

TK50

TK50 DATA RELIAB TST
CZTKBB0

AH-T774B-MC
1 OF 1 OCT 1985
COPYRIGHT © 1985

digital
MADE IN USA

The image displays a grid of 140 small data tables, arranged in 10 rows and 14 columns. Each table contains technical data, likely test results, with various columns and rows of text and numbers. The text is very small and difficult to read, but the layout is consistent across the grid. The tables appear to be organized into sections, with some tables having headers or sub-headers. The overall appearance is that of a data log or test report page.

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

.REM @

IDENTIFICATION

PRODUCT CODE: AC - T773B - MC
PRODUCT NAME: CZTKBBO TK50 DATA RELIABILITY
PRODUCT DATE: JULY - 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 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 XXDP+ environment.

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

The PDP11 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 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 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 Mass 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) /74500 ?
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
313 ENABLE TIME OF DAY CLOCK (L) N ?
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. Anytime 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
335 CHANGE CONTROLLER PARAMETERS (L) N ?
336

337 The default answer (no) prohibits the asking of the controller
338 parameter questions. To change the controller parameters type a Y.
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
356
357
358
359
360
361
362
363
364
365
366
367
368
369
370
371
372
373
374
375
376
377
378
379
380
381
382
383
384
385
386
387
388
389
390
391
392
393
394
395
396
397
398
399
400
401
402
403
404
405
406
407
408
409

If answered "yes" (default) the program will enable the controller's pad blocking algorithms to assist in streaming

CHANGE PRINTING PARAMETERS (L) N ?

The default answer (no) prohibits the asking of the printing parameter questions. To change the printing parameters type a Y.

ENABLE SOFT ERROR REPORTS (L) N ?

The default answer (no) inhibits the printing, but not the tallying of soft errors as reported by the subsystem. Answering the question "yes" will result in detailed error reports on the terminal for each recoverable data error.

ENABLE READ SOFT ERRORS ONLY (L) N ?

This question will only be asked when the above question is answered no. This question allows the operator to enable print outs on read soft errors only. The default answer is to inhibit all soft error printouts.

CLEAR MEDIA TABLE ON EVERY PASS (L) N ?

The default answer (no) allows the tallying of media defects over multiple passes. By answering the question yes, the operator can then print the table on every pass and see how the defects are affected by passing over the heads.

ENABLE PRINTING OF MEDIA DEFECTS TABLE (L) N ?

The default answer (no) inhibits the printing, but not the tallying of media defects as reported in the soft error reports by the subsystem. If the default answer is used the table may still be printed by giving the PRINT command at the supervisor prompt (DS>) after the termination of the program. Answering the question "yes" will cause the printing of the table after every pass and after a control C (C) is issued.

ENABLE CLEAR STATS ON FATAL ERROR (L) N ?

The default answer (no) allows the accumulation of statistics from pass to pass. An answer of "yes" results in the clearing of a device's statistical matrix following any error that results in the unit's being dropped from the test sequence for the rest of the current pass. This action is intended for use primarily by Springfield volume manufacturing.

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 (O) 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 cause 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

524 ways. For a branch command, the user would specify the command number
 525 to which (s)he wishes to branch. For the delay command, the value
 526 entered here is the relative delay length, with larger numbers
 527 producing longer delays. User experimentation may be required to
 528 produce desired delay.
 529

530
 531 ITERATION COUNT (D) 1 ?
 532

533 This field allows the user to specify how many times the command
 534 should be issued before the program issues the next command. The
 535 value is entered in decimal.
 536

537
 538 Additional Commands
 539

540 This same sequence of four questions will be repeated up to 6 more
 541 times, allowing the operator to create a command table with seven
 542 unique commands. The only noticeable difference in question format is
 543 that each time the command question is asked, its relative position in
 544 the Test 7 command table is identified.
 545

546 2.5 Allowable Commands

547
 548 The following commands are supported by Test 7. Please remember
 549 that the octal values are defined by the program and have no numerical
 550 correlation to TMSCP command opcodes. Also note that the diagnostic
 551 does not check for legality of the value entered or for valid command
 552 sequences. Operator error in either of these cases could result in
 553 bizarre program behavior.
 554

555 Octal	Command	Description
556 10	RD	Read forward
557 20	WR	Write
558 30	CMP	Compare host data
559 40	ACC	Access
560 50	SPC	Space records
561 51	SCR	Space records reverse
562 60	SKP	Skip tape marks
563 61	SKR	Skip tape marks reverse
564 70	SPO	Space objects
565 71	SPR	Space objects reverse
566 100	WTM	Write tape mark
567 160	REW	Rewind
568 300	BR	Branch - item count specifies destination
569 310	DLY	Delay - item count specifies relative delay
570 377	END	End of sequence - necessary if sequence has less than 7 commands

571 2.6 Supervisor Run Time Flags

572
 573 This program will support all of the PDP11 Diagnostic Supervisor
 574 flags except for those mentioned here.
 575

576
 577 LOE - Loop on Error - This flag will not be supported by this program.
 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
669

670 3.5 Error Log Packets

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

673 Retriable Data
674 Hard CRC
675 Data Underrun
676 Data Overrun
677 ECC Corrected
678 CRC Error on ECC Block
679
680
681
682
683
684

685 3.6 Program Detected Error Conditions

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

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

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

	Octal	Hex	Description
727			
728			
729			
730	1	01	Write lock violation
731	2	02	Drive fault
732	4	04	Communication exception (timeout, etc.)
733	6	06	Wrong track error (following a turnaround)
734	10	08	No cable or drive powered off
735	20	10	Synchronization failure - write/read
736	23	13	
737	44	22	
738	45	23	
739	47	27	
740	201	81	Failure to load to BOT
741	202	82	Failure to unload tape into cartridge
742	203	83	General motor or tach failure
743	204	84	Motor A failure
744	205	85	Motor B failure
745	206	86	Drive lost control of tape or bad tach
746	207	87	Excessive drag in tape transport
747	210	88	Failure to stop tape or remain stopped
748	211	89	Cartridge insert error
749	212	8A	Cartridge extract error
750	213	8B	CU attempted to move tape with drive in error
751	214	8C	Deceleration timeout error
752	215	8D	Second attempt to balance reels in init failed
753	220	90	8155 RAM memory failure in self-test
754	221	91	8155 timer failure
755	222	92	Read amplit (Hd 1) too low in calibrate
756	223	93	Read amplit (Hd 2) too low in calibrate
757	225	95	EOT sensed in R/W/S
758	226	96	BOT sensed in R/W/S
759	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)
000050(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.

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(0)      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: 15(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 23. Skip a tape mark,
- 985
- 986 24. Read the sixth record set,
- 987
- 988 25. Skip two tape marks,
- 989
- 990 26. Space objects reverse to the end of the second file set,
- 991
- 992 27. Skip a tape mark,
- 993
- 994 28. Read the third file set,
- 995
- 996 29. Rewind tape,
- 997
- 998
- 999

5.2 Test 2 Quick Verify Read/Write Test

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

- 1000
- 1001
- 1002
- 1003
- 1004 1. Write record set,
- 1005
- 1006 2. Write LEOT,
- 1007
- 1008 3. Rewind,
- 1009
- 1010 4. Reposition to just written record set,
- 1011
- 1012 5. Read the current record set,
- 1013
- 1014 6. Skip to LEOT.
- 1015

1016 for 5 iterations or until fatal error is encountered. This test
1017 permits retries, fixed record length (4096 bytes decimal), fixed
1018 number of records/set (250), and predetermined data patterns. This
1019 test will execute in a round-robin manner.
1020

5.3 Test 3 Complex Read/Write Test

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

- 1021
- 1022
- 1023
- 1024
- 1025
- 1026
- 1027
- 1028 1. Write N records,
- 1029
- 1030 2. Write a tape mark,
- 1031
- 1032 3. Repeat 1 and 2 until EOT is reached,
- 1033
- 1034 4. Write 2 tape marks (LEOT),
- 1035
- 1036 5. Rewind,
- 1037
- 1038 6. Read N records,
- 1039
- 1040 7. Space 1 record (should see unexpected tape mark)

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$DEPO::      ;0
      002011      060      .ASCII /O/
      002012      L$UNIT::      ;NUMBER OF UNITS
      002012      000001      .WORD T$PTHV
      002014      L$TIML::      ;LONGEST TEST TIME
      002014      015000      .WORD 15000
      002016      L$HPCP::      ;POINTER TO H.W. QUES.
      002016      046244      .WORD L$HARD
      002020      L$SPCP::      ;POINTER TO S.W. QUES.
      002020      046332      .WORD L$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$LADP::      ;DIAG. END ADDRESS
      002026      115744      .WORD L$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::	.WORD	0	;FLAGS DESCRIBE HOW IT WAS SETUP
002044	000000		.WORD	0	
002046		L\$EXP1::	.WORD	0	;EXPANSION WORD
002046	000000		.WORD	0	
002050		L\$MREV::	.BYTE		;SVC REV AND EDIT *
002050	004		.BYTE		C\$REVISION
002051	000		.BYTE		C\$EDIT
002052		L\$EF::	.WORD	0	;DIAG. EVENT FLAGS
002052	000000		.WORD	0	
002054	000000		.WORD	0	
002056		L\$SPC::	.WORD	0	
002056	000000		.WORD	0	
002060		L\$DEVP::	.WORD		; POINTER TO DEVICE TYPE LIST
002060	002174		.WORD	L\$DVTYP	
002062		L\$REPP::	.WORD		;PTR. TO REPORT CODE
002062	034216		.WORD	L\$RPT	
002064		L\$EXP4::	.WORD	0	
002064	000000		.WORD	0	
002066		L\$EXP5::	.WORD	0	
002066	000000		.WORD	0	
002070		L\$AUT::	.WORD		;PTR. TO ADD UNIT CODE
002070	000000		.WORD	0	
002072		L\$DUT::	.WORD		;PTR. TO DROP UNIT CODE
002072	040706		.WORD	L\$DU	
002074		L\$LUN::	.WORD		;LUN FOR EXERCISERS TO FILL
002074	000000		.WORD	0	
002076		L\$DESP::	.WORD		;POINTER TO DIAG. DESCRIPTION
002076	002142		.WORD	L\$DESC	
002100		L\$LOAD::	EMT	E\$LOAD	;GENERATE SPECIAL AUTOLOAD EMT
002100	104035		EMT	E\$LOAD	
002102		L\$ETP::	.WORD		;POINTER TO ERR_TBL
002102	013262		.WORD	L\$ERR_TBL	
002104		L\$ICP::	.WORD		;PTR. TO INIT CODE
002104	037012		.WORD	L\$INIT	
002106		L\$CCP::	.WORD		;PTR. TO CLEAN-UP CODE
002106	040400		.WORD	L\$CLEAN	
002110		L\$ACP::	.WORD		;PTR. TO AUTO CODE
002110	040376		.WORD	L\$AUTO	
002112		L\$PRT::	.WORD		;PTR. TO PROTECT TABLE
002112	021036		.WORD	L\$PROT	
002114		L\$TEST::	.WORD		;TEST NUMBER
002114	000000		.WORD	0	
002116		L\$DLY::	.WORD		;DELAY COUNT
002116	000000		.WORD	0	
002120		L\$HIME::	.WORD		;PTR. TO HIGH MEM
002120	000000		.WORD	0	


```

1187          .SBTTL DISPATCH TABLE
1188
1189          ;++
1190          ; THE DISPATCH TABLE CONTAINS THE STARTING ADDRESS OF EACH TEST.
1191          ; IT IS USED BY THE SUPERVISOR TO DISPATCH TO EACH TEST.
1192          ;--
1193
1194          DISPATCH 7
          002122 000007          .WORD 7
          002124          L$DISPATCH::
          002124 040722          .WORD T1
          002126 043124          .WORD T2
          002130 043620          .WORD T3
          002132 044276          .WORD T4
          002134 044656          .WORD T5
          002136 045176          .WORD T6
          002140 045654          .WORD T7
1195
1196
1197          002142          DESCRIPT          <CZTKBAO DATA RELIABILITY>
          002142          L$DESC::
          002142 103 132 124          .ASCIZ /CZTKBAO DATA RELIABILITY/
          .EVEN
1198
1199          ;
1200          ; NAMES OF DEVICES SUPPORTED BY PROGRAM
1201          ;
1202
1203          002174          DEVTYP <TK50>
          002174          L$DVTYP::
          002174 124 113 065          .ASCIZ *TK50*
          .EVEN
1204

```

```
1206          .SBTTL  DEFAULT HARDWARE P-TABLE
1207
1208          ;**
1209          ; THE DEFAULT HARDWARE P TABLE CONTAINS DEFAULT VALUES OF
1210          ; THE TEST-DEVICE PARAMETERS.  THE STRUCTURE OF THIS TABLE
1211          ; IS IDENTICAL TO THE STRUCTURE OF THE HARDWARE P-TABLES.
1212          ;--
1213
1214 002202          BGNHW  DFPTBL
          002202 000002          .WORD  L10000-L$HW/2
          002204          L$HW::
          002204          DFPTBL::
1215
1216 002204 174500          174500          ;TKIP ADDRESS
1217 002206 000000          0          ;T/MSCP UNIT NUMBER
1218
1219 002210          ENDPHW
          002210          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-L1$SW/2
          002212          L$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          %COND::          .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			

1301
1312
1313
1388
1389 002314
1390
1391
1392
1393
1394
1395
1396
1397
1398
1399
1400
1401
1402 002314

```
.TITLE GLOBAL AREAS
.SBTTL GLOBAL EQUATES SECTION

      BGNMOD

;/ ** */
; 1.0 SUPERVISOR DEFINED LITERALS
;/ ** */
```

```
:**
; THE GLOBAL EQUATES SECTION CONTAINS PROGRAM EQUATES THAT
; ARE USED IN MORE THAN ONE TEST.
;
```

EQUALS

```
;
; BIT DIFINITIONS
;
```

100000	BIT15==	100000
040000	BIT14==	40000
020000	BIT13==	20000
010000	BIT12==	10000
004000	BIT11==	4000
002000	BIT10==	2000
001000	BIT09==	1000
000400	BIT08==	400
000200	BIT07==	200
000100	BIT06==	100
000040	BIT05==	40
000020	BIT04==	20
000010	BIT03==	10
000004	BIT02==	4
000002	BIT01==	2
000001	BIT00==	1

```
;
BIT9== BIT09
BIT8== BIT08
BIT7== BIT07
BIT6== BIT06
BIT5== BIT05
BIT4== BIT04
BIT3== BIT03
BIT2== BIT02
BIT1== BIT01
BIT0== BIT00
;
```

```
; EVENT FLAG DEFINITIONS
; EF32:EF17 RESERVED FOR SUPERVISOR TO PROGRAM COMMUNICATION
;
```

000040	EF.START==	32.	; BIT POSITION IN SECOND STATUS WORD
000037	EF.RESTART==	31.	; (100000) START COMMAND WAS ISSUED
000036	EF.CONTINUE==	30.	; (040000) RESTART COMMAND WAS ISSUED
000035	EF.NEW==	29.	; (020000) CONTINUE COMMAND WAS ISSUED
			; (010000) A NEW PASS HAS BEEN STARTED

```
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
```

1403

```

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
    
```

GLOBAL EQUATES SECTION

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					


```

1478 ;/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*
1479 ; 3.0 TMSCP END PACKET LITERALS
1480 ;/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*
1481
1482 ; 3.1 END PACKET FLAGS
1483 EF.LOG == 000040 ;ERROR LOG GENERATED
1484 EF.SEX == 000020 ;SERIOUS EXCEPTION
1485 EF.EOT == 000010 ;EOT ENCOUNTERED
1486
1487 ; 3.2 CONTROLLER FLAGS
1488 CF.ATN == 000200 ;ENABLE ATTENTION INTERRUPTS
1489 CF.MSC == 000100 ;ENABLE MISCELLANEOUS ERROR LOG MESSAGES
1490 CF.OTH == 000040 ;ENABLE OTHER HOSTS ERROR LOG MESSAGES
1491 CF.THS == 000020 ;ENABLE THIS HOSTS ERROR LOG MESSAGES
1492
1493 ; 3.3 UNIT FLAGS
1494 UF.WPH == 020000 ;WRITE PROTECT (HARDWARE)
1495 UF.VSU == 020000 ;VARIABLE SPEED UNIT
1496 UF.WPS == 010000 ;SOFTWARE WRITE PROTECT
1497 UF.RMV == 000200 ;REMOVABLE MEDIA
1498 UF.VSS == 000040 ;VARIABLE SPEED MODE SUPPRESSION
1499 UF.CMW == 000002 ;COMPARE WRITES
1500 UF.CMR == 000001 ;COMPARE READS
1501
1502 ; 3.4 GENERIC END MESSAGE OFFSETS
1503 P.CRF == 0 ;COMMAND REFERENCE NUMBER
1504 P.UNIT == 4 ;UNIT NUMBER
1505 P.OPCD == 10 ;ENDCODE
1506 P.FLGS == 11 ;END MESSAGE FLAGS
1507 P.STS == 12 ;STATUS
1508 P.BCNT == 14 ;BYTE COUNT
1509
1510 ; 3.5 ACCESS,COMPARE HOST DATA,READ,REPOSITION,WRITE,-
1511 ; AND WRITE TAPE MARK END MESSAGE OFFSETS
1512 P.POS == 34 ;POSITION (OBJECT COUNT)
1513 P.TRBC == 40 ;TAPE RECORD BYTE COUNT
1514
1515 ; 3.6 GET COMMAND STATUS END PACKET OFFSETS
1516 P.OTRF == 14 ;OUTSTANDING COMMAND REFERENCE NUMBER
1517 P.CMST == 20 ;COMMAND STATUS
1518
1519 ; 3.7 GET UNIT STATUS END MESSAGE OFFSETS
1520 P.MLUN == 14 ;MULTI-UNIT CODE
1521 P.UNFL == 16 ;UNIT FLAGS
1522 P.UNTI == 24 ;UNIT IDENTIFIER
1523 P.MEDI == 34 ;MEDIA IDENTIFIER
1524 P.FORM == 40 ;TAPE FORMAT
1525 P.SPED == 42 ;SPEED
1526 P.FMEM == 44 ;FORMAT MENU
1527 P.CSVR == 50 ;CONTROLLER SOFTWARE VERSION
1528 P.CHVR == 51 ;CONTROLLER HARDWARE VERSION
1529 P.USVR == 52 ;UNIT SOFTWARE VERSION
1530 P.UHVR == 53 ;UNIT HARDWARE VERSION
1531
1532 ; 3.8 OP.ONL, OP.AVL AND OP.SUC MESSAGE OFFSETS
1533 P.MXWR == 44 ;MAXIMUM WRITE RECORD SIZE
1534 P.NREC == 50 ;NOISE RECORD
    
```

```
1535
1536 ; 3.9 REPOSITION MESSAGE OFFSETS
1537 P.RCSK == 14 ;RECORDS SKIPPED
1538 P.TMSK == 20 ;TAPE MARKS SKIPPED
1539
1540 ; 3.10 SET CONTROLLER CHARACTERISTICS MESSAGE OFFSETS
1541 P.VRSN == 14 ;MSCP VERSION
1542 P.CNTF == 16 ;CONTROLLER FLAGS
1543 P.HTMO == 20 ;HOST TIMEOUT
1544 P.TIME == 24 ;QUAD-WORD TIME AND DATE
```

```

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 ;CONTROLLER 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 ;UNJT 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 TRANSMISION 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	;UNRECORVERABLE 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	RDTB	**	000002	;RECORD DATA TRUNCATED
1712	000004	SEXB	**	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

Address	Value	Symbol	Offset	Description
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 DROP 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	OVERRUN	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

```

1861 ;/★\★\★\★\★\★\★\★\★\★\★\★\★\★\★\★\★\★\★\★\★\★\★\★\★\★\
1862 : 7.0 PROGRAM PRIMITIVES
1863 ;/★\★\★\★\★\★\★\★\★\★\★\★\★\★\★\★\★\★\★\★\★\★\★\★\★\★
1864
1865 ; 7.1 COMMAND PRIMITIVE LITERALS
1866 NUL == 000 ;NULL
1867 RD == 010 ;READ
1868 RDR == 011 ;READ REVERSE
1869 WR == 020 ;WRITE
1870 CMP == 030 ;COMPARE HOST DATA
1871 CMR == 031 ;COMPARE HOST DATA REVERSE
1872 ACC == 040 ;ACCESS
1873 ACR == 041 ;ACCESS REVERSE
1874 SPC == 050 ;SPACE RECORDS
1875 SCR == 051 ;SPACE RECORDS REVERSE
1876 SCD == 052 ;SPACE TO LEOT
1877 SKP == 060 ;SKIP TAPE MARKS
1878 SKR == 061 ;SKIP TAPE MARKS REVERSE
1879 SKD == 062 ;SKIP TO LEOT
1880 SPO == 070 ;SPACE OBJECTS
1881 SPR == 071 ;SPACE OBJECTS REVERSE
1882 WTM == 100 ;WRITE TAPE MARKS
1883 ERS == 110 ;ERASE
1884 ERI == 113 ;ERASE IMMEDIATE
1885 ERG == 120 ;ERASE GAPS
1886 AVL == 130 ;AVAILABLE
1887 AVU == 134 ;AVAILABLE UNLOAD
1888 ONL == 140 ;ONLINE
1889 SUC == 150 ;SET UNIT CHARACTERISTICS
1890 SUW == 155 ;SET UNIT CHARA. W/WRITE PROTECT
1891 REW == 160 ;REWIND
1892 RWI == 163 ;REWIND IMMEDIATE
1893 INT == 170 ;INITIALIZATION
1894 ABO == 200 ;ABORT
1895 GCS == 210 ;GET COMMAND STATUS
1896 GUS == 220 ;GET UNIT STATUS
1897 SCC == 230 ;SET CONTROLLER CHARACTERISTICS
1898
1899 ; 7.2 COMMAND PRIMITIVE MODIFIER LITERALS
1900 REVBIT == 1 ;REVERSE MODIFIER
1901 EOTBIT == 2 ;DETECT LEOT MODIFIER
1902 IMMBIT == 3 ;IMMEDIATE MODIFIER
1903 UNLBIT == 4 ;UNLOAD MOIFIER
1904 WPRBIT == 5 ;WRITE PROTECT MODIFIER
1905
    
```

```
1907      .SBTTL GLOBAL DATA SECTION
1908
1909      ;/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*
1910      ; 8.0 PROGRAM AND DRIVER DATA STRUCTURES
1911      ;/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*
1912
1913      ; 8.1 UNIT LUN BLOCKS
1914 002314 000000 000000 000000 LUN0:: .WORD 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
1915 002404 000000 000000 000000          DRSPB0
1916 002472 004032          DRBEN0
1917 002474 005072          RDSRG0
1918 002476 010232          RDREN0
1919 002500 010272          CDSRG0
1920 002502 010272          CDREN0
1921 002504 010312
1922
1923 002506 000000 000000 000000 LUN1:: .WORD 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
1924 002576 000000 000000 000000          DRSPB1
1925 002664 005072          DRBEN1
1926 002666 006132          RDSRG1
1927 002670 010316          RDREN1
1928 002672 010356          CDSRG1
1929 002674 010356          CDREN1
1930 002676 010376
1931
1932 002700 000000 000000 000000 LUN2:: .WORD 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
1933 002770 000000 000000 000000          DRSPB2
1934 003056 006132          DRBEN2
1935 003060 007172          RDSRG2
1936 003062 010402          RDREN2
1937 003064 010442          CDSRG2
1938 003066 010442          CDREN2
1939 003070 010462
1940
1941 003072 000000 000000 000000 LUN3:: .WORD 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
1942 003162 000000 000000 000000          DRSPB3
1943 003250 007172          DRBEN3
1944 003252 010232          RDSRG3
1945 003254 010466          RDREN3
1946 003256 010526          CDSRG3
1947 003260 010526          CDREN3
1948 003262 010546
1949
1950      ; 8.2 PROGRAM COMMAND BUFFERS
1951 003264 PCMDBF:: .BLKW 24. ;PROGRAM COMMAND RING
1952      003344 PCBEND= ;PROGRAM COMMAND RING END
1953 003344 DUMPKT:: .BLKW 2. ;PROGRAM DUMMY COMMAND PACKET
1954 003350 DRINUS:: .BLKW 4. ;DRIVE IN USE TABLE
1955 003360 UNTTBL:: .BLKW 8. ;TEMPORY MEDIA STAT STORAGE
1956
1957      ; 8.3 PROGRAM VARIABLES
1958 003400 CMDCNT:: .WORD 0 ;COMMAND COUNT
1959 003402 RSPCNT:: .WORD 0 ;RESPONSE COUNT
1960 003404 COUNT:: .WORD 0 ;COMMAND LOOP COUNTER
1961 003406 RESPON:: .WORD 0 ;RESPONSE STATUS
1962 003410 BRcnt: .WORD 0 ;BRANCH COUNTER
1963 003412 HNDLRP:: .WORD 0 ;NUMBER OF RESPONSES
```

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	CMPEER::	.WORD	0 ;NUMBER OF BYTES IN ERROR
1981	003446		BYTADD::	.BLKW	10. ;SAVE TABLE FOR BYTE IN ERROR ADDRESS
1982		003472	TBLEND ==		
1983	003472		DATBL::	.BLKW	10. ;END OF BYTE ADDRESS TABLE
1984	003516	000000	PCFLAG::	.WORD	0 ;SAVE TABLE FOR BYTE IN ERROR DATA
1985					
1986	003520	000000	OBJECT::	.WORD	0 ;PROGRAM CONTROL FLAGS
1987	003522	000000	PASCNT::	.WORD	0 ;OBJECT COUNTER FOR TEST 2
1988	003524	000000	UDROP::	.WORD	0 ;PASS COUNTER
1989	003526	000000	UEOT::	.WORD	0 ;NUMBER OF DROPPED UNITS
1990					
1991	003530	000000	R8::	.WORD	0 ;COUNT OF UNITS AT EOT
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 CDPEN1::      ;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 000J00 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           ; I/O STATUS ERROR INFORMATION TABLE
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  SEQER
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           ; PROGRAM DETECTED ERROR INFORMATION TABLE
2119
2120 010652     001      CMDT: .BYTE  DEVFAT      ;INVALID COMMAND ISSUED
2121 010653     102      .BYTE  OTHER
2122 010654     000007   .WORD  7
2123 010656     011502   .WORD  CMDER
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	POL:: .BYTE	DEVFAT	;TAPE POSITION LOST
2217	011053	076	.BYTE	RMSP0S	
2218	011054	000027	.WORD	23.	
2219	011056	012045	.WORD	P0LER	
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
2255 011132 002
2256 011133 106
2257 011134 000035
2258 011136 013172
2259 011140 014502
2260
2261 011142
2262 011142 002
2263 011143 106
2264 011144 000036
2265 011146 013233
2266 011150 014502
2267
2268 011152
2269 011152 002
2270 011153 070
2271 011154 000037
2272 011156 013030
2273 011160 014502
2274
2275 011162
2276 011162 002
2277 011163 106
2278 011164 000040
2279 011166 013125
2280 011170 014502
2281
2282 011172
2283 011172 002
2284 011173 106
2285 011174 000041
2286 011176 013145
2287 011200 014502
2288
2289 011202
2290 011202 002
2291 011203 106
2292 011204 000042
2293 011206 013057
2294 011210 014502
2295
2296 011212
2297 011212 002
2298 011213 054
2299 011214 000043
2300 011216 013103
2301 011220 014502
2302
2303 011222
2304 011222 002
2305 011223 060
2306 011224 000043
2307 011226 013103
    
```

<pre> CNTLER: .BYTE HARD .BYTE NOERR .WORD 29. .WORD CNTLEL .WORD ERLGER BADERL: .BYTE HARD .BYTE NOERR .WORD 30. .WORD BADEL .WORD ERLGER CMPERL: .BYTE HARD .BYTE DTCPER .WORD 31. .WORD CMPEL .WORD ERLGER DRVERL: .BYTE HARD .BYTE NOERR .WORD 32. .WORD DRVEL .WORD ERLGER CNTERL: .BYTE HARD .BYTE NOERR .WORD 33. .WORD CNTEL .WORD ERLGER LGPERL: .BYTE HARD .BYTE NOERR .WORD 34. .WORD LGPEL .WORD ERLGER RDSTEL: .BYTE HARD .BYTE H1READ .WORD 35. .WORD DSTEL .WORD ERLGER WDSTEL: .BYTE HARD .BYTE H1WRIT .WORD 35. .WORD DSTEL </pre>	<pre> ;CONTROLLER (LAST FAIL) ERROR LOG ;HOST MEMORY ACCSESS ERROR ;COMPARE ERROR ;DRIVE ERROR LOG ;CONTROLLER ERROR ;LONG GAP ENCOUNTERED ;READ DATA SYNC NOT FOUND ;WRITE DATA SYNC NOT FOUND </pre>
--	---

2308	011230	014502	.WORD	ERLGER	
2309					
2310	011232				
2311	011232	003	CORERL:	.BYTE	SOFT
2312	011233	044		.BYTE	S1WRIT
2313	011234	000044		.WORD	36.
2314	011236	012627		.WORD	COREL
2315	011240	014502		.WORD	ERLGER
2316					
2317	011242		UWEERL:	.BYTE	HARD
2318	011242	002		.BYTE	H1WRIT
2319	011243	060		.WORD	37.
2320	011244	000045		.WORD	UWEEL
2321	011246	012760		.WORD	ERLGER
2322	011250	014502			
2323					
2324	011252		RTYERL:	.BYTE	SOFT
2325	011252	003		.BYTE	S1READ
2326	011253	040		.WORD	42.
2327	011254	000052		.WORD	RTYEL
2328	011256	012574		.WORD	ERLGER
2329	011260	014502			
2330					
2331	011262		UREERL:	.BYTE	HARD
2332	011262	002		.BYTE	H1READ
2333	011263	054		.WORD	38.
2334	011264	000046		.WORD	UREEL
2335	011266	012740		.WORD	ERLGER
2336	011270	014502			
2337					
2338	011272		ECBERL:	.BYTE	SOFT
2339	011272	003		.BYTE	RTYEC1
2340	011273	064		.WORD	39.
2341	011274	000047		.WORD	ECBEL
2342	011276	013001		.WORD	ERLGER
2343	011300	014502			
2344					
2345	011302		ECTERL:	.BYTE	SOFT
2346	011302	003		.BYTE	EC1COR
2347	011303	050		.WORD	40.
2348	011304	000050		.WORD	ECTEL
2349	011306	012707		.WORD	ERLGER
2350	011310	014502			
2351					
2352	011312		ECDERL:	.BYTE	SOFT
2353	011312	003		.BYTE	EC1COR
2354	011313	050		.WORD	41.
2355	011314	000051		.WORD	ECDEL
2356	011316	012663		.WORD	ERLGER
2357	011320	014502			
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:  .ASCII? /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  THER:   .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  DRJEL:  .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          ERRTBL                                ;GENERIC ERROR TABLE
          L$ERRTBL::
          ERRTP::                                .WORD    0
          ERRNBR::                               .WORD    0
          ERRMSG::                               .WORD    0
          ERRBLK::                              .WORD    0
2482
2483          BGNMSG DEVERR
          DEVERR::
          PUSH    <R1,R5>                        ;SAVE R1 AND R5
          MOV     R1,-(SP)                        ;;PUSH R1 ON STACK
          MOV     R5,-(SP)                        ;;PUSH R5 ON STACK
          MOV     R3SAVE,R3                      ;RESTORE R3
          MOV     R4SAVE,R4                      ;RESTORE R4
          MOV     CMD(R2),R5                     ;GET THE COMMAND PRIMITIVE
          BIC     #177407,R5                     ;CLEAR MODIFIERS
          ASR     R5                             ;THE PRIMITIVE
          ADD     #CMDASC,R5                      ;PUT ADDRESS IN R5
          MOV     CMD(R2),R8                     ;GET THE PRIMITIVE AGAIN
          BIC     #177770,R8                     ;SAVE THE LAST 3 BITS
          BNE     1$                             ;BRANCH IF NOT ZERO
          PRINTB #ERR00,R5,TKUNIT(R4)
          MOV     TKUNIT(R4),-(SP)
          MOV     R5,-(SP)
          MOV     #ERR00,-(SP)
          MOV     #3,-(SP)
          MOV     SP,R0
          TRAP   C$PNTB
          ADD     #10,SP
          BR     6$                               ;GO PRINT THE REST OF THE MESSAGE
          CMP     #REVBIT,R8                     ;IS IT A REVERSE ?
          BNE     2$                             ;BRANCH IF NOT
          PRINTB #ERR01,R5,TKUNIT(R4)
          MOV     TKUNIT(R4),-(SP)
          MOV     R5,-(SP)
          MOV     #ERR01,-(SP)
          MOV     #3,-(SP)
          MOV     SP,R0
          TRAP   C$PNTB
          ADD     #10,SP
          BR     6$                               ;GO PRINT THE REST OF THE MESSAGE
          BIT     #EOTBIT,R8                     ;IS IT A DETECT LEOT ?
          BEQ     3$                             ;BRANCH IF NOT
          PRINTB #ERR02,R5,TKUNIT(R4)
          MOV     TKUNIT(R4),-(SP)
          MOV     R5,-(SP)
          MOV     #ERR02,-(SP)
          MOV     #3,-(SP)
          MOV     SP,R0
2484 013262 013262 000000
          013264 000000
          013266 000000
          013270 000000
2485 013272 013272 010146
          013274 010546
          013276 013703 003560
          013302 013704 003562
          013306 116205 000000
          013312 042705 177407
          013316 006205
          013320 062705 011322
          013324 016237 000000 003530
          013332 042737 177770 003530
          013340 001014
          013342 016446 000004
          013346 010546
          013350 012746 015770
          013354 012746 000003
          013360 010600
          013362 104414
          013364 062706 000010
          013370 000513
          013372 022737 000001 003530 1$:
          013400 001014
          013402 016446 000004
          013406 010546
          013410 012746 016040
          013414 012746 000003
          013420 010600
          013422 104414
          013424 062706 000010
          013430 000473
          013432 032737 000002 003530 2$:
          013440 001414
          013442 016446 000004
          013446 010546
          013450 012746 016116
          013454 012746 000003
          013460 010600
    
```

	013462	104414			TRAP	C#PNTB	
	013464	062706	000010		ADD	#10,SP	
2503	013470	000453			BR	6#	;GO PRINT THE REST OF THE MESSAGE
2504	013472	022737	000003	003530	3#:	CMP	#IMMBIT,R8
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,R0	
	013522	104414			TRAP	C#PNTB	
	013524	062706	000010		ADD	#10,SP	
2507	013530	000433			BR	6#	;GO PRINT THE REST OF THE MESSAGE
2508	013532	022737	000004	003530	4#:	CMP	#UNLBIT,R8
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,R0	
	013562	104414			TRAP	C#PNTB	
	013564	062706	000010		ADD	#10,SP	
2511	013570	000413			BR	6#	;GO PRINT THE REST OF THE MESSAGE
2512	013572				5#:	PRINTB	#ERR05,R5,TKUNIT(R4)
	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,R0	
	013612	104414			TRAP	C#PNTB	
	013614	062706	000010		ADD	#10,SP	
2513	013620				6#:	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,R0	
	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,R0	
	013674	104414			TRAP	C#PNTB	
	013676	062706	000006		ADD	#6,SP	
2517	013702				7#:	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,R0	
	013724	104414			TRAP	C#PNTB	

	013726	062706	000010			ADD	#10,SP	
2518	013732	032737	000010	003516		BIT	#DRERFL,PCFLAG	;WAS IT A DRIVE ERROR ?
2519	013740	001417				BEQ	8\$;NO, GO PRINT TIME
2520	013742	042737	000010	003516		BIC	#DRERFL,PCFLAG	;CLEAR THE DRIVE ERROR FLAG
2521	013750					PRINTB	#ERR09,HIHEX,LOHEX	;PRINT THE DRIVE ERROR CODE
	013750	013746	003552			MOV	LOHEX,-(SP)	
	013754	013746	003554			MOV	HIHEX,-(SP)	
	013760	012746	016556			MOV	#ERR09,-(SP)	
	013764	012746	000003			MOV	#3,-(SP)	
	013770	010600				MOV	SP,RO	
	013772	104414				TRAP	C\$PNTB	
	013774	062706	000010			ADD	#10,SP	
2522	014000	122737	000006	010546	8\$:	CMPB	#INTERR,IOSTAT	;IS IT A PORT INIT FAILURE ?
2523	014006	001001				BNE	9\$;KEEP GOING IF IT ISN'T
2524	014010	000404				BR	10\$;GO PRINT SA CONTENTS
2525	014012	122737	000003	010546	9\$:	CMPB	#IOPDRE,IOSTAT	;IS IT A PORT DETECTED FAILURE ?
2526	014020	001014				BNE	11\$;KEEP GOING IF IT ISN'T
2527	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,RO	
	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	CMPERR	;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,RO	
	014100	104415				TRAP	C\$PNTX	
	014102	062706	000004			ADD	#4,SP	
2534	014106					PRINTX	#ERR12	
	014106	012746	016675			MOV	#ERR12,-(SP)	
	014112	012746	000001			MOV	#1,-(SP)	
	014116	010600				MOV	SP,RO	
	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,RO	
	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

    BNE PRIPCK
    TST CMPERR ;KEEP PRINTING TILL ALL DONE
    BEQ DEVEXT ;WAS THIS A COMPARE ERROR ?
    PUSH <R2> ;GET OUT IF IT WASN'T
    MOV #BYTADD,R1 ;SAVE R2
    MOV #DATBL,R2 ;POINT R1 TO THE BYTE ADDRESS TABLE
    MOV CMPERR,R5 ;POINT R2 TO THE WRITE DATA TABLE
    PRINTX #ERR14 ;LET R5 = THE NUMBER OF BYTES IN ERROR
    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$: PRINTX #ERR16,(R1),<B,ONE(R2)>,<B,(R2)>
    CLR -(SP)
    BISB (R2),(SP)
    CLR -(SP)
    BISB ONE(R2),(SP)
    MOV (R1),-(SP)
    MOV #ERR16,-(SP)
    MOV #4,-(SP)
    MOV SP,R0
    TRAP C#PNTX
    ADD #12,SP
    DEC CMPERR ;SUBTRACT 1 FROM NUMBER OF ERRORS
    BEQ CPRIEX ;GO PRINT TOTAL NUMBER IN ERROR
    TST (R1)+ ;POINT R1 TO THE NEXT ADDRESS
    TST (R2)+ ;POINT R2 TO THE NEXT DATA
    CMP #TBLEND,R1 ;HAVE WE PRINTED THE WHOLE TABLE ?
    BNE 1$ ;NO CONTINUE
CPRIEX: PRINTX #ERR17,R5
    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,R0
	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	MOV	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	MOV	R11,R8		;GET THE PRIMITIVE AGAIN
2580	014542	042737	177770	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	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	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	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	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	JMP	MSGEXT	;GET OUT
2615	015476		PKPRNT: PRINTB	#ERR08,OBOFFH(R2),OBOFFL(R2)	
	015476	016246	MOV	OBOFFL(R2),-(SP)	
	015502	016246	MOV	OBOFFH(R2),-(SP)	
	015506	012746	MOV	#ERR08,-(SP)	
	015512	012746	MOV	#3,-(SP)	
	015516	010600	MOV	SP,R0	
	015520	104414	TRAP	C#PNTB	
	015522	062706	ADD	#10,SP	
2616	015526		PRINTX	#ERL08	
	015526	012746	MOV	#ERL08,-(SP)	
	015532	012746	MOV	#1,-(SP)	
	015536	010600	MOV	SP,R0	
	015540	104415	TRAP	C#PNTX	
	015542	062706	ADD	#4,SP	
2617	015546		PRINTX	#ERR11	
	015546	012746	MOV	#ERR11,-(SP)	
	015552	012746	MOV	#1,-(SP)	
	015556	010600	MOV	SP,R0	
	015560	104415	TRAP	C#PNTX	
	015562	062706	ADD	#4,SP	
2618	015566		PRINTX	#ERR12	
	015566	012746	MOV	#ERR12,-(SP)	
	015572	012746	MOV	#1,-(SP)	
	015576	010600	MOV	SP,R0	
	015600	104415	TRAP	C#PNTX	
	015602	062706	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	ADD	#2,R1	;R1 POINT TO SECOND WORD OF PACKET
2622	015616	005763	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	MOV	#ERR13,-(SP)	
	015634	012746	MOV	#3,-(SP)	
	015640	010600	MOV	SP,R0	
	015642	104415	TRAP	C#PNTX	
	015644	062706	ADD	#10,SP	
2625	015650	062701	ADD	#4,R1	;GET THE NEXT WORD
2626	015654	062705	ADD	#4,R5	;AND A .AIN
2627	015660	162763	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	MOV	#LINE,-(SP)	
	015674	012746	MOV	#1,-(SP)	
	015700	010600	MOV	SP,R0	
	015702	104417	TRAP	C#PNTF	
	015704	062706	ADD	#4,SP	
2630	015710	105737	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)	

015720	153716	002215		BISB	SECOND,(SP)
015724	005046			CLR	-(SP)
015726	153716	002214		BISB	MINUTE,(SP)
015732	005046			CLR	-(SP)
015734	153716	002213		BISB	HOURS,(SP)
015740	012746	020044		MOV	#TIME,-(SP)
015744	012746	000004		MOV	#4,-(SP)
015750	010600			MOV	SP,RO
015752	104417			TRAP	C#PNTF
015754	062706	000012		ADD	#12,SP
2633 015760			1#:	POP	<R5,R1>
2634 015764				EXIT	MSG
015764	000167			.WORD	J#JMP
015766	003044			.WORD	L10003-2-
2635					
2636 015770	045	101	103	ERR00::	.ASCIZ ?#ACOMMAND: #T#S3#AT/MSCP UNIT: #03#A(O)?
2637 016040	045	101	103	ERR01::	.ASCIZ ?#ACOMMAND: #T#A-REV#S3#AT/MSCP UNIT: #03#A(O)?
2638 016116	045	101	103	ERR02::	.ASCIZ ?#ACOMMAND: #T#A-LEOT#S3#AT/MSCP UNIT: #03#A(O)?
2639 016175	045	101	103	ERR03::	.ASCIZ ?#ACOMMAND: #T#A-IMM#S3#AT/MSCP UNIT: #03#A(O)?
2640 016253	045	101	103	ERR04::	.ASCIZ ?#ACOMMAND: #T#A-UNLD#S3#AT/MSCP UNIT: #03#A(O)?
2641 016332	045	101	103	ERR05::	.ASCIZ ?#ACOMMAND: #T#A-WRPR#S3#AT/MSCP UNIT: #03#A(O)?
2642 016411	045	116	045	ERR06::	.ASCIZ ?#N#APASS: #D3#A(D) DATA PAT: #02#A(O)?
2643 016461	045	116	045	ERR07::	.ASCIZ ?#N#ARECORD BYTE COUNT: #D6#A(D)?
2644 016521	045	116	045	ERR08::	.ASCIZ ?#N#AOBJECT CNT : #06#06#A(O)?
2645 016556	045	116	045	ERR09::	.ASCIZ ?#N#ADRIVE ERROR CODE : #T#T#A(H)?
2646 016617	045	116	045	ERR10::	.ASCIZ ?#N#ASA CONTENTS: #06#A(O)?
2647 016651	045	116	045	ERR11::	.ASCIZ ?#N#ARESPONSE PACKET?
2648 016675	045	116	045	ERR12::	.ASCIZ ?#N#S5#AHIGH WORD#S6#ALOW WORD?
2649 016733	045	116	045	ERR13::	.ASCIZ ?#N#S5#06#A(O)#S6#06#A(O)?
2650 016764	045	116	045	ERR14::	.ASCIZ ?#N#S2#ABYTE#S10#ADATA?
2651 017012	045	116	045	ERR15::	.ASCIZ ?#N#S2#AADDR#S3#AEXPECTED#S3#ARECEIVED#N?
2652 017062	045	123	061	ERR16::	.ASCIZ ?#S1#06#S5#03#S8#03#N?
2653 017107	045	101	124	ERR17::	.ASCIZ ?#ATOTAL BYTES IN ERROR : #D4#N?
2654 017146	045	116	045	ERL00::	.ASCIZ ?#N#ATAP OBJ CNT: #06#06#A(O)?
2655 017203	045	116	045	ERL01::	.ASCIZ ?#N#ATRK NUM: #D2#A(D) LEVEL: #D2#A(D) RETRIES: #D2#A(D)?
2656 017301	045	116	045	ERL02::	.ASCIZ ?#N#ALOG BLK NUM: #D3#A(D) PHYS BLK NUM: #D6#A(D)?
2657 017365	045	116	045	ERL03::	.ASCIZ ?#N#ADRV CODE: #03#A(O) DRV FLGS: #03#A(O)?
2658 017454	045	116	045	ERL04::	.ASCIZ ?#N#ADRV STATE: #06#A(O) INTERN STATUS: #03#A(O)?
2659 017540	045	116	045	ERL05::	.ASCIZ ?#N#ATAP CNT 0: #03#A(O) TAP CNT 1: #03#A(O)?
2660 017627	045	116	045	ERL06::	.ASCIZ ?#N#ATAP CNT 2: #03#A(O) RD/WR STATE: #06#A(O)?
2661 017713	045	116	045	ERL07::	.ASCIZ ?#N#AOPER FLGS: #06#A(O)?
2662 017743	045	116	045	ERL08::	.ASCIZ ?#N#AERROR LOG PACKET?
2663 017770	045	116	045	NCLK::	.ASCIZ ?#N#A**/**/* CLOCK NOT PRESENT **/**/*#N?
2664 020044	045	116	045	TIME::	.ASCIZ ?#N#A**/**/* TIME #Z2#A:#Z2#A:#Z2#A **/**/*#N?
2665 020125	045	116	045	DAY::	.ASCIZ ?#N#A**/**/* START OF DAY #02#A **/**/*#N?
2666 020202	045	101	102	COUNTS::	.ASCIZ ?#N#ABYTE COUNT: #D#A(D) FILE SIZE: #D#A(D)#N?
2667 020256	045	116	045	UNTEOT::	.ASCIZ ?#N#AUNIT: #01#A IS AT EOT#N?
2668 020312	045	116	045	UNTLOT::	.ASCIZ ?#N#AUNIT: #01#A IS AT LEOT#N?
2669 020347	045	116	045	DUMP::	.ASCIZ ?#N#AR1:#06#A R2:#06#A R3:#06#A R4:#06#A R5:#06#N?
2670 020430	045	116	045	DUMP1::	.ASCIZ ?#N#N#ABYTES X-FERRED: #D5#A(D)#N?
2671 020471	045	117	066	DUMP2::	.ASCIZ ?#06#S3#06#S3#06#S3#06#S3#06#N?
2672 020527	045	116	000	LINE::	.ASCIZ ?#N?
2673 020532	045	116	045	CONTRV::	.ASCII /#N#AUNIT: #D1/
2674 020547	045	116	045		.ASCII /#N#ACONTROLLER SOFTWARE VERSION : #D3/
2675 020614	045	116	045		.ASCIZ /#N#ACONTROLLER HARDWARE VERSION : #D3/
2676 020662	045	116	045	UNITRV::	.ASCII /#N#ADRIVE SOFTWARE VERSION : #D3/
2677 020727	045	116	045		.ASCIZ /#N#ADRIVE HARDWARE VERSION : #D3#N#N/

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  017770          MOV    @NCLK, (SP)
      021062    012746  000001          MOV    @1, (SP)
      021066    010600          MOV    SP,RO
      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
      021100    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,RO
      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 SCHED::
2738 021240 005001 CLR R1 ;SET R1 TO FIRST 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,CRD LIM ;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 ;Reg'ister 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 033742              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 027432                      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 033250                      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                      50$:   POP      <R5,R1>            ;RESTORE REGISTERS R1 AND R5
      021774 012605              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      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                      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 022212          PUSH    <R1,R5>          ;SAVE R1 AND R5
2940 022216 105765 000001  TSTB   DATPAT(R5)      ;TEST DATPAT FOR A TEST PATTERN
2941 022222 001445          BEQ    20$          ;BRANCH IF WE DON'T NEED ONE
2942 022224 105737 002233  TSTB   PATERN          ;PATTERN SPECIFIED IN SOFTWARE P-TABLE ?
2943 022230 001404          BEQ    1$          ;NO, KEEP GOING
2944 022232 113764 002233 000024  MOVB  PATERN,PATSAV(R4) ;PUT THE PATTERN IN THE SAVE LOCATION
2945 022240 000420          BR     10$
2946
2947 022242 105765 000001  1$:  TSTB   DATPAT(R5)      ;DO WE WANT ROTATING DATA PATTERNS ?
2948 022246 100404          BMI    5$          ;IF NEGATIVE GO TO 5$
2949 022250 116564 000001 000024  MOVB  DATPAT(R5),PATSAV(R4) ;LET PATSAV EQUAL DATPAT
2950 022256 000411          BR     10$          ;BRANCH
2951
2952 022260 005264 000024  5$:  INC    PATSAV(R4)      ;ADD 1 TO PATSAV
2953 022264 026427 000024 000010  CMP   PATSAV(R4),#ENDPAT ;ARE WE AT THE END OF THE PATTERN TABLE ?
2954 022272 001003          BNE   10$          ;NO, KEEP GOING
2955 022274 012764 000001 000024  MOV   #1.,PATSAV(R4)    ;AT THE END, LET PATSAV EQUAL 1
2956
2957 022302 013705 003416 10$:  MOV   BYTES,R5        ;PUT THE BYTE COUNT IN R5
2958 022306 032705 000001  BIT   #BIT0,R5        ;IS THE BYTE COUNT ODD ?
2959 022312 001401          BEQ   15$          ;BRANCH IF NOT
2960 022314 005205          INC   R5            ;MAKE BYTE COUNT EVEN FOR PATGEN
2961
2962 022316 012703 070560 15$:  MOV   #WRTBUF,R3      ;POINT R3 TO THE WRITE BUFFER
2963 022322 116401 000024  MOVB  PATSAV(R4),R1   ;SAVE PATSAV IN R1
2964 022326 005301          DEC   R1            ;ADJUST FOR TABLE STEP
2965 022330 006301          ASL   R1            ;MAKE IT MOD 2 OFFSET
2966
2967 022332 004771 022344  20$:  JSR   PC,@PATTBL(R1) ;GO FILL THE BUFFER
2968 022336          POP   <R5,R1>    ;RESTORE R5 AND R1
2969 022342 000207          RTS   PC          ;RETURN
2970
2971
2972 PATTBL::
2973 022344 022362          .WORD  PATGN1      ;ALL 1'S
2974 022346 022376          .WORD  PATGN2      ;ALL 0'S
2975 022350 022410          .WORD  PATGN3      ;WORST CASE MFM
2976 022352 022450          .WORD  PATGN4      ;ALTERNATE 1'S AND 0'S
2977 022354 022464          .WORD  PATGN5      ;RANDOM DATA
  
```



```

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
    
```

```
3035 022560 162702 000002      SUB      #2,R2      ;HAVE WE DONE A FULL BLOCK YET
3036 022564 001410      BEQ      5$        ;YES DO NEXT BLOCK IN PATTERN 5
3037 022566 012723 066666      MOV      #066666,(R3)+ ;PUT THE NUMBER INTO THE BUFFER
3038 022572 162705 000002      SUB      #2,R5      ;SUBTRACT TWO FROM R5
3039 022576 001420      BEQ      10$       ;KEEP GOING IF WE AREN'T AT 0
3040 022600 162702 000002      SUB      #2,R2      ;HAVE WE DONE A FULL BLOCK YET
3041 022604 001350      BNE      3$        ;YES DO NEXT BLOCK IN PATTERN 5
3042 022606 012702 001000      5$:     MOV      #512.,R2
3043 022612 004737 022716      6$:     JSR      PC,RANGEN
3044 022616 013723 003426      MOV      RANWRD,(R3)+ ;GO GENERATE RANDOM PATTERN
3045 022622 162705 000002      SUB      #2,R5      ;PUT THE NUMBER INTO THE BUFFER
3046 022626 001404      BEQ      10$       ;SUBTRACT TWO FROM R5
3047 022630 162702 000002      SUB      #2,R2      ;KEEP GOING IF WE AREN'T AT 0
3048 022634 001366      BNE      6$        ;HAVE WE DONE A FULL BLOCK YET
3049 022636 000731      BR       1$        ;YES DO NEXT BLOCK IN PATTERN 5
3050 022640      10$:    POP      <R2>
3051 022642 000207      RTS      PC
3052
3053      ;RETURN
```

```

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 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 SEED INTO RAN3
3096 022742 013705 003430          MOV      RAN1,R5                ;MOV THE LAST 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      1$:      MOV      #DUMPKT,R5    ;POINT R5 TO THE DUMMY PACKET
3135 023066 116563 000000 000000  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     ;MO. 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 000020 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 070556  MOV  @BOFFL(R3),WRTBUF-2 ;PUT LOW ORDER OBJECT COUNT IN WRTBUF
3167      023174 012737 000004 003424  MOV  @N,SUBCNT         ;REINIT SUBCNT
3168      023202 012737 050560 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 070556 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 AN ACCESS DATA COMMAND
3181      023266 012737 070560 003422  7$:  MOV  @WRTBUF,BUFADR  ;SET WRTBUF ADDRESS IN BUFADR
3182      023274 000207              10$: RTS  PC          ;RETURN

```

```

3184 .SBTTL OBJCT COUNT HANDLER
3185 :*****
3186 :
3187 : OBJCT 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 OBJCTHD::
3197 023276          PUSH    <R1>                ;SAVE R1
3198 023300 116301 000001  MOVB   CMD(R3),R1          ;PUT THE COMMAND PRIMITIVE INTO R1
3199 023304 042701 000000  BIC    07,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    0REW,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$:  CMP    0SPC,R1          ;IS IT A NON-DATA TRANSFER COMMAND ?
3209 023340 101007          BHI    2$                    ;BRANCH IF IT IS
3210 023342 022701 000100  CMP    0WTM,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$:  MOV    01,R8            ;PUT A 1 IN THE TEMP REGISTER
3216 023366 032763 000002 000000  3$:  BIT    0EOTBIT,CMD(R3)  ;IS IT AN LEOT COMMAND ?
3217 023374 001021          BNE    6$                    ;GET OUT IF IT IS
3218 023376 032763 000001 000000  BIT    0REVBIT,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          6$:  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,PRTCCLR      ;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 COMMAND OLD POINTER SAVE
3299 023702          POP   <R5,R3>    ;RESTORE R3, AND R5
3300 023706 000207          RTS   PC            ;RETURN
```

```

3302 .SBTTL CLASS DRIVER RECEIVE
3303 ;*****
3304 ;
3305 ;Class Driver Receive
3306 ;
3307 ;Called By      : CLSDVR
3308 ;Calls To      : PDRECV, PRTCLR
3309 ;Inputs       : RESP - The number of RESPONSEs found.
3310 ;              : GCSREF - Get Command Status reference number.
3311 ;              : RNUSAV - New response buffer save
3312 ;              : CMSTSV - Command progress count.
3313 ;              : ELBSAV - Error log buffer pointer.
3314 ;Register Inputs: R2 - Old pointer to program command ring.
3315 ;              : R3 - New pointer to program command ring.
3316 ;              : R4 - Lun block pointer.
3317 ;              : R5 - Old pointer to driver command ring.
3318 ;Registers Used: R1 - Old pointer to driver RESPONSE ring.
3319 ;
3320
3321 023710 CDRECV::
3322 023710      PUSH      <R1,R2>          ;SAVE R1,R2
3323 023714      MOV       RNUSAV(R4),R1    ;LET R1 = NEW RESPONSE BUFFER SAVE
3324 023720      JSR      PC,PDRECV        ;CALL PORT DRIVER RECEIVE
3325 023724      TST      RESP            ;DID WE GET A RESPONSE ?
3326 023730      BEQ      35$              ;NO, GET OUT OF HERE
3327 023732      MOV      RESP,HNDLRP     ;SAVE A COPY FOR RSPHDL
3328
3329 023740      022761 000022 000012 5$:  CMP      #ST.SEX,P.STS(R1)      ;IS IT A SERIOUS EXCEPTION ?
3330 023746      001425                                BEQ      10$                  ;YES, CONTINUE
3331 023750      005761 000000                                TST      P.CRF(R1)           ;IS IT AN UNSOLICITED ERROR LOG ?
3332 023754      001422                                BEQ      10$                  ;YES, GO HANDLE ERROR LOG
3333 023756      026165 000000 000000      CMP      P.CRF(R1),P.CRF(R5)   ;IS THIS THE COMMAND THAT IS EXPECTED ?
3334 023764      001416                                BEQ      10$                  ;YES, CONTINUE
3335 023766      023761 010552 000000      CMP      GCSREF,P.CRF(R1)     ;IS IT THE GCS END PACKET
3336 023774      001003                                BNE      7$                   ;NO, GO DO RESPONSE OUT OF SEQUENCE
3337 023776      004737 026534                                JSR      PC,GCSHDL           ;GO TO THE GCS HANDLING ROUTINE
3338 024002      000461                                BR       35$                  ;GET OUT
3339
3340 024004      112737 000005 010546 7$:  MOVB    #MISSEQ,IOSTAT       ;SET MISSING SEQUENCE IN I/O STATUS
3341 024012      004737 033742                                JSR      PC,CORDMP
3342 024016      000240                                NOP
3343 024020      000452                                BR       35$                  ;EXIT
3344
3345 024022      032761 000200 000010 10$: BIT      #OP.END,P.OPCD(R1)    ;YES, IS IT AN END PACKET ?
3346 024030      001427                                BEQ      20$                  ;NO, GO HANDLE ERROR LOG
3347 024032      052737 100000 010546      BIS      #NURESP,IOSTAT      ;SET A NEW RESPONSE IN THE I/O STATUS
3348 024040      105237 010571                                INCB    CRDLIM               ;ADD 1 TO THE CREDIT LIMIT
3349 024044      062705 000050                                ADD     #DCBSTP,R5           ;ADJUST THE OLD COMMAND POINTER
3350 024050      022705 004032                                CMP     #DCBEND,R5          ;IS IT AT THE END OF THE RING ?
3351 024054      001002                                BNE     15$                  ;NO, BRANCH
3352 024056      012705 003572                                MOV     #DCMDBF,R5          ;YES, SET IT TO THE TOP OF THE RING
3353
3354 024062      016162 000012 000010 15$: MOV     P.STS(R1),XFERST(R2)   ;PUT REPSONCE STATUS IN THE HOST PACKET
3355 024070      062702 000014                                ADD     #PCBSTP,R2          ;ADJUST R2 TO POINT AT THE NEXT SLOT
3356 024074      022702 003344                                CMP     #PCBEND,R2          ;IS IT AT THE END OF THE RING ?
3357 024100      001006                                BNE     25$                  ;NO, BRANCH
3358 024102      012702 003264                                MOV     #PCMDBF,R2          ;YES, SET IT BACK TO TOP OF THE RING

```

```
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 005037 010564      CLR     PCKSIZ          ;CLEAR PACKET SIZE
3395 024174 016401 000014      MOV     CNUSAV(R4),R1   ;LET R1 EQUAL THE NEW COMMAND POINTER
3396 024200 016461 000006 000000      MOV     CMDSEQ(R4),P.CRF(R1) ;PUT COMMAND SEQUENCE NUMBER INTO PACKET
3397 024206 005061 000002          CLR     P.CRF+2(R1)    ;CLEAR THE UPPER WORD
3398 024212 016461 000004 000004      MOV     TKUNIT(R4),P.UNIT(R1) ;PUT THE UNIT NUMBER INTO THE PACKET
3399 024220 005061 000006          CLR     P.UNIT+2(R1)   ;CLEAR THE UPPER WORD
3400 024224 005061 000012          CLR     P.MOD(R1)      ;CLEAR MODIFIERS FIELD
3401 024230 032737 000020 003516      BIT     #GCSNFL,PCFLAG ;ARE WE IN GCS COMMAND MODE ?
3402 024236 001402          BEQ     5$            ;NO, CONTINUE
3403 024240 000137 025400          JMP     GCMDST         ;YES, GO DO A GET COMMAND STATUS
3404
3405 024244 016361 000002 000014 5$:      MOV     ITMOFF(R3),P.BCNT(R1) ;PUT THE BYTE COUNT INTO THE PACKET
3406 024252 005061 000016          CLR     P.BCNT+2(R1)   ;CLEAR THE UPPER WORD
3407 024256 016337 000000 003530      MOV     CMD(R3),R8     ;PUT THE PRIMITIVE IN R8
3408 024264 042737 177770 003530      BIC     #177770,R8    ;GET JUST THE MODIFIERS
3409 024272 022737 000001 003530      CMP     #REVBIT,R8    ;IS THE COMMAND A REVERSE ?
3410 024300 001003          BNE     10$           ;NO, BRANCH
3411 024302 052761 000010 000012      BIS     #MD.REV,P.MOD(R1) ;YES, SET REVERSE IN THE MODIFIER FIELD
3412
3413 024310 032764 000002 000026 10$:      BIT     #SEREXC,LUNFLG(R4) ;IS IT A SERIOUS EXCEPTION CONDITION ?
3414 024316 001406          BEQ     15$           ;NO, BRANCH
3415 024320 052761 020000 000012      BIS     #MD.CSE,P.MOD(R1) ;YES, SET CLEAR SERIOUS EXCEPTION
3416 024326 042764 000002 000026      BIC     #SEREXC,LUNFLG(R4) ;CLEAR SERIOUS EXCEPTION FLAG
3417
3418 024334 116302 000000          15$:      MOVVB  CMD(R3),R2     ;PUT THE COMMAND PRIMITIVE INTO R1
3419 024340 006202          ASR     R2            ;
3420 024342 006202          ASR     R2            ;
3421 024344 042702 177701          BIC     #C76,R2       ;ADJUST FOR THE CASE STATEMENT
3422 024350 022702 000046          CMP     #46,R2        ;ARE WE IN THE RANGE ?
3423 024354 103002          BHIS   20$           ;YES, KEEP GOING
3424 024356 000137 025604          JMP     ILCMD         ;NO, HANDLE AN ILLEGAL COMMAND
3425 024362 000172 024366          20$:      JMP     @CMDTBL(R2)   ;SELECT
3426
3427 024366 024436      CMDTBL: .WORD  NULL
3428 024370 024442      .WORD  READ
3429 024372 024462      .WORD  WRITE
3430 024374 024502      .WORD  CHODAT
3431 024376 024522      .WORD  ACCESS
3432 024400 024542      .WORD  SPCREC
    
```

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	GCMDST	
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
  
```


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


```

3620 .SBTTL PORT DRIVER TRANSMIT
3621 ;*****
3622 ;
3623 ;Port Driver Transmit
3624 ;
3625 ;Called By      : CLSDVR
3626 ;Inputs        : CMDSSV - Command descriptor ring pointer.
3627 ;              : DCDSAV - Driver command ring pointer.
3628 ;              : CRDLIM - Number of open slots in the driver command ring.
3629 ;              : PCKSIZ - Length in bytes of the command packet.
3630 ;Register Inputs: R4 - Lun block pointer.
3631 ;Registers Used : R2 - Pointer to driver command ring.
3632 ;              : R1 - Pointer to driver command descriptor ring.
3633 ;
3634 ;
3635 025730 PRTDRV::
3636 025730      PUSH    <R3,R2,R1>          ;SAVE R3, R2 AND R1
3637 025736 016402 000014      MOV     CNUSAV(R4),R2      ;POINT R2 AT NEW COMMAND BUFFER SLOT
3638 025742 016401 000012      MOV     CMDSSV(R4),R1    ;LET R1 POINT TO THE COMMAND DESCRIPTOR
3639 025746 022702 003572      CMP     #DCMDBF,R2      ;IS R2 AT TOP OF DRIVER COMMAND Ring
3640 025752 001403      BEQ    1$              ;YES, BRANCH
3641 025754 162702 000050      SUB     #DCBSTP,R2      ;NO, SUBTRACT DCBSTP FROM R2
3642 025760 000402      BR     5$              ;
3643 ;
3644 025762 062702 000170      1$:   ADD     #DCB3SP,R2    ;YES, ADD DCB3SP TO R2
3645 025766 113762 010571 177776 5$:   MOVB   CRDLIM,CRD(R2)    ;PUT THE CREDIT LIMIT INTO THE PACKET
3646 025774 112762 000001 177777      MOVB   #1,CONID(R2)     ;PUT THE CONNECTION TYPE INTO THE PACKET
3647 026002 013762 010564 177774      MOV     PCKSIZ,MSGLEN(R2) ;PUT THE PACKET SIZE INTO THE PACKET
3648 026010 010211      MOV     R2,(R1)         ;PUT THE PACKET ADDRESS INTO THE DESCRIPTOR
3649 026012 052761 100000 000002      BIS    #OWN,HIADDR(R1)  ;SET THE OWNERSHIP BIT OF THE DESCRIPTOR
3650 026020 042761 040000 000002      BIC    #FLAG,HIADDR(R1) ;CLEAR TO DESCRIPTOR FLAG BIT
3651 026026 005774 000000      TST    #TKIP(R4)        ;READ THE IP REGISTER
3652 026032 017437 000002 010566      MOV     #TKSA(R4),SAERR ;SAVE THE SA FOR THE ERROR PRINTOUT
3653 026040 005737 010566      TST    SAERR            ;READ THE SA REGISTER
3654 026044 100003      BPL    10$             ;BRANCH IF NO ERRORS
3655 026046 052737 000003 010546      BIS    #IOPDRE,IOSTAT   ;SET PORT DETECTED ERROR IN I/O STATUS
3656 ;
3657 026054 062701 000004      10$:  ADD     #DSPSTP,R1      ;ADVANCE THE DESCRIPTOR POINTER
3658 026060 026401 000170      CMP     UCDEND(R4),R1   ;ARE WE AT END OF THE DESCRIPTOR RING
3659 026064 001002      BNE    15$             ;NO, BRANCH
3660 026066 016401 000166      MOV     UCDSRG(R4),R1   ;YES, SET POINTER TO START OF THE RING
3661 ;
3662 026072 010164 000012      15$:  MOV     R1,CMDSSV(R4)    ;SAVE THE POINTER
3663 026076      POP    <R1,R2,R3>      ;RESTORE R1, R2 AND R3
3664 026104 000207      RTS    PC              ;RETURN
    
```

```

3666 .SBTTL PORT DRIVER RECEIVE
3667 ;*****
3668 ;
3669 ;Port Driver Receive
3670 ;
3671 ;Called By : CDRECV
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 026106 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 #BIT0,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 RCR SLTUSE(R4) ;MOVE BIT0 TO CARY BIT
3726 026266 103003 BCC 5$ ;BRANCH IF SLOT NOT USED
3727 026270 012761 100000 000002 MOV #0WN,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 026526 MOV URDSRG(R4),INTTBL*2 ;PUT RESP DESCRIPTOR ADDRESS IN TABLE
3750 026330 012703 026524 MOV @INTTBL,R3 ;PUT THE TABLE ADDRESS INTO R3
3751 026334 012701 104000 MOV @S1!ERR,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 005202 ILOOP: INC R2 ;INCREMENT HI TIME OUT VALUE ?
3757 026352 001003 BNE 2$ ;IF NOT, BRANCH
3758 026354 005337 010554 DEC CNTHI ;ELSE, INCREMENT HI TIMEOUT
3759 026360 001447 BEQ TKERR ;GET OUT, WE'VE TIMED OUT
3760 ;
3761 026362 037401 000002 2$: BIT @TKSA(R4),R1 ;TEST FOR STEP BIT FROM DRIVE
3762 026366 001770 BEQ ILOOP ;LOOP UNTIL SOMETHING SETS
3763 026370 017437 000002 010566 MOV @TKSA(R4),SAERR ;SAVE THE SA FOR THE ERROR PRINTOUT
3764 026376 005737 010566 TST SAERR ;CHECK FOR ERROR
3765 026402 100436 BMI TKERR ;GET OUT ON ERROR
3766 026404 012374 000002 3$: MOV (R3)-,@TKSA(R4) ;WRITE WORD FROM TABLE TO CONTROLLER
3767 026410 006301 ASL R1 ;SHIFT TO NEXT STEP
3768 026412 100403 BMI 4$ ;GET OUT AFTER THE FOURTH STEP
3769 026414 052701 100000 BIS @ERR,R1 ;ALSO CHECK FOR ERROR BIT
3770 026420 000747 BR LOOP ;IF NOT AT LAST STEP LOOP
3771 ;
3772 026422 016402 000162 4$: MOV URDSRG(R4),R2 ;PUT THE RESPONSE DESCRIPTOR ADD IN R2
3773 026426 016403 000156 MOV URSPBF(R4),R3 ;PUT THE RESPONSE BUFFER ADDRESS IN R3
3774 026432 010322 5$: MOV R3,(R2) ;PUT THE BUFF ADD IN THE DESCRIPTOR
3775 026434 005022 CLR (R2) ;CLEAR THE NEXT WORD
3776 026436 062703 000104 ADD @DRBSTP,R3 ;STEP TO THE NEXT BUFFER SLOT
3777 026442 026403 000160 CMP URBEND(R4),R3 ;ARE WE AT THE END OF THE BUFFER ?
3778 026446 001371 BNE 5$ ;NO, KEEP GOING
3779 ;
3780 026450 016402 000166 MOV UCDSRG(R4),R2 ;PUT THE CMD DESCRIPTOR ADDRESS IN R2
3781 026454 012703 003572 MOV @DCMDBF,R3 ;PUT THE CMD BUFFER ADDRESS IN R3
3782 026460 010322 10$: MOV R3,(R2) ;PUT THE BUFF ADD IN THE DESCRIPTOR
3783 026462 005022 CLR (R2) ;CLEAR THE NEXT WORD
3784 026464 062703 000050 ADD @DCBSTP,R3 ;STEP TO THE NEXT BUFFER SLOT
3785 026470 022703 004032 CMP @DCBEND,R3 ;ARE WE AT THE END OF THE BUFFER ?
3786 026474 001371 BNE 10$ ;NO, KEEP GOING
3787 026476 000403 BR IDONE ;ALL DONE
3788 ;
3789 026500 012737 000006 010546 TKERR: MOV @INTERR,IOSTAT ;SET UP FOR A FATAL ERROR
3790 ;
    
```

```
3791 026506 005337 003402      IDONE: DEC   RSPCNT
3792 026512                POP   <R5,R3,R2,R1>      ;RESTORE THE REGISTERS
3793 026522 000207                RTS   PC           ;RETURN
3794
3795                ;INIT DATA TABLE
3796
3797 026524 111400      INTTBL:  .WORD  TKINIT
3798 026526 000000                .WORD  0
3799 026530 000000                .WORD  0
3800 026532 000001                .WORD  GO
```

```

3802      .SBTTL  GCS RESPONSE HANDLER
3803      ;*****
3804      ;
3805      ;GCS RESPONSE HANDLER
3806      ;
3807      ;Called By      :
3808      ;Calls To      :
3809      ;Register Inputs :
3810      ;
3811      ;Register Inputs :
3812      ;
3813
3814 026534      GCSHDL::
3815 026534      023761 010550 000020      CMP      CMSTSV,P.CMST(R1)      ;ANY PROGRESS ?
3816 026542      101017                      BHI      5$                      ;YES, CLKEAN UP THE MESS
3817 026544      042737 000040 003516      BIC      #GCSRFL,PCFLAG      ;CLEAR THE GCS MODE FLAG
3818 026552      005037 003414                      CLR      RESP                  ;TAKE OFF THE RESPONSE
3819 026556      005037 003412                      CLR      HNDLRP              ;TAKE OFF THE RESPONSE
3820 026562      112737 000002 010546      MOVVB   #IOHUNG,IOSTAT      ;SET HUNG CONTROLLER BIT
3821 026570      004737 033742                      JSR      PC,CORDMP
3822 026574      000240                      NOP
3823 026576      000137 027320                      JMP      GCSEXT              ;GET OUT
3824
3825 026602      5$:      PUSH     <R1,R2>          ;
3826 026606      016401 000016                      MOV     COLSAV(R4),R1      ;PUT THE OLD POINTER IN R1
3827 026612      162701 000004                      SUB     #4,R1              ;ADJUST TO INCLUDE DESCRIPTOR WORDS
3828 026616      016402 000014                      MOV     CNUSAV(R4),R2      ;PUT THE NEW POINTER IN R2
3829 026622      162702 000004                      SUB     #4,R2              ;ADJUST TO INCLUDE DESCRIPTOR WORDS
3830 026626      022701 003566                      CMP     #CMDBF1,R1        ;OLD POINTER AT BF1 ?
3831 026632      001407                      BEQ     OLD1              ;YES, GO HANDLE IT
3832 026634      022701 003636                      CMP     #CMDBF2,R1        ;OLD POINTER AT BF2 ?
3833 026640      001434                      BEQ     OLD2              ;YES, GO HANDLE IT
3834 026642      022701 003706                      CMP     #CMDBF3,R1        ;OLD POINTER AT BF3 ?
3835 026646      001461                      BEQ     OLD3              ;YES, GO HANDLE IT
3836 026650      000510                      BR      OLD4              ;NO, GO HANDLE BF4
3837
3838 026652      022702 003706      OLD1:    CMP     #CMDBF3,R2        ;NEW POINTER AT BF3 ?
3839 026656      001004                      BNE     5$                ;NO, TRY AGAIN
3840 026660      004737 027322                      JSR     PC,EXC1A2          ;GO MOVE COMMAND 1 TO 2
3841 026664      000137 027146                      JMP     ADJUST            ;GO ADJUST THE OLD POINTER
3842 026670      022702 003756      5$:     CMP     #CMDBF4,R2        ;NEW POINTER AT BF4 ?
3843 026674      001006                      BNE     10$              ;NO, TRY AGAIN
3844 026676      004737 027344                      JSR     PC,EXC2A3          ;GO MOVE COMMAND 2 TO 3
3845 026702      004737 027322                      JSR     PC,EXC1A2          ;GO MOVE COMMAND 1 TO 2
3846 026706      000137 027146                      JMP     ADJUST            ;GO ADJUST THE OLD POINTER
3847 026712      004737 027366      10$:   JSR     PC,EXC3A4          ;GO MOVE COMMAND 3 TO 4
3848 026716      004737 027344                      JSR     PC,EXC2A3          ;GO MOVE COMMAND 2 TO 3
3849 026722      004737 027322                      JSR     PC,EXC1A2          ;GO MOVE COMMAND 1 TO 2
3850 026726      000137 027146                      JMP     ADJUST            ;GO ADJUST THE OLD POINTER
3851
3852 026732      022702 003756      OLD2:    CMP     #CMDBF4,R2        ;NEW POINTER AT BF4 ?
3853 026736      001004                      BNE     5$                ;NO, TRY AGAIN
3854 026740      004737 027344                      JSR     PC,EXC2A3          ;GO MOVE COMMAND 2 TO 3
3855 026744      000137 027146                      JMP     ADJUST            ;GO ADJUST THE OLD POINTER
3856 026750      022702 003566      5$:     CMP     #CMDBF1,R2        ;NEW POINTER AT BF1 ?
3857 026754      001006                      BNE     10$              ;NO, TRY AGAIN
3858 026756      004737 027366                      JSR     PC,EXC3A4          ;GO MOVE COMMAND 3 TO 4

```

```

3859 026762 004737 027344      JSR    PC,EXC2A3      ;GO MOVE COMMAND 2 TO 3
3860 026766 000137 027146      JMP    ADJUST        ;GO ADJUST THE OLD POINTER
3861 026772 004737 027410      10$:  JSR    PC,EXC4A1      ;GO MOVE COMMAND 4 TO 1
3862 026776 004737 027366      JSR    PC,EXC3A4      ;GO MOVE COMMAND 3 TO 4
3863 027002 004737 027344      JSR    PC,EXC2A3      ;GO MOVE COMMAND 2 TO 3
3864 027006 000137 027146      JMP    ADJUST        ;GO ADJUST THE OLD POINTER
3865
3866 027012 022702 003566      OLD3:  CMP    #CMDBF1,R2  ;NEW POINTER AT BF1 ?
3867 027016 001004              BNE    5$            ;NO, TRY AGAIN
3868 027020 004737 027366      JSR    PC,EXC3A4      ;GO MOVE COMMAND 3 TO 4
3869 027024 000137 027146      JMP    ADJUST        ;GO ADJUST THE OLD POINTER
3870 027030 022702 003636      5$:    CMP    #CMDBF2,R2  ;NEW POINTER AT BF2 ?
3871 027034 001006              BNE    10$           ;NO, TRY AGAIN
3872 027036 004737 027410      JSR    PC,EXC4A1      ;GO MOVE COMMAND 4 TO 1
3873 027042 004737 027366      JSR    PC,EXC3A4      ;GO MOVE COMMAND 3 TO 4
3874 027046 000137 027146      JMP    ADJUST        ;GO ADJUST THE OLD POINTER
3875 027052 004737 027322      10$:  JSR    PC,EXC1A2      ;GO MOVE COMMAND 1 TO 2
3876 027056 004737 027410      JSR    PC,EXC4A1      ;GO MOVE COMMAND 4 TO 1
3877 027062 004737 027366      JSR    PC,EXC3A4      ;GO MOVE COMMAND 3 TO 4
3878 027066 000137 027146      JMP    ADJUST        ;GO ADJUST THE OLD POINTER
3879
3880 027072 022702 003636      OLD4:  CMP    #CMDBF2,R2  ;NEW POINTER AT BF2 ?
3881 027076 001004              BNE    5$            ;NO, TRY ACAIN
3882 027100 004737 027410      JSR    PC,EXC4A1      ;GO MOVE COMMAND 4 TO 1
3883 027104 000137 027146      JMP    ADJUST        ;GO ADJUST THE OLD POINTER
3884 027110 022702 003706      5$:    CMP    #CMDBF3,R2  ;NEW POINTER AT BF3 ?
3885 027114 001006              BNE    10$           ;NO, TRY AGAIN
3886 027116 004737 027322      JSR    PC,EXC1A2      ;GO MOVE COMMAND 1 TO 2
3887 027122 004737 027410      JSR    PC,EXC4A1      ;GO MOVE COMMAND 4 TO 1
3888 027126 000137 027146      JMP    ADJUST        ;GO ADJUST THE OLD POINTER
3889 027132 004737 027344      10$:  JSR    PC,EXC2A3      ;GO MOVE COMMAND 2 TO 3
3890 027136 004737 027322      JSR    PC,EXC1A2      ;GO MOVE COMMAND 1 TO 2
3891 027142 004737 027410      JSR    PC,EXC4A1      ;GO MOVE COMMAND 4 TO 1
3892
3893 027146              ADJUST: POP    <R2,R1>      ;
3894 027152 042737 000040 003516  BIC    #GCSRFL,PCFLAG  ;CLEAR THE GCS MODE FLAG
3895 027160 016137 000020 010550  MOV    P,CMST(R1),CMSTSV ;PUT THE CMD STATUS INTO THE SAVE LOC
3896 027166 005037 003414              CLR    RESP          ;TAKE OFF THE RESPONSE
3897 027172 005037 003412              CLR    HNDLRP        ;TAKE OFF THE RESPONSE
3898 027176 005364 000006              DEC    CMDSEQ(R4)     ;ADJUST THE CMDSEQ NUMBER BACK 1
3899 027202 042737 100000 010546  BIC    #NURESP,IOSTAT  ;CLEAR THE NEW RESPONSE FLAG IN IOSTAT
3900 027210 105237 010571              INCB   CRDLIM        ;ADD 1 TO THE CREDIT LIMIT
3901
3902 027214 062764 000050 000016  ADD    #DCBSTP,COLSAV(R4) ;ADJUST THE OLD COMMAND POINTER
3903 027222 022764 004032 000016  CMP    #DCBEND,COLSAV(R4) ;IS IT AT THE END OF THE RING ?
3904 027230 001003              BNE    5$            ;NO, BRANCH
3905 027232 012764 003572 000016  MOV    #DCMDBF,COLSAV(R4) ;YES, SET IT TO THE TOP OF THE RING
3906
3907 027240 062705 000050      5$:    ADD    #DCBSTP,R5      ;ADJUST THE OLD COMMAND POINTER
3908 027244 022705 004032      CMP    #DCBEND,R5      ;IS IT AT THE END OF THE RING ?
3909 027250 001002              BNE    10$           ;NO, BRANCH
3910 027252 012705 003572      MOV    #DCMDBF,R5      ;YES, SET IT TO THE TOP OF THE RING
3911
3912 027256 062764 000104 000022  10$:  ADD    #DRBSTP,ROLSAV(R4) ;ADJUST THE OLD RESPONSE POINTER
3913 027264 026464 000160 000022  CMP    URBEND(R4),ROLSAV(R4) ;IS IT AT THE END OF THE BUFFER ?
3914 027272 001003              BNE    15$           ;NO, KEEP GOING
3915 027274 016464 000156 000022  MOV    URSPBF(R4),ROLSAV(R4) ;YES, SET IT TO BEGINNING OF THE BUFFER
    
```

```

3916
3917 027302 062701 000104      15$:  ADD    #DRBSTP,R1      ;ADJUST R1
3918 027306 026401 000160      CMP    URBEND(R4),R1    ;IS IT AT THE END OF THE BUFFER ?
3919 027312 001002                BNE    GCSEXT           ;NO, GET OUT
3920 027314 016401 000156      MOV    URSPBF(R4),R1    ;YES, SET IT TO BEGINNING OF THE BUFFER
3921
3922 027320                GCSEXT:
3923 027320 000207      RTS    PC                ;RETURN
3924
3925
3926 027322 012701 003566      EXC1A2: MOV   #CMDBF1,R1  ;SET R1 TO BF1
3927 027326 012702 003636      MOV   #CMDBF2,R2        ;SET R2 TO BF2
3928 027332 012122                5$:  MOV   (R1)+,(R2)+    ;MOV BF1 CONTENTS TO BF2
3929 027334 022701 003636      CMP   #CMDBF2,R1        ;HAVE WE MOVED THEM ALL
3930 027340 001374                BNE   5$                ;NO, KEEP MOVING IT
3931 027342 000207      RTS    PC                ;YES, GET OUT
3932
3933 027344 012701 003636      EXC2A3: MOV   #CMDBF2,R1  ;SET R1 TO BF2
3934 027350 012702 003706      MOV   #CMDBF3,R2        ;SET R2 TO BF3
3935 027354 012122                5$:  MOV   (R1)+,(R2)+    ;MOV BF2 CONTENTS TO BF3
3936 027356 022701 003706      CMP   #CMDBF3,R1        ;HAVE WE MOVED THEM ALL
3937 027362 001374                BNE   5$                ;NO, KEEP MOVING IT
3938 027364 000207      RTS    PC                ;YES, GET OUT
3939
3940 027366 012701 003706      EXC3A4: MOV   #CMDBF3,R1  ;SET R1 TO BF3
3941 027372 012702 003756      MOV   #CMDBF4,R2        ;SET R2 TO BF4
3942 027376 012122                5$:  MOV   (R1)+,(R2)+    ;MOV BF3 CONTENTS TO BF4
3943 027400 022701 003756      CMP   #CMDBF4,R1        ;HAVE WE MOVED THEM ALL
3944 027404 001374                BNE   5$                ;NO, KEEP MOVING IT
3945 027406 000207      RTS    PC                ;YES, GET OUT
3946
3947 027410 012701 003756      EXC4A1: MOV   #CMDBF4,R1  ;SET R1 TO BF4
3948 027414 012702 003566      MOV   #CMDBF1,R2        ;SET R2 TO BF1
3949 027420 012122                5$:  MOV   (R1)+,(R2)+    ;MOV BF4 CONTENTS TO BF1
3950 027422 022702 003636      CMP   #CMDBF2,R2        ;HAVE WE MOVED THEM ALL
3951 027426 001374                BNE   5$                ;NO, KEEP MOVING IT
3952 027430 000207      RTS    PC                ;YES, GET OUT

```



```

3954 .SBTTL RESPONSE HANDLER
3955 ;*****
3956 ;
3957 ;RESPONSE HANDLER
3958 ;
3959 ;Called By      : CMDSEQ
3960 ;Calls To      : ERRDEI, ERRDEL, ERRDEC, CMPDAT, DQCMD
3961 ;Register Inputs : R1 - UNIT NUMBER
3962 ;              : R4 - LUN BLOCK POINTER
3963 ;Register Inputs : R3  POINTER TO CURRENT RESPONSE PACKET
3964 ;
3965
3966 RSPHDL::
3967     PUSH    <R3>
3968     CLR     RESPON      ;0 HERE TELLS CMDSEQ ALL'S OKAY
3969     TSTB   IOSTAT      ;DID WE HAVE I/O TYPE FAILURE?
3970     BEQ    5$          ;BRANCH AROUND IF NOT
3971     JSR    PC,ERRDEI   ;ELSE DECODE AND PRINT IT
3972     BR     60$         ;GET OUT NOW
3973
3974     5$:     MOV     ROLSAV(R4),R3      ;GET OLD RESPONSE BUFFER POINTER
3975           TST     HNDLRP             ;DID WE HAVE ANY RESPONSES ?
3976           BEQ    60$                ;NO, GET OUT OF HERE
3977     10$:    BIT     #OP.END,P.OPCD(R3) ;IS IT AN END PACKET?
3978           BNE   15$                ;YES, BRANCH
3979           JSR    PC,ERRDEL          ;GO HANDLE ERROR LOG PACKET
3980           BR     50$                ;SEE IF THERE'S MORE RESPONSES
3981
3982     15$:    TST     P.STS(R3)         ;WAS STATUS "NORMAL"?
3983           BEQ    25$                ;YES - BRANCH
3984           CMP     #ST.SEX,P.STS(R3) ;IS IT SERIOUS EXCEPTION STATUS?
3985           BNE   20$                ;NO - GO HANDLE THE ERROR
3986           BIS     #NOTALY,LUNFLG(R4) ;SET THE NO-TALLY FLAG
3987           BR     45$                ;YES, GO DE-QUE THE COMMANDS
3988     20$:    JSR    PC,ERRDEC          ;GO HANDLE ERROR STATUS
3989           BR     30$
3990     25$:    CLC
3991           ROL     LEOTFL(R4)         ;CLEAR THE CARRY BIT
3992           ;ROTATE THE CARRY INTO THE LEOT FLAG
3993
3993     30$:    BIT     #AVB,DRINUS(R1)   ;HAVE WE DROPPED THE UNIT ?
3994           BEQ    60$                ;YES - GET OUT
3995           CMP     #OP.END!OP.RD,P.OPCD(R3) ;DID WE READ THIS TIME ?
3996           BNE   45$                ;NO - SKIP DATA COMPARE
3997           CMP     #5,L$TEST         ;ARE WE IN TEST 5 ?
3998           BNE   35$                ;NO - DO DATA COMPARE
3999           TSTB   T5CMP              ;DO DATA COMPARES IN TEST 5 ?
4000           BEQ    40$                ;NO, SKIP DATA COMPARE
4001     35$:    JSR    PC,CMPDAT         ;DO COMPARE DATA
4002           CMP     #5,L$TEST         ;ARE WE IN TEST 5 ?
4003           BNE   45$                ;NO - DON'T DUMP BUFFER
4004     40$:    BITB   #BITO,DMPBUF      ;SHOULD WE DUMP RDBUF ?
4005           BEQ    45$                ;NO - BRANCH
4006           JSR    PC,BUFDMP
4007
4008     45$:    JSR    PC,DQCMD          ;DEQUEUE THE COMMAND
4009     50$:    ADD     #DRBSTP,R3        ;ADJUST POINTER TO NEXT PACKET
4010           CMP     URBEND(R4),R3     ;END OF RESPONSE BUFFER?
    
```

4011	027650	001002		BNE	55\$;NO - BRANCH AROUND
4012	027652	016403	000156	MOV	URSPBF(R4),R3		;PUT POINTER AT BEGINNING OF BUFFER
4013							
4014	027656	005337	003412	55\$:	DEC	HNDLRP	;DECREMENT RESPONSE COUNTER
4015	027662	001301			BNE	10\$;GO HANDLE ANOTHER ONE
4016							
4017	027664	010364	000022	60\$:	MOV	R3,ROLSAV(R4)	;SAVE OLD RESPONSE BUFFER POINTER
4018	027670				POP	<R3>	
4019	027672	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 027674 DQCMD::
4033 027674 004737 027732 JSR PC, LGSTAT ;CALL LOG STATS
4034 027700 062702 000014 ADD #PCBSTP, R2 ;ADJUST THE OLD COMMAND POINTER
4035 027704 022702 003344 CMP #PCBEND, R2 ;ARE WE AT THE END OF THE BUFFER ?
4036 027710 001002 BNE 5$ ;NO, KEEP GOING
4037 027712 012702 003264 MOV #PCMDBF, R2 ;YES, SET IT BACK TO THE TOP
4038 ;
4039 027716 005337 003402 5$: DEC RSPCNT ;DECREMENT THE RESPONSE COUNTER
4040 027722 012737 177777 010550 MOV #-1, CMSTSV ;RESET THE GCS PROGRESS COUNT
4041 027730 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 027732 LGSTAT::
4055 027732 032764 000004 000026 BIT #NOTALY,LUNFLG(R4) ;IS THIS NOT TO BE TALLIED ?
4056 027740 001162 BNE 10$ ;YES, GET OUT
4057 027742 122762 000040 000000 CMPB #ACC,CMD(R2) ;SEE IF COMMAND A READ OR WRITE
4058 027750 103556 BLO 10$ ;NO, EXIT SUBROUTINE
4059 027752 105762 000000 TSTB CMD(R2) ;IS IT A NULL ?
4060 027756 001553 BEQ 10$ ;YES, EXIT SUBROUTINE
4061 027760 122762 000020 000000 CMPB #WR,CMD(R2) ;IS IT A WRITE ?
4062 027766 001056 BNE 5$ ;NO, HANDLE READ
4063 027770 066264 000002 000120 ADD ITMOFF(R2),WRBYT1(R4) ;YES, ADD THE BYTES WRITTEN TO TOTAL
4064 027776 026427 000120 001747 1$: CMP WRBYT1(R4),#999. ;IS IT HIGER THAN 999. ?
4065 030004 003406 BLE 2$ ;BRANCH IF IT'S NOT
4066 030006 162764 001750 000120 SUB #1000.,WRBYT1(R4) ;SUBTRACT 1000. FROM THE LOWER ORDER WORD
4067 030014 005264 000122 INC WRBYT2(R4) ;INCREMENT THE SECOND WORD
4068 030020 000766 BR 1$
4069 030022 026427 000122 001747 2$: CMP WRBYT2(R4),#999. ;IS IT HIGER THAN 999. ?
4070 030030 003406 BLE 3$ ;BRANCH IF IT'S NOT
4071 030032 162764 001750 000122 SUB #1000.,WRBYT2(R4) ;SUBTRACT 1000. FROM THE LOWER ORDER WORD
4072 030040 005264 000124 INC WRBYT3(R4) ;INCREMENT THE SECOND WORD
4073 030044 000766 BR 2$
4074 030046 026427 000124 001747 3$: CMP WRBYT3(R4),#999. ;IS IT HIGER THAN 999. ?
4075 030054 003406 BLE 4$ ;BRANCH IF IT'S NOT
4076 030056 162764 001750 000124 SUB #1000.,WRBYT3(R4) ;SUBTRACT 1000. FROM THE LOWER ORDER WORD
4077 030064 005264 000126 INC WRBYT4(R4) ;INCREMENT THE SECOND WORD
4078 030070 000766 BR 3$
4079 030072 026427 000126 001747 4$: CMP WRBYT4(R4),#999. ;IS IT HIGER THAN 999. ?
4080 030100 003502 BLE 10$ ;BRANCH IF IT'S NOT
4081 030102 005064 000120 CLR WRBYT1(R4) ;
4082 030106 005064 000122 CLR WRBYT2(R4) ;
4083 030112 005064 000124 CLR WRBYT3(R4) ;
4084 030116 005064 000126 CLR WRBYT4(R4) ;
4085 030122 000471 BR 10$ ;EXIT
4086 ;
4087 030124 022763 000016 000012 5$: CMP #ST.TM,P.STS(R3) ;WAS THIS A TAPE MARK DURING READ
4088 030132 001465 BEQ 10$ ;YES, GET OUT
4089 030134 022763 000010 000012 CMP #10,P.STS(R3) ;WAS THIS A DATA ERROR DURING READ
4090 030142 001461 BEQ 10$ ;YES, GET OUT
4091 030144 022763 000350 000012 CMP #350,P.STS(R3) ;WAS THIS A DATA ERROR DURING READ
4092 030152 001455 BEQ 10$ ;YES, GET OUT
4093 030154 066364 000040 000110 ADD P.TRBC(R3),RDBYT1(R4) ;YES, ADD THE BYTES WRITTEN TO TOTAL
4094 030162 026427 000110 001747 6$: CMP RDBYT1(R4),#999. ;IS IT HIGER THAN 999. ?
4095 030170 003406 BLE 7$ ;BRANCH IF IT'S NOT
4096 030172 162764 001750 000110 SUB #1000.,RDBYT1(R4) ;SUBTRACT 1000. FROM THE LOWER ORDER WORD
4097 030200 005264 000112 INC RDBYT2(R4) ;INCRFMENT THE SECOND WORD
4098 030204 000766 BR 6$
4099 030206 026427 000112 001747 7$: CMP RDBYT2(R4),#999. ;IS IT HIGER THAN 999. ?
4100 030214 003406 BLE 8$ ;BRANCH IF IT'S NOT

```

4101	030216	162764	001750	000112	SUB	#1000.,RDBYT2(R4)	;SUBTRACT 1000. FROM THE LOWER ORDER WORD	
4102	030224	005264	000114		INC	RDBYT3(R4)	;INCREMENT THE SECOND WORD	
4103	030230	000766			BR	7\$		
4104	030232	026427	000114	001747	8\$:	CMP	RDBYT3(R4),#999.	;IS IT HIGER THAN 999. ?
4105	030240	003406			BLE	9\$;BRANCH IF IT'S NOT	
4106	030242	162764	001750	000114	SUB	#1000.,RDBYT3(R4)	;SUBTRACT 1000. FROM THE LOWER ORDER WORD	
4107	030250	005264	000116		INC	RDBYT4(R4)	;INCREMENT THE SECOND WORD	
4108	030254	000766			BR	8\$		
4109	030256	026427	000116	001747	9\$:	CMP	RDBYT4(R4),#999.	;IS IT HIGER THAN 999. ?
4110	030264	003410			BLE	10\$;BRANCH IF IT'S NOT	
4111	030266	005064	000110		CLR	RDBYT1(R4)	:	
4112	030272	005064	000112		CLR	RDBYT2(R4)	:	
4113	030276	005064	000114		CLR	RDBYT3(R4)	:	
4114	030302	005064	000116		CLR	RDBYT4(R4)	:	
4115	030306	042764	000004	000026	10\$:	BIC	#NOTALY,LUNFLG(R4)	;CLEAR THE NO-TALLY FLAG BEFORE EXITING
4116	030314	000207			RTS	PC	;RETURN	

```

4118 .SBTTL ERROR DECODE
4119 ;*****
4120 ;
4121 ; ERROR DECODE
4122 ;
4123 ;Called By      : RSPHDL
4124 ;Calls To       : ERRPLY, PRIERR
4125 ;Register Inputs : R2 - OLD PROGRAM COMMAND BUFFER POINTER
4126 ;
4127
4128 030316 ERRDEC::
4129 030316          PUSH    <R5>                ;SAVE R5
4130 030320 016205 000010          MOV     XFERST(R2),R5          ;PUT THE COMMAND STATUS IN R5
4131 030324 022705 000400          CMP     #ST.ONL,R5          ;IS IT A UNIT ONLINE ERROR ?
4132 030330 001005                BNE     997$                ;BRANCH IF IT ISN'T
4133 030332 012737 000100 003442          MOV     #ONLB,WRKMSK        ;SET THE ERROR BIT IN THE MASK
4134 030340 000137 030774          JMP     MSKTST              ;GO TEST IF IT'S O.K.
4135 030344 042705 177740          997$: BIC     #177740,R5        ;CLEAR THE UNWANTED BITS
4136 030350 022762 002000 000010          CMP     #2000,XFERST(R2)    ;IS EOT SET IN TRANSFER STATUS ?
4137 030356 001037                BNE     1$                  ;BRANCH IF IT ISN'T
4138 030360 052761 000004 003350          BIS     #EOT,DRINUS(R1)    ;SET THE DRIVE TO EOT
4139 030366 005237 003526          INC     UEOT                ;INC THE EOT FLAG
4140 030372 163737 003400 003402          SUB     CMDCNT,RSPCNT      ;SET RESPONSE COUNT TO NUMBER OUT
4141 030400 005037 003400          CLR     CMDCNT              ;ISSUE NO MORE COMMANDS
4142 030404 032764 000010 000026          BIT     #EOTPR,LUNFLG(R4)  ;HAS EOT BEEN PRINTED FOR THIS DRIVE ?
4143 030412 001017                BNE     999$                ;DON'T PRINT IT AGAIN
4144 030414                PUSH    <R1>                ;SAVE R1
4145 030416 006001                ROR     R1                  ;DIVIDE R1 BY 2
4146 030420          PRINTF #UNTEOT,R1      ;PRINT UNIT AT EOT MESSAGE
         030420 010146          MOV     R1,-(SP)
         030422 012746 020256          MOV     #UNTEOT,-(SP)
         030426 012746 000002          MOV     #2,-(SP)
         030432 010600          MOV     SP,R0
         030434 104417          TRAP   C$PNTF
         030436 062706 000006          ADD     #6,SP
4147 030442          POP     <R1>                ;RESTORE R1
4148 030444 052764 000010 000026          BIS     #EOTPR,LUNFLG(R4)  ;EOT BEEN PRINTED FOR THIS DRIVE
4149 030452 000137 031244          999$: JMP     EDCEXT              ;GET OUT
4150
4151 030456 012737 010652 003530 1$:  MOV     #CMDT,R8            ;PUT THE ERROR TABLE ADDRESS IN R8
4152 030464 022705 000013          CMP     #13,R5              ;IS IT A DRIVE ERROR ?
4153 030470 001005                BNE     4$                  ;NO, CONTINUE
4154 030472 052737 000010 003516          BIS     #DRERFL,PCFLAG     ;SET THE DRIVE ERROR FLAG
4155 030500 004737 033376          JSR     PC,OCTHEX           ;GO DECODE THE STATUS
4156 030504 022705 000023          4$:  CMP     #23,R5            ;IS IT A VALID STATUS ?
4157 030510 103003                BHS     5$                  ;IT'S VALID, BRANCH
4158 030512 012705 000024          MOV     #24,R5              ;MAKE SURE ITS NOT MORE THAN 24
4159 030516 000543                BR      ERREXT              ;TAKE THE ERROR EXIT
4160
4161 030520 012737 000002 003406 5$:  MOV     #SEREXC,RESPON      ;SET SERIOUS EXCEPTION
4162 030526 016337 000000 003542          MOV     P.CRF(R3),SECRNS   ;SAVE THE CURRENT COMMAND REF #
4163 030534 022705 000006          CMP     #ST.WPR,R5          ;IS IT A WRITE PROTECT ERROR ?
4164 030540 001004                BNE     10$                 ;BRANCH IF IT ISN'T
4165 030542 012737 000020 003442          MOV     #WPRB,WRKMSK       ;SET THE ERROR BIT IN THE MASK
4166 030550 000511                BR      MSKTST              ;GO TEST IF IT'S O.K.
4167
4168 030552 022705 000016          10$: CMP     #ST.TM,R5        ;IS IT A TAPE MARK ERROR ?

```

```
4169 030556 001061          BNE      15$          ;BRANCH IF IT ISN'T
4170 030560 052764 000002 000026  BIS      @SEREXC,LUNFLG(R4) ;SET SERIOUS EXCEPTION
4171 030566 000261          SEC
4172 030570 006164 000030  ROL      LEOTFL(R4)      ;
4173 030574 042764 177774 000030  BIC      @177774,LEOTFL(R4) ;
4174 030602 022764 000003 000030  CMP      @3,LEOTFL(R4)    ;
4175 030610 001032          BNE      13$          ;
4176 030612 052761 000004 003350  BIS      @EOT,DRINUS(R1) ;
4177 030620 005237 003526  INC      UEOT           ;INC THE EOT FLAG
4178 030624          PUSH     <R1>         ;SAVE R1
4179 030626 006001          ROR      R1            ;DIVIDE R1 BY 2
4180 030630          PRINTF  @UNTLOT,R1    ;PRINT UNIT AT EOT MESSAGE
      030630 010146          MOV      R1,-(SP)
      030632 012746 020312  MOV      @UNTLOT,-(SP)
      030636 012746 000002  MOV      @2,-(SP)
      030642 010600          MOV      SP,R0
      030644 104417          TRAP    C$PNTF
      030646 062706 000006  ADD      @6,SP
4181 030652          POP      <R1>         ;RESTORE R1
4182 030654 163737 003400 003402  SUB      CMDCNT,RSPCNT  ;SET RESPONSE COUNT TO NUMBER OUT
4183 030662 005037 003400          CLR      CMDCNT       ;ISSUE NO MORE COMMANDS
4184 030666 005037 003406          CLR      RESPON      ;MAKE SURE WE
4185 030672 000137 031244          JMP      EDCEXT       ;GET OUT
4186
4187 030676 132763 000010 000011 13$:  BITB    @EF.EOT,P.FLGS(R3) ;IS THE TAPE MARK AT EOT ?
4188 030704 001402          BEQ     14$          ;NO, KEEP ON GOING
4189 030706 000137 031244          JMP     EDCEXT       ;YES, GET OUT
4190 030712 012737 000010 003442 14$:  MOV     @TMB,WRKMSK    ;SET THE ERROR BIT IN THE MASK
4191 030720 000425          BR      MSKTST       ;GO TEST IF IT'S O.K.
4192
4193 030722 022705 000020          15$:  CMP     @ST.RDT,R5    ;IS IT A RECORD DATA TRUNCATED ERROR ?
4194 030726 001007          BNE    20$          ;BRANCH IF IT ISN'T
4195 030730 000241          CLC
4196 030732 006164 000030  ROL     LEOTFL(R4)    ;CLEAR THE CARRY BIT
4197 030736 012737 000002 003442  MOV     @RDTB,WRKMSK  ;ROTATE THE CARRY INTO THE LEOT FLAG
4198 030744 000413          BR     MSKTST       ;SET THE ERROR BIT IN THE MASK
4199
4200 030746 022705 000023          20$:  CMP     @ST.LED,R5    ;IS IT A LOGICAL END OF TAPE ERROR ?
4201 030752 001002          BNE    25$          ;BRANCH IF IT ISN'T
4202 030754 000137 031244          JMP     EDCEXT       ;GET OUT IF LEOT DETECTED
4203
4204 030760 022705 000004          25$:  CMP     @ST.AVL,R5    ;IS IT A UNIT AVAILABLE ERROR ?
4205 030764 001020          BNE    ERREXT       ;BRANCH IF IT ISN'T
4206 030766 012737 000040 003442  MOV     @AVLB,WRKMSK  ;SET THE ERROR BIT IN THE MASK
4207
4208 030774 033737 003442 003440  MSKTST: BIT     WRKMSK,TSTMSK ;IS IT AN ACCEPTABLE ERROR ?
4209 031002 001120          BNE    EDCEXT       ;GET OUT IF IT IS
4210 031004 000410          BR     ERREXT       ;OTHERWISE PRINT THE ERROR
4211
4212 031006          ERRDEI::
4213 031006          PUSH   <R5>         ;SAVE R5
4214 031010 113705 010546          MOVB   IOSTAT,R5     ;PUT THE I/O ERROR CODE INTO R5
4215 031014 012737 010572 003530  MOV     @IOERTB,R8    ;SET THE ERROR TABLE ADDRESS IN R8
4216 031022 042705 177770          BIC    @177770,R5    ;CLEAR OFF ALL UNWANTED BITS
4217
4218 031026 005305          ERREXT: DEC     R5    ;SUBTRACT 1 FROM R5
4219 031030 006305          ASL    R5           ;MULTIPLY R5 BY 10(8)
```

```
4220 031032 006305          ASL      R5          ;
4221 031034 006305          ASL      R5          ;
4222 031036 063705 003530    ADD      R8,R5      ;ADD THE TABLE ADDRESS TO R5
4223 031042          PUSH     <R3>
4224 031044 012703 013262    MOV      @L$ERRTBL,R3 ;SET R3 TO THE GENERIC ERROR TABLE
4225 031050 012523          MOV      (R5),.(R3). ;MOVE ERROR TABLE CONTENTS
4226 031052 012523          MOV      (R5),.(R3). ;MOVE ERROR TABLE CONTENTS
4227 031054 012523          MOV      (R5),.(R3). ;MOVE ERROR TABLE CONTENTS
4228 031056 011513          MOV      (R5),(R3)   ;MOVE ERROR TABLE CONTENTS
4229 031060          POP      <R3>
4230
4231 031062 022762 000010 000010    CMP      @EV.LGP,XFERST(R2) ;IS IS A LONG GAP ENCOUNTERED ?
4232 031070 001031          BNE     ERTLY       ;NO, KEEP GOING
4233 031072 112737 000001 013262    MOVB     @DEVFAT,ERRTYP ;YES, DROP THE UNIT
4234 031100 004737 033742    JSR      PC,CORDMP   ;;;;GO DO IT
4235 031104 000240          NOP
4236 031106 122762 000020 000000    CMPB     @WR,CMD(R2)   ;IS IT A WRITE ?
4237 031114 001004          BNE     1$         ;NO TRY AGAIN
4238 031116 112737 000060 013263    MOVB     @H1WRIT,ERRTYP*1 ;AND TALLY THE HARD WRITE ERROR
4239 031124 000413          BR      ERTLY      ;GO TALLY THE ERROR
4240 031126 122762 000100 000000 1$:  CMPB     @WTM,CMD(R2)   ;IS IT A WRITE TAPE MARK ?
4241 031134 001004          BNE     5$         ;NO, MUST BE A READ
4242 031136 112737 000060 013263    MOVB     @H1WRIT,ERRTYP*1 ;AND TALLY THE HARD WRITE ERROR
4243 031144 000403          BR      ERTLY      ;TALLY THE ERROR
4244 031146 112737 000054 013263 5$:  MOVB     @H1READ,ERRTYP*1 ;TALLY THE HARD READ ERROR
4245 031154 004737 032050          JSR      PC,ERTLY    ;TALLY THE ERROR
4246 031160 105037 013263          CLRB    ERRTYP*1    ;CLEAR UPPER BYTE
4247 031164 105737 002224          TSTB    SOERRP      ;ARE SOFT ERRORS ENABLED ?
4248 031170 001004          BNE     6$         ;YES, GO PRINT THE ERROR
4249 031172 122737 000003 013262    CMPB     @SOFT,L$ERRTBL ;IS IT A SOFT ERROR ?
4250 031200 001402          BEQ     8$         ;YES, DON'T PRINT IT
4251 031202 004737 032550          JSR      PC,PRIERR   ;GO PRINT THE ERROR
4252
4253 031206 132737 000001 002231 8$:  BITB     @BITO,DMPFLG ;SHOULD WE DUMP PROGRAM TABLES?
4254 031214 001403          BEQ     10$        ;NO BRANCH
4255 031216 004737 033742    JSR      PC,CORDMP   ;GO DO IT
4256 031222 000240          NOP
4257 031224 022737 000001 013262 10$:  CMP      @DEVFAT,ERRTYP ;IS IT A FATAL ERROR ?
4258 031232 001004          BNE     EDCEXT     ;NO EXIT
4259 031234 010100          MOV     R1,R0      ;MOVE UNIT # * 2 TO R0
4260 031236 006000          ROR     R0         ;DIVIDE BY 2
4261 031240 004737 040450          JSR     PC,DROPUN   ;DROP DRIVE FROM TESTING
4262 031244          EDCEXT: POP     <R5> ;RESTORE REGISTERS
4263 031246 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 ERRDEL::
4279 031250      PUSH    <R3,R5>
4280 031254 116237 000000 003536      MOVB   CMD(R2),R11      ;GET THE COMMAND PRIMITIVE FOR LATER USE
4281 031262 042737 177407 003536      BIC   #177407,R11      ;GET JUST THE ROOT PRIMITIVE
4282 031270      PUSH    <R1>      ;SAVE R1
4283 031272 122763 000005 000010      CMPB  #FM.TPE,L.FMT(R3) ;TAPE TRANSFER ERROR LOG?
4284 031300 001420      BEQ   5$              ;YES, DECODE IT
4285 031302 122763 000000 000010      CMPB  #FM.CNT,L.FMT(R3) ;CONTROLLER ERROR LOG?
4286 031310 001004      BNE   1$              ;NO, SEE WHAT IT IS
4287 031312 012705 011132      MOV   #CNTLER,R5      ;PRINT PACKET
4288 031316 000137 031736      JMP   PRTEXT          ;GET OUT
4289 031322 122763 000001 000010 1$:  CMPB  #FM.BAD,L.FMT(R3) ;HOST MEMORY ACCESS ERROR LOG?
4290 031330 001004      BNE   5$              ;NO, GET OUT
4291 031332 012705 011142      MOV   #BADERL,R5     ;SET UP TO PRINT HOST MEM ACC ERL
4292 031336 000137 031736      JMP   PRTEXT          ;GO PRINT IT
4293 031342 016337 000012 003556 5$:  MOV   L.EVNT(R3),EVENT ;GET A COPY OF THE EVENT CODE
4294 031350 042737 177740 003556      BIC   #177740,EVENT   ;GET RID OF THE SUBCODES
4295 031356 022737 000007 003556      CMP   #ST.CMP,EVENT   ;IS IT A COMPARE ERROR ?
4296 031364 001004      BNE   10$             ;NO, KEEP GOING
4297 031366 012705 011152      MOV   #CMPERL,R5     ;SET UP TO PRINT ERROR
4298 031372 000137 031736      JMP   PRTEXT          ;GO PRINT THE ERROR
4299 031376 022737 000013 003556 10$:  CMP   #ST.DRV,EVENT   ;IS IT A DRIVE ERROR ?
4300 031404 001004      BNE   15$             ;NO, KEEP GOING
4301 031406 012705 011162      MOV   #DRVERL,R5     ;SET UP TO PRINT ERROR
4302 031412 000137 031736      JMP   PRTEXT          ;GO PRINT THE ERROR
4303 031416 022737 000012 003556 15$:  CMP   #ST.CNT,EVENT   ;IS IT A CONTROLLER ERROR ?
4304 031424 001004      BNE   DATAEL        ;NO, MUST BE A DATA ERROR
4305 031426 012705 011172      MOV   #CNTERL,R5     ;SET UP TO PRINT ERROR
4306 031432 000137 031736      JMP   PRTEXT          ;GO PRINT THE ERROR
4307 031436 022763 000010 000012 DATAEL:  CMP   #EV.LGP,L.EVNT(R3) ;IS IT A LONG GAP ENCOUNTERED ?
4308 031444 001004      BNE   1$              ;NO, KEEP TRYING
4309 031446 012705 011202      MOV   #LGPERL,R5     ;SET UP TO PRINT LGE ERROR
4310 031452 000137 031736      JMP   PRTEXT          ;GO PRINT IT
4311 031456 012701 003360 1$:  MOV   #UNTTBL,R1      ;PUT THE TEMP TABLE ADDRESS IN R1
4312 031462 116361 000055 000000      MOVB  L.TRK(R3),TRK(R1) ;PUT THE TRACK NUMBER IN TEMP
4313 031470 016361 000056 000002      MOV   L.PBLK(R3),PHB(R1) ;PHYSICAL BLOCK NUMBER TO TEMP
4314 031476 022737 000020 003536      CMP   #WR,R11        ;WAS IT A WRITE COMMAND ?
4315 031504 001404      BEQ   WRERL          ;GO HANDLE WRITE ERROR
4316 031506 022737 000100 003536      CMP   #WTM,R11       ;IS IT A WRITE TAPE MARK COMMAND ?
4317 031514 001026      BNE   RDERL          ;GO HANDLE READ ERROR LOG
4318 031516 022763 000150 000012 WRERL:  CMP   #EV.COR,L.EVNT(R3) ;IS IT A SOFT ERROR ?
4319 031524 001005      BNE   1$              ;NO, INCREMENT HARD ERROR
4320 031526 105261 000004      INCB  SWR(R1)        ;YES, INCREMENT SOFT WRITE ERROR
4321 031532 012705 011232      MOV   #CORERL,R5     ;SET UP TO PRINT ERROR
4322 031536 000413      BR    10$            ;KEEP GOING
    
```

4323	031540	105261	000006		1\$:	INCB	HWR(R1)		;INCREMENT HARD WRITE ERROR
4324	031544	022763	000050	000012		CMP	#EV.DST,L.EVNT(R3)		;IS IT A DATA SYNC TIMEOUT ?
4325	031552	001003				BNE	5\$;NO, KEEP TRYING
4326	031554	012705	011222			MOV	#WDSTEL,R5		;SET UP TO PRINT DST ERROR
4327	031560	000402				BR	10\$;GO PRINT IT
4328	031562	012705	011242		5\$:	MOV	#UWEERL,R5		;SET UP TO PRINT ERROR
4329	031566	000137	031732		10\$:	JMP	SRTBL		;GO SORT THE ERROR AND PRINT
4330	031572	022763	000350	000012	RDERL:	CMP	#EV.URE,L.EVNT(R3)		;IS IT A HARD READ ERROR ?
4331	031600	001006				BNE	1\$;NO, HANDLE SOFT READ
4332	031602	105261	000007			INCB	HRD(R1)		;INC THE HARD READ ERROR
4333	031606	012705	011262			MOV	#UREERL,R5		;SET UP TO PRINT ERROR
4334	031612	000137	031732			JMP	SRTBL		;GO SORT AND PRINT IT
4335	031616	022763	000050	000012	1\$:	CMP	#EV.DST,L.EVNT(R3)		;IS IT A DATA SYNC TIMEOUT ?
4336	031624	001006				BNE	5\$;NO, KEEP TRYING
4337	031626	105261	000007			INCB	HRD(R1)		;INC THE HARD READ ERROR
4338	031632	012705	011212			MOV	#RDSTEL,R5		;SET UP TO PRINT DST ERROR
4339	031636	000137	031732			JMP	SRTBL		;GO SORT AND PRINT IT
4340	031642	105261	000005		5\$:	INCB	SRD(R1)		;INC THE SOFT READ ERROR
4341	031646	105763	000042			TSTB	L.LVL(R3)		;WAS IT RECOVERED BY ECC ?
4342	031652	001025				BNE	20\$;NO, KEEP TRYING
4343	031654	122763	000036	000052		CMPB	#ECCBC,L.STS(R3)		;WAS THE ERROR ON AN ECC BLOCK ?
4344	031662	001003				BNE	10\$;NO TRY AGAIN
4345	031664	012705	011272			MOV	#ECBERL,R5		;SET UP TO PRINT ERROR
4346	031670	000420				BR	SRTBL		;GO SORT AND PRINT ERROR
4347	031672	122763	000030	000052	10\$:	CMPB	#ECCTC,L.STS(R3)		;WAS ERROR ON A TAPE MARK ?
4348	031700	001003				BNE	15\$;TRY AGAIN
4349	031702	012705	011302			MOV	#ECTERL,R5		;SET UP TO PRINT ERROR
4350	031706	000411				BR	SRTBL		;GO SORT AND PRINT ERROR
4351	031710	122763	000026	000052	15\$:	CMPB	#ECCDC,L.STS(R3)		;WAS ERROR ON DATA BLOCK ?
4352	031716	001003				BNE	20\$;NO, TRY AGAIN
4353	031720	012705	011312			MOV	#ECDERL,R5		;SET UP TO PRINT ERROR
4354	031724	000402				BR	SRTBL		;GO SORT AND PRINT ERROR
4355	031726	012705	011252		20\$:	MOV	#RTYERL,R5		;SET UP RETRY RECOVERED
4356	031732	004737	032174		SRTBL:	JSR	PC,TBLSRT		;GO TALLY ERROR IN MEDIA TABLE
4357	031736	116237	000000	003536	PRTEXT:	MOVB	CMD(R2),R11		;GET THE COMMAND PRIMITIVE FOR LATER USE
4358	031744	012701	013262			MOV	#L\$ERRTBL,R1		;R1 = SUPERVISORS ERROR TABLE
4359	031750	012521				MOV	(R5)+,(R1)+		;COPY PROGRAM'S ERROR TABLE
4360	031752	012521				MOV	(R5)+,(R1)+		; TO SUPERVISOR'S
4361	031754	012521				MOV	(R5)+,(R1)+		; ERROR
4362	031756	011511				MOV	(R5),(R1)		; TABLE
4363	031760					POP	<R1>		;RESTORE R1
4364	031762	004737	032132			JSR	PC,ETLLY		;TALLY ERROR FIRST
4365	031766	105037	013263			CLRB	ERRTYP+1		;DISCARD INFO USED BY ERRTLLY
4366	031772	105737	002224			TSTB	SOERRP		;ARE SOFT ERRORS ENABLED ?
4367	031776	001017				BNE	1\$;YES, GO PRINT THE ERROR
4368	032000	122737	000003	013262		CMPB	#SOFT,L\$ERRTBL		;IS IT A SOFT ERROR ?
4369	032006	001013				BNE	1\$;NO, PRINT IT
4370	032010	022737	000020	003536		CMP	#WR,R11		;IS IT A WRITE ?
4371	032016	001411				BEQ	EDLEXT		;DON'T PRINT IT
4372	032020	022737	000100	003536		CMP	#WTM,R11		;IS IT A WRITE TAPE MARK ?
4373	032026	001405				BEQ	EDLEXT		;DON'T PRINT IT
4374	032030	105737	002225			TSTB	RDSOER		;ARE WE PRINTING SOFT READ ERRORS ?
4375	032034	001402				BEQ	EDLEXT		;NO, DON'T PRINT IT
4376	032036	004737	032550		1\$:	JSR	PC,PRIERR		;GO PRINT IT
4377	032042				EDLEXT:	POP	<R5,R3>		
4378	032046	000207				RTS	PC		

```

4380 .SBTTL ERROR TALLY
4381 ;*****
4382 ;
4383 ; ERROR TALLY
4384 ;
4385 ;Called By      : ERRDEC, ERRDEI, ERDEL
4386 ;
4387
4388 032050 ERRPLY::
4389 032050          PUSH    <R1>          ;SAVE R1
4390 032052 122737 000106 013263          CMPB   #NOERR,ERRTYP+1      ;IS IT A STATUS ERROR ?
4391 032060 001422          BEQ     15$          ;YES, GET OUT
4392 032062 113701 013263          MOVB   ERRTYP+1,R1        ;PUT THE ERROR TYPE IN R1
4393 032066 122737 000076 013263          CMPB   #RMSPOS,ERRTYP+1    ;IS IT A MISPOSITION ERROR
4394 032074 001012          BNE     10$          ;NO KEEP GOING
4395 032076 122762 000020 000000          CMPB   #WR,CMD(R2)        ;IS IT A WRITE ?
4396 032104 001404          BEQ     5$          ;NO TRY AGAIN
4397 032106 122762 000100 000000          CMPB   #WTM,CMD(R2)       ;IS IT A WRITE TAPE MARK ?
4398 032114 001002          BNE     10$          ;NO, MUST BE A READ
4399 032116 112701 000100          5$:  MOVB   #WMSPOS,R1      ;SET IT TO A WRITE MISPOSITON
4400 032122 060401          10$:  ADD    R4,R1          ;ADD THE OFFSET TO THE LUN POINTER
4401 032124 005211          INC    (R1)            ;INC THE ERROR COUNT
4402 032126          15$:  POP    <R1>          ;RESTORE R1 AND R4
4403 032130 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 032132      ETLTY::
4419 032132      PUSH    <R1>                ;SAVE R1
4420 032134 122737 000106 013263      CMPB    #NOERR,ERRTYP+1          ;IS IT A STATUS ERROR ?
4421 032142 001412                BEQ     ELTEXT                  ;YES, GET OUT
4422 032144 113701 013263      MOVB    ERRTYP+1,R1            ;LET R1 = THE ERROR TYPE
4423 032150 060401                ADD     R4,R1                  ;ADD THE LUN BLOCK POINTER TO R1
4424 032152 132763 000001 000055      BITB    #BIT0,L.TRK(R3)        ;IS IT AN ODD TRACK NUMBER ?
4425 032160 001402                BEQ     1$                    ;YES, GO TALLY THE ERROR
4426 032162 062701 000002                ADD     #2,R1                  ;NO, TALLY THE EVEN TRACK
4427 032166 005211                1$: INC    (R1)                ;INC THE ERROR COUNT
4428 032170                ELTEXT: POP   <R1>            ;RESTORE R1
4429 032172 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 032174 TBLSORT::PUSH <R1,R2,R3,R5> ;SAVE R1,R2,R3,R5 ON STACK
4441 032204 012701 003360 MOV #UNTTBL,R1 ;INITIALIZE R1,R2,R3,R5
4442 032210 016402 000130 MOV TBLTOP(R4),R2 ;FOR LOCAL USAGE
4443 032214 016403 000134 MOV LSTENT(R4),R3
4444 032220 010205 MOV R2,R5
4445 032222 020203 CMP R2,R3 ;FIRST ENTRY?
4446 032224 001524 BEQ 30$ ;YES, PUT IT IN THE TABLE
4447
4448 032226 126162 000000 000000 1$: CMPB TRK(R1),TRK(R2) ;GET TO THE RIGHT TRACK
4449 032234 101405 BLOS 5$
4450 032236 062702 000010 ADD #8.,R2
4451 032242 020203 CMP R2,R3 ;NEXT UNUSED SLOT ?
4452 032244 001514 BEQ 30$ ;YES, PUT IT IN THE TABLE
4453 032246 000767 BR 1$
4454
4455 032250 126162 000000 000000 5$: CMPB TRK(R1),TRK(R2) ;MAKE SURE ITS THE SAME TRACK
4456 032256 001024 BNE 15$
4457 032260 026162 000002 000002 CMP PHB(R1),PHB(R2) ;IF OUR OBJ CNT HIGHER
4458 032266 101405 BLOS 10$
4459 032270 062702 000010 ADD #8.,R2 ;GO TO NEXT TABLE LOCATION
4460 032274 020203 CMP R2,R3 ;NEXT UNUSED SLOT ?
4461 032276 001477 BEQ 30$ ;YES, PUT IT IN THE TABLE
4462 032300 000763 BR 5$
4463
4464 032302 026162 000002 000002 10$: CMP PHB(R1),PHB(R2) ;IF THE ENTRY ALREADY EXIST
4465 032310 001007 BNE 15$
4466 032312 066162 000004 000004 ADD SWR(R1),SWR(R2) ;THEN ADD THE ERROR COUNT
4467 032320 066162 000006 000006 ADD HWR(R1),HWR(R2)
4468 032326 000477 BR EXTSRT
4469
4470 032330 026464 000134 000132 15$: CMP LSTENT(R4),TBLBTM(R4) ;IF THE TABLES FULL, GET OUT
4471 032336 001022 BNE 20$
4472 032340 032764 000040 000026 17$: BIT #MTBLOV,LUNFLG(R4)
4473 032346 001067 BNE EXTSRT
4474 032350 052764 000040 000026 BIS #MTBLOV,LUNFLG(R4)
4475 032356 PRINTF #TBLFUL,L$LUN
032356 013746 002074 MOV L$LUN,-(SP)
032362 012746 036724 MOV #TBLFUL,-(SP)
032366 012746 000002 MOV #2,-(SP)
032372 010600 MOV SP,R0
032374 104417 TRA? C$PNTF
032376 062706 000006 ADD #6,SP
4476 032402 000451 BR EXTSRT
4477
4478 032404 016405 000134 20$: MOV LSTENT(R4),R5 ;MOVE THE TABLE CONTENTS
4479 032410 010503 MOV R5,R3 ;DOWN TO PUT IN THE ENTRY
4480 032412 162703 000010 SUB #8.,R3
4481

```

```

4482 032416 116365 000000 000000 25$:  MOVB  TRK(R3),TRK(R5)
4483 032424 016365 000002 000002      MOV   PHB(R3),PHB(R5)
4484 032432 116365 000004 000004      MOVB  SWR(R3),SWR(R5)
4485 032440 116365 000005 000005      MOVB  SRD(R3),SRD(R5)
4486 032446 116365 000006 000006      MOVB  HWR(R3),HWR(R5)
4487 032454 116365 000007 000007      MOVB  HRD(R3),HRD(R5)
4488 032462 162703 000010      SUB   #8.,R3
4489 032466 162705 000010      SUB   #8.,R5
4490 032472 020502      CMP   R5,R2      ;KEEP GOING TILL THE RIGHT SLOT
4491 032474 101350      BHI   25$        ;IS OPEN
4492
4493 032476 026464 000134 000132 30$:  CMP   LSTENT(R4),TBLBTM(R4) ;IF THE TABLES FULL, GET OUT
4494 032504 001715      BEQ   17$
4495 032506 020301      CMP   R3,R1
4496 032510 103403      BLO   35$
4497 032512 010103      MOV   R1,R3
4498 032514 010205      MOV   R2,R5
4499 032516 000737      BR    25$
4500 032520 062764 000010 000134 35$:  ADD   #8.,LSTENT(R4)      ;ADJUST THE LAST ENTRY POINTER
4501 032526 005061 000004      EXTSRT: CLR  SWR(R1)      ;CLEAR SOFT ERROR COUNTERS
4502 032532 005061 000006      CLR  HWR(R1)      ;CLEAR HARD ERROR COUNTERS
4503 032536      POP   <R5,R3,R2,R1>    ;RESTORE REGISTERS
4504 032546 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 032550 PRIERR::
4517 032550 010337 003560 MOV R3,R3SAVE ;SAVE R3
4518 032554 010437 003562 MOV R4,R4SAVE ;SAVE R4
4519 032560 ERROR ;ERROR MACRO
4520 032560 104460 TRAP C$ERROR
4521 032562 000207 RTS PC ;RETURN
```

```

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 CMPDAT::
4537 032564 PUSH <R1,R2,R3,R5> ;SAVE R1,R2,R3,R5
032564 010146 MOV R1,-(SP) ;;PUSH R1 ON STACK
032566 010246 MOV R2,-(SP) ;;PUSH R2 ON STACK
032570 010346 MOV R3,-(SP) ;;PUSH R3 ON STACK
032572 010546 MOV R5,-(SP) ;;PUSH R5 ON STACK
4538 032574 026363 000040 000014 CMP P.TRBC(R3),P.BCNT(R3) ;AS MANY BYTES READ AS WRITTEN ?
4539 032602 001424 BEQ 5$ ;BRANCH IF YES
4540 032604 012705 011122 MOV #RLST,R5 ;PUT THE RLS TABLE ADDRESS IN R5
4541 032610 012702 013262 MOV #L$ERRTBL,R2 ;PUT THE ERROR TABLE ADDRESS IN R2
4542 032614 012522 MOV (R5)+,(R2)+ ;MOV THE RLS TABLE TO THE ERROR TABLE
4543 032616 012522 MOV (R5)+,(R2)+ ;
4544 032620 012522 MOV (R5)+,(R2)+ ;
4545 032622 011512 MOV (R5),(R2) ;
4546 032624 POP <R5,R3,R2,R1> ;RESTORE REGISTERS
032624 012605 MOV (SP)+,R5 ;;POP STACK INTO R5
032626 012603 MOV (SP)+,R3 ;;POP STACK INTO R3
032630 012602 MOV (SP)+,R2 ;;POP STACK INTO R2
032632 012601 MOV (SP)+,R1 ;;POP STACK INTO R1
4547 032634 004737 032050 JSR PC,ERRTLY ;GO TALLY THE ERROR
4548 032640 105037 013263 CLR#B ERRTP+1 ;CLEAR THE LUN POINTER
4549 032644 004737 032550 JSR PC,PRIERR ;GO PRINT THE ERROR
4550 032650 000137 033246 JMP 45$ ;GET OUT IF THERE WAS AN ERROR
4551 032654 005037 003444 5$: CLR CMPERR ;CLEAR LOCATION CMPERR
4552 032660 042764 000020 000026 BIC #ODDFLG,LUNFLG(R4) ;CLEAR THE ODD BYTE COUNT FLAG
4553 032666 016337 000014 003530 MOV P.BCNT(R3),R8 ;PUT THE TAPE RECORD BYTE COUNT IN R8
4554 032674 005037 003532 CLR R9 ;CLEAR THE BYTE ADDRESS COUNTER
4555 032700 032737 000001 003530 BIT #BIT0,R8 ;IS THE BYTE COUNT ODD
4556 032706 001406 BEQ 10$ ;BRANCH IF NOT
4557 032710 042737 000001 003530 BIC #BIT0,R8 ;MAKE THE COUNT EVEN
4558 032716 052764 000020 000026 BIS #ODDFLG,LUNFLG(R4) ;SET THE ODD BYTE FLAG
4559 032724 012701 003446 10$: MOV #BYTADD,R1 ;LET R1 POINT TO THE ADDRESS TABLE
4560 032730 022737 000003 002114 CMP #3,L$TEST ;ARE WE IN TEST 3 ?
4561 032736 001003 BNE 11$ ;NO, SO JUST SET RDBUF IN BUFADR
4562 032740 012702 070556 MOV #WRTBUF-2,R2 ;LET R2 POINT TO THE WRITE BUFFER
4563 032744 000402 BR 12$
4564 032746 012702 070560 11$: MOV #WRTBUF,R2 ;LET R2 POINT TO THE WRITE BUFFER
4565 032752 012703 050560 12$: MOV #RDBUF,R3 ;LET R3 POINT TO THE READ BUFFER
4566 032756 012705 003472 MOV #DATBL,R5 ;LET R5 POINT TO THE ERROR DATA TABLE
4567 032762 022322 14$: CMP (R3)+,(R2)+ ;COMPARE THE FIRST WORD OF DATA
4568 032764 001447 BEQ 25$ ;BRANCH IF THEY ARE EQUAL
4569 032766 126362 177776 177776 CMP#B LOBYTE(R3),LOBYTE(R2) ;COMPARE THE LOW BYTE
4570 032774 001415 BEQ 15$ ;BRANCH IF EQUAL
4571 032776 005237 003444 INC CMPERR ;ADD 1 TO THE ERROR COUNT

```



```

4572 033002 022701 003472      CMP      #TBLEND,R1      ;IS THERE ROOM TO SAVE THIS DATA ?
4573 033006 001410              BEQ      15$             ;BRANCH IF NOT
4574 033010 116315 177776              MOV      LOBYTE(R3),(R5) ;LOW READ BYTE IN SAVE BYTE
4575 033014 116265 177776 000001      MOV      LOBYTE(R2),ONE(R5) ;LOW WRITE BYTE IN SAVE BYTE
4576 033022 013721 003532              MOV      R9,(R1)+       ;SAVE THE ADDRESS FOR PRINTING
4577 033026 005725              TST      (R5)+          ;POINT R5 TO NEXT TABLE LOCATION
4578 033030 005237 003532 177777 177777 177777 177777 177777 177777 ;ADD 1 TO THE BYTE COUNT
4579 033034 126362 177777 177777 177777 177777 177777 177777 177777 ;COMPARE THE HIGH BYTE
4580 033042 001415              BEQ      20$             ;BRANCH IF THEY ARE THE SAME
4581 033044 005237 003444              INC      CMPERR          ;ADD 1 TO THE ERROR COUNT
4582 033050 022701 003472      CMP      #TBLEND,R1      ;ROOM TO SAVE THIS DATA ?
4583 033054 001410              BEQ      20$             ;BRANCH IF NOT
4584 033056 116315 177777              MOV      HIBYTE(R3),(R5) ;HI READ BYTE TO LOW SAVE BYTE
4585 033062 116265 177777 000001      MOV      HIBYTE(R2),ONE(R5) ;HIGH WRITE BYTE TO HIGH SAVE BYTE
4586 033070 013721 003532              MOV      R9,(R1)+       ;SAVE THE ADDRESS TO PRINT
4587 033074 005725              TST      (R5)+          ;POINT R5 TO THE NEXT TABLE LOCATION
4588 033076 005237 003532              INC      R9              ;ADD 1 TO THE BYTE COUNTER
4589 033102 000403              BR       30$             ;BRANCH
4590 033104 062737 000002 003532 25$: ADD      #2,R9            ;ADD 2 TO THE BYTE COUNT
4591 033112 023737 003532 003530 30$: CMP      R9,R8            ;HAVE WE COMPARED THEM ALL ?
4592 033120 001320              BNE      14$             ;NO, GO DO SOME MORE
4593 033122 032764 000020 000026      BIT      #ODDFLG,LUNFLG(R4) ;WAS IT AN ODD BYTE COUNT ?
4594 033130 001414              BEQ      35$             ;BRANCH IF NOT
4595 033132 121312              CMP      (R3),(R2)       ;COMPARE THE LOW BYTES
4596 033134 001412              BEQ      35$             ;BRANCH IF THEY MATCH
4597 033136 005237 003444              INC      CMPERR          ;ADD 1 TO THE ERROR COUNT
4598 033142 022701 003472      CMP      #TBLEND,R1      ;IS THERE ROOM FOR THIS DATA ?
4599 033146 001405              BEQ      35$             ;BRANCH IF NOT
4600 033150 111315              MOV      (R3),(R5)       ;LOW READ BYTE IN THE LOW SAVE BYTE
4601 033152 111265 000001      MOV      (R2),ONE(R5)    ;LOW WRITE BYTE TO HIGH SAVE BYTE
4602 033156 013721 003532              MOV      R9,(R1)+       ;SAVE THE ADDRESS TO PRINT
4603 033162 005737 003444              TST      CMPERR          ;DID WE HAVE ANY ERRORS ?
4604 033166 001423              BEQ      40$             ;BRANCH TO EXIT IF NOT
4605 033170 012705 011112              MOV      #DCMPT,R5       ;POINT R5 TO THE DATA COMPARE ERROR TABLE
4606 033174 012702 013262              MOV      #LERRTBL,R2     ;POINT R2 TO THE ERROR TABLE
4607 033200 012522              MOV      (R5)+,(R2)+     ;MOVE THE DATA COMPARE TABLE
4608 033202 012522              MOV      (R5)+,(R2)+     ;TO THE ERROR TABLE
4609 033204 012522              MOV      (R5)+,(R2)+     ;
4610 033206 011512              MOV      (R5),(R2)       ;
4611 033210              POP      <R5,R3,R2,R1>   ;RESTORE THE REGISTERS
      033210 012605              MOV      (SP)+,R5        ;;POP STACK INTO R5
      033212 012603              MOV      (SP)+,R3        ;;POP STACK INTO R3
      033214 012602              MOV      (SP)+,R2        ;;POP STACK INTO R2
      033216 012601              MOV      (SP)+,R1        ;;POP STACK INTO R1
4612 033220 004737 032050              JSR      PC,ERRTLY        ;GO TALLY THE ERROR
4613 033224 105037 013263              CLRB     ERRTP+1         ;CLEAR THE UPPER BYTE OF ERROR TYPE
4614 033230 004737 032550              JSR      PC,PRIERR        ;GO PRINT THE ERROR
4615 033234 000404              BR       45$             ;EXIT
4616 033236              POP      <R5,R3,R2,R1>   ;RESTORE THE REGISTERS
      033236 012605              MOV      (SP)+,R5        ;;POP STACK INTO R5
      033240 012603              MOV      (SP)+,R3        ;;POP STACK INTO R3
      033242 012602              MOV      (SP)+,R2        ;;POP STACK INTO R2
      033244 012601              MOV      (SP)+,R1        ;;POP STACK INTO R1
4617 033246 000207 45$:      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 033250 005737 003344 TST DUMPKT ;ARE WE ISSUEING NULL COMMANDS
4634 033254 001444 BEQ 15$ ;YES , THEN EXIT
4635 033256 023764 003542 000006 CMP SECRNS,CMDSEQ(R4) ;IS IT THE ONLY COMMAND OUT ?
4636 033264 001440 BEQ 15$ ;YES , THEN EXIT
4637 033266 016437 000006 003436 MOV CMDSEQ(R4),SAVDIF ;SET UP TO UNJAM THE QUEUES
4638 033274 163737 003542 003436 SUB SECRNS,SAVDIF ;SUBTRACT CURRENT FROM THE HIGHEST
4639 033302 063737 003436 003400 ADD SAVDIF,CMDCNT ;ADJUST THE COMMAND COUNT
4640 033310 063737 003436 003402 ADD SAVDIF,RSPCNT ;ADJUST THE RESPONSE COUNT
4641 033316 042737 000200 003516 BIC #CMDONE,PCFLAG ;CLEAR THE ALL COMMANDS ISSUED FLAG
4642 033324 163764 003436 000034 SUB SAVDIF,OBJFDL(R4) ;ADJUST THE OBJECT COUNT
4643 033332 103002 BCC 5$ ;GET OUT IF NO CARRY
4644 033334 005364 000036 DEC OBJFDH(R4) ;OTHERWISE, ADJUST THE HIGH WORD
4645 033340 022737 000004 003424 5$: CMP #N,SUBCNT
4646 033346 001002 BNE 10$
4647 033350 005037 003424 CLR SUBCNT
4648 033354 005237 003424 10$: INC SUBCNT
4649 033360 005337 003436 DEC SAVDIF
4650 033364 001365 BNE 5$
4651 033366 052764 000002 000026 15$: BIS #SEREXC,LUNFLG(R4) ;SET THE SERIOUS EXCEPTION FLAG
4652 033374 000207 RTS PC ;RETURN
4653 033376 ENDMOD
4654
```

```
4656 .SBTTL OCTAL -> HEX CONVERTER
4657 ;*****
4658 ;
4659 ; OCTAL -> HEX CONVERTER
4660 ;
4661
4662 033376 OCTHEX::
4663 033376 116337 000013 003552 MOV B P.STS+1(R3),LOHEX ;GET THE UPPER STATUS BYTE
4664 033404 013737 003552 003554 MOV LOHEX,HIHEX ;GET A SECOND COPY
4665 033412 042737 177760 003552 BIC #177760,LOHEX ;CLEAR THE UNWANTED BITS
4666 033420 042737 177417 003554 BIC #177417,HIHEX ;CLEAR THE UNWANTED BITS
4667 033426 006037 003554 ROR HIHEX ;SHIFT THE
4668 033432 006037 003554 ROR HIHEX ; UPPER NIBBLE
4669 033436 006037 003554 ROR HIHEX ; OVER 3 PLACES
4670 033442 006137 003552 ROL LOHEX ;SHIFT THE LOWER NIBBLE OVER 1
4671 033446 062737 011442 003552 ADD #HEXTBL,LOHEX ;ADD ASCII TABLE BASE ADDRESS
4672 033454 062737 011442 003554 ADD #HEXTBL,HIHEX ;ADD ASCII TABLE BASE ADDRESS
4673 033462 000207 RTS PC ;RETURN
```

```

4675          .SBTTL CLEAR EOT
4676          ;*****
4677          ;
4678          ; CLEAR EOT
4679          ;
4680
4681 033464    CLREOT::
4682 033464    PUSH    <R1,R2,R4>          ;SAVE R1, R2, AND R4
          033464 010146    MOV     R1,-(SP)          ;;PUSH R1 ON STACK
          033466 010246    MOV     R2,-(SP)          ;;PUSH R2 ON STACK
          033470 010446    MOV     R4,-(SP)          ;;PUSH R4 ON STACK
4683 033472 005002    CLR     R2          ;CLEAR OUT R2
4684 033474 012701 177776    MOV     @-2.,R1          ;SET R1 TO THE FIRST UNIT
4685 033500 012704 002314    MOV     @LUN0,R4        ;LET R4 EQUAL THE FIRDT LUN
4686 033504 062701 000002    5$:   ADD     @2.,R1          ;ADD 2 TO THE UNIT POINTER
4687 033510 062702 000001    ADD     @1.,R2          ;ADD 1 TO R2
4688 033514 112764 177777 000154    MOVB   @-1,OLDTRK(R4)    ;INITIALIZE OLD TRACK
4689 033522 012764 177777 000152    MOV     @-1,OLDBLK(R4)  ;INITIALIZE OLD BLOCK
4690 033530 062704 000172    ADD     @LUNSTP,R4      ;SET R4 TO THE NEXT LUN
4691 033534 042761 000004 003350    BIC    @EOT,DRINUS(R1)  ;CLEAR THE EOT BIT IN DRINUS
4692 033542 005037 003526    CLR    UEOT            ;CLEAR THE EOT FLAG
4693 033546 023702 002012    10$:  CMP     L$UNIT,R2      ;HAVE WE DONE THEM ALL
4694 033552 001354    BNE    5$              ;NO, KEEP GOING TILL ALL DONE
4695 033554    POP     <R4,R2,R1>      ;RESTORE R4, R2, AND R1
          033554 012604    MOV     (SP)+,R4        ;;POP STACK INTO R4
          033556 012602    MOV     (SP)+,R2        ;;POP STACK INTO R2
          033560 012601    MOV     (SP)+,R1        ;;POP STACK INTO R1
4696 033562 000207    RTS     PC              ;RETURN

```

```

4698 .SBTTL SEED SETUP AND SAVE
4699 ;*****
4700 ;
4701 ; SEED SETUP
4702 ;
4703
4704 033564 SDSTUP::
4705 033564          PUSH   <R1,R4>          ;
          033564 010146      MOV    R1,-(SP)      ;;PUSH R1 ON STACK
          033566 010446      MOV    R4,-(SP)      ;;PUSH R4 ON STACK
4706 033570 013701 002012      MOV    L$UNIT,R1          ;
4707 033574 012704 002314      MOV    @LUNO,R4          ;
4708 033600 016464 000136 000144 5$:  MOV    SED1(R4),SEED1(R4)      ;
4709 033606 016464 000140 000146      MOV    SED2(R4),SEED2(R4)      ;
4710 033614 016464 000142 000150      MOV    SED3(R4),SEED3(R4)      ;
4711 033622 062704 000172      ADD    @LUNSTP,R4          ;
4712 033626 005301          DEC    R1          ;
4713 033630 001363          BNE   S$          ;
4714 033632          POP    <R4,R1>          ;
          033632 012604      MOV    (SP)+,R4      ;;POP STACK INTO R4
          033634 012601      MOV    (SP)+,R1      ;;POP STACK INTO R1
4715 033636 000207          RTS    PC          ;
4716
4717 ;*****
4718 ;
4719 ; SEED SAVE
4720 ;
4721
4722 033640 SDSAVE::
4723 033640          PUSH   <R1,R4>          ;
          033640 010146      MOV    R1,-(SP)      ;;PUSH R1 ON STACK
          033642 010446      MOV    R4,-(SP)      ;;PUSH R4 ON STACK
4724 033644 013701 002012      MOV    L$UNIT,R1          ;
4725 033650 012704 002314      MOV    @LUNO,R4          ;
4726 033654 016464 000144 000136 5$:  MOV    SEED1(R4),SED1(R4)      ;
4727 033662 016464 000146 000140      MOV    SEED2(R4),SED2(R4)      ;
4728 033670 016464 000150 000142      MOV    SEED3(R4),SED3(R4)      ;
4729 033676 062704 000172      ADD    @LUNSTP,R4          ;
4730 033702 005301          DEC    R1          ;
4731 033704 001363          BNE   S$          ;
4732 033706          POP    <R4,R1>          ;
          033706 012604      MOV    (SP)+,R4      ;;POP STACK INTO R4
          033710 012601      MOV    (SP)+,R1      ;;POP STACK INTO R1
4733 033712 000207          RTS    PC          ;
4734

```

4736
4737
4738
4739
4740
4741
4742
4743
4744 033714
4745 033714 012704 002314
4746 033720 005064 000024
4747 033724 022704 003072
4748 033730 001403
4749 033732 062704 000172
4750 033736 000770
4751 033740 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      .SBTTL  CORE DUMP
4757      :*****
4758      :
4759      : CORE DUMP
4760      :
4761      :THIS ROUTINE IS DESIGNED TO DUMP ALL CRITICAL MEMORY LOCATIONS ON
4762      :OCCURRENCE OF ERRORS, WHEN ENEABLED BY THE OPERATOR VIA THE SOFTWARE
4763      :QUESTIONS. IT IS INTENDED PRIMARILY AS AN AID TO DEBUGGING THE
4764      :PROGRAM, BUT MAY PROVE USEFUL IN ANALYZING CERTAIN DEVICE ERRORS
4765      :AS WELL.
4766
4767 033742  CORDMP:
4768 033742 010146      PUSH   <R1,R2>
          033742 010246      MOV    R1,-(SP)          ;;PUSH R1 ON STACK
          033744 010246      MOV    R2,-(SP)          ;;PUSH R2 ON STACK
4769 033746      PRINTF  #DUMP,R1,R2,R3,R4,R5
          033746 010546      MOV    R5,-(SP)
          033750 010446      MOV    R4,-(SP)
          033752 010346      MOV    R3,-(SP)
          033754 010246      MOV    R2,-(SP)
          033756 010146      MOV    R1,-(SP)
          033760 012746 020347      MOV    #DUMP,-(SP)
          033764 012746 000006      MOV    #6,-(SP)
          033770 010600      MOV    SP,R0
          033772 104417      TRAP  C#PNTF
          033774 062706 000016      ADD    #16,SP
4770 034000 012701 002314      MOV    #LUN0,R1          ;PUT STARING ADDRESS IN R1
4771 034004 012702 002314      MOV    #LUN0,R2          ;AND ANOTHER COPY IN R2
4772 034010      1$: PRINTF  #DUMP2,R1,(R2),2(R2),4(R2),6(R2)
          034010 016246 000006      MOV    6(R2),-(SP)
          034014 016246 000004      MOV    4(R2),-(SP)
          034020 016246 000002      MOV    2(R2),-(SP)
          034024 011246      MOV    (R2),-(SP)
          034026 010146      MOV    R1,-(SP)
          034030 012746 020471      MOV    #DUMP2,-(SP)
          034034 012746 000006      MOV    #6,-(SP)
          034040 010600      MOV    SP,R0
          034042 104417      TRAP  C#PNTF
          034044 062706 000016      ADD    #16,SP
4773
4774 034050 062701 000010      ADD    #10,R1          ;UPDATE R1
4775 034054 062702 000010      ADD    #10,R2          ;UPDATE R2
4776 034060 022701 010572      CMP    #IOERTB,R1      ;ARE WE AT THE END OF DUMP AREA
4777 034064 101351      BHI    1$              ;KEEP GOING IF NOT
4778 034066      POP    <R2,R1>
4779 034072 000207      RTS    PC
4780

```

```

4782          .SBTTL BUFFER DUMP
4783          ;*****
4784          ;
4785          ; BUFFER DUMP
4786          ;
4787          ;THIS ROUTINE WILL PRINT THE READ BUFFER FOR EVERY RECGRD READ IN
4788          ;TEST 5.
4789
4790 034074     BUFDMF:
4791 034074
4792 034100     016301 000014
4793 034104     012702 050560
4794 034110
           034110     010146
           034112     012746 020430
           034116     012746 000002
           034122     010600
           034124     104417
           034126     062706 000006
4795
4796 034132     1$: PRINTF #DUMP2,(R2),2(R2),4(R2),6(R2),10(R2)
           034132     016246 000010
           034136     016246 000006
           034142     016246 000004
           034146     016246 000002
           034152     011246
           034154     012746 020471
           034160     012746 000006
           034164     010600
           034166     104417
           034170     062706 000016
4797 034174     062702 000012
4798 034200     162701 000012
4799 034204     001401
4800 034206     100351
4801
4802 034210     5$: POP <R2,R1>
4803 034214     000207     RTS PC
4804

```



```

4816 .TITLE MISCELLANEOUS SECTIONS
4817 .SBTTL REPORT CODING SECTION
4845
4846 034216 BGNMOD
4847
4848
4849 ;++
4850 ; THE REPORT CODING SECTION CONTAINS THE
4851 ; 'PRINTS' CALLS THAT GENERATE STATISTICAL REPORTS.
4852 ;--
4853 034216 BGNRPT
4854 034216 L$RPT::
4860 034216 PUSH <R1,R4,R5>
      034216 010146 MOV R1,-(SP) ;;PUSH R1 ON STACK
      034220 010446 MOV R4,-(SP) ;;PUSH R4 ON STACK
      034222 010546 MOV R5,-(SP) ;;PUSH R5 ON STACK
4861 034224 032737 000400 003516 BIT #DROPT,PCFLAG ;ARE WE DROPPING A UNIT
4862 034232 001015 BNE 5$ ;YES, ONLY PRINT STATS FOR THIS UNIT
4863 034234 005001 CLR R1 ;SET R1 TO FIRST UNIT
4864 034236 005037 002074 CLR L$LUN ;START WITH UNIT 0
4865 034242 012704 002314 MOV #LUNO,R4 ;START WITH LUN BLOCK FOR UNIT 0
4866 034246 013705 002012 MOV L$UNIT,R5 ;INIT UNIT COUNTER
4867 034252 032761 000010 003350 1$: BIT #DROP,DRINUS(R1) ;HAS THE DRIVE BEEN DROPPED ?
4868 034260 001402 BEQ 5$ ;NO, PRINT ITS STATS
4869 034262 000137 035354 JMP 15$ ;OTHERWISE GET THE NEXT DRIVE
4870
4871 034266 5$: PRINTS #STAT0
      034266 012746 035472 MOV #STAT0,-(SP)
      034272 012746 000001 MOV #1,-(SP)
      034276 010600 MOV SP,R0
      034300 104416 TRAP C$PNTS
      034302 062706 000004 ADD #4,SP
4872
4873 034306 PRINTS #STAT1,L$LUN
      034306 013746 002074 MOV L$LUN,-(SP)
      034312 012746 035475 MOV #STAT1,-(SP)
      034316 012746 000002 MOV #2,-(SP)
      034322 010600 MOV SP,R0
      034324 104416 TRAP C$PNTS
      034326 062706 000006 ADD #6,SP
4874
4875 034332 PRINTS #STAT2
      034332 012746 035541 MOV #STAT2,-(SP)
      034336 012746 000001 MOV #1,-(SP)
      034342 010600 MOV SP,R0
      034344 104416 TRAP C$PNTS
      034346 062706 000004 ADD #4,SP
4876
4877 034352 PRINTS #STAT3
      034352 012746 035606 MOV #STAT3,-(SP)
      034356 012746 000001 MOV #1,-(SP)
      034362 010600 MOV SP,R0
      034364 104416 TRAP C$PNTS
      034366 062706 000004 ADD #4,SP
4878
4879 034372 PRINTS #STAT4

```

	034372	012746	035666	MOV	#STAT4,-(SP)
	034376	012746	000001	MOV	#1,-(SP)
	034402	010600		MOV	SP,R0
	034404	104416		TRAP	C#PNTS
	034406	062706	000004	ADD	#4,SP
4880					
4881	034412			PRINTS	#STAT5,S1READ(R4),S2READ(R4),S1WRIT(R4),S2WRIT(R4)
	034412	016446	000046	MOV	S2WRIT(R4),-(SP)
	034416	016446	000044	MOV	S1WRIT(R4),-(SP)
	034422	016446	000042	MOV	S2READ(R4),-(SP)
	034426	016446	000040	MOV	S1READ(R4),-(SP)
	034432	012746	035713	MOV	#STAT5,-(SP)
	034436	012746	000005	MOV	#5,-(SP)
	034442	010600		MOV	SP,R0
	034444	104416		TRAP	C#PNTS
	034446	062706	000014	ADD	#14,SP
4882					
4883	034452			PRINTS	#STAT6,EC1COR(R4),EC2COR(R4)
	034452	016446	000052	MOV	EC2COR(R4),-(SP)
	034456	016446	000050	MOV	EC1COR(R4),-(SP)
	034462	012746	035760	MOV	#STAT6,-(SP)
	034466	012746	000003	MOV	#3,-(SP)
	034472	010600		MOV	SP,R0
	034474	104416		TRAP	C#PNTS
	034476	062706	000010	ADD	#10,SP
4884					
4885	034502			PRINTS	#STAT7,H1READ(R4),H2READ(R4),H1WRIT(R4),H2WRIT(R4)
	034502	016446	000062	MOV	H2WRIT(R4),-(SP)
	034506	016446	000060	MOV	H1WRIT(R4),-(SP)
	034512	016446	000056	MOV	H2READ(R4),-(SP)
	034516	016446	000054	MOV	H1READ(R4),-(SP)
	034522	012746	036032	MOV	#STAT7,-(SP)
	034526	012746	000005	MOV	#5,-(SP)
	034532	010600		MOV	SP,R0
	034534	104416		TRAP	C#PNTS
	034536	062706	000014	ADD	#14,SP
4886					
4887	034542			PRINTS	#STAT8,RTYEC1(R4),RTYEC2(R4)
	034542	016446	000066	MOV	RTYEC2(R4),-(SP)
	034546	016446	000064	MOV	RTYEC1(R4),-(SP)
	034552	012746	036077	MOV	#STAT8,(SP)
	034556	012746	000003	MOV	#3,-(SP)
	034562	010600		MOV	SP,R0
	034564	104416		TRAP	C#PNTS
	034566	062706	000010	ADD	#10,SP
4888					
4889	034572			PRINTS	#STAT9,DTCPER(R4)
	034572	016446	000070	MOV	DTCPER(R4),-(SP)
	034576	012746	036151	MOV	#STAT9,-(SP)
	034602	012746	000002	MOV	#2,-(SP)
	034606	010600		MOV	SP,R0
	034610	104416		TRAP	C#PNTS
	034612	062706	000006	ADD	#6,SP
4890					
4891	034616			PRINTS	#STAT10,UNDRUN(R4)
	034616	016446	000072	MOV	UNDRUN(R4),-(SP)
	034622	012746	036230	MOV	#STAT10,(SP)

	034626	012746	000002	MOV	#2,-(SP)
	034632	010600		MOV	SP,RO
	034634	104416		TRAP	C#PNTS
	034636	062706	000006	ADD	#6,SP
4892					
4893	034642			PRINTS	#STAT11,OVRUN(R4)
	034642	016446	000074	MOV	OVRUN(R4),-(SP)
	034646	012746	036300	MOV	#STAT11,-(SP)
	034652	012746	000002	MOV	#2,-(SP)
	034656	010600		MOV	SP,RO
	034660	104416		TRAP	C#PNTS
	034662	062706	000006	ADD	#6,SP
4894					
4895	034666			PRINTS	#STAT12,RMSPOS(R4),WMSPOS(R4)
	034666	016446	000100	MOV	WMSPOS(R4),-(SP)
	034672	016446	000076	MOV	RMSPOS(R4),-(SP)
	034676	012746	036352	MOV	#STAT12,-(SP)
	034702	012746	000003	MOV	#3,-(SP)
	034706	010600		MOV	SP,RO
	034710	104416		TRAP	C#PNTS
	034712	062706	000010	ADD	#10,SP
4896					
4897	034716			PRINTS	#STAT13,OTHER(R4)
	034716	016446	000102	MOV	OTHER(R4),-(SP)
	034722	012746	036413	MOV	#STAT13,-(SP)
	034726	012746	000002	MOV	#2,-(SP)
	034732	010600		MOV	SP,RO
	034734	104416		TRAP	C#PNTS
	034736	062706	000006	ADD	#6,SP
4898					
4899	034742			PRINTS	#STAT14,DROPPD(R4)
	034742	016446	000104	MOV	DROPPD(R4),-(SP)
	034746	012746	036436	MOV	#STAT14,-(SP)
	034752	012746	000002	MOV	#2,-(SP)
	034756	010600		MOV	SP,RO
	034760	104416		TRAP	C#PNTS
	034762	062706	000006	ADD	#6,SP
4900					
4901	034766			PRINTS	#STAT15,WRBYT4(R4),WRBYT3(R4),WRBYT2(R4),WRBYT1(R4)
	034766	016446	000120	MOV	WRBYT1(R4),-(SP)
	034772	016446	000122	MOV	WRBYT2(R4),-(SP)
	034776	016446	000124	MOV	WRBYT3(R4),-(SP)
	035002	016446	000126	MOV	WRBYT4(R4),-(SP)
	035006	012746	036461	MOV	#STAT15,-(SP)
	035012	012746	000005	MOV	#5,-(SP)
	035016	010600		MOV	SP,RO
	035020	104416		TRAP	C#PNTS
	035022	062706	000014	ADD	#14,SP
4902					
4903	035026			PRINTS	#STAT16,RDBYT4(R4),RDBYT3(R4),RDBYT2(R4),RDBYT1(R4)
	035026	016446	000110	MOV	RDBYT1(R4),-(SP)
	035032	016446	000112	MOV	RDBYT2(R4),-(SP)
	035036	016446	000114	MOV	RDBYT3(R4),-(SP)
	035042	016446	000116	MOV	RDBYT4(R4),-(SP)
	035046	012746	036533	MOV	#STAT16,-(SP)
	035052	012746	000005	MOV	#5,-(SP)
	035056	010600		MOV	SP,RO

	035060	104416			TRAP	C#PNTS	
	035062	062706	000014		ADD	#14,SP	
4904							
4905	035066				PRINTS	#STATO	
	035066	012746	035472		MOV	#STATO,-(SP)	
	035072	012746	000001		MOV	#1,-(SP)	
	035076	010600			MOV	SP,R0	
	035100	104416			TRAP	C#PNTS	
	035102	062706	000004		ADD	#4,SP	
4906							
4907	035106	032737	000100	003516	BIT	#NOMDTB,PCFLAG	;IS THE TABLE TO BE PRINTED
4908	035114	001117			BNE	15#	;NO, GET OUT
4909	035116				PUSH	<R1,R2,R3>	
4910	035124	016401	000130		MOV	TBLTOP(R4),R1	;POINT R1 TO THE TOP OF THE TABLE
4911	035130	016402	000134		MOV	LSTENT(R4),R2	;POINT R2 TO THE LAST ENTRY IN TABLE
4912							
4913	035134				PRINTS	#STAT17,L#LUN	
	035134	013746	002074		MOV	L#LUN,-(SP)	
	035140	012746	036605		MOV	#STAT17,-(SP)	
	035144	012746	000002		MOV	#2,-(SP)	
	035150	010600			MOV	SP,R0	
	035152	104416			TRAP	C#PNTS	
	035154	062706	000006		ADD	#6,SP	
4914							
4915	035160	06103	000000		MOV	TRK(R1),R3	;MOV THE TRACK # TO TEMP STORAGE
4916	035164	020102		10#:	CMP	R1,R2	;HAVE WE PRINTED ALL ENTRIES ?
4917	035166	001451			BEQ	12#	;YES, GET OUT
4918	035170	026103	000000		CMP	TRK(R1),R3	;ARE WE STILL ON THE SAME TRACK ?
4919	035174	001412			BEQ	11#	;YES, KEEP GOING
4920	035176	016103	000000		MOV	TRK(R1),R3	;MOV THE TRACK # TO TEMP STORAGE
4921							
4922	035202				PRINTS	#STATO	
	035202	012746	035472		MOV	#STATO,-(SP)	
	035206	012746	000001		MOV	#1,-(SP)	
	035212	010600			MOV	SP,R0	
	035214	104416			TRAP	C#PNTS	
	035216	062706	000004		ADD	#4,SP	
4923							
4924	035222			11#:	PRINTS	#STAT18,<B,TRK(R1)>,PHB(R1),<B,HWR(R1)>,<B,HRD(R1)>,<B,SWR(R1)>,<B,SRD(R1)>	
	035222	005046			CLR	-(SP)	
	035224	156116	000005		BISB	SRD(R1),(SP)	
	035230	005046			CLR	-(SP)	
	035232	156116	000004		BISB	SWR(R1),(SP)	
	035236	005046			CLR	-(SP)	
	035240	156116	000007		BISB	HRD(R1),(SP)	
	035244	005046			CLR	-(SP)	
	035246	156116	000006		BISB	HWR(R1),(SP)	
	035252	016146	000002		MOV	PHB(R1),-(SP)	
	035256	005046			CLR	-(SP)	
	035260	156116	000000		BISB	TRK(R1),(SP)	
	035264	012746	036655		MOV	#STAT18,-(SP)	
	035270	012746	000007		MOV	#7,-(SP)	
	035274	010600			MOV	SP,R0	
	035276	104416			TRAP	C#PNTS	
	035300	062706	000020		ADD	#20,SP	
4925							
4926	035304	062701	000010		ADD	#8.,R1	;STEP R1 TO THE NEXT TABLE LOCATION

```

4927 035310 000725          BR      10$          ;NO, KEEP PRINTING
4928 035312          POP      <R3,R2,R1>
4929 035320 032764 000040 000026 12$: BIT      #MTBLOV,LUNFLG(R4) ;IS THE TABLE FULL ?
4930 035326 001412          BEQ      15$          ;NO, GET NEXT DRIVE
4931 035330          PRINTF   #TBLFUL,L$LUN ;PRINT TABLE FULL MESSAGE
      035330 013746 002074          MOV      L$LUN,-(SP)
      035334 012746 036724          MOV      #TBLFUL,-(SP)
      035340 012746 000002          MOV      #2,-(SP)
      035344 010600          MOV      SP,R0
      035346 104417          TRAP    C$PNTF
      035350 062706 000006          ADD      #6,SP

4932
4933 035354 032737 000400 003516 15$: BIT      #DROPI,PCFLAG ;ARE WE DROPPING A UNIT
4934 035362 001036          BNE      25$          ;YES, ONLY PRINT STATS FOR THIS UNIT
4935 035364 062704 000172          ADD      #LUNSTP,R4 ;R4 POINTS TO NEXT LUN BLOCK
4936 035370 062701 000002          ADD      #2,R1 ;POINT R1 TO THE NEXT UNIT
4937 035374 005237 002074          INC      L$LUN ;POINTS TO NEXT UNIT NUMBER
4938 035400 005305          DEC      R5 ;ANY UNITS LEFT TO REPORT?
4939 035402 001402          BEQ      20$          ;BRANCH IF NOT
4940 035404 000137 034252          JMP      1$          ;ELSE, DO IT AGAIN
4941 035410 105737 002212          TSTB    CLOCK ;IS THE CLOCK ENABLED
4942 035414 001421          BEQ      25$          ;NO, THEN CAN'T PRINT TIME
4943 035416          PRINTF   #TIME,<B,HOURS>,<B,MINUTE>,<B,SECOND>,
      035416 005046          CLR      -(SP)
      035420 153716 002215          BISB    SECOND,(SP)
      035424 005046          CLR      -(SP)
      035426 153716 002214          BISB    MINUTE,(SP)
      035432 005046          CLR      -(SP)
      035434 153716 002213          BISB    HOURS,(SP)
      035440 012746 020044          MOV      #TIME,-(SP)
      035444 012746 000004          MOV      #4,-(SP)
      035450 010600          MOV      SP,R0
      035452 104417          TRAP    C$PNTF
      035454 062706 000012          ADD      #12,SP

4944 035460          POP      <R5,R4,R1> ;RESTORE REGS
4945 035466          EXIT    RPT
      035466 000167          .WORD   J$JMP
      035470 001316          .WORD   L10007 2 .

4946
4958

```

```

4960          ;FORMAT STATEMENTS FOR PRINT CALLS
4961
4962 035472    045    116    000  STAT0:  .ASCIZ  ?N?
4963 035475    045    116    045  STAT1:  .ASCIZ  ?N#ASTATISTICAL REPORT FOR UNIT #D1?
4964 035541    045    116    045  STAT2:  .ASCIZ  ?N#S8#S8#S8#S8#S2#AREAD#S4#S8#AWRITE?
4965 035606    045    116    045  STAT3:  .ASCIZ  ?N#S8#S8#S8#S6#ACH 1#S4#ACH 2#S4#ACH 1#S4#ACH 2?
4966 035666    045    116    045  STAT4:  .ASCIZ  ?N#ASOFT DATA ERRORS?
4967 035713    045    116    045  STAT5:  .ASCIZ  ?N#A  RETRY RECOVERED#S8#D8#D8#D8#D8?
4968 035760    045    116    045  STAT6:  .ASCIZ  ?N#A  ECC CORRECTED  #S8#D8#D8#S8#A  N.A.?
4969 036032    045    116    045  STAT7:  .ASCIZ  ?N#AHARD DATA ERRORS #S8#D8#D8#D8#D8?
4970 036077    045    116    045  STAT8:  .ASCIZ  ?N#ACRC ON ECC BLOCKS#S8#D8#D8#S8#A  N.A.?
4971 036151    045    116    045  STAT9:  .ASCIZ  ?N#ADATA COMPARE ERRORS  #S8#D8#S8#A  N.A.?
4972 036230    045    116    045  STAT10: .ASCIZ  ?N#ADATA UNDERRUNS#S7#S7#S7#AN.A.#S6#D8?
4973 036300    045    116    045  STAT11: .ASCIZ  ?N#ADATA OVERRUNS#S8#S8#D8#S8#A  N.A.?
4974 036352    045    116    045  STAT12: .ASCIZ  ?N#AMISPOSITIONS #S8#S8#D8#S8#D8?
4975 036413    045    116    045  STAT13: .ASCIZ  ?N#AOTHERS  #S8#D8?
4976 036436    045    116    045  STAT14: .ASCIZ  ?N#ADROPS  #S8#D8?
4977 036461    045    116    045  STAT15: .ASCIZ  ?N#ABYTES WRITTEN  #D3#A,#Z3#A,#Z3#A,#Z3?
4978 036533    045    116    045  STAT16: .ASCIZ  ?N#ABYTES READ   #D3#A,#Z3#A,#Z3#A,#Z3?
4979 036605    045    116    045  STAT17: .ASCIZ  ?N#A TRK PHY BLK HWR HRD SWR SRD#N?
4980 036655    045    116    045  STAT18: .ASCIZ  ?N#S2#D2#S3#D5#S3#D3#S2#D3#S2#D3#S2#D3?
4981 036724    045    116    045  TBLFUL: .ASCIZ  ?N#AUNIT #D1#A MEDIA DATA TABLE HAS OVERFLOWED#N#N?
4982          .EVEN
4983
4984
4985 037010          ENDRPT
      037010          L10007:
      037010 104425    TRAP    C#RPT
  
```

```

4987          .SBTTL  INITIALIZE SECTION
4988
4989          ;++
4990          ; THE INITIALIZE SECTION CONTAINS THE CODING THAT IS PERFORMED
4991          ; AT THE BEGINNING OF EACH PASS.
4992          ;--
4993
4994 037012          BGNINIT
4995 037012          L$INIT::
4996
4997 037012          STINIT::
4998 037012          READEF  #EF.START
4999 037012          MOV      #EF.START,R0
5000 037012          TRAP   C$REFG
5001 037016          BCMPLETE  START
5002 037020          BCS     START
5003
5004 037022          READEF  #EF.RESTART
5005 037022          MOV      #EF.RESTART,R0
5006 037026          TRAP   C$REFG
5007 037030          BCMPLETE  START
5008 037030          BCS     START
5009
5010 037032          READEF  #EF.NEW
5011 037032          MOV      #EF.NEW,R0
5012 037036          TRAP   C$REFG
5013 037040          BCMPLETE  NUPASS
5014 037040          BCS     NUPASS
5015
5016 037042          READEF  #EF.CONTINUE
5017 037042          MOV      #EF.CONTINUE,R0
5018 037046          TRAP   C$REFG
5019 037050          BCMPLETE  NUPASS
5020 037050          BCS     NUPASS
5021
5022 037052          112737 000001 003564  START:  MOVB   #1,DAYS          ;SET TO FIRST DAY
5023 037060          005037 003522          CLR    PASCNT          ;CLEAR THE PASS COUNTER
5024 037064          005037 003444          CLR    CMPERR         ;CLEAR THE COMPARE ERROR COUNTER
5025
5026 037070          012701 002314          MOV    #LUNO,R1       ;SET R1 TO THE FIRST LUN
5027 037074          010102          5$:    MOV    R1,R2          ;LET R2 = R1
5028 037076          062702 000156          ADD   #URSPBF,R2     ;LET R2 = THE END OF THE CLEAR AREA
5029 037102          005021          10$:   CLR    (R1)+          ;CLEAR THE LUN LOCATION
5030 037104          020201          CMP    R2,R1         ;ARE WE AT THE END OF THE CLEAR AREA
5031 037106          001375          BNE   10$           ;NO, KEEP CLEARING
5032 037110          062701 000014          ADD   #14,R1        ;SET R1 TO THE NEXT LUN BLOCK
5033 037114          022701 003264          CMP    #PCMDBF,R1   ;HAVE WE DONE THEM ALL
5034 037120          001365          BNE   5$            ;GO CLEAR THE NEXT LUN BLOCK
5035
5036 037122          005021          15$:   CLR    (R1)+          ;CLEAR THE LOCATION AND GET THE NEXT
5037 037124          022701 003400          CMP    #CMDCNT,R1  ;HAVE WE CLEARED THEM ALL ?
5038 037130          001374          BNE   15$          ;NO, KEEP GOING
5039
5040 037132          012701 110560          MOV    #MEDIAS,R1   ;PUT THE STATS TABLE ADDRESS INTO R1
5041 037136          005021          20$:   CLR    (R1)+          ;CLEAR THE LOCATION AND GET THE NEXT
5042 037140          022701 115620          CMP    #PATCH,R1  ;HAVE WE CLEARED THEM ALL ?
5043 037144          001374          BNE   20$          ;NO, KEEP GOING
    
```

```

5031
5032 037146 012704 002314      MOV    #LUN0,R4      ;SET R4 TO THE FIRST LUN
5033 037152 013702 002012      MOV    L$UNIT,R2    ;SET UP R2
5034 037156 012764 001233 000136 25$:  MOV    #RS1,SED1(R4) ;SET UP THE SEED IN THE LUN BLOCK
5035 037164 012764 007622 000140      MOV    #RS2,SED2(R4) ;SET UP THE SEED IN THE LUN BLOCK
5036 037172 012764 000000 000142      MOV    #RS3,SED3(R4) ;SET UP THE SEED IN THE LUN BLOCK
5037
5038 037200 062704 000172      30$:  ADD    #LUNSTP,R4    ;SET UP THE NEXT LUN BLOCK
5039 037204 005302      DEC    R2            ;DECREMENT R2
5040 037206 001363      BNE    25$          ;DID YOU DO THEM ALL
5041
5042 037210      NUPASS: BRESET
      037210 104433      TRAP   C$RESET
5043 037212 005737 003522      TST    PASCNT        ;IS THIS THE FIRST PASS ?
5044 037216 001403      BEQ    1$            ;YES GO CLEAR THE MEDIA TABLE
5045 037220 105737 002226      TSTB   CLRMED        ;ARE WE CLEARING THE MEDIA TABLE ?
5046 037224 001457      BEQ    25$          ;NO GET OUT
5047
5048 037226      1$:
5049 037226 013702 002012      MOV    L$UNIT,R2    ;SET UP R2
5050 037232 022702 000001      CMP    #1,R2        ;IS THERE 1 UNIT ?
5051 037236 001004      BNE    3$            ;NO, CHECK AGAIN
5052 037240 012737 005040 003544      MOV    #U1OFF,TBLOFF ;SET OFFSET FOR 1 UNIT IN TBLOFF
5053 037246 000421      BR     15$          ;GO GET STARTED
5054
5055 037250 022702 000002      3$:  CMP    #2,R2        ;IS THERE 2 UNITS ?
5056 037254 001004      BNE    5$            ;NO, CHECK AGAIN
5057 037256 012737 002420 003544      MOV    #U2OFF,TBLJFF ;SET OFFSET FOR 2 UNITS IN TBLOFF
5058 037264 000412      BR     15$          ;GO GET STARTED
5059
5060 037266 022702 000003      5$:  CMP    #3,R2        ;IS THERE 3 UNITS ?
5061 037272 001004      BNE    10$           ;NO, CHECK AGAIN
5062 037274 012737 001540 003544      MOV    #U3OFF,TBLOFF ;SET OFFSET FOR 3 UNITS IN TBLOFF
5063 037302 000403      BR     15$          ;GO GET STARTED
5064
5065 037304 012737 001210 003544 10$:  MOV    #U4OFF,TBLOFF ;SET OFFSET FOR 4 UNITS IN TBLOFF
5066
5067 037312 012704 002314      15$:  MOV    #LUN0,R4      ;SET R4 TO THE FIRST LUN
5068 037316 012737 110560 003530      MOV    #MEDIAS,R8    ;PUT THE STAT TABLE ADDRESS IN R8
5069
5070 037324 013764 003530 000130 20$:  MOV    R8,TBLTOP(R4) ;PUT THE TOP TABLE ADDR IN LUN BLOCK
5071 037332 013764 003530 000134      MOV    R8,LSTENT(R4) ;PUT THE TABLE ENTRY ADDR IN LUN BLOCK
5072 037340 063737 003544 003530      ADD    TBLOFF,R8     ;ADD THE LENGTH FOR EACH UNIT
5073 037346 013764 003530 000132      MOV    R8,TBLBTM(R4) ;PUT THE TABLE BOTTOM POINTER IN LUNBLK
5074 037354 062704 000172      ADD    #LUNSTP,R4    ;SET UP THE NEXT LUN BLOCK
5075 037360 005302      DEC    R2            ;DECREMENT R2
5076 037362 001360      BNE    20$          ;DID YOU DO THEM ALL
5077
5078 037364 005037 003516      25$:  CLR    PCFLAG        ;CLEAR THE PROGRAM CONTROL FLAG.
5079 037370 005037 003526      CLR    UEOT          ;CLEAR THE EOT FLAG
5080 037374 013737 002012 003524      MOV    L$UNIT,UDROP  ;SET UP THE DROP UNIT FLAG
5081 037402 005237 003522      INC    PASCNT        ;ADD 1 TO PASS COUNTER
5082 037406 105737 002227      TSTB   MEDTBL        ;IS THE MEDIA TABLE TO BE PRINTED
5083 037412 001003      BNE    30$          ;YES, DON'T DO ANYTHING
5084 037414 052737 000100 003516      BIS    #NOMDTB,PCFLAG ;NO, SET THE PROGRAM CONTROL FLAG BIT
5085
5086 037422 012702 003566      30$:  MOV    #CMDBF1,R2    ;PUT COMMAND BUFFER ADDRESS IN R2
    
```



```

5087 037426 005022          35$: CLR      (R2)+          ;CLEAR THE BUFFERS
5088 037430 022702 010226  CMP      #DSRNG0,R2    ;ARE WE AT THE END OF THE BUFFER ?
5089 037434 001374          BNE      35$          ;KEPP GOING TILL WE ARE
5090
5091 037436 012704 002314  MOV      #LUN0,R4      ;SET R4 TO THE FIRST LUN
5092 037442 013702 002012  MOV      L$UNIT,R2    ;SET UP R2
5093 037446 005001          CLR      R1           ;CLEAR R1
5094 037450 005003          CLR      R3           ;CLEAR R3
5095
5096 037452 032761 000020 003350 40$: BIT      #FAIL,DRINUS(R1) ;HAS THIS DRIVE FAILED ?
5097 037460 001054          BNE      60$          ;YES, GET NEXT UNIT
5098
5099 037462 032761 000010 003350 45$: BIT      #DROP,DRINUS(R1) ;DID THIS DRIVE DROP LAST TIME
5100 037470 001002          BNE      50$          ;YES, KEEP GOING
5101 037472 005064 000032          CLR      UNDROP(R4)   ;OTHERWISE CLEAR THE DROP COUNTER
5102 037476 012761 000001 003350 50$: MOV      #AVB,DRINUS(R1) ;SET UP ALL DRIVES TO AVAILABLE
5103
5104 037504 012764 003572 000014 55$: MOV      #DCMDBF,CNUSAV(R4) ;SET UP NEW COMMAND BUFFER SAVE
5105 037512 012764 003572 000016  MOV      #DCMDBF,COLSAV(R4) ;SET UP OLD COMMAND BUFFER SAVE
5106 037520 016464 000166 000012  MOV      UCDSRG(R4),CMDSSV(R4) ;SET UP COMMAND DESCRIPTOR SAVE
5107 037526 016464 000156 000020  MOV      URSPBF(R4),RNUSA(R4) ;SET UP NEW RESPONSE BUFFER SAVE
5108 037534 016464 000156 000022  MOV      URSPBF(R4),ROLSAV(R4) ;SET UP OLD RESPONSE BUFFER SAVE
5109
5110 037542 005064 000006          CLR      CMDSEQ(R4)   ;CLEAR THE COMMAND REFERENCE NUMBER
5111 037546 005064 000034          CLR      OBJFDL(R4)   ;CLEAR THE LOW OBJECT FIELD
5112 037552 005064 000036          CLR      OBJFDH(R4)   ;CLEAR THE HIGH OBJECT FIELD
5113 037556 005064 000010          CLR      SLTUSE(R4)   ;CLEAR THE SLOT IN USE FLAG
5114 037562 010300          MOV      R3,R0
5115
5116 037564          GPHARD  R0,R0
5117 037566 104442          TRAP    C$GPHRD
5118 037566 103011          BNCOMPLETE 60$
5119 037570 011064 000000          BCC    60$
5120 037574 012064 000002          MOV      (R0),TKIP(R4) ;
5121 037600 062764 000002 000002  MOV      (R0)+,TKSA(R4) ;
5122 037606 011064 000004          ADD     #2,TKSA(R4)   ;
5123 037606 011064 000004          MOV      (R0),TKUNIT(R4) ;
5124 037612 062701 000002          60$: ADD     #2,R1          ;SET R1 TO THE NEXT UNIT
5125 037616 062703 000001          ACD     #1.,R3        ;GET NEXT UNIT
5126 037622 062704 000172          ADD     #LUNSTP,R4   ;SET UP THE NEXT LUN BLOCK
5127 037626 005302          DEC     R2           ;DECREMENT R2
5128 037630 001310          BNE     40$          ;DID YOU DO THEM ALL
5129
5130 037632 042737 000002 003516  BIC     #NCLKFL,PCFLAG ;GET READY TO TEST FOR CLOCK PRESENT
5131 037640          SETVEC  #4,#NOCLK,#PRI00 ;SET VECTOR 4 IN CASE NO CLOCK
5132 037640 012746 000000          MOV     #PRI00,(SP)
5133 037644 012746 021044          MOV     #NOCLK,-(SP)
5134 037650 012746 000004          MOV     #4,-(SP)
5135 037654 012746 000003          MOV     #3,-(SP)
5136 037660 104437          TRAP    C$SVEC
5137 037662 062706 000010          ADD     #10,SP
5138 037666 005737 177546          TST     KWCSR        ;IS THE CLOCK THERE ?
5139 037672 000240          NOP
5140 037674 000240          NOP
5141

```

```

5136 037676          CLRVEC  #4          ;RETURN VECTOR TO TRAP CATCHER
      037676 012700 000004  MOV      #4,R0
      037702 104436  TRAP     C%CVEC
5137 037704 032737 000002 003516  BIT      #NCLKFL,PCFLAG ;WAS A CLOCK PRESENT ?
5138 037712 001016          BNE     ISTART      ;NO CLOCK, START REGULAR INIT
5139 037714          SETVEC  #100,#KWHDL,#PRI00 ;SET UP THE CLOCK VECTOR
      037714 012746 000000  MOV      #PRI00,-(SP)
      037720 012746 021100  MOV      #KWHDL,-(SP)
      037724 012746 000100  MOV      #100,-(SP)
      037730 012746 000003  MOV      #3,-(SP)
      037734 104437  TRAP     C%SVEC
      037736 062706 000010  ADD      #10,SP
5140 037742 012737 000100 177546  MOV      #100,KWCSR ;ENABLE THE CLOCK INTERRUPTS
5141
5142 037750 005001          ISTART: CLR     R1      ;SET R1 TO FIST UNIT
5143 037752 005037 002074  CLR     L$LUN      ;SET L$LUN TO FIRST UNIT
5144 037756 012704 002314  MOV      #LUN0,R4   ;SET R4 TO THE FIRST LUN BLOCK
5145
5146 037762 032761 000001 003350 1$:  BIT      #AVB,DRINUS(R1) ;SEE IF DRIVE IS PRESENT AND AVAILABLE
5147 037770 001547          BEQ     20$         ;GET THE NEXT DRIVE IF IT ISN'T
5148 037772 032761 000004 003350  BIT      #EOT,DRINUS(R1) ;CHECK IF THE DRIVE IS AT EOT
5149 040000 001143          BNE     20$         ;GET NEXT DRIVE IF IT IS
5150
5151 040002 012764 000377 000010  MOV      #377,SLTUSE(R4) ;SET ALL RESPONSE SLOTS TO THE PORT
5152 040010 004737 026230  JSR     PC,PRTCLR   ;GO DO IT
5153 040014 112737 000004 010571  MOVB    #4,CRDLIM  ;CREDITS START AT 4 FOR NEW LUN
5154
5155 040022 012705 040344          MOV      #INITIT,R5 ;PUT INIT TEST TABLE ADDRESS IN R5
5156 040026 004737 021406  JSR     PC,CMMDSQ   ;GO DO INIT ON THIS DRIVE
5157 040032 032761 000001 003350  BIT      #AVB,DRINUS(R1) ;SEE IF DRIVE IS PRESENT AND AVAILABLE
5158 040040 001523          BEQ     20$         ;GET THE NEXT DRIVE IF IT ISN'T
5159
5160 040042 012764 000377 000010  MOV      #377,SLTUSE(R4) ;SET ALL RESPONSE SLOTS TO THE PORT
5161 040050 004737 026230  JSR     PC,PRTCLR   ;GO DO IT
5162 040054 112737 000004 010571  MOVB    #4,CRDLIM  ;CREDITS START AT 4 FOR NEW LUN
5163 040062 062705 000006          ADD      #TSTSTP,R5 ;POINT R5 TO THE SCC COMMAND
5164 040066 004737 021406  JSR     PC,CMMDSQ   ;GO DO SCC ON THIS DRIVE
5165 040072 032761 000001 003350  BIT      #AVB,DRINUS(R1) ;SEE IF DRIVE IS PRESENT AND AVAILABLE
5166 040100 001503          BEQ     20$         ;GET THE NEXT DRIVE IF IT ISN'T
5167
5168 040102 012764 000377 000010 5$:  MOV      #377,SLTUSE(R4) ;SET ALL RESPONSE SLOTS TO THE PORT
5169 040110 004737 026230  JSR     PC,PRTCLR   ;GO DO IT
5170 040114 112737 000004 010571  MOVB    #4,CRDLIM  ;CREDITS START AT 4 FOR NEW LUN
5171 040122 062705 000006          ADD      #TSTSTP,R5 ;POINT R5 TO THE ONL COMMAND
5172 040126 004737 021406  JSR     PC,CMMDSQ   ;GO DO ONLINE ON THIS DRIVE
5173
5174 040132 012764 000377 000010 10$: MOV      #377,SLTUSE(R4) ;SET ALL RESPONSE SLOTS TO THE PORT
5175 040140 004737 026230  JSR     PC,PRTCLR   ;GO DO IT
5176 040144 112737 000004 010571  MOVB    #4,CRDLIM  ;CREDITS START AT 4 FOR NEW LUN
5177 040152 062705 000006          ADD      #TSTSTP,R5 ;POINT R5 TO THE GUS COMMAND
5178 040156 012737 000040 003440  MOV      #AVLB,TSTMSK ;ALLOW UNIT AVAILABLE ERRORS
5179 040164 004737 021406  JSR     PC,CMMDSQ   ;GO DO GUS ON THIS DRIVE
5180 040170 005037 003440          CLR     TSTMSK     ;ALLOW NO ERRORS
5181 040174 032761 000001 003350  BIT      #AVB,DRINUS(R1) ;SEE IF DRIVE IS PRESENT AND AVAILABLE
5182 040202 001442          BEQ     20$         ;GET THE NEXT DRIVE IF IT ISN'T
5183
5184 040204 016403 000022          15$: MOV      ROLSAV(R4),R3 ;LET R3 = OLD RESPONSE BUFFER POINTER
    
```

```

5185 040210 162703 000104      SUB      @DRBSTP,R3          ;ADJUST R3 BACK TO THE LAST RESPONCE
5186 040214      PRINTF  @CONTRV,L$LUN,<B,P.CSVR(R3)>,<B,P.CHVR(R3)>
      040214 005046      CLR      (SP)
      040216 156316 000051      BISB    P.CHVR(R3),(SP)
      040222 005046      CLR      (SP)
      040224 156316 000050      BISB    P.CSVR(R3),(SP)
      040230 013746 002074      MOV     L$LUN,-(SP)
      040234 012746 020532      MOV     @CONTRV,-(SP)
      040240 012746 000004      MOV     @4,-(SP)
      040244 010600      MOV     SP,R0
      040246 104417      TRAP   C$PNTF
      040250 062706 000012      ADD     @12,SP
5187 040254      PRINTF  @UNITRV,<B,P.USVR(R3)>,<B,P.UHVR(R3)>
      040254 005046      CLR      -(SP)
      040256 156316 000053      BISB    P.UHVR(R3),(SP)
      040262 005046      CLR      -(SP)
      040264 156316 000052      BISB    P.USVR(R3),(SP)
      040270 012746 020662      MOV     @UNITRV,-(SP)
      040274 012746 000003      MOV     @3,-(SP)
      040300 010600      MOV     SP,R0
      040302 104417      TRAP   C$PNTF
      040304 062706 000010      ADD     @10,SP

5188
5189 040310 022701 000006      20$:   CMP     @6.,R1          ;HAVE WE DONE THEM ALL ?
5190 040314 001411      BEQ     EXTINT          ;GET OUT
5191 040316 062701 000002      ADD     @UNTSTP,R1      ;GET NEXT UNIT
5192 040322 062704 000172      ADD     @LUNSTP,R4      ;SET UP THE NEXT LUN BLOCK
5193 040326 005237 002074      INC     L$LUN          ;GET NEXT UNIT
5194 040332      BREAK
      040332 104422      TRAP   C$BRK
5195 040334 000137 037762      JMP     1$              ;GO DO THE NEXT ONE
5196
5197 040340      EXTINT: EXIT  INIT
      040340 104432      TRAP   C$EXIT
      040342 000032      .WORD  L10010-..

5198
5199      ;INIT TEST TABLE
5200 040344      170      INITIT:: .BYTE  INT      ;INITIALIZATION TABLE
5201 040345      000      .BYTE  NULPAT
5202 040346 000000      .WORD  0
5203 040350 000001      .WORD  1
5204 040352      230      .BYTE  SCC      ;SET CONTROLLER CHARACTERISTICS TABLE
5205 040353      000      .BYTE  NULPAT
5206 040354 000000      .WORD  0
5207 040356 000001      .WORD  1
5208 040360      140      .BYTE  ONL      ;ONLINE TABLE
5209 040361      000      .BYTE  NULPAT
5210 040362 000000      .WORD  0
5211 040364 000001      .WORD  1
5212 040366      220      .BYTE  GUS      ;GET UNIT STATUS TABLE
5213 040367      000      .BYTE  NULPAT
5214 040370 000000      .WORD  0
5215 040372 000001      .WORD  1
5216
5217      .EVEN
5218 040374      ENDINIT
      040374      L10010:
    
```

040374 104411

TRAP C\$INIT

5220 040376
040376
5221 040376
040376
040376 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 040400      BGNCLN
040400      L$CLEAN::
5229
5236 040400 032737 000002 003516      BIT      #NCLKFL,PCFLA^      ;WAS A CLOCK PRESENT ?
5237 040406 001005      BNE      5$      ;NO CLOCK, DO REPORT
5238 040410 005037 177546      CLR      KWCSR
5239 040414      CLRVEC  #100
040414 012700 000100      MOV      #100,R0
040420 104436      TRAP    C$CVEC
5240 040422      5$:      DORPT
040422 104424      TRAP    C$DRPT
5241 040424 032737 000400 003516      BIT      #DROPT,PCFLAG
5242 040432 001003      BNE      EXTCLN
5243 040434 042737 000100 003516      BIC      #NOMDTB,PCFLAG
5244 040442      EXTCLN: EXIT  CLN
040442 104432      TRAP    C$EXIT
040444 000002      .WORD  L10012-.
5245
5257
5258      .EVEN
5259
5260 040446      ENDCLN
040446      L10012:
040446 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 040450      DROPUN::
5270 040450      PUSH    <R1>                ;SAVE R1
                    MOV    R1,-(SP)          ;;PUSH R1 ON STACK
5271 040450 010146  MOV    R0,R1            ;POINT R1 TO THE DRINUS TABLE
5272 040454 006101  ROL    R1                ;MULTIPLY BY 2
5273 040456 005264 000104  INC    DROPPD(R4)          ;INC DROPPED FLAG FOR THIS UNIT
5274 040462 052737 000400 003516  BIS    #DROPI,PCFLAG      ;SET THE UNIT DROP FLAG
5275 040470      PUSH    <R0>                ;SAVE R0
                    MOV    R0,-(SP)          ;;PUSH R0 ON STACK
5276 040472      DORPT  C$DRPT              ;GO PRINT UNIT STATS
                    TRAP   C$DRPT
5277 040474      POP    <R0>                ;RESTORE R0
                    MOV    (SP)+,R0         ;;POP STACK INTO R0
5278 040476 042737 000400 003516  BIC    #DROPI,PCFLAG      ;CLEAR THE DROP FLAG
5279 040504 105737 002230      TSTB  NOCLR              ;DO WE WANT TO CLEAR STATS ON ERROR ?
5280 040510 001415      BEQ    3$                ;NO, DON'T CLEAR THE STATS
5281 040512      PUSH    <R2,R3>
                    MOV    R2,-(SP)          ;;PUSH R2 ON STACK
                    MOV    R3,-(SP)          ;;PUSH R3 ON STACK
5282 040516 012702 000040      MOV    #S1READ,R2          ;STARTING ADDRESS OF STATS IN R2
5283 040522 012703 000130      MOV    #TBLTOP,R3         ;END ADDRESS OF STATS IN R3
5284 040526 060402      ADD    R4,R2              ;ADD THE LUN BLOCK ADDRESS TO R2
5285 040530 060403      ADD    R4,R3              ;ADD THE LUN BLOCK ADDRESS TO R3
5286 040532 005022      CLR    (R2)+              ;CLEAR THE LOCATION
5287 040534 020203      CMP    R2,R3              ;ARE WE AT THE END OF THE STATS
5288 040536 001375      BNE    2$                ;NO, KEEP CLEARING
5289 040540      POP    <R3,R2>
                    MOV    (SP)+,R3         ;;POP STACK INTO R3
                    MOV    (SP)+,R2         ;;POP STACK INTO R2
5290 040544 042761 000001 003350 3$:  BIC    #AVB,DRINUS(R1)    ;CLEAR THE AVB BIT IN DRIVE IN USE TABLE
5291 040552 032761 000004 003350  BIT    #EOT,DRINUS(R1)    ;IS THE DRIVE AT EOT ?
5292 040560 001404      BEQ    5$                ;BRANCH IF NOT
5293 040562 042761 000004 003350  BIC    #EOT,DRINUS(R1)    ;CLEAR THE EOT BIT IN DRIVE IN USE TABLE
5294 040570 000402      BR    10$              ;GET OUT
5295 040572 005337 003524      DEC    UDROP              ;SUBTRACT 1 TO DROPPED FLAG
5296 040576 052761 000010 003350 10$: BIS    #DROP,DRINUS(R1)    ;SET DRIVE IN USE TABLE TO DROPPED
5297 040604 005264 000032      INC    UNDROP(R4)        ;ADD 1 TO THE UNIT DROP COUNT
5298 040610 022764 000012 000032  CMP    #10.,UNDROP(R4)    ;DO WE HAVE 10. ERRORS ?
5299 040616 001004      BNE    15$              ;NO, GET OUT
5300 040620 052761 000020 003350  BIS    #FAIL,DRINUS(R1)   ;SET THE DRIVE TO FAIL
5301 040626      DODU  R0
                    TRAP   C$DODU
5302 040630 005037 003406      CLR    RESPON            ;CLEAR THE RESPONSE STATUS
5303 040634      POP    <R1>                ;RESTORE R1
                    MOV    (SP)+,R1         ;;POP STACK INTO R1
5304 040636      DELAY 20.                ;DELAY FOR AWHILE
                    MOV    #20.,(PC)+
                    .WORD 0
                    MOV    L$DLY,(PC)+
                    .WORD 0

```

040652	005367	177772		DEC	-6(PC)	
040656	001375			BNE	.-4	
040660	005367	177756		DEC	-22(PC)	
040664	001367			BNE	.-20	
5305	040666	010174	000000	MOV	R1,0TKIP(R4)	;FLUSH THE DRIVE
5306	040672	012764	000377 000010	MOV	#377,SLTUSE(R4)	;SET ALL RESPONCE SLOTS TO PORT
5307	040700	004737	026230	JSR	PC,PRTCLR	;GO CLEAR THE PORT
5308	040704	000207		RTS	PC	;RETURN
5309						
5310						
5311	040706			BGNDU		
	040706			L\$DU::		
5312	040706			EXIT	DU	
	040706	000167		.WORD	J\$JMP	
	040710	000000		.WORD	L10013-2-	
5313						
5325						
5326				.EVEN		
5327						
5328	040712			ENDDU		
	040712			L10013:		
	040712	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 040714      BGNAU
5339 040714      L$AU::
5345
5346 040714      EXIT      AU
5347 040714 000167  .WORD    J$JMP
5348 040716 000000  .WORD    L10014-2-.
5349
5359
5360      .EVEN
5361
5362 040720      ENDAU
5363 040720      L10014:
5364 040720 104452  TRAP      C$AU
5365      ENDMOD
```

```

5368
5369
5380
5381
5417
5418 040722
5419
5420
5421
5422
5423
5424
5425
5426
5433
5439
5440 040722
      040722
5441 040722 105737 002234
5442 040726 001414
5443 040730
      040730 013746 002114
      040734 012746 021001
      040740 012746 000002
      040744 010600
      040746 104417
      040750 062706 000006
5444 040754 000137 042632
5445
5446 040760 005737 003524
5447 040764 001014
5448 040766
      040766 013746 002114
      040772 012746 021001
      040776 012746 000002
      041002 010600
      041004 104417
      041006 062706 000006
5449 041012 000137 042632
5450
5451 041016 105737 002212
5452 041022 001421
5453 041024
      041024 005046
      041026 153716 002215
      041032 005046
      041034 153716 002214
      041040 005046
      041042 153716 002213
      041046 012746 020044
      041052 012746 000004
      041056 010600
      041060 104417
      041062 062706 000012
5454
5455 041066 004737 033464
5456 041072 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.
; --
      BGNTST
T1::
      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$:
      TSTB    CLOCK          ;IS THE CLOCK ENABLED
      BEQ     G01           ;NO, THEN CAN'T PRINT TIME
      PRINTF  #TIME,<B,HOURS>,<B,MINUTE>,<B,SECOND>,
      CLR     -(SP)
      BISB   SECOND,(SP)
      CLR     -(SP)
      BISB   MINUTE,(SP)
      CLR     -(SP)
      BISB   HOURS,(SP)
      MOV     #TIME,-(SP)
      MOV     #4,-(SP)
      MOV     SP,R0
      TRAP   C$PNTF
      ADD     #12,SP

G01:
      JSR     PC,CLREOT     ;MAKE SURE EOT STATUS IS CLEAR
      MOV     #ONLB,TSTMSK ;ALLOW ALREADY ONLINE STATUS

```

```

5457
5458 041100 012705 042636      MOV      #T10NL,R5      ;SET UP TO DO AN ONLINE
5459 041104 004737 021240      JSR      PC,SCHED      ;GO ISSUE THE COMMAND
5460 041110 005737 003524      TST      UDROP          ;HAVE ALL UNITS BEEN DROPPED ?
5461 041114 001002                BNE      5$             ;NO, CONTINUE
5462 041116 000137 042632      JMP      T1EXIT        ;GET OUT IF NONE LEFT TO TEST
5463
5464 041122 005037 003440      5$:     CLR      TSTMSK    ;ALLOW NO ERRORS
5465 041126 012705 042644      MOV      #T1REW,R5     ;SET UP TO DO A REWIND
5466 041132 004737 021240      JSR      PC,SCHED      ;GO ISSUE THE COMMAND
5467 041136 005737 003524      TST      UDROP          ;HAVE ALL UNITS BEEN DROPPED ?
5468 041142 001002                BNE      10$           ;NO, CONTINUE
5469 041144 000137 042632      JMP      T1EXIT        ;GET OUT IF NONE LEFT TO TEST
5470
5471 041150 012705 042652      10$:    MOV      #T1LEOT,R5  ;SET UP TO DO 2 TAPE MARK COMMANDS
5472 041154 004737 021240      JSR      PC,SCHED      ;GO ISSUE THE COMMAND
5473 041160 005737 003524      TST      UDROP          ;HAVE ALL UNITS BEEN DROPPED ?
5474 041164 001002                BNE      15$           ;NO, CONTINUE
5475 041166 000137 042632      JMP      T1EXIT        ;GET OUT IF NONE LEFT TO TEST
5476
5477 041172 012705 042674      15$:    MOV      #T1SKR,R5   ;SET UP TO SKIP REVERSE 2 TAPE MARKS
5478 041176 004737 021240      JSR      PC,SCHED      ;GO ISSUE THE COMMAND
5479 041202 005737 003524      TST      UDROP          ;HAVE ALL UNITS BEEN DROPPED ?
5480 041206 001002                BNE      20$           ;NO, CONTINUE
5481 041210 000137 042632      JMP      T1EXIT        ;GET OUT IF NONE LEFT TO TEST
5482
5483 041214 052737 000001 003440 20$:    BIS      #LEDB,TSTMSK  ;SET UP TO ALLOW LEOT DETECTED
5484 041222 012705 043004      MOV      #T1SKD,R5     ;SET UP TO DO A SPACE TO LEOT
5485 041226 004737 021240      JSR      PC,SCHED      ;GO ISSUE THE COMMAND
5486 041232 042737 000001 003440      BIC      #LEDB,TSTMSK  ;DISALLOW LEOT DETECTED
5487 041240 005737 003524      TST      UDROP          ;HAVE ALL UNITS BEEN DROPPED ?
5488 041244 001002                BNE      25$           ;NO, CONTINUE
5489 041246 000137 042632      JMP      T1EXIT        ;GET OUT IF NONE LEFT TO TEST
5490
5491 041252 012705 042644      25$:    MOV      #T1REW,R5     ;SET UP TO DO A REWIND
5492 041256 004737 021240      JSR      PC,SCHED      ;GO ISSUE THE COMMAND
5493 041262 005737 003524      TST      UDROP          ;HAVE ALL UNITS BEEN DROPPED ?
5494 041266 001002                BNE      30$           ;NO, CONTINUE
5495 041270 000137 042632      JMP      T1EXIT        ;GET OUT IF NONE LEFT TO TEST
5496
5497 041274 012705 043012      30$:    MOV      #T1WR1,R5    ;WRITE 99, 512 BYTE RECORDS
5498 041300 004737 033564      JSR      PC,SDSTUP     ;RESET THE RANDOM SEEDS
5499 041304 004737 021240      JSR      PC,SCHED      ;GO ISSUE THE COMMAND
5500 041310 005737 003524      TST      UDROP          ;HAVE ALL UNITS BEEN DROPPED ?
5501 041314 001002                BNE      35$           ;NO, CONTINUE
5502 041316 000137 042632      JMP      T1EXIT        ;GET OUT IF NONE LEFT TO TEST
5503
5504 041322 012705 042660      35$:    MOV      #T1WTM,R5    ;SET UP TO WRITE A TAPE MARK
5505 041326 004737 021240      JSR      PC,SCHED      ;GO ISSUE THE COMMAND
5506 041332 005737 003524      TST      UDROP          ;HAVE ALL UNITS BEEN DROPPED ?
5507 041336 001002                BNE      40$           ;NO, CONTINUE
5508 041340 000137 042632      JMP      T1EXIT        ;GET OUT IF NONE LEFT TO TEST
5509
5510 041344 012705 043020      40$:    MOV      #T1WR2,R5    ;WRITE 84, 525 BYTE RECORDS
5511 041350 004737 033564      JSR      PC,SDSTUP     ;RESET THE RANDOM SEEDS
5512 041354 004737 021240      JSR      PC,SCHED      ;GO ISSUE THE COMMAND
5513 041360 005737 003524      TST      UDROP          ;HAVE ALL UNITS BEEN DROPPED ?

```

```

5514 041364 001002          BNE    45$      ;NO, CONTINUE
5515 041366 000137 042632  JMP    T1EXIT  ;GET OUT IF NONE LEFT TO TEST
5516
5517 041372 012705 042660  45$:  MOV    #T1WTM,R5 ;SET UP TO WRITE A TAPE MARK
5518 041376 004737 021240  JSR    PC,SCHED ;GO ISSUE THE COMMAND
5519 041402 005737 003524  TST    UDROP   ;HAVE ALL UNITS BEEN DROPPED ?
5520 041406 001002          BNE    50$      ;NO, CONTINUE
5521 041410 000137 042632  JMP    T1EXIT  ;GET C 'IT IF NONE LEFT TO TEST
5522
5523 041414 012705 043026  50$:  MOV    #T1WR3,R5 ;WRITE 69, 1038 BYTE RECORDS
5524 041420 004737 033564  JSR    PC,SDSTUP ;RESET THE RANDOM SEEDS
5525 041424 004737 021240  JSR    PC,SCHED ;GO ISSUE THE COMMAND
5526 041430 005737 003524  TST    UDROP   ;HAVE ALL UNITS BEEN DROPPED ?
5527 041434 001002          BNE    55$      ;NO, CONTINUE
5528 041436 000137 042632  JMP    T1EXIT  ;GET OUT IF NONE LEFT TO TEST
5529
5530 041442 012705 042660  55$:  MOV    #T1WTM,R5 ;SET UP TO WRITE A TAPE MARK
5531 041446 004737 021240  JSR    PC,SCHED ;GO ISSUE THE COMMAND
5532 041452 005737 003524  TST    UDROP   ;HAVE ALL UNITS BEEN DROPPED ?
5533 041456 001002          BNE    60$      ;NO, CONTINUE
5534 041460 000137 042632  JMP    T1EXIT  ;GET OUT IF NONE LEFT TO TEST
5535
5536 041464 012705 043034  60$:  MOV    #T1WR4,R5 ;WRITE 54, 1551 BYTE RECORDS
5537 041470 004737 033564  JSR    PC,SDSTUP ;RESET THE RANDOM SEEDS
5538 041474 004737 021240  JSR    PC,SCHED ;GO ISSUE THE COMMAND
5539 041500 005737 003524  TST    UDROP   ;HAVE ALL UNITS BEEN DROPPED ?
5540 041504 001002          BNE    65$      ;NO, CONTINUE
5541 041506 000137 042632  JMP    T1EXIT  ;GET OUT IF NONE LEFT TO TEST
5542
5543 041512 012705 042660  65$:  MOV    #T1WTM,R5 ;SET UP TO WRITE A TAPE MARK
5544 041516 004737 021240  JSR    PC,SCHED ;GO ISSUE THE COMMAND
5545 041522 005737 003524  TST    UDROP   ;HAVE ALL UNITS BEEN DROPPED ?
5546 041526 001002          BNE    70$      ;NO, CONTINUE
5547 041530 000137 042632  JMP    T1EXIT  ;GET OUT IF NONE LEFT TO TEST
5548
5549 041534 012705 043042  70$:  MOV    #T1WR5,R5 ;WRITE 39, 2064 BYTE RECORDS
5550 041540 004737 033564  JSR    PC,SDSTUP ;RESET THE RANDOM SEEDS
5551 041544 004737 021240  JSR    PC,SCHED ;GO ISSUE THE COMMAND
5552 041550 005737 003524  TST    UDROP   ;HAVE ALL UNITS BEEN DROPPED ?
5553 041554 001002          BNE    75$      ;NO, CONTINUE
5554 041556 000137 042632  JMP    T1EXIT  ;GET OUT IF NONE LEFT TO TEST
5555
5556 041562 012705 042660  75$:  MOV    #T1WTM,R5 ;SET UP TO WRITE A TAPE MARK
5557 041566 004737 021240  JSR    PC,SCHED ;GO ISSUE THE COMMAND
5558 041572 005737 003524  TST    UDROP   ;HAVE ALL UNITS BEEN DROPPED ?
5559 041576 001002          BNE    80$      ;NO, CONTINUE
5560 041600 000137 042632  JMP    T1EXIT  ;GET OUT IF NONE LEFT TO TEST
5561
5562 041604 012705 043050  80$:  MOV    #T1WR6,R5 ;WRITE 24, 2577 BYTE RECORDS
5563 041610 004737 033564  JSR    PC,SDSTUP ;RESET THE RANDOM SEEDS
5564 041614 004737 021240  JSR    PC,SCHED ;GO ISSUE THE COMMAND
5565 041620 005737 003524  TST    UDROP   ;HAVE ALL UNITS BEEN DROPPED ?
5566 041624 001002          BNE    85$      ;NO, CONTINUE
5567 041626 000137 042632  JMP    T1EXIT  ;GET OUT IF NONE LEFT TO TEST
5568
5569 041632 012705 042652  85$:  MOV    #T1LEOT,R5 ;SET UP TO WRITE LOGICAL END OF TAPE
5570 041636 004737 021240  JSR    PC,SCHED ;GO ISSUE THE COMMAND

```

5571	041642	005737	003524		TST	UDROP		;HAVE ALL UNITS BEEN DROPPED ?
5572	041646	001002			BNE	90\$;NO, CONTINUE
5573	041650	000137	042632		JMP	T1EXIT		;GET OUT IF NONE LEFT TO TEST
5574								
5575	041654	012705	042644	90\$:	MOV	0T1REW,R5		;SET UP TO REWIND
5576	041660	004737	021240		JSR	PC,SCHED		;GO ISSUE THE COMMAND
5577	041664	005737	003524		TST	UDROP		;HAVE ALL UNITS BEEN DROPPED ?
5578	041670	001002			BNE	95\$;NO, CONTINUE
5579	041672	000137	042632		JMP	T1EXIT		;GET OUT IF NONE LEFT TO TEST
5580								
5581	041676	012705	043056	95\$:	MOV	0T1RD1,R5		;SET UP TO READ 100 RECORDS
5582	041702	004737	033564		JSR	PC,SDSTUP		;RESET THE RANDOM SEEDS
5583	041706	004737	021240		JSR	PC,SCHED		;GO ISSUE THE COMMAND
5584	041712	005737	003524		TST	UDROP		;HAVE ALL UNITS BEEN DROPPED ?
5585	041716	001002			BNE	100\$;NO, CONTINUE
5586	041720	000137	042632		JMP	T1EXIT		;GET OUT IF NONE LEFT TO TEST
5587								
5588	041724	012705	042666	100\$:	MOV	0T1SKP,R5		;SET UP TO SKIP A TAPE MARK
5589	041730	004737	021240		JSR	PC,SCHED		;GO ISSUE THE COMMAND
5590	041734	005737	003524		TST	UDROP		;HAVE ALL UNITS BEEN DROPPED ?
5591	041740	001002			BNE	105\$;NO, CONTINUE
5592	041742	000137	042632		JMP	T1EXIT		;GET OUT IF NONE LEFT TO TEST
5593								
5594	041746	012705	042702	105\$:	MOV	0T1SPC1,R5		;SET UP TO SPACE 84 RECORDS
5595	041752	004737	021240		JSR	PC,SCHED		;GO ISSUE THE COMMAND
5596	041756	005737	003524		TST	UDROP		;HAVE ALL UNITS BEEN DROPPED ?
5597	041762	001002			BNE	110\$;NO, CONTINUE
5598	041764	000137	042632		JMP	T1EXIT		;GET OUT IF NONE LEFT TO TEST
5599								
5600	041770	012705	042666	110\$:	MOV	0T1SKP,R5		;SET UP TO SKIP A TAPE MARK
5601	041774	004737	021240		JSR	PC,SCHED		;GO ISSUE THE COMMAND
5602	042000	005737	003524		TST	UDROP		;HAVE ALL UNITS BEEN DROPPED ?
5603	042004	001002			BNE	115\$;NO, CONTINUE
5604	042006	000137	042632		JMP	T1EXIT		;GET OUT IF NONE LEFT TO TEST
5605								
5606	042012	012705	042710	115\$:	MOV	0T1SPC2,R5		;SET UP TO SPACE 69 RECORDS
5607	042016	004737	021240		JSR	PC,SCHED		;GO ISSUE THE COMMAND
5608	042022	005737	003524		TST	UDROP		;HAVE ALL UNITS BEEN DROPPED ?
5609	042026	001002			BNE	120\$;NO, CONTINUE
5610	042030	000137	042632		JMP	T1EXIT		;GET OUT IF NONE LEFT TO TEST
5611								
5612	042034	012705	042724	120\$:	MOV	0T1SP01,R5		;SET UP TO SPACE 56 OBJECTS
5613	042040	004737	021240		JSR	PC,SCHED		;GO ISSUE THE COMMAND
5614	042044	005737	003524		TST	UDROP		;HAVE ALL UNITS BEEN DROPPED ?
5615	042050	001002			BNE	125\$;NO, CONTINUE
5616	042052	000137	042632		JMP	T1EXIT		;GET OUT IF NONE LEFT TO TEST
5617								
5618	042056	012705	043106	125\$:	MOV	0T1RD5,R5		;SET UP TO READ 39 RECORDS
5619	042062	004737	033564		JSR	PC,SDSTUP		;RESET THE RANDOM SEEDS
5620	042066	004737	021240		JSR	PC,SCHED		;GO ISSUE THE COMMAND
5621	042072	005737	003524		TST	UDROP		;HAVE ALL UNITS BEEN DROPPED ?
5622	042076	001002			BNE	130\$;NO, CONTINUE
5623	042100	000137	042632		JMP	T1EXIT		;GET OUT IF NONE LEFT TO TEST
5624								
5625	042104	012705	042732	130\$:	MOV	0T1SKR1,R5		;SET UP TO SKIP REVERSE 4 TAPE MARKS
5626	042110	004737	021240		JSR	PC,SCHED		;GO ISSUE THE COMMAND
5627	042114	005737	003524		TST	UDROP		;HAVE ALL UNITS BEEN DROPPED ?

5628	042120	001002		BNE	135‡		;NO, CONTINUE
5629	042122	000137	042632	JMP	T1EXIT		;GET OUT IF NONE LEFT TO TEST
5630							
5631	042126	012705	042666	135‡:	MOV	‡T1SKP,R5	;SET UP TO SKIP TAPE MARK FORWARD
5632	042132	004737	021240		JSR	PC,SCHED	;GO ISSUE THE COMMAND
5633	042136	005737	003524		TST	UDROP	;HAVE ALL UNITS BEEN DROPPED ?
5634	042142	001002			BNE	140‡	;NO, CONTINUE
5635	042144	000137	042632		JMP	T1EXIT	;GET OUT IF NONE LEFT TO TEST
5636							
5637	042150	012705	043064	140‡:	MOV	‡T1RD2,R5	;SET UP TO READ 84 RECORDS
5638	042154	004737	033564		JSR	PC,SDSTUP	;RESET THE RANDOM SEEDS
5639	042160	004737	021240		JSR	PC,SCHED	;GO ISSUE THE COMMAND
5640	042164	005737	003524		TST	UDROP	;HAVE ALL UNITS BEEN DROPPED ?
5641	042170	001002			BNE	145‡	;NO, CONTINUE
5642	042172	000137	042632		JMP	T1EXIT	;GET OUT IF NONE LEFT TO TEST
5643							
5644	042176	012705	042740	145‡:	MOV	‡T1SP02,R5	;SET UP TO SPACE 71 OBJECTS
5645	042202	004737	021240		JSR	PC,SCHED	;GO ISSUE THE COMMAND
5646	042206	005737	003524		TST	UDROP	;HAVE ALL UNITS BEEN DROPPED ?
5647	042212	001002			BNE	150‡	;NO, CONTINUE
5648	042214	000137	042632		JMP	T1EXIT	;GET OUT IF NONE LEFT TO TEST
5649							
5650	042220	012705	043100	150‡:	MOV	‡T1RD4,R5	;SET UP TO READ 54 RECORDS
5651	042224	004737	033564		JSR	PC,SDSTUP	;RESET THE RANDOM SEEDS
5652	042230	004737	021240		JSR	PC,SCHED	;GO ISSUE THE COMMAND
5653	042234	005737	003524		TST	UDROP	;HAVE ALL UNITS BEEN DROPPED ?
5654	042240	001002			BNE	155‡	;NO, CONTINUE
5655	042242	000137	042632		JMP	T1EXIT	;GET OUT IF NONE LEFT TO TEST
5656							
5657	042246	012705	042746	155‡:	MOV	‡T1SP03,R5	;SET UP TO SPACE 66 OBJECTS
5658	042252	004737	021240		JSR	PC,SCHED	;GO ISSUE THE COMMAND
5659	042256	005737	003524		TST	UDROP	;HAVE ALL UNITS BEEN DROPPED ?
5660	042262	001002			BNE	160‡	;NO, CONTINUE
5661	042264	000137	042632		JMP	T1EXIT	;GET OUT IF NONE LEFT TO TEST
5662							
5663	042270	012705	042754	160‡:	MOV	‡T1SPR1,R5	;SET UP TO SPACE REVERSE 375 OBJECTS
5664	042274	004737	021240		JSR	PC,SCHED	;GO ISSUE THE COMMAND
5665	042300	005737	003524		TST	UDROP	;HAVE ALL UNITS BEEN DROPPED ?
5666	042304	001002			BNE	165‡	;NO, CONTINUE
5667	042306	000137	042632		JMP	T1EXIT	;GET OUT IF NONE LEFT TO TEST
5668							
5669	042312	012705	042762	165‡:	MOV	‡T1SKP1,R5	;SET UP TO SKIP FORWARD 4 TAPE MARKS
5670	042316	004737	021240		JSR	PC,SCHED	;GO ISSUE THE COMMAND
5671	042322	005737	003524		TST	UDROP	;HAVE ALL UNITS BEEN DROPPED ?
5672	042326	001002			BNE	170‡	;NO, CONTINUE
5673	042330	000137	042632		JMP	T1EXIT	;GET OUT IF NONE LEFT TO TEST
5674							
5675	042334	012705	042716	170‡:	MOV	‡T1SPC3,R5	;SET UP TO SPACE 39 RECORDS
5676	042340	004737	021240		JSR	PC,SCHED	;GO ISSUE THE COMMAND
5677	042344	005737	003524		TST	UDROP	;HAVE ALL UNITS BEEN DROPPED ?
5678	042350	001002			BNE	175‡	;NO, CONTINUE
5679	042352	000137	042632		JMP	T1EXIT	;GET OUT IF NONE LEFT TO TEST
5680							
5681	042356	012705	042666	175‡:	MOV	‡T1SKP,R5	;SET UP TO SKIP A TAPE MARK
5682	042362	004737	021240		JSR	PC,SCHED	;GO ISSUE THE COMMAND
5683	042366	005737	003524		TST	UDROP	;HAVE ALL UNITS BEEN DROPPED ?
5684	042372	001002			BNE	180‡	;NO, CONTINUE

```

5685 042374 000137 042632          JMP      T1EXIT          ;GET OUT IF NONE LEFT TO TEST
5686
5687 042400 012705 043114          180$:  MOV      #T1RD6,R5      ;SET UP TO READ 824 RECORDS
5688 042404 004737 033564          JSR      PC,SDSTUP        ;RESET THE RANDOM SEEDS
5689 042410 004737 021240          JSR      PC,SCHED        ;GO ISSUE THE COMMAND
5690 042414 005737 003524          TST      UDROP           ;HAVE ALL UNITS BEEN DROPPED ?
5691 042420 001002          BNE      185$           ;NO, CONTINUE
5692 042422 000137 042632          JMP      T1EXIT          ;GET OUT IF NONE LEFT TO TEST
5693
5694 042426 012705 042770          185$:  MOV      #T1SKP2,R5     ;SET UP TO SKIP 2 TAPE MARKS
5695 042432 004737 021240          JSR      PC,SCHED        ;GO ISSUE THE COMMAND
5696 042436 005737 003524          TST      UDROP           ;HAVE ALL UNITS BEEN DROPPED ?
5697 042442 001002          BNE      190$           ;NO, CONTINUE
5698 042444 000137 042632          JMP      T1EXIT          ;GET OUT IF NONE LEFT TO TEST
5699
5700 042450 012705 042776          190$:  MOV      #T1SPR2,R5    ;SET UP TO SPACE REVERSE 192 OBJECTS
5701 042454 004737 021240          JSR      PC,SCHED        ;GO ISSUE THE COMMAND
5702 042460 005737 003524          TST      UDROP           ;HAVE ALL UNITS BEEN DROPPED ?
5703 042464 001002          BNE      195$           ;NO, CONTINUE
5704 042466 000137 042632          JMP      T1EXIT          ;GET OUT IF NONE LEFT TO TEST
5705
5706 042472 012705 042666          195$:  MOV      #T1SKP,R5     ;SET UP TO SKIP A TAPE MARK
5707 042476 004737 021240          JSR      PC,SCHED        ;GO ISSUE THE COMMAND
5708 042502 005737 003524          TST      UDROP           ;HAVE ALL UNITS BEEN DROPPED ?
5709 042506 001002          BNE      200$           ;NO, CONTINUE
5710 042510 000137 042632          JMP      T1EXIT          ;GET OUT IF NONE LEFT TO TEST
5711
5712 042514 012705 043072          200$:  MOV      #T1RD3,R5     ;SET UP TO READ 69 RECORDS
5713 042520 004737 033564          JSR      PC,SDSTUP        ;RESET THE RANDOM SEEDS
5714 042524 004737 021240          JSR      PC,SCHED        ;GO ISSUE THE COMMAND
5715 042530 005737 003524          TST      UDROP           ;HAVE ALL UNITS BEEN DROPPED ?
5716 042534 001002          BNE      205$           ;NO, CONTINUE
5717 042536 000137 042632          JMP      T1EXIT          ;GET OUT IF NONE LEFT TO TEST
5718
5719 042542 012705 042644          205$:  MOV      #T1REW,R5     ;SET UP TO REWIND
5720 042546 004737 021240          JSR      PC,SCHED        ;GO ISSUE THE COMMAND
5721 042552 005737 003524          TST      UDROP           ;HAVE ALL UNITS BEEN DROPPED ?
5722 042556 001002          BNE      210$           ;NO, CONTINUE
5723 042560 000137 042632          JMP      T1EXIT          ;GET OUT IF NONE LEFT TO TEST
5724
5725 042564 052737 000001 003440 210$:  BIS      #LEDB,TSTMSK    ;SET UP TO ALLOW LEOT DETECTED
5726 042572 012705 043004          MOV      #T1SKD,R5     ;SET UP TO SKIP TO LEOT
5727 042576 004737 021240          JSR      PC,SCHED        ;GO ISSUE THE COMMAND
5728 042602 042737 000001 003440  BIC      #LEDB,TSTMSK    ;DISALLOW LEOT DETECTED
5729 042610 005737 003524          TST      UDROP           ;HAVE ALL UNITS BEEN DROPPED ?
5730 042614 001002          BNE      215$           ;NO, CONTINUE
5731 042616 000137 042632          JMP      T1EXIT          ;GET OUT IF NONE LEFT TO TEST
5732
5733 042622 012705 042644          215$:  MOV      #T1REW,R5     ;SET UP TO REWIND
5734 042626 004737 021240          JSR      PC,SCHED        ;GO ISSUE THE COMMAND
5735
5736 042632          T1EXIT: EXIT          TST
          042632 104432          TRAP      C$EXIT
          042634 000266          .WORD    L10015-.
    
```

5738					
5739	042636	140	T1ONL:	.BYTE ONL	;ONLINE COMMAND
5740	042637	000		.BYTE NULPAT	;NO DATA NEEDED
5741	042640	000000		.WORD 0	;NO ITEM COUNT
5742	042642	000001		.WORD 1	;DO IT ONE TIME
5743					
5744	042644	160	T1REW:	.BYTE REW	;REWIND COMMAND
5745	042645	000		.BYTE NULPAT	;NO DATA NEEDED
5746	042646	000000		.WORD 0	;NO ITEM COUNT
5747	042650	000001		.WORD 1	;DO IT ONE TIME
5748					
5749	042652	100	T1LEOT:	.BYTE WTM	;WRITE TAPE MARK
5750	042653	000		.BYTE NULPAT	;NO DATA NEEDED
5751	042654	000000		.WORD 0	;NO ITEM COUNT
5752	042656	000002		.WORD 2	;DO IT TWICE
5753					
5754	042660	100	T1WTM:	.BYTE WTM	;WRITE TAPE MARK
5755	042661	000		.BYTE NULPAT	;NO DATA NEEDED
5756	042662	000000		.WORD 0	;NO ITEM COUNT
5757	042664	000001		.WORD 1	;DO IT ONE TIME
5758					
5759	042666	060	T1SKP:	.BYTE SKP	;SKIP TAPE MARK
5760	042667	000		.BYTE NULPAT	;NO DATA NEEDED
5761	042670	000001		.WORD 1	;SKIP 1 TAPE MARK
5762	042672	000001		.WORD 1	;DO IT ONE TIME
5763					
5764	042674	061	T1SKR:	.BYTE SKR	;SKIP TAPE MARK REVERSE
5765	042675	000		.BYTE NULPAT	;NO DATA NEEDED
5766	042676	000002		.WORD 2	;SKIP REVERSE 2 TAPE MARKS
5767	042700	000001		.WORD 1	;DO IT ONCE
5768					
5769	042702	050	T1SPC1:	.BYTE SPC	;SPACE RECORDS
5770	042703	000		.BYTE NULPAT	;NO DATA NEEDED
5771	042704	000124		.WORD 84.	;SPACE 84 RECORDS
5772	042706	000001		.WORD 1	;DO IT ONE TIME
5773					
5774	042710	050	T1SPC2:	.BYTE SPC	;SPACE RECORDS
5775	042711	000		.BYTE NULPAT	;NO DATA NEEDED
5776	042712	000105		.WORD 69.	;SPACE 69 RECORDS
5777	042714	000001		.WORD 1	;DO IT ONE TIME
5778					
5779	042716	050	T1SPC3:	.BYTE SPC	;SPACE RECORDS
5780	042717	000		.BYTE NULPAT	;NO DATA NEEDED
5781	042720	000047		.WORD 39.	;SPACE 39 RECORDS
5782	042722	000001		.WORD 1	;DO IT ONE TIME
5783					
5784	042724	070	T1SP01:	.BYTE SPO	;SPACE OBJECTS
5785	042725	000		.BYTE NULPAT	;NO DATA NEEDED
5786	042726	000070		.WORD 56.	;SPACE 56 OBJECTS
5787	042730	000001		.WORD 1	;DO IT ONE TIME
5788					
5789	042732	061	T1SKR1:	.BYTE SKR	;SKIP TAPE MARK REVERSE
5790	042733	000		.BYTE NULPAT	;NO DATA NEEDED
5791	042734	000004		.WORD 4	;4 TAPE MARKS
5792	042736	000001		.WORD 1	;DO IT ONCE
5793					
5794	042740	070	T1SP02:	.BYTE SPO	;SPACE OBJECTS

5795	042741	000	.BYTE	NULPAT	;NO DATA NEEDED
5796	042742	000107	.WORD	71.	;SPACE 71 OBJECTS
5797	042744	000001	.WORD	1	;DO IT ONE TIME
5798					
5799	042746	070	T1SP03: .BYTE	SPO	;SPACE OBJECTS
5800	042747	000	.BYTE	NULPAT	;NO DATA NEEDED
5801	042750	000102	.WORD	66.	;SPACE 66 OBJECTS
5802	042752	000001	.WORD	1	;DO IT ONE TIME
5803					
5804	042754	071	T1SPR1: .BYTE	SPR	;SPACE OBJECTS REVERSE
5805	042755	000	.BYTE	NULPAT	;NO DATA NEEDED
5806	042756	000567	.WORD	375.	;SPACE 375 OBJECTS
5807	042760	000001	.WORD	1	;DO IT ONE TIME
5808					
5809	042762	060	T1SKP1: .BYTE	SKP	;SKIP TAPE MARKS
5810	042763	000	.BYTE	NULPAT	;NO DATA NEEDED
5811	042764	000004	.WORD	4.	;SKIP 4 TAPE MARKS
5812	042766	000001	.WORD	1	;DO IT ONE TIME
5813					
5814	042770	060	T1SKP2: .BYTE	SKP	;SKIP TAPE MARKS
5815	042771	000	.BYTE	NULPAT	;NO DATA NEEDED
5816	042772	000002	.WORD	2.	;SKIP 2 TAPE MARKS
5817	042774	000001	.WORD	1	;DO IT ONE TIME
5818					
5819	042776	071	T1SPR2: .BYTE	SPR	;SPACE OBJECTS REVERSE
5820	042777	000	.BYTE	NULPAT	;NO DATA NEEDED
5821	043000	000300	.WORD	192.	;SPACE 192 OBJECTS
5822	043002	000001	.WORD	1	;DO IT ONE TIME
5823					
5824	043004	062	T1SKD: .BYTE	SKD	;SKIP TO LEOT
5825	043005	000	.BYTE	NULPAT	;NO DATA NEEDED
5826	043006	000004	.WORD	4	;NO ITEM COUNT
5827	043010	000001	.WORD	1	;DO IT ONE TIME
5828					
5829	043012	020	T1WR1: .BYTE	WR	;WRITE RECORD
5830	043013	001	.BYTE	PAT1	;DATA PATERN 1 (ALL 1'S)
5831	043014	000026	.WORD	22.	;BYTE COUNT OF 512.
5832	043016	000143	.WORD	99.	;DO IT 99 TIMES
5833					
5834	043020	020	T1WR2: .BYTE	WR	;WRITE RECORD
5835	043021	002	.BYTE	PAT2	;DATA PATERN 2 (ALL 0'S)
5836	043022	001015	.WORD	525.	;BYTE COUNT OF 525
5837	043024	000124	.WORD	84.	;DO IT 84 TIMES
5838					
5839	043026	020	T1WR3: .BYTE	WR	;WRITE RECORD
5840	043027	003	.BYTE	PAT3	;DATA PATERN 3 (WORST MFM)
5841	043030	002016	.WORD	1038.	;BYTE COUNT OF 1038
5842	043032	000105	.WORD	69.	;DO IT 69 TIMES
5843					
5844	043034	020	T1WR4: .BYTE	WR	;WRITE RECORD
5845	043035	004	.BYTE	PAT4	;DATA PATERN 4 (ALTERNATE 1'S AND 0'S)
5846	043036	003017	.WORD	1551.	;BYTE COUNT OF 1551
5847	043040	000066	.WORD	54.	;DO IT 54 TIMES
5848					
5849	043042	020	T1WR5: .BYTE	WR	;WRITE RECORD
5850	043043	003	.BYTE	PAT3	;DATA PATERN 3 (WORST MFM)
5851	043044	004020	.WORD	2064.	;BYTE COUNT OF 2064

```

5852 043046 000047          .WORD  39.          ;DO IT 39 TIMES
5853
5854 043050      020      T1WR6: .BYTE  WR          ;WRITE RECORD
5855 043051      001          .BYTE  PAT1          ;DATA PATTERN 1 (ALL 1'S)
5856 043052 005021          .WORD  2577.         ;BYTE COUNT OF 2577
5857 043054 000030          .WORD  24.          ;DO IT 24 TIMES
5858
5859 043056      010      T1RD1: .BYTE  RD          ;READ RECORD
5860 043057      001          .BYTE  PAT1          ;DATA PATTERN 1 (ALL 1'S)
5861 043060 000026          .WORD  22.          ;BYTE COUNT OF 512.
5862 043062 000143          .WORD  99.          ;DO IT 99 TIMES
5863
5864 043064      010      T1RD2: .BYTE  RD          ;READ RECORD
5865 043065      002          .BYTE  PAT2          ;DATA PATTERN 2 (ALL 0'S)
5866 043066 001015          .WORD  525.         ;BYTE COUNT OF 525
5867 043070 000124          .WORD  84.          ;DO IT 84 TIMES
5868
5869 043072      010      T1RD3: .BYTE  RD          ;READ RECORD
5870 043073      003          .BYTE  PAT3          ;DATA PATTERN 3 (WORST MFM)
5871 043074 002016          .WORD  1038.        ;BYTE COUNT OF 1038
5872 043076 000105          .WORD  69.          ;DO IT 69 TIMES
5873
5874 043100      010      T1RD4: .BYTE  RD          ;READ RECORD
5875 043101      004          .BYTE  PAT4          ;DATA PATTERN 4 (ALTERNATE 1'S AND 0'S)
5876 043102 003017          .WORD  1551.        ;BYTE COUNT OF 1551
5877 043104 000066          .WORD  54.          ;DO IT 54 TIMES
5878
5879 043106      010      T1RD5: .BYTE  RD          ;READ RECORD
5880 043107      003          .BYTE  PAT3          ;DATA PATTERN 3 (WORST MFM)
5881 043110 004020          .WORD  2064.        ;BYTE COUNT OF 2064
5882 043112 000047          .WORD  39.          ;DO IT 39 TIMES
5883
5884 043114      010      T1RD6: .BYTE  RD          ;READ RECORD
5885 043115      001          .BYTE  PAT1          ;DATA PATTERN 1 (ALL 1'S)
5886 043116 005021          .WORD  2577.        ;BYTE COUNT OF 2577
5887 043120 000030          .WORD  24.          ;DO IT 24 TIMES
5888
5889          .EVEN
5890
5891 043122          ENDTST
          043122      L10015:
          043122 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 043124      BGNTST
5913 043124      T2::
5914 043124 105737 002234      TSTB      T3ONLY          ;CHECK THE TEST 3 ONLY FLAG
5915 043130 001414      BEQ        START2        ;BRANCH IF IT IS CLEAR
5916 043132      PRINTF     #BYPASS,L$TEST      ;PRINT THE TEST BYPASSED MESSAGE
5917 043132 013746 002114      MOV        L$TEST,-(SP)
5918 043136 012746 021001      MOV        #BYPASS,-(SP)
5919 043142 012746 000002      MOV        #2,-(SP)
5920 043146 010600      MOV        SP,R0
5921 043150 104417      TRAP     C$PNTF
5922 043152 062706 000006      ADD        #6,SP
5923 043156 000137 043542      JMP        T2EXIT          ;GET OUT
5924
5925 043162 005737 003524      START2: TST        UDROP          ;HAVE ALL UNITS BEEN DROPPED ?
5926 043166 001014      BNE        5$              ;GO START THE TEST
5927 043170      PRINTF     #BYPASS,L$TEST      ;PRINT THE TEST BYPASSED MESSAGE
5928 043170 013746 002114      MOV        L$TEST,-(SP)
5929 043174 012746 021001      MOV        #BYPASS,-(SP)
5930 043200 012746 000002      MOV        #2,-(SP)
5931 043204 010600      MOV        SP,R0
5932 043206 104417      TRAP     C$PNTF
5933 043210 062706 000006      ADD        #6,SP
5934 043214 000137 043542      JMP        T2EXIT          ;GET OUT IF NONE LEFT TO TEST
5935
5936 043220 105737 002212      5$:   TSTB      CLOCK          ;IS THE CLOCK ENABLED
5937 043224 001421      BEQ        G02              ;NO, THEN CAN'T PRINT TIME
5938 043226      PRINTF     #TIME,<B,HOURS>,<B,MINUTE>,<B,SECOND>,
5939 043226 005046      CLR        -(SP)
5940 043230 153716 002215      BISB     SECOND,(SP)
5941 043234 005046      CLR        -(SP)
5942 043236 153716 002214      BISB     MINUTE,(SP)
5943 043242 005046      CLR        -(SP)
5944 043244 153716 002213      BISB     HOURS,(SP)
5945 043250 012746 020044      MOV        #TIME,-(SP)
5946 043254 012746 000004      MOV        #4,-(SP)
5947 043260 010600      MOV        SP,R0
5948 043262 104417      TRAP     C$PNTF
5949 043264 062706 000012      ADD        #12,SP
5950
5951 043270 004737 033464      G02:  JSR        PC,CLREOT      ;MAKE SURE EOT STATUS IS CLEAR
5952 043274 005037 003520      CLR        OBJECT          ;CLEAR THE OBJECT COUNTER
5953 043300 005037 003440      CLR        TSTMSK         ;ALLOW NO ERRORS
5954

```

```

5931 043304 012705 043552      MOV      #T2REW,R5      ;SET UP TO DO A REWIND
5932 043310 004737 021240      JSR      PC,SCHED      ;GO ISSUE A REWIND TO ALL DRIVES
5933 043314 005737 003524      TST      UDROP         ;HAVE ALL UNITS BEEN DROPPED ?
5934 043320 001510                BEQ      T2EXIT        ;GET OUT IF NONE LEFT TO TEST
5935
5936
5937 043322 004737 033564      5$:     JSR      PC,SDSTUP ;RESET THE RANDOM SEEDS
5938 043326 012705 043560      MOV      #T2WRT,R5    ;SET UP TO DO A WRITE ITERATION
5939 043332 004737 021240      JSR      PC,SCHED      ;GO ISSUE WRITES TO ALL DRIVES
5940 043336 005737 003524      TST      UDROP         ;HAVE ALL UNITS BEEN DROPPED ?
5941 043342 001477                BEQ      T2EXIT        ;GET OUT IF NONE LEFT TO TEST
5942
5943 043344 012705 043566      MOV      #T2LEOT,R5   ;SET UP TO DO A WRITE LEOT
5944 043350 004737 021240      JSR      PC,SCHED      ;GO ISSUE WRITES TO ALL DRIVES
5945 043354 005737 003524      TST      UDROP         ;HAVE ALL UNITS BEEN DROPPED ?
5946 043360 001470                BEQ      T2EXIT        ;GET OUT IF NONE LEFT TO TEST
5947
5948 043362 012705 043552      MOV      #T2REW,R5    ;SET UP TO DO A REWIND
5949 043366 004737 021240      JSR      PC,SCHED      ;GO ISSUE A REWIND TO ALL DRIVES
5950 043372 005737 003524      TST      UDROP         ;HAVE ALL UNITS BEEN DROPPED ?
5951 043376 001461                BEQ      T2EXIT        ;GET OUT IF NONE LEFT TO TEST
5952
5953 043400 005737 003520      TST      OBJECT        ;IS THIS THE FIRST TIME THROUGH ?
5954 043404 001412                BEQ      10$           ;YES, DON'T DO THE SPACE FORWARD
5955 043406 012705 043610      MOV      #T2SPO,R5    ;SET UP TO SPACE OBJECTS
5956 043412 013765 003520 000002  MOV      OBJECTS,ITMCNT(R5) ;SET UP # OF OBJECTS TO SPACE FORWARD
5957 043420 004737 021240      JSR      PC,SCHED      ;GO ISSUE A REWIND TO ALL DRIVES
5958 043424 005737 003524      TST      UDROP         ;HAVE ALL UNITS BEEN DROPPED ?
5959 043430 001444                BEQ      T2EXIT        ;GET OUT IF NONE LEFT TO TEST
5960
5961 043432 004737 033564      10$:    JSR      PC,SDSTUP    ;RESET THE RANDOM SEEDS
5962 043436 012705 043574      MOV      #T2RD,R5     ;SET UP TO DO A READITERATION
5963 043442 004737 021240      JSR      PC,SCHED      ;GO ISSUE READS TO ALL DRIVES
5964 043446 005737 003524      TST      UDROP         ;HAVE ALL UNITS BEEN DROPPED ?
5965 043452 001433                BEQ      T2EXIT        ;GET OUT IF NONE LEFT TO TEST
5966 043454 066537 000004 003520  ADD      ITRCNT(R5),OBJECTS ;ADD THE # OF RECORDS TO OBJECTS
5967
5968 043462 052737 000001 003440  BIS      #LEDB,TSTMSK  ;SET UP TO ALLOW LEOT DETECTED
5969 043470 012705 043602      MOV      #T2SKD,R5    ;SET UP TO DO A SKIP TO LEOT
5970 043474 004737 021240      JSR      PC,SCHED      ;GO ISSUE READS TO ALL DRIVES
5971 043500 042737 000001 003440  BIC      #LEDB,TSTMSK  ;DISALLOW LEOT DETECTED
5972 043506 005737 003524      TST      UDROP         ;HAVE ALL UNITS BEEN DROPPED ?
5973 043512 001413                BEQ      T2EXIT        ;GET OUT IF NONE LEFT TO TEST
5974 043514 066537 000004 003520  ADU      ITRCNT(R5),OBJECTS ;ADD THE # OF RECORDS TO OBJECTS
5975
5976 043522 022737 004716 003520  CMP      #T2END,OBJECTS ;HAVE WE DONE 2 TRACKS ?
5977 043530 001274                BNE      5$           ;NO, KEEP GOING
5978 043532 012705 043552      MOV      #T2REW,R5    ;SET UP TO DO A REWIND
5979 043536 004737 021240      JSR      PC,SCHED      ;GO ISSUE A REWIND TO ALL DRIVES
5980 043542 004737 033640      T2EXIT: JSR      PC,SDSAVE ;RESET THE RANDOM SEEDS
5981 043546                EXIT      TST
          043546 104432      TRAP     C$EXIT
          043550 000046      .WORD    L10016-.
5982
5983
5984 043552      160      T2REW:  .BYTE   REW
5985 043553      000      .BYTE   NULPAT      ;REWIND

```

5986	043554	000000		.WORD	0	
5987	043556	000001		.WORD	1	
5988						
5989	043560	020	T2WRT:	.BYTE	WR	;WRITE RECORDS
5990	043561	003		.BYTE	PAT3	
5991	043562	010000		.WORD	4096.	
5992	043564	000372		.WORD	250.	
5993						
5994	043566	100	T2LEOT:	.BYTE	WTM	;WRITE TAPE MARK
5995	043567	000		.BYTE	NULPAT	;NO DATA NEEDED
5996	043570	000000		.WORD	0	;NO ITEM COUNT
5997	043572	000002		.WORD	2	;DO IT TWICE
5998						
5999	043574	010	T2RD:	.BYTE	RD	;READ RECORDS
6000	043575	003		.BYTE	PAT3	
6001	043576	010000		.WORD	4096.	
6002	043600	000372		.WORD	250.	
6003						
6004	043602	062	T2SKD:	.BYTE	SKD	;SKIP TAPE MARK TO LEOT
6005	043603	000		.BYTE	NULPAT	;NO DATA NEEDED
6006	043604	000062		.WORD	50.	;SKIP 50 TAPE MARKS
6007	043606	000001		.WORD	1	;DO IT ONE TIME
6008						
6009	043610	070	T2SPO:	.BYTE	SPO	;SPACE OBJECTS
6010	043611	000		.BYTE	NULPAT	
6011	043612	000001		.WORD	1	
6012	043614	000001		.WORD	1	
6013						
6014				.EVEN		
6015	043616			ENDTST		
	043616		L10016:			
	043616	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 043620 BGNTST
        043620
6038
6039 043620 005737 003524 START3: TST UDROP ;HAVE ALL UNITS BEEN DROPPED ?
6040 043624 001014 BNE 5$ ;GO START THE TEST
6041 043626 PRINTF #BYPASS,L$TEST ;PRINT THE TEST BYPASSED MESSAGE
        043626 013746 002114 MOV L$TEST,-(SP)
        043632 012746 021001 MOV #BYPASS,-(SP)
        043636 012746 000002 MOV #2,-(SP)
        043642 010600 MOV SP,RO
        043644 104417 TRAP C$PNTF
        043646 062706 000006 ADD #6,SP
6042 043652 000137 044226 JMP T3EXIT ;GET OUT IF NONE LEFT TO TEST
6043
6044 043656 105737 002212 5$: TSTB CLOCK ;IS THE CLOCK ENABLED
6045 043662 001421 BEQ G03 ;NO, THEN CAN'T PRINT TIME
6046 043664 PRINTF #TIME,<B,HOURS>,<B,MINUTE>,<B,SECOND>,
        043664 005046 CLR -(SP)
        043666 153716 002215 BISB SECOND,(SP)
        043672 005046 CLR -(SP)
        043674 153716 002214 BISB MINUTE,(SP)
        043700 005046 CLR -(SP)
        043702 153716 002213 BISB HOURS,(SP)
        043706 012746 020044 MOV #TIME,-(SP)
        043712 012746 000004 MOV #4,-(SP)
        043716 010600 MOV SP,RO
        043720 104417 TRAP C$PNTF
        043722 062706 000012 ADD #12,SP
6047
6048 043726 004737 033714 G03: JSR PC,PATCLR ;MAKE SURE WE START WITH PATTERN 1
6049 043732 004737 033564 JSR PC,SDSTUP ;RESET THE RANDOM SEEDS
6050 043736 004737 033464 JSR PC,CLREOT ;CLEAR ALL EOT INDICATIONS
6051
6052 043742 005037 003440 CLR TSTMSK ;ALLOW NO ERRORS
6053 043746 005737 003524 TST UDROP ;HAVE ALL UNITS BEEN DROPPED ?
6054 043752 001525 BEQ T3EXIT ;GET OUT IF NONE LEFT TO TEST
6055
6056 043754 012705 044236 MOV #T3REW,R5 ;SET UP TO DO REWIND

```

6057	043760	004737	021240			JSR	PC,SCHED		;GO ISSUE TO ALL DRIVES
6058	043764	005737	003524			TST	UDROP		;HAVE ALL UNITS BEEN DROPPED ?
6059	043770	001516				BEQ	T3EXIT		;GET OUT IF NONE LEFT TO TEST
6060									
6061	043772	012705	044244	5\$:		MOV	#T3WRT,R5		;SET UP A WRITE ITERATION
6062	043776	004737	021240			JSR	PC,SCHED		;GO DO IT ON ALL DRIVES
6063	044002	005737	003524			TST	UDROP		;HAVE ALL UNITS BEEN DROPPED ?
6064	044006	001507				BEQ	T3EXIT		;GET OUT IF NONE LEFT TO TEST
6065	044010	023737	003526	003524		CMP	UEOT,UDROP		;ARE ALL UNITS AT EOT ?
6066	044016	001413				BEQ	10\$;YES, WRITE ONE MORE REC AND LEOT
6067									
6068	044020	012705	044252			MOV	#T3WTM,R5		;SET UP TO WRITE A TAPE MARK
6069	044024	004737	021240			JSR	PC,SCHED		;GO DO IT ON ALL DRIVES
6070	044030	005737	003524			TST	UDROP		;HAVE ALL UNITS BEEN DROPPED ?
6071	044034	001474				BEQ	T3EXIT		;GET OUT IF NONE LEFT TO TEST
6072	044036	023737	003526	003524		CMP	UEOT,UDROP		;ARE ALL UNITS AT EOT ?
6073	044044	001352				BNE	5\$;NO, KEEP WRITING
6074									
6075	044046	012737	000004	010560	10\$:	MOV	#4,LOOPS		;SET UP TO DO 4 TAPE MARKS
6076	044054	004737	033464		15\$:	JSR	PC,CLREOT		;CLEAR THE EOT INDICATORS
6077	044060	012705	044252			MOV	#T3WTM,R5		;SET UP TO WRITE A TAPE MARK
6078	044064	004737	021240			JSR	PC,SCHED		;GO DO IT ON ALL DRIVES
6079	044070	005737	003524			TST	UDROP		;HAVE ALL UNITS BEEN DROPPED ?
6080	044074	001454				BEQ	T3EXIT		;GET OUT IF NONE LEFT TO TEST
6081	044076	005337	010560			DEC	LOOPS		;SUBTRACT 1 FROM THE TAPE MARK COUNT
6082	044102	001364				BNE	15\$;KEEP GOING TIL THEY'RE ALL WRITTEN
6083									
6084	044104	004737	033464			JSR	PC,CLREOT		;CLEAR THE EOT INDICATORS
6085	044110	012705	044236			MOV	#T3REW,R5		;SET UP TO REWIND ALL DRIVES
6086	044114	004737	021240			JSR	PC,SCHED		;GO DO IT
6087	044120	005737	003524			TST	UDROP		;HAVE ALL UNITS BEEN DROPPED ?
6088	044124	001440				BEQ	T3EXIT		;GET OUT IF NONE LEFT TO TEST
6089	044126	004737	033714			JSR	PC,PATCLR		;START AT PATTERN 1
6090	044132	004737	033564			JSR	PC,SDSTUP		;RESET THE RANDOM SEEDS
6091									
6092	044136	012705	044260	20\$:		MOV	#T3RD,R5		;SET UP TO READ AN ITERATION SET
6093	044142	004737	021240			JSR	PC,SCHED		;GO ISSUE TO ALL DRIVES
6094	044146	005737	003524			TST	UDROP		;HAVE ALL UNITS BEEN DROPPED ?
6095	044152	001425				BEQ	T3EXIT		;GET OUT IF NONE LEFT TO TEST
6096	044154	023737	003526	003524		CMP	UEOT,UDROP		;ALL UNITS AT EOT ?
6097	044162	001413				BEQ	EX3REW		;YES, GO REWIND ALL DRIVES
6098									
6099	044164	012705	044266			MOV	#T3SPO,R5		;SPACE 1 OBJECT (TAPE MARK)
6100	044170	004737	021240			JSR	PC,SCHED		;GO DO IT AN ALL DRIVES
6101	044174	005737	003524			TST	UDROP		;HAVE ALL UNITS BEEN DROPPED ?
6102	044200	001412				BEQ	T3EXIT		;GET OUT IF NONE LEFT TO TEST
6103	044202	023737	003526	003524		CMP	UEOT,UDROP		;ALL UNITS AT EOT ?
6104	044210	001352				BNE	20\$;NO, KEEP READING
6105									
6106	044212	004737	033464			EX3REW:	JSR	PC,CLREOT	;CLEAR THE EOT INDICATORS
6107	044216	012705	044236			MOV	#T3REW,R5		;SET UP TO REWIND ALL DRIVES
6108	044222	004737	021240			JSR	PC,SCHED		;GO DO IT
6109									
6110	044226	004737	033640			T3EXIT:	JSR	PC,SDSAVE	;RESET THE RANDOM SEEDS
6111	044232					EXIT	TST		
	044232	104432				TRAP	C\$EXIT		
	044234	000040				.WORD	L10017-		

6112

6114	044236	160	T3REW:	.BYTE	REW	;REWIND
6115	044237	000		.BYTE	NULPAT	
6116	044240	000000		.WORD	0	
6117	044242	000001		.WORD	1	
6118						
6119	044244	020	T3WRT:	.BYTE	WR	;WRITE RECORDS
6120	044245	200		.BYTE	ALLPAT	
6121	044246	000000		.WORD	RNDBYT	
6122	044250	000000		.WORD	RNDITR	
6123						
6124	044252	100	T3WTM:	.BYTE	WTM	;WRITE TAPE MARK
6125	044253	000		.BYTE	NULPAT	
6126	044254	000000		.WORD	0	
6127	044256	000001		.WORD	1	
6128						
6129	044260	010	T3RD:	.BYTE	RD	;READ RECORDS
6130	044261	200		.BYTE	ALLPAT	
6131	044262	000000		.WORD	RNDBYT	
6132	044264	000000		.WORD	RNDITR	
6133						
6134	044266	070	T3SPO:	.BYTE	SPO	;SPACE OBJECT (TAPE MARK)
6135	044267	000		.BYTE	NULPAT	
6136	044270	000001		.WORD	1	
6137	044272	000001		.WORD	1	
6138						
6139				.EVEN		
6140	044274			ENDTST		
	044274		L10C17:			
	044274	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 044276      BGNTST
        T4::
6151 044276 105737 002234      TSTB    T3ONLY          ;CHECK THE TEST 3 ONLY FLAG
6152 044302 001414      BEQ     START4         ;BRANCH IF IT IS CLEAR
6153 044304      PRINTF   #BYPASS,L$TEST      ;PRINT THE TEST BYPASSED MESSAGE
        044304 013746 002114      MOV     L$TEST,-(SP)
        044310 012746 021001      MOV     #BYPASS,-(SP)
        044314 012746 000002      MOV     #2,-(SP)
        044320 010600      MOV     SP,R0
        044322 104417      TRAP   C$PNTF
        044324 062706 000006      ADD     #6,SP
6154 044330 000137 044626      JMP     T4EXIT          ;GET OUT
6155
6156 044334 005737 003524      START4: TST    UDROP          ;HAVE ALL UNITS BEEN DROPPED ?
6157 044340 001014      BNE     5$              ;GO START THE TEST
6158 044342      PRINTF   #BYPASS,L$TEST      ;PRINT THE TEST BYPASSED MESSAGE
        044342 013746 002114      MOV     L$TEST,-(SP)
        044346 012746 021001      MOV     #BYPASS,-(SP)
        044352 012746 000002      MOV     #2,-(SP)
        044356 010600      MOV     SP,R0
        044360 104417      TRAP   C$PNTF
        044362 062706 000006      ADD     #6,SP
6159 044366 000137 044626      JMP     T4EXIT          ;GET OUT IF NONE LEFT TO TEST
6160
6161 044372 105737 002212      5$:    TSTB    CLOCK          ;IS THE CLOCK ENABLED
6162 044376 001421      BEQ     G04             ;NO, THEN CAN'T PRINT TIME
6163 044400      PRINTF   #TIME,<B,HOURS>,<B,MINUTE>,<B,SECOND>,
        044400 005046      CLR     -(SP)
        044402 153716 002215      BISB    SECOND,(SP)
        044406 005046      CLR     -(SP)
        044410 153716 002214      BISB    MINUTE,(SP)
        044414 005046      CLR     -(SP)
        044416 153716 002213      BISB    HOURS,(SP)
        044422 012746 020044      MOV     #TIME,-(SP)
        044426 012746 000004      MOV     #4,-(SP)
        044432 010600      MOV     SP,R0
        044434 104417      TRAP   C$PNTF
        044436 062706 000012      ADD     #12,SP
6164
6165 044442 004737 033714      G04:   JSR     PC,PATCLR
6166 044446 004737 033564      JSR     PC,SDSTUP
6167 044452 004737 033464      JSR     PC,CLREOT
6168 044456 005037 003440      CLR     TSTMSK
6169
6170 044462 012705 044632      MOV     #T4REW,R5
6171 044466 004737 021240      JSR     PC,SCHED
6172 044472 005737 003524      TST    UDROP
6173 044476 001453      BEQ     T4EXIT
6174
    
```



```

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 044656          BGNTST
        044656
6231 044656 105737 002234
6232 044662 001414
6233 044664          T5::
        044664 013746 002114          TSTB      T3ONLY          ;CHECK THE TEST 3 ONLY FLAG
        044670 012746 021001          BEQ       START5          ;BRANCH IF IT IS CLEAR
        044674 012746 000002          PRINTF   #BYPASS,L$TEST ;PRINT THE TEST BYPASSED MESSAGE
        044700 010600          MOV      L$TEST,-(SP)
        044702 104417          MOV      #BYPASS,-(SP)
        044704 062706 000006          MOV      #2,-(SP)
        044710 000137 045154          MOV      SP,R0
        044710 000137 045154          TRAP     C$PNTF
        044710 000137 045154          ADD      #6,SP
        044710 000137 045154          JMP      TSEXIT          ;GET OUT

6235          START5: TST      UDROP          ;HAVE ALL UNITS BEEN DROPPED ?
6236 044714 005737 003524          BNE      5$              ;GO START THE TEST
6237 044720 001014
6238 044722          PRINTF   #BYPASS,L$TEST ;PRINT THE TEST BYPASSED MESSAGE
        044722 013746 002114          MOV      L$TEST,-(SP)
        044726 012746 021001          MOV      #BYPASS,-(SP)
        044732 012746 000002          MOV      #2,(SP)
        044736 010600          MOV      SP,R0
        044740 104417          TRAP     C$PNTF
        044742 062706 000006          ADD      #6,SP
6239 044746 000137 045154          JMP      TSEXIT          ;GET OUT IF NONE LEFT TO TEST
6240
6241 044752 105737 002212          5$:   TSTB      CLOCK          ;IS THE CLOCK ENABLED
6242 044756 001421          BEQ      G05              ;NO, THEN CAN'T PRINT TIME
6243 044760          PRINTF   #TIME,<B,HOURS>,<B,MINUTE>,<B,SECOND>,
        044760 005046          CLR      -(SP)
        044762 153716 002215          BISB     SECOND,(SP)
        044766 005046          CLR      -(SP)
        044770 153716 002214          BISB     MINUTE,(SP)
        044774 005046          CLR      -(SP)
        044776 153716 002213          BISB     HOURS,(SP)
        045002 012746 020044          MOV      #TIME,-(SP)
        045006 012746 000004          MOV      #4,-(SP)
        045012 010600          MOV      SP,R0
        045014 104417          TRAP     C$PNTF
        045016 062706 000012          ADD      #12,SP

6244
6245 045022 004737 033714          G05:   JSR      PC,PATCLR          ;MAKE SURE WE START WITH PATTERN 1
6246 045026 004737 033564          JSR      PC,SDSTUP          ;SET UP THE RANDOM SEEDS
6247 045032 004737 033464          JSR      PC,CLREOT          ;CLEAR THE EOT FLAG
6248
6249 045036 132737 000001 002236          BITB     #BIT0,DMPBUF          ;ARE WE DUMPING ALL RECORDS?
6250 045044 001403          BEQ      1$              ;NO - BRANCH
6251 045046 012737 000001 045172          MOV      #1,T5RD+4          ;CHANGE ITER COUNT TO 1
6252
6253 045054 005037 003440          1$:   CLR      T5MSK          ;NO ALLOWABLE ERRORS
    
```

6254	045060	012705	045160		MOV	0T5REW,R5		;POINT R5 TO THE REWIND TABLE
6255	045064	004737	021240		JSR	PC,SCHED		;GO START THE TEST
6256	045070	005737	003524		TST	UDROP		;HAVE ALL UNITS BEEN DROPPED ?
6257	045074	001427			BEQ	T5EXIT		;GET OUT IF NONE LEFT TO TEST
6258								
6259	045076	012737	000012	003440	MOV	0TMB!RDTB,T5TMSK		;TAPE MARKS AND TRUNC. RECORDS OK
6260	045104	012705	045166	5:	MOV	0T5RD,R5		;POINT R5 TO THE TEST TABLE
6261	045110	004737	021240		JSR	PC,SCHED		;GO START THE TEST
6262	045114	005737	003524		TST	UDROP		;HAVE ALL UNITS BEEN DROPPED ?
6263	045120	001415			BEQ	T5EXIT		;GET OUT IF NONE LEFT TO TEST
6264	045122	023737	003526	003524	CMP	UEOT,UDROP		;ARE THEY ALL AT EOT ?
6265	045130	001365			BNE	5:		;BRANCH IF THEY ARE NOT
6266								
6267	045132	004737	033464	10:	JSR	PC,CLREOT		
6268	045136	012705	045160		MOV	0T5REW,R5		;POINT R5 TO THE REWIND TABLE
6269	045142	004737	021240		JSR	PC,SCHED		;GO REWIND ALL UNITS
6270	045146	012737	001751	045172	MOV	01001.,T5RD-4		;RESTORE ITER COUNT
6271	045154				T5EXIT:	EXIT		
	045154	104432			TRAP	C#EXIT		
	045156	000016			.WORD	L10021-		
6272								
6273	045160	160			TSREW:	.BYTE	REW	;REWIND
6274	045161	000				.BYTE	NULPAT	
6275	045162	000000				.WORD	0	
6276	045164	000001				.WORD	1	
6277								
6278	045166	010			T5RD:	.BYTE	RD	;READ RECORDS
6279	045167	200				.BYTE	ALLPAT	
6280	045170	010000				.WORD	4096.	
6281	045172	001751				.WORD	1001.	
6282								
6283						.EVEN		
6284	045174				ENDTST			
	045174				L10021:			
	045174	104401			TRAP	C#ETST		

```

6286 .SBTTL TEST 6: Start/Stop Write/Read Test
6287
6288
6289 ;**
6290 ;This test rewinds the tape, and executes the following sequence:
6291 ;
6292 ; 1. Write 1300 records one at a time,
6293 ; 2. Write 2 file marks (LEOT),
6294 ; 3. Rewind,
6295 ; 4. Read 1300 records one at a time,
6296 ; 5. Skip to LEOT.
6297 ; 6. Rewind,
6298 ;
6299 ;This sequence will permit hardware retries, if not user disabled.
6300 ;This test will run until exhaustion of the command count or fatal error
6301 ;is detected. All data patterns including random data will be used
6302 ;in this test.
6303 ;--
6303 045176 BGNTST
6304 045176 105737 002234 T6::
6305 045202 001414 TSTB T3ONLY ;CHECK THE TEST 3 ONLY FLAG
6306 045204 013746 002114 BEQ START6 ;BRANCH IF IT IS CLEAR
6307 045204 012746 021001 PRINTF #BYPASS,L$TEST ;PRINT THE TEST BYPASSED MESSAGE
6308 045210 012746 000002 MOV L$TEST,-(SP)
6309 045214 012746 000002 MOV #BYPASS,-(SP)
6310 045220 010600 MOV #2,-(SP)
6311 045222 104417 MOV SP,R0
6312 045224 062706 000006 TRAP C$PNTF
6313 045230 000137 045604 ADD #6,SP
6314 045234 005737 003524 JMP T6EXIT ;GET OUT
6315 045240 001014 START6: TST UDROP ;HAVE ALL UNITS BEEN DROPPED ?
6316 045242 013746 002114 BNE 5$ ;GO START THE TEST
6317 045242 012746 021001 PRINTF #BYPASS,L$TEST ;PRINT THE TEST BYPASSED MESSAGE
6318 045246 012746 000002 MOV L$TEST,-(SP)
6319 045252 012746 000002 MOV #BYPASS,-(SP)
6320 045256 010600 MOV #2,-(SP)
6321 045260 104417 MOV SP,R0
6322 045262 062706 000006 TRAP C$PNTF
6323 045266 000137 045604 ADD #6,SP
6324 045272 105737 002212 JMP T6EXIT ;GET OUT IF NONE LEFT TO TEST
6325 045276 001421 5$: TSTB CLOCK ;IS THE CLOCK ENABLED
6326 045300 005046 BEQ G06 ;NO, THEN CAN'T PRINT TIME
6327 045302 153716 002215 PRINTF #TIME,<B,HOURS>,<B,MINUTE>,<B,SECOND>,
6328 045306 005046 CLR -(SP)
6329 045310 153716 002214 BISB SECOND,(SP)
6330 045314 005046 CLR -(SP)
6331 045316 153716 002213 BISB MINUTE,(SP)
6332 045322 012746 020044 CLR -(SP)
6333 045326 012746 000004 BISB HOURS,(SP)
6334 045332 010600 MOV #TIME,-(SP)
6335 045334 104417 MOV #4,(SP)
6336 045336 062706 000012 MOV SP,R0
6337 JSR PC,PATCLR
6338 045342 004737 033714 TRAP C$PNTF
6339 ADD #12,SP
6340 ;MAKE SURE WE START WITH PATTERN 1

```

6319	045346	004737	033564		JSR	PC,SDSTUP	;RESET THE RANDOM SEEDS
6320	045352	004737	033464		JSR	PC,CLREOT	;CLEAR ALL EOT INDICATIONS
6321							
6322	045356	005037	003440		CLR	TSTMSK	;ALLOW NO ERRORS
6323	045362	005737	003524		TST	UDROP	;HAVE ALL UNITS BEEN DROPPED ?
6324	045366	001506			BEQ	T6EXIT	;GET OUT IF NONE LEFT TO TEST
6325							
6326	045370	012705	045614		MOV	#T6REW,R5	;SET UP TO DO REWIND
6327	045374	004737	021240		JSR	PC,SCHED	;GO ISSUE TO ALL DRIVES
6328	045400	005737	003524		TST	UDROP	;HAVE ALL UNITS BEEN DROPPED ?
6329	045404	001477			BEQ	T6EXIT	;GET OUT IF NONE LEFT TO TEST
6330	045406	012737	002424	003404	MOV	#1300.,COUNT	;STEP UP THE WRITE ITERATION COUNT
6331							
6332	045414	012705	045622	5\$:	MOV	#T6WRT,R5	;SET UP A WRITE ITERATION
6333	045420	004737	021240		JSR	PC,SCHED	;GO DO IT ON ALL DRIVES
6334	045424	005737	003524		TST	UDROP	;HAVE ALL UNITS BEEN DROPPED ?
6335	045430	001465			BEQ	T6EXIT	;GET OUT IF NONE LEFT TO TEST
6336	045432	005337	003404		DEC	COUNT	;SUBTRACT 1 FROM THE ITERATION COUNT
6337	045436	001366			BNE	5\$;MORE COMMANDS TO DO, KEEP GOING
6338							
6339	045440	012705	045630		MOV	#T6WTM,R5	;SET UP TO WRITE A TAPE MARK
6340	045444	004737	021240		JSR	PC,SCHED	;GO DO IT ON ALL DRIVES
6341	045450	005737	003524		TST	UDROP	;HAVE ALL UNITS BEEN DROPPED ?
6342	045454	001453			BEQ	T6EXIT	;GET OUT IF NONE LEFT TO TEST
6343							
6344	045456	012705	045614		MOV	#T6REW,R5	;SET UP TO REWIND ALL DRIVES
6345	045462	004737	021240		JSR	PC,SCHED	;GO DO IT
6346	045466	005737	003524		TST	UDROP	;HAVE ALL UNITS BEEN DROPPED ?
6347	045472	001444			BEQ	T6EXIT	;GET OUT IF NONE LEFT TO TEST
6348							
6349	045474	004737	033714		JSR	PC,PATCLR	;START AT PATTERN 1
6350	045500	004737	033564		JSR	PC,SDSTUP	;RESET THE RANDOM SEEDS
6351	045504	012737	002424	003404	MOV	#1300.,COUNT	;STEP UP THE WRITE ITERATION COUNT
6352							
6353	045512	012705	045636	10\$:	MOV	#T6RD,R5	;SET UP TO READ AN ITERATION SET
6354	045516	004737	021240		JSR	PC,SCHED	;GO DO IT ON ALL DRIVES
6355	045522	005737	003524		TST	UDROP	;HAVE ALL UNITS BEEN DROPPED ?
6356	045526	001426			BEQ	T6EXIT	;GET OUT IF NONE LEFT TO TEST
6357	045530	005337	003404		DEC	COUNT	;SUBTRACT 1 FROM THE ITERATION COUNT
6358	045534	001366			BNE	10\$;MORE COMMANDS TO DO, KEEP GOING
6359							
6360	045536	052737	000001	003440	BIS	#LEDB,TSTMSK	;SET UP TO ALLOW LEOT DETECTED
6361	045544	012705	045644		MOV	#T6SKD,R5	;SKIP TO LEOT
6362	045550	004737	021240		JSR	PC,SCHED	;GO DO IT AN ALL DRIVES
6363	045554	042737	000001	003440	BIC	#LEDB,TSTMSK	;DISALLOW LEOT DETECTED
6364	045562	005737	003524		TST	UDROP	;HAVE ALL UNITS BEEN DROPPED ?
6365	045566	001406			BEQ	T6EXIT	;GET OUT IF NONE LEFT TO TEST
6366							
6367	045570	004737	033464		JSR	PC,CLREOT	;CLEAR THE EOT INDICATORS
6368	045574	012705	045614		MOV	#T6REW,R5	;SET UP TO REWIND ALL DRIVES
6369	045600	004737	021240		JSR	PC,SCHED	;GO DO IT
6370							
6371	045604	004737	033640	T6EXIT:	JSR	PC,SDSAVE	;RESET THE RANDOM SEEDS
6372	045610				EXIT	TST	
	045610	104432			TRAP	C\$EXIT	
	045612	000040			.WORD	L10022-.	
6373							

6375 045614 160	T6REW:	.BYTE	REW	;REWIND
6376 045615 000		.BYTE	NULPAT	
6377 045616 000000		.WORD	0	
6378 045620 000001		.WORD	1.	
6379				
6380 045622 020	T6WRT:	.BYTE	WR	;WRITE RECORDS
6381 045623 200		.BYTE	ALLPAT	
6382 045624 020000		.WORD	8192.	
6383 045626 000001		.WORD	1.	
6384				
6385 045630 100	T6WTM:	.BYTE	WTM	;WRITE TAPE MARK
6386 045631 000		.BYTE	NULPAT	
6387 045632 000000		.WORD	0	
6388 045634 000002		.WORD	2.	
6389				
6390 045636 010	T6RD:	.BYTE	RD	;READ RECORDS
6391 045637 200		.BYTE	ALLPAT	
6392 045640 020000		.WORD	8192.	
6393 045642 000001		.WORD	1.	
6394				
6395 045644 062	T6SKD:	.BYTE	SKD	;SKIP TO LEOT
6396 045645 000		.BYTE	NULPAT	
6397 045646 000001		.WORD	1	
6398 045650 000001		.WORD	1	
6399				
6400		.EVEN		
6401 045652		ENDTST		
045652	L10022:			
045652 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 045654      BGNTST
6415 045654      T7::
6416 045654 105737 002234      TSTB      T3ONLY          ;CHECK THE TEST 3 ONLY FLAG
6417 045660 001414      BEQ      START7          ;BRANCH IF IT IS CLEAR
6418 045662      PRINTF    #BYPASS,L$TEST      ;PRINT THE TEST BYPASSED MESSAGE
6419 045662 013746 002114      MOV      L$TEST,-(SP)
6420 045666 012746 021001      MOV      #BYPASS,-(SP)
6421 045672 012746 000002      MOV      #2,-(SP)
6422 045676 010600      MOV      SP,R0
6423 045700 104417      TRAP    C$PNTF
6424 045702 062706 000006      ADD      #6,SP
6425 045706 000137 046230      JMP      T7EXIT          ;GET OUT
6426
6427 045712 005737 003524      START7: TST      UDROP          ;HAVE ALL UNITS BEEN DROPPED ?
6428 045716 001014      BNE      5$              ;GO START THE TEST
6429 045720      PRINTF    #BYPASS,L$TEST      ;PRINT THE TEST BYPASSED MESSAGE
6430 045720 013746 002114      MOV      L$TEST,-(SP)
6431 045724 012746 021001      MOV      #BYPASS,-(SP)
6432 045730 012746 000002      MOV      #2,-(SP)
6433 045734 010600      MOV      SP,R0
6434 045736 104417      TRAP    C$PNTF
6435 045740 062706 000006      ADD      #6,SP
6436 045744 000137 046230      JMP      T7EXIT          ;GET OUT IF NONE LEFT TO TEST
6437
6438 045750 105737 002212      5$:      TSTB      CLOCK          ;IS THE CLOCK ENABLED
6439 045754 001421      BEQ      G07            ;NO, THEN CAN'T PRINT TIME
6440 045756      PRINTF    #TIME,<B,HOURS>,<B,MINUTE>,<B,SECOND>,
6441 045756 005046      CLR      -(SP)
6442 045760 153716 002215      BISB     SECOND,(SP)
6443 045764 005046      CLR      -(SP)
6444 045766 153716 002214      BISB     MINUTE,(SP)
6445 045772 005046      CLR      -(SP)
6446 045774 153716 002213      BISB     HOURS,(SP)
6447 046000 012746 020044      MOV      #TIME,-(SP)
6448 046004 012746 000004      MOV      #4,-(SP)
6449 046010 010600      MOV      SP,R0
6450 046012 104417      TRAP    C$PNTF
6451 046014 062706 000012      ADD      #12,SP
6452
6453 046020 005037 003410      G07:    CLR      BRCNT          ; CLEAR THE BRANCH COUNTER
6454 046024 012705 002232      MOV      #T7TBL,R5      ;POINT R5 TO TEST 7 TABLE
6455 046030 005037 003440      CLR      TSTMSK         ;NO ALLOWABLE ERRORS
6456 046034 052737 000001 003440      BIS      #LEDB,TSTMSK   ;SET UP TO ALLOW LEOT DETECTED
6457
6458 046042 004737 033564      10$:    JSR      PC,SDSTUP    ;SET UP THE RANDOM SEEDS
6459 046046 004737 033714      JSR      PC,PATCLR      ;USE THE SAME PATTERN
    
```



```
6484
6486
6497
6498
6526
6527 046242
6528
6529
6530
6531
6532
6533
6534
6535
6536
6537
6538 046242
        046242 000032
        046244
6539
6540 046244
        046244 000031
        046246 046270
        046250 160002
        046252 177564
6541 046254
        046254 001032
        046256 046305
        046260 000777
        046262 000000
        046264 000251
6542
6543 046266
        046266 021004
6544
6545
6546 046270      124      113      111 TKIPAD: .ASCIZ ?TKIP ADDRESS?
6547 046305      124      057      115 TKUNT:  .ASCIZ ?T/MSCP UNIT NUMBER?
6548
6549 046330
        046330
        046330
```

.TITLE PARAMETER CODING

.SBTTL HARDWARE PARAMETER CODING SECTION

BGNMOD

;++
; THE HARDWARE PARAMETER CODING SECTION CONTAINS MACROS
; THAT ARE USED BY THE SUPERVISOR TO BUILD P-TABLES. THE
; MACROS ARE NOT EXECUTED AS MACHINE INSTRUCTIONS BUT ARE
; INTERPRETED BY THE SUPERVISOR AS DATA STRUCTURES. THE
; MACROS ALLOW THE SUPERVISOR TO ESTABLISH COMMUNICATIONS
; WITH THE OPERATOR.
;--

BGNHRD
.WORD L10024-L\$HARD/2
L\$HARD::

GPRMA TKIPAD,0,0,160002,177564,YES
.WORD T\$CODE
.WORD TKIPAD
.WORD T\$LOLIM
.WORD T\$HILIM
GPRMD TKUNT,2,0,777,0,251,YES
.WORD T\$CODE
.WORD TKUNT
.WORD 777
.WORD T\$LOLIM
.WORD T\$HILIM

EXIT HRD
.WORD T\$CODE

ENDHRD
.EVEN

L10024:

```

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    046330                    BGNSFT
                              .WORD L10025-L$SOFT/2
                              L$SOFT::
6563
6564    046332                    GPRML    ECLK,0,1,YES
                              .WORD    T$CODE
                              046332    000130                    .WORD    ECLK
                              046334    047216                    .WORD    1
                              046336    000001                    XFERF    5$
6565    046340                    .WORD    T$CODE
                              046340    013044                    GPRMD    HOUR,0,D,177400,0,24.,YES
6566    046342                    .WORD    T$CODE
                              046342    000052                    .WORD    HOUR
                              046344    047247                    .WORD    177400
                              046346    177400                    .WORD    T$LOLIM
                              046350    000000                    .WORD    T$HILIM
                              046352    000030                    GPRMD    MINT,2,D,000377,0,60.,YES
6567    046354                    .WORD    T$CODE
                              046354    001052                    .WORD    MINT
                              046356    047331                    .WORD    000377
                              046360    047331                    .WORD    T$LOLIM
                              046362    000000                    .WORD    T$HILIM
                              046364    000074                   
6568
6569    046366                    5$:    GPRML    CTPA,4,400,YES
                              046366    002130                    .WORD    T$CODE
                              046370    047376                    .WORD    CTPA
                              046372    000400                    .WORD    400
6570    046374                    XFERF    10$
                              046374    012044                    .WORD    T$CODE
6571    046376                    GPRML    SERR,6,1,YES
                              046376    003130                    .WORD    T$CODE
                              046400    047433                    .WORD    SERR
                              046402    000001                    .WORD    1
6572    046404                    GPRML    SERC,6,400,YES
                              046404    003130                    .WORD    T$CODE
                              046406    047477                    .WORD    SERC
                              046410    000400                    .WORD    400
6573    046412                    GPRML    PADD,10,1,YES
                              046412    004130                    .WORD    T$CODE
                              046414    047541                    .WORD    PADD
                              046416    000001                    .WORD    1
6574
6575    046420                    10$:    GPRML    PRPA,10,400,YES
                              046420    004130                    .WORD    T$CODE
                              046422    047570                    .WORD    PRPA
                              046424    000400                    .WORD    400

```

6576	046426		XFERF	15\$
	046426	024044	.WORD	T\$CODE
6577	046430		GPRML	SOER,12,1,YES
	046430	005130	.WORD	T\$CODE
	046432	047623	.WORD	SOER
	046434	000001	.WORD	1
6578	046436		XFERT	11\$
	046436	004024	.WORD	T\$CODE
6579	046440		GPRML	SRER,12,400,YES
	046440	005130	.WORD	T\$CODE
	046442	047666	.WORD	SRER
	046444	000400	.WORD	400
6580	046446		11\$: GPRML	CLMD,14,1,YES
	046446	006130	.WORD	T\$CODE
	046450	047725	.WORD	CLMD
	046452	000001	.WORD	1
6581	046454		GPRML	MDTB,14,400,YES
	046454	006130	.WORD	T\$CODE
	046456	047774	.WORD	MDTB
	046460	000400	.WORD	400
6582	046462		GPRML	NOCL,16,1,YES
	046462	007130	.WORD	T\$CODE
	046464	050034	.WORD	NOCL
	046466	000001	.WORD	1
6583	046470		GPRML	PDMP,16,400,YES
	046470	007130	.WORD	T\$CODE
	046472	050077	.WORD	PDMP
	046474	000400	.WORD	400
6584				
6585	046476		15\$: GPRML	TSPA,20,1,YES
	046476	010130	.WORD	T\$CODE
	046500	050142	.WORD	TSPA
	046502	000001	.WORD	1
6586	046504		XFERF	20\$
	046504	023044	.WORD	T\$CODE
6587	046506		GPRMD	PATE,20,0,177400,0,7,YES
	046506	010032	.WORD	T\$CODE
	046510	050171	.WORD	PATE
	046512	177400	.WORD	177400
	046514	000000	.WORD	T\$LOLIM
	046516	000007	.WORD	T\$HILIM
6588	046520		GPRML	T3ON,22,1,YES
	046520	011130	.WORD	T\$CODE
	046522	050207	.WORD	T3ON
	046524	000001	.WORD	1
6589	046526		GPRML	T5CP,22,400,YES
	046526	011130	.WORD	T\$CODE
	046530	050230	.WORD	T5CP
	046532	000400	.WORD	400
6590	046534		GPRML	BDMP,24,1,YES
	046534	012130	.WORD	T\$CODE
	046536	050270	.WORD	BDMP
	046540	000001	.WORD	1
6591	046542		GPRML	CHGF,24,400,YES
	046542	012130	.WORD	T\$CODE
	046544	050334	.WORD	CHGF
	046546	000400	.WORD	400

6592	046550		XFERT	25\$
	046550	002024	.WORD	T\$CODE
6593	046552		20\$: XFER	SFTEX1
	046552	076004	.WORD	T\$CODE
6594				
6595	046554		25\$: GPRMD	CMD1,26,0,000377,0,377,YES
	046554	013032	.WORD	T\$CODE
	046556	050476	.WORD	CMD1
	046560	000377	.WORD	000377
	046562	000000	.WORD	T\$LOLIM
	046564	000377	.WORD	T\$HILIM
6596	046566		GPRMD	DPAT,26,0,177400,0,7,YES
	046566	013032	.WORD	T\$CODE
	046570	050370	.WORD	DPAT
	046572	177400	.WORD	177400
	046574	000000	.WORD	T\$LOLIM
	046576	000007	.WORD	T\$HILIM
6597	046600		GPRMD	ICNT,30,D,177777,0,MAXBUF,YES
	046600	014052	.WORD	T\$CODE
	046602	050407	.WORD	ICNT
	046604	177777	.WORD	177777
	046606	000000	.WORD	T\$LOLIM
	046610	020000	.WORD	T\$HILIM
6598	046612		GPRMD	ITER,32,D,177777,0,65000,YES
	046612	015052	.WORD	T\$CODE
	046614	050451	.WORD	ITER
	046616	177777	.WORD	177777
	046620	000000	.WORD	T\$LOLIM
	046622	065000	.WORD	T\$HILIM
6599				
6600	046624		GPRMD	CMD2,34,0,000377,0,377,YES
	046624	016032	.WORD	T\$CODE
	046626	050504	.WORD	CMD2
	046630	000377	.WORD	000377
	046632	000000	.WORD	T\$LOLIM
	046634	000377	.WORD	T\$HILIM
6601	046636		GPRMD	DPAT,34,0,177400,0,7,YES
	046636	016032	.WORD	T\$CODE
	046640	050370	.WORD	DPAT
	046642	177400	.WORD	177400
	046644	000000	.WORD	T\$LOLIM
	046646	000007	.WORD	T\$HILIM
6602	046650		GPRMD	ICNT,36,D,177777,0,MAXBUF,YES
	046650	017052	.WORD	T\$CODE
	046652	050407	.WORD	ICNT
	046654	177777	.WORD	177777
	046656	000000	.WORD	T\$LOLIM
	046660	020000	.WORD	T\$HILIM
6603	046662		GPRMD	ITER,40,D,177777,0,65000,YES
	046662	020052	.WORD	T\$CODE
	046664	050451	.WORD	ITER
	046666	177777	.WORD	177777
	046670	000000	.WORD	T\$LOLIM
	046672	065000	.WORD	T\$HILIM
6604				
6605	046674		GPRMD	CMD3,42,0,000377,0,377,YES
	046674	021032	.WORD	T\$CODE

	046676	0505i2	.WORD	CMD3
	046700	000377	.WORD	000377
	046702	000000	.WORD	T\$LOLIM
	046704	000377	.WORD	T\$HILIM
6606	046706		GPRMD	DPAT,42,0,177400,0,7,YES
	046706	021032	.WORD	T\$CODE
	046710	050370	.WORD	DPAT
	046712	177400	.WORD	177400
	046714	000000	.WORD	T\$LOLIM
	046716	000007	.WORD	T\$HILIM
6607	046720		GPRMD	ICNT,44,D,177777,0,MAXBUF,YES
	046720	022052	.WORD	T\$CODE
	046722	050407	.WORD	ICNT
	046724	177777	.WORD	177777
	046726	000000	.WORD	T\$LOLIM
	046730	020000	.WORD	T\$HILIM
6608	046732		GPRMD	ITER,46,D,177777,0,65000,YES
	046732	023052	.WORD	T\$CODE
	046734	050451	.WORD	ITER
	046736	177777	.WORD	177777
	046740	000000	.WORD	T\$LOLIM
	046742	065000	.WORD	T\$HILIM
6609	046744		XFER	CONT1
	046744	002004	.WORD	T\$CODE
6610				
6611	046746		SFTEX1: XFER	SFTEX2
	046746	076004	.WORD	T\$CODE
6612				
6613	046750		CONT1: GPRMD	CMD4,50,0,000377,0,377,YES
	046750	024032	.WORD	T\$CODE
	046752	050520	.WORD	CMD4
	046754	000377	.WORD	000377
	046756	000000	.WORD	T\$LOLIM
	046760	000377	.WORD	T\$HILIM
6614	046762		GPRMD	DPAT,50,0,177400,0,7,YES
	046762	024032	.WORD	T\$CODE
	046764	050370	.WORD	DPAT
	046766	177400	.WORD	177400
	046770	000000	.WORD	T\$LOLIM
	046772	000007	.WORD	T\$HILIM
6615	046774		GPRMD	ICNT,52,D,177777,0,MAXBUF,YES
	046774	025052	.WORD	T\$CODE
	046776	050407	.WORD	ICNT
	047000	177777	.WORD	177777
	047002	000000	.WORD	T\$LOLIM
	047004	020000	.WORD	T\$HILIM
6616	047006		GPRMD	ITER,54,D,177777,0,65000,YES
	047006	026052	.WORD	T\$CODE
	047010	050451	.WORD	ITER
	047012	177777	.WORD	177777
	047014	000000	.WORD	T\$LOLIM
	047016	065000	.WORD	T\$HILIM
6617				
6618	047020		GPRMD	CMD5,56,0,000377,0,377,YES
	047020	027032	.WORD	T\$CODE
	047022	050526	.WORD	CMD5
	047024	000377	.WORD	000377

	047026	000000	.WORD	T\$LOLIM
	047030	000377	.WORD	T\$HILIM
6619	047032		GPRMD	DPAT,56,0,177400,0,7,YES
	047032	027032	.WORD	T\$CODE
	047034	050370	.WORD	DPAT
	047036	177400	.WORD	177400
	047040	000000	.WORD	T\$LOLIM
	047042	000007	.WORD	T\$HILIM
6620	047044		GPRMD	ICNT,60,D,177777,0,MAXBUF,YES
	047044	030052	.WORD	T\$CODE
	047046	050407	.WORD	ICNT
	047050	177777	.WORD	177777
	047052	000000	.WORD	T\$LOLIM
	047054	020000	.WORD	T\$HILIM
6621	047056		GPRMD	ITER,62,D,177777,0,65000,YES
	047056	031052	.WORD	T\$CODE
	047060	050451	.WORD	ITER
	047062	177777	.WORD	177777
	047064	000000	.WORD	T\$LOLIM
	047066	065000	.WORD	T\$HILIM
6622				
6623	047070		GPRMD	CMD6,64,0,000377,0,377,YES
	047070	032032	.WORD	T\$CODE
	047072	050534	.WORD	CMD6
	047074	000377	.WORD	000377
	047076	000000	.WORD	T\$LOLIM
	047100	000377	.WORD	T\$HILIM
6624	047102		GPRMD	DPAT,64,0,177400,0,7,YES
	047102	032032	.WORD	T\$CODE
	047104	050370	.WORD	DPAT
	047106	177400	.WORD	177400
	047110	000000	.WORD	T\$LOLIM
	047112	000007	.WORD	T\$HILIM
6625	047114		GPRMD	ICNT,66,D,177777,0,MAXBUF,YES
	047114	033052	.WORD	T\$CODE
	047116	050407	.WORD	ICNT
	047120	177777	.WORD	177777
	047122	000000	.WORD	T\$LOLIM
	047124	020000	.WORD	T\$HILIM
6626	047126		GPRMD	ITER,70,D,177777,0,65000,YES
	047126	034052	.WORD	T\$CODE
	047130	050451	.WORD	ITER
	047132	177777	.WORD	177777
	047134	000000	.WORD	T\$LOLIM
	047136	065000	.WORD	T\$HILIM
6627	047140		XFER	CONT2
	047140	002004	.WORD	T\$CODE
6628				
6629	047142		SFTEX2: XFER	SFTEX3
	047142	025004	.WORD	T\$CODE
6630				
6631	047144		CONT2: GPRMD	CMD7,72,0,000377,0,377,YES
	047144	035032	.WORD	T\$CODE
	047146	050542	.WORD	CMD7
	047150	000377	.WORD	000377
	047152	000000	.WORD	T\$LOLIM
	047154	000377	.WORD	T\$HILIM

6632	047156				GPRMD	DPAT,72,0,177400,0,7,YES
	047156	035032			.WORD	T\$CODE
	047160	050370			.WORD	DPAT
	047162	177400			.WORD	177400
	047164	000000			.WORD	T\$LOLIM
	047166	000007			.WORD	T\$HILIM
6633	047170				GPRMD	ICNT,74,D,177777,0,MAXBUF,YES
	047170	036052			.WORD	T\$CODE
	047172	050407			.WORD	ICNT
	047174	177777			.WORD	177777
	047176	000000			.WORD	T\$LOLIM
	047200	020000			.WORD	T\$HILIM
6634	047202				GPRMD	ITER,76,D,177777,0,65000,YES
	047202	037052			.WORD	T\$CODE
	047204	050451			.WORD	ITER
	047206	177777			.WORD	177777
	047210	000000			.WORD	T\$LOLIM
	047212	065000			.WORD	T\$HILIM
6635						
6636	047214				SFTEX3: XFER	SFTEX4
	047214	070004			.WORD	T\$CODE
6637						
6638	047216	105	116	101	ECLK: .ASCIZ	/ENABLE TIME OF DAY CLOCK/
6639	047247	040	111	116	HOUR: .ASCIZ	/ INPUT HOUR IN 24 HOUR FORMAT (OMIT LEADING ZERO)/
6640	047331	040	111	116	MINT: .ASCIZ	/ INPUT MINUTES (OMIT LEADING ZERO)/
6641					.EVEN	
6642						
6643	047374				SFTEX4: XFER	SFTEX5
	047374	075004			.WORD	T\$CODE
6644						
6645	047376	103	110	101	CTPA: .ASCIZ	/CHANGE CONTROLLER PARAMETERS/
6646	047433	040	105	116	SERR: .ASCIZ	/ ENABLE CONTROLLER ERROR CORRECTION/
6647	047477	040	105	116	SERC: .ASCIZ	/ ENABLE CONTROLLER ERROR RECOVERY/
6648	047541	040	105	116	PADD: .ASCIZ	/ ENABLE PAD BLOCKING/
6649					.EVEN	
6650						
6651	047566				SFTEX5: XFER	SFTEX6
	047566	102004			.WORD	T\$CODE
6652						
6653	047570	103	110	101	PRPA: .ASCIZ	/CHANGE PRINTING PARAMETERS/
6654	047623	040	105	116	SOER: .ASCIZ	/ ENABLE SOFT ERROR REPORT PRINTING/
6655	047666	040	040	105	SRER: .ASCIZ	/ ENABLE READ SOFT ERRORS ONLY/
6656	047725	040	105	116	CLMD: .ASCIZ	/ ENABLE CLEAR MEDIA TABLE EVERY PASS/
6657					.EVEN	
6658						
6659	047772				SFTEX6: XFER	SFTEX7
	047772	063004			.WORD	T\$CODE
6660						
6661	047774	040	105	116	MDTB: .ASCIZ	/ ENABLE PRINTING OF MEDIA TABLE/
6662	050034	040	105	116	NOCL: .ASCIZ	/ ENABLE CLEAR STATS ON FATAL ERROR/
6663	050077	040	105	116	PDMP: .ASCIZ	/ ENABLE VARIABLES DUMP ON ERROR/
6664					.EVEN	
6665						
6666	050140				SFTEX7: XFER	SFTEX8
	050140	113004			.WORD	T\$CODE
6667						
6668	050142	103	110	101	TSPA: .ASCIZ	/CHANGE TEST PARAMETERS/

```

6669 050171    040    104    101 PATE:  .ASCIZ / DATA PATTERN/
6670 050207    040    122    125 T3ON:  .ASCIZ / RUN TEST 3 ONLY/
6671 050230    040    105    116 T5CP:  .ASCIZ / ENABLE DATA COMPARES IN TEST 5/
6672 050270    040    105    116 BDMP:  .ASCIZ / ENABLE PRINT READ BUFFER IN TEST 5/
6673 050334    040    103    110 CHGF:  .ASCIZ / CHANGE COMMAND SEQUENCE/
6674
6675
6676 050366          SFTEX8: XFER    SFTEX9
      050366 043004      .WORD    T$CODE
6677
6678 050370    040    040    104 DPAT:  .ASCIZ / DATA PATTERN/
6679 050407    040    040    111 ICNT:  .ASCIZ / ITEM COUNT (BYTE,RECORD,OBJECT)/
6680 050451    040    040    111 ITER:  .ASCIZ / ITERATION COUNT/
6681
6682
6683 050474          SFTEX9: XFER    SFTEXT
      050474 026004      .WORD    T$CODE
6684
6685 050476    103    115    104 CMD1:  .ASCIZ "CMD/1"
6686 050504    103    115    104 CMD2:  .ASCIZ "CMD/2"
6687 050512    103    115    104 CMD3:  .ASCIZ "CMD/3"
6688 050520    103    115    104 CMD4:  .ASCIZ "CMD/4"
6689 050526    103    115    104 CMD5:  .ASCIZ "CMD/5"
6690 050534    103    115    104 CMD6:  .ASCIZ "CMD/6"
6691 050542    103    115    104 CMD7:  .ASCIZ "CMD/7"
6692
6693
6694 050550          SFTEXT:
6695 050550          ENDSFT
          .EVEN
          050550          L10025:
6696
6703
6704 050550          ENDMOD
6705
6706 050550
6707 050560          RDBUF::  .BLKW   4
6708 070560          WRTBUF:: .BLKW  10000 ;READ BUFFER
6709
6710 110560          MEDIAS:: .BLKW  1296. ;MEDIA STATISTICS TABLE
6711
6712 115620          PATCH::  .BLKW   50 ;PATCH SPACE
6713
6714
6715 115740          LASTAD
          .EVEN
          115740 115754 .WORD T$FREE
          115742 000004 .WORD T$SIZE
          115744          L$LAST::
6716
6717 115744          BGNSETUP      1 ;NUMBER OF P-TABLES
6718 115744          BGNPTAB
          115744 000000 .WORD 0
          115746 000002 .WORD L10030-./2-1
          115750          L10026:
6719 115750 174500 .WORD 174500 ;IP ADDRESS
6720 115752 000000 .WORD 0 ;UNIT NUMBER
    
```

```
6721 115754                    ENDPTAB  
      115754                    L10030:  
6722 115754                    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	BYTES	003416	G	CNTER	011674	G	C\$GPRI=	000040	DROP	=	000010	G
ABOT	010662	G	CDRECV	023710	G	CNTERL	011172		C\$INIT=	000011	DRGPIT=	000400	G	
ACC	=	000040	CDRENO	010312	G	CNTFLG	010562	G	C\$INLP=	000020	DROPPD=	000104	G	
ACCESS	024522		CDREN1	010376	G	CNTMI	010554	G	C\$MANI=	000050	DROPUN	040450	G	
ACR	=	000041	CDREN2	010462	G	CNTLEL	013172		C\$MAP	=	000102	DRSPB0	004032	G
ADJUST	027146		CDREN3	010546	G	CNTLER	011132		C\$MEM	=	000031	DRSPB1	005072	G
ADR	=	000020	CDSRG0	010272	G	CNTT	010762	G	C\$MMU	=	000103	DRSPB2	006132	G
ALLPAT	=	000200	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	CF.ATN=	000200	G	CONID	=	177777	G	C\$PNTB=	000014	DRVERL	011162	
AVL	=	000130	CF.MSC=	000100	G	CONTPA	002217	G	C\$PNTF=	000017	DRVT	010772	G	
AVLB	=	000040	CF.OTH=	000040	G	CONTRV	020532	G	C\$PNTS=	000016	DSPSTP=	000004	G	
AVLER	011556		CF.THS=	000020	G	CONT1	046750		C\$PNTX=	000015	DSRNG0	010226	G	
AVLT	010702	G	CHGF	050334		CONT2	047144		C\$PUTB=	000072	DSRNG1	010312	G	
AVU	=	000134	CHGFLG	002237	G	CORDMP	033742		C\$PUTW=	000073	DSRNG2	010376	G	
BADEL	013233		CHODAT	024502		COREL	012627		C\$QIO	=	000377	DSRNG3	010462	G
BADER	011643		CLMD	047725		CORERL	011232		C\$RDBU=	000007	DSTEL	013103		
BADERL	011142		CLOCK	002212	G	COUNT	003404	G	C\$REFG=	000047	DTCPER=	000070	G	
BDMP	050270		CLREOT	033464	G	COUNTS	020202	G	C\$REL	=	000077	DUMP	020347	G
BIT0	=	000001	CLRMED	002226	G	CPRIEX	014350		C\$RESE=	000033	DUMPKT	003344	G	
BIT00	=	000001	CLSDRV	023444	G	CRD	=	177776	G	C\$REVI=	000004	DUMP1	020430	G
BIT01	=	000002	CMD	=	000000	CRDLIM	010571	G	C\$RFLA=	000021	DUMP2	020471	G	
BIT02	=	000004	CMDASC	011322	G	CTPA	047376		C\$RPT	=	000025	ECBEL	013001	
BIT03	=	000010	CMDBF1	003566	G	C\$AU	=	000052	C\$SEFG=	000046	ECBERL	011272		
BIT04	=	000020	CMDBF2	003636	G	C\$AUTO=	000061		C\$SPRI=	000041	ECCBC	=	000036	G
BIT05	=	000040	CMDBF3	003706	G	C\$BRK	=	000022	C\$SVEC=	000037	ECCDC	=	000026	G
BIT06	=	000100	CMDBF4	003756	G	C\$BSEG=	000004		C\$TOME=	000076	ECCFLG=	000100	G	
BIT07	=	000200	CMDBLD	022002	G	C\$BSUB=	000002		DATAEL	031436	ECCTC	=	000030	G
BIT08	=	000400	CMDCNT	003400	G	C\$CLCK=	000062		DATBL	003472	ECDEL	012663		
BIT09	=	001000	CMDER	011502		C\$CLEA=	000012		DATER	011630	ECDERL	011312		
BIT1	=	000002	CMDLST=	000001	G	C\$CLOS=	000035		DATPAT=	000001	ECLK	047216		
BIT10	=	002000	CMDONE=	000200	G	C\$CLP1=	000006		DATT	010742	ECTEL	012707		
BIT11	=	004000	CMDSEQ=	000006	G	C\$CPBF=	000074		DAYS	020125	ECTERL	011302		
BIT12	=	010000	CMDSSV=	000012	G	C\$CPME=	000075		DCBEND	004032	EC1COR=	000050	G	
BIT13	=	020000	CMDT	010652	G	C\$CVEC=	000036		DCBSTP=	000050	EC2COR=	000052	G	
BIT14	=	040000	CMDTBL	024366		C\$DCLN=	000044		DCB3SP=	000170	EDCEXT	031244		
BIT15	=	100000	CMD1	050476		C\$DODU=	000051		DCMDF	003572	EDLEXT	032042		
BIT2	=	000004	CMD2	050504		C\$DRPT=	000024		DCMPER	012507	EF.CON=	000036	G	
BIT3	=	000010	CMD3	050512		C\$DU	=	000053	DCMPT	011112	EF.EOT=	000010	G	
BIT4	=	000020	CMD4	050520		C\$EDIT=	000000		DEVERR	013272	EF.LOG=	000040	G	
BIT5	=	000040	CMD5	050526		C\$ERDF=	000055		DEVEXT	014400	EF.NEW=	000035	G	
BIT6	=	000100	CMD6	050534		C\$ERHR=	000056		DEVFAT=	000001	EF.PWR=	000034	G	
BIT7	=	000200	CMD7	050542		C\$ERRO=	000060		DFPTBL	002204	EF.RES=	000037	G	
BIT8	=	000400	CMLSER	012213		C\$ERSF=	000054		DIAGMC=	000000	EF.SEX=	000020	G	
BIT9	=	001000	CMMDSQ	021406	G	C\$ERSO=	000057		DMPBUF	002236	EF.STA=	000040	G	
BOE	=	000400	CMP	=	000030	C\$ESCA=	000010		DMPFLG	002231	ELTEXT	032170		
BOTER	011751		CMPDAT	032564	G	C\$ESEG=	000005		DPAT	050370	ENDPAT=	000010	G	
BOTT	011012	G	CMPEL	013030		C\$ESUB=	000003		DQCMD	027674	EOT	=	000004	G
BRCNT	003410	G	CMPER	012464		C\$ETST=	000001		DRBENO	005072	EOTBIT=	000002	G	
BUFADR	003422	G	CMPEP	011152		C\$EXIT=	000032		DRBEN1	006132	EOTPR	=	000010	G
BUFDMP	034074		CMPEP	011152		C\$FREQ=	000101		DRBEN2	007172	ERASE	024750		
BUFDSC	025616		CMPEP	011152		C\$FRME=	000100		DRBEN3	010232	ERASGP	025006		
BUFOFF	=	000012	CMPT	010732	G	C\$GETB=	000026				ERG	=	000120	G

PARAMETER CODING
Symbol table

ERI = 000113 G	EXC3A4 027366	G\$EXCP= 000400	IONORM= 000000 G	LUNG 002314 G
ERLGER 014502 G	EXC4A1 027410	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 040442	G\$NO = 000000	IOTIME= 000004 G	LUN3 003072 G
ERL02 017301 G	EXTINT 040340	G\$OFFS= 000400	ISR = 000100 G	L\$ACP 002110 G
ERL03 017365 G	EXTSRT 032526	G\$OFFSI= 000376	ISTART 037750	L\$APT 002036 G
ERL04 017454 G	EX3REW 044212	G\$PRMA= 000001	ITER 050451	L\$AU 040714 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 040376 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 040400 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 030316 G	FM.BAD= 000001 G	G\$RADO= 000020	IVST2 011032 G	L\$DESC 002142 G
ERRDEI 031006 G	FM.CNT= 000000 G	G\$XFER= 000004	IVST3 011102 G	L\$DESP 002076 G
ERRDEL 031250 G	FM.TPE= 000005 G	G\$YES = 000010	IXE = 004000 G	L\$DEVP 002060 G
ERREXT 031026	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 032050 G	F\$DU = 000016	HIADDR= 000002 G	I\$HRD = 000041	L\$DU 040706 G
ERRTYP 013262 G	F\$END = 000041	HIBYTE= 177777 G	I\$INIT= 000041	L\$DUT 002072 G
ERR00 015770 G	F\$HARD= 000004	HIHEX 003554 G	I\$MOD = 000041	L\$DVTY 002174 G
ERR01 016040 G	F\$HW = 000013	HNDLRP C03412 G	I\$MSG = 000041	L\$EF 002052 G
ERR02 016116 G	F\$INIT= 000006	HOE = 100000 G	I\$PROT= 000040	L\$ENVI 002044 G
ERR03 016175 G	F\$JMP = 000050	HOUR 047247	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 046244 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 037012 G
ERR15 017012 G	GCMDST 025400	IBE = 010000 G	LEDB = 000001 G	L\$LADP 002026 G
ERR16 017062 G	GCS = 000210 G	ICNT 050407	LEDER 012105	L\$LAST 115744 G
ERR17 017107 G	GCSCFL= 000020 G	IDONE 026506	LEDT 011072 G	L\$LOAD 002100 G
ERS = 000110 G	GCSEXT 027320	IDU = 000040 G	LEOTFL= 000030 G	L\$LUN 002074 G
ERTLY 031154	GCSHDL 026534 G	IER = 020000 G	LF.CON= 000100 G	L\$MREV 002050 G
ETLLY 032132 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	G0 = 000001 G	ILOOP 026350	LGPEL 013057	L\$PROT 021036 G
EV.COR= 000150 G	G01 041066	IMM = 000200 G	LGPERL 011202	L\$PRT 002112 G
EV.CTO= 000052 G	G02 043270	IMMBIT= 000003 G	LGSTAT 027732 G	L\$REPP 002062 G
EV.DST= 000050 G	G03 043726	INIT 025342	LINE 020527 G	L\$REV 002010 G
EV.HER= 000213 G	G04 044442	INITER 012400	LOBYTE= 177776 G	L\$RPT 034216 G
EV.IDS= 000152 G	G05 045022	INITIT 040344 G	LOE = 040000 G	L\$SOFT 046332 G
EV.LGP= 000010 G	G06 045342	INT = 000170 G	LOHEX 003552 G	L\$SPC 002056 G
EV.SER= 000153 G	G07 046020	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 026524	LOT = 000010 G	L\$STA 002030 G
EV.URE= 000350 G	G\$CNT0= 000200	IOERTB 010572 G	LSTENT= 000134 G	L\$SW 002212 G
EXC1A2 027322	G\$DELM= 000372	IOHUNG= 000002 G	LUNFLG= 000026 G	L\$TEST 002114 G
EXC2A3 027344	G\$DISP= 000003	IOICRD= 020000 G	LUNSTP= 000172 G	L\$TIML 002014 G

PARAMETER CODING
Symbol table

L\$UNIT	002012	G	L10026	115750	OFLT	010672	G	PATGN7	022520	P.MXWR=	000044	G
L.BADR=	000030	G	L10030	115754	OLDBLK=	000152	G	PATSAV=	000024	P.NREC=	000050	G
L.CHVR=	000025	G	MAXBUF=	020000	OLDTRK=	000154	G	PATTBL	022344	P.OPCD=	000010	G
L.CNTI=	000014	G	MAXITR=	003720	OLD1	026652		PAT1	= 000001	P.OTRF=	000014	G
L.CNT0=	000061	G	MDTB	047774	OLD2	026732		PAT2	= 000002	P.POS =	000034	G
L.CNT1=	000062	G	MD.ALL=	000002	OLD3	027012		PAT3	= 000003	P.RCSK=	000014	G
L.CNT2=	000063	G	MD.CMP=	040000	OLD4	027072		PAT4	= 000004	P.REDD=	000014	G
L.CRF =	000000	G	MD.CSE=	020000	ONE	= 000001	G	PAT5	= 000005	P.SPED=	000042	G
L.CSVR=	000024	G	MD.DLE=	000200	ONEFIL=	000001		PAT6	= 000006	P.STS =	000012	G
L.DFLG=	000054	G	MD.EXC=	000040	ONL	= 000140	G	PAT7	= 000007	P.TIME=	000024	G
L.DRVC=	000053	G	MD.IMM=	000100	ONLB	= 000100	G	PCBEND=	003344	P.TMGC=	000020	G
L.DRVS=	000064	G	MD.NXU=	000001	ONLINE	025064		PCBSTP=	000014	P.TMSK=	000020	G
L.EVNT=	000012	G	MD.OBC=	000004	OP.ABO=	000001	G	PCB3SP=	000044	P.TRBC=	000040	G
L.FHVR=	000050	G	MD.REV=	000010	OP.ACC=	000020	G	PCFLAG	010516	P.UHVR=	000053	G
L.FLGS=	000011	G	MD.RWD=	000002	OP.ACP=	000102	G	PKSIZ	010564	P.UNFL=	000016	G
L.FMT =	000010	G	MD.SEC=	001000	OP.AVA=	000100	G	PCMDBF	003264	P.UNIT=	000004	G
L.FMTD=	000042	G	MD.SER=	000400	OP.AVL=	000010	G	PDMP	050077	P.UNTI=	000024	G
L.FSVR=	000051	G	MD.SPD=	000001	OP.CMP=	000040	G	PDRECV	026106	P.USVR=	000052	G
L.GPCT=	000044	G	MD.SWP=	000004	OP.DAP=	000013	G	PHB	= 000002	P.VRSN=	000014	G
L.LBLK=	000060	G	MD.UNL=	000020	OP.END=	000200	G	PKPRNT	015476	QCMD	023050	G
L.LVL =	000042	G	MEDIAS	110560	OP.ERG=	000026	G	PNT	= 001000	RANGEN	022716	G
L.MLUN=	000026	G	MEDTBL	002227	OP.ERS=	000022	G	POLER	012045	RANWRD	003426	G
L.OPFL=	000070	G	MINBUF=	000024	OP.GCS=	000002	G	POLT	011052	RAN1	003430	G
L.PBLK=	000056	G	MINITR=	000144	OP.GUS=	000003	G	PORTER	012266	RAN2	003432	G
L.PSTN=	000044	G	MINLIM	010570	OP.ONL=	000011	G	PRI	= 002000	RAN3	003434	G
L.RTRY=	000043	G	MINT	047331	OP.RD =	000041	G	PRIERR	032550	RD	= 000010	G
L.RWST=	000066	G	MINUTE	002214	OP.REP=	000045	G	PRIPCK	014136	RDBUF	050560	G
L.SEQN=	000006	G	MISSEQ=	000005	OP.SCC=	000004	G	PRI00	= 000000	RDBYT1=	000110	G
L.STI =	000050	G	MSCPVR=	000000	OP.SUC=	000012	G	PRI01	= 000040	RDBYT2=	000112	G
L.STS =	000052	G	MSGEXT	015670	OP.WR =	000042	G	PRI02	= 000100	RDBYT3=	000114	G
L.TRK =	000055	G	MSGLEN=	177774	OP.WTM=	000044	G	PRI03	= 000140	RDBYT4=	000116	G
L.UHVR=	000041	G	MSKTST	030774	OTHER =	000102	G	PRI04	= 000200	RDERL	031572	
L.UNIT=	000004	G	MTBLOV=	000040	OVERUN	012557		PRI05	= 000240	RDR	= 000011	G
L.UNTI=	000030	G	N	= 000004	OVRUN=	000074	G	PRI06	= 000300	RDRENO	010272	G
L.USVR=	000040	G	NCLK	017770	OWN	= 100000	G	PRI07	= 000340	RDREN1	010356	G
L.VSER=	000044	G	NCLKFL=	000002	O\$APTS=	000000		PRNTPA	002223	RDREN2	010442	G
L10000	002210		NOCL	050034	O\$AU =	000000		PRPA	047570	RDREN3	010526	G
L10001	002314		NOCLK	021044	O\$BGNR=	000001		PRTCLR	026230	RDSOER	002225	G
L10002	014500		NOCLR	002230	O\$BGNS=	000001		PRTDRV	025730	RDSRG0	010232	G
L10003	021034		NOERR =	000106	O\$DU =	000001		PRTXT	031736	RDSRG1	010316	G
L10005	021076		NOMDTB=	000100	O\$ERRT=	000001		PRTINT	026306	RDSRG2	010402	G
L10006	021236		NOTALY=	000004	O\$GNSW=	000001		P.BCNT=	000014	RDSRG3	010466	G
L10007	037010		NRDY =	000002	O\$POIN=	000001		P.BUFF=	000020	RDSTEL	011212	
L10010	040374		NUL =	000000	O\$SETU=	000001		P.CHVR=	000051	RDTB	= 000002	G
L10011	040376		NULL	024436	PADD	047541		P.CMST=	000020	RDTER	012017	
L10012	040446		NULPAT=	000000	PADING	002222	G	P.CNTF=	000016	RDTT	011042	G
L10013	040712		NUPASS	037210	PASCNT	003522	G	P.CRF =	000000	READ	024442	
L10014	040720		NURESP=	100000	PATCH	115620	G	P.CSVR=	000050	RECCNT	003546	G
L10015	043122		OBCTHD	023276	PATCLR	033714		P.CTPM=	000034	RESP	003414	G
L10016	043616		OBJECT	003520	PATE	050171		P.DVPM=	000034	RESPON	003406	G
L10017	044274		OBJFDH=	000036	PATERN	002233	G	P.FLGS=	000011	REVBIT=	000001	G
L10020	044654		OBJFDL=	000034	PATGN1	022362		P.FMEM=	000044	REW	= 000160	G
L10021	045174		OBOFFH=	000006	PATGN2	022376		P.FORM=	000040	REWIND	025254	
L10022	045652		OBOFFL=	000004	PATGN3	022410		P.HTMO=	000020	RLSER	012421	
L10023	046240		OCTHEX	033376	PATGN4	022450		P.MEDI=	000034	RLST	011122	G
L10024	046330		ODDFLG=	000020	PATGN5	022464		P.MLUN=	000014	RMSPOS=	000076	G
L10025	050550		OFLER	011541	PATGN6	022504		P.MOD =	000012	RNDBYT=	000000	G

RNDITR= 000000 G	SFTEX4 047374	ST.CMD= 000001 G	TKINIT= 111400 G	T\$\$SRV= 010006
RNUSAV= 000020 G	SFTEX5 047566	ST.CMP= 000007 G	TKIP = 000000 G	T\$\$SW = 010001
ROLSAV= 000022 G	SFTEX6 047772	ST.CNT= 000012 G	TKIPAD 046270	T\$\$TES= 010023
RSPBFO 004026 G	SFTEX7 050140	ST.DAT= 000010 G	TKSA = 000002 G	T1 040722 G
RSPBF1 005066 G	SFTEX8 050366	ST.DIA= 000037 G	TKUNIT= 000004 G	T1EXIT 042632
RSPBF2 006126 G	SFTEX9 050474	ST.DRV= 000013 G	TKUNT 046305	T1LEOT 042652
RSPBF3 007166 G	SKD = 000062 G	ST.FNT= 000014 G	TMB = 000010 G	T1ONL 042636
RSPCNT 003402 G	SKP = 000060 G	ST.HST= 000011 G	TMCNT 003550 G	T1RD1 043056
RSPHDL 027432 G	SKPTMK 024610	ST.LED= 000023 G	TMER 011771	T1RD2 043064
RS1 = 001233 G	SKR = 000061 G	ST.MFE= 000005 G	TMT 011022 G	T1RD3 043072
RS2 = 007622 G	SLTUSE= 000010 G	ST.MSK= 000037 G	TRK = 000000 G	T1RD4 043100
RS3 = 000000 G	SOER 047623	ST.OFL= 000003 G	TSPA 050142	T1RD5 043106
RTYEC1= 000064 G	SOERRP 002224 G	ST.ONL= 000400 G	TSTMSK 003440 G	T1RD6 043114
RTYEC2= 000066 G	SOFT = 000003 G	ST.POL= 000021 G	TSTSTP= 000006 G	T1REW 042644
RTYEL 012574	SPC = 000050 G	ST.RDT= 000020 G	T\$ARGC= 000004	T1SKD 043004
RTYERL 011252	SPCASC 011346	ST.SEX= 000022 G	T\$CODE= 026004	T1SKP 042666
RWI = 000163 G	SPCOBJ 024664	ST.SUB= 000040 G	T\$ERRN= 000000	T1SKP1 042762
R10 0.3534 G	SPCREC 024542	ST.SUC= 000000 G	T\$EXCP= 000000	T1SKP2 042770
R11 003536 G	SPO = 000070 G	ST.T:I = 000016 G	T\$FLAG= 000041	T1SKR 042674
R12 003540 G	SPR = 000071 G	ST.WPR= 000006 G	T\$FREE= 115754	T1SKR1 042732
R3SAVE 003560 G	SRD = 000005 G	SUBCNT 003424 G	T\$GMAN= 000000	T1SPC1 042702
R4SAVE 003562 G	SRER 047666	SUBITR 023142 G	T\$HILI= 065000	T1SPC2 042710
R8 003530 G	SRTBL 031732	SUBSEC 002216 G	T\$LAST= 000001	T1SPC3 042716
R9 003532 G	START 037052	SUC = 000150 G	T\$LOLI= 000000	T1SP01 042724
SAERR 010566 G	START1 040760	SUCCE= 000001 G	T\$LSYM= 010000	T1SP02 042740
SAVDIF 003436 G	START2 043162	SUNCHR 025162	T\$LTNO= 000007	T1SP03 042746
SCC = 000230 G	START3 043620	SUPRES 025650	T\$NEST= 177777	T1SPR1 042754
SCD = 000052 G	START4 044334	SUW = 000155 G	T\$NSO = 000000	T1SPR2 042776
SCHED 021240 G	START5 044714	SVCGBL= 000000	T\$NS1 = 000005	T1WR1 043012
SCNTCH 025466	START6 045234	SVCINS= 000000	T\$PCNT= 000000	T1WR2 043020
SCR = 000051 G	START7 045712	SVCSUB= 000000	T\$PTAB= 010027	T1WR3 043026
SDSAVE 033640 G	STATER 012445	SVCTAG= 000000	T\$PTHV= 000001	T1WR4 043034
SDSTUP 033564 G	STATUS= 000004 G	SVCTST= 000000	T\$PTNU= 000001	T1WR5 043042
SECOND 002215 G	STAT0 035472	SWR = 000004 G	T\$SAVL= 177777	T1WR6 043050
SECRNS 003542 G	STAT1 035475	SYSFAT= 000000 G	T\$SEGL= 177777	T1WTM 042660
SED1 = 000136 G	STAT10 036230	S\$LSYM= 010000	T\$SIZE= 000004	T2 043124 G
SED2 = 000140 G	STAT11 036300	S1 = 004000 G	T\$SUBN= 000000	T2END = 004716 G
SED3 = 000142 G	STAT12 036352	S1READ= 000040 G	T\$TAGL= 177777	T2EXIT 043542
SEED1 = 000144 G	STAT13 036413	S1WRIT= 000044 G	T\$TAGN= 010031	T2LEOT 043566
SEED2 = 000146 G	STAT14 036436	S2READ= 000042 G	T\$TEMP= 000000	T2RD 043574
SEED3 = 000150 G	STAT15 036461	S2WRIT= 000046 G	T\$TEST= 000007	T?REW 043552
SELDAT 022212 G	STAT16 036533	TBLBTM= 000132 G	T\$TSTM= 177777	T2SKD 043602
SELREC 022644 G	STAT17 036605	TBLEND= 003472 G	T\$TSTS= 000001	T2SPO 043610
SEQR 012347 G	STAT18 036655	TBLFUL 036724 G	T\$AU = 010014	T2WRT 043560
SERC 047477	STAT2 035541	TBLOFF 003544 G	T\$AUT= 010011	T3 043620 G
SERCCR 002221 G	STAT3 035606	TBLSRT 032174 G	T\$CLE= 010012	T3EXIT 044226
SEREXC= 000002 G	STAT4 035666	TBLTOP= 000130 G	T\$DAT= 010030	T3ON 050207
SERR 047433	STAT5 035713	TCNTFL= 000004 G	T\$DU = 010013	T3ONLY 002234 G
SERREC 002220 G	STAT6 035760	TESTPA 002232 G	T\$HAR= 010024	T3RD 044260
SEXB = 000004 G	STAT7 036032	TF.BLK= 000010 G	T\$HW = 010000	T3REW 044236
SEXER 012063	STAT8 036077	TF.GCR= 000004 G	T\$INI= 010010	T3SPO 044266
SEXT 011062 G	STAT9 036151	TF.PE = 000002 G	T\$MSG= 010003	T3WRT 044244
SFPTBL 002212 G	STFPCK 024164 G	TF.800= 000001 G	T\$PC = 000001	T3WTM 044252
SFTEXT 050550	STINIT 037012 G	TIME 020044 G	T\$PRO= 010004	T4 044276 G
SFTEX1 046746	ST.ABO= 000002 G	TIMER 010556 G	T\$PTA= 010027	T4EXIT 044626
SFTEX2 047142	ST.AVL= 000004 G	TIMERR 012312	T\$RPT= 010007	T4REW 044632
SFTEX3 047214	ST.BOT= 000015 G	TKERR 026500	T\$SOF= 010025	T4WRT 044640

T4WTM	044646	T7CMD2	002246	UF.RMV=	000200 G	URDSRG=	000162 G	WR	=	000020 G
T5	044656 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	050230	T7CMD5	002270	UF.WPH=	020000 G	URSPBF=	000156 G	WRBYT3=	000124 G	
T5EXIT	045154	T7CMD6	002276	UF.WPS=	010000 G	UWEEL	012760	WRBYT4=	000126 G	
T5RD	045166	T7CMD7	002304	UNDROP=	000032 G	UWEERL	011242	WRERL	031516	
T5REW	045160	T7END	002312	UNDRUN=	000072 G	U1OFF =	005040 G	WRITE	024462	
T6	045176 G	T7EXIT	046230	UNITRV	020662 G	U2OFF =	002420 G	WRKMSK	003442 G	
T6EXIT	045604	T7TBL	002232 G	UNJAM	033250 G	U3OFF =	001540 G	WRTBUF	070560 G	
T6RD	045636	UAM	= 000200 G	UNLBIT=	000004 G	U4OFF =	001210 G	WTAPMK	024722	
T6REW	045614	UCDEND=	000170 G	UNTEOT	020256 G	WDSTEL	011222	WTM	= 000100 G	
T6SKD	045644	UCDSRG=	000166 G	UNTLOT	020312 G	WMSPOS=	000100 G	XFERST=	000010 G	
T6WRT	045622	UDRNER	012541	UNTSTP=	000002 G	WPRB	= 000020 G	X\$ALWA=	000000	
T6WTM	045630	UDROP	003524 G	UNTTBL	003360 G	WPRBIT=	000005 G	X\$FALS=	000040	
T7	045654 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. 115754 000 (RW,I,GBL,ABS,OVR)
000000 001 (RW,I,LCL,REL,CON)

Errors detected: 0

*** Assembler statistics

Work file reads: 335
Work file writes: 332
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: 00:10:37.33
CZTKBB.BIN,CZTKBB.LST/-SP=SVC40R.MLB/ML,CZTKBB