

RK611  
RK06, RK07

UNIBUS RK6 DR-3  
CZR6JF0

AH-9126F-MC  
FICHE 1 OF 1

APR 1982  
COPYRIGHT © 76-82  
MADE IN USA



A large grid of data tables, likely a technical specification or test results document. The grid consists of approximately 15 columns and 15 rows of smaller tables. Each cell in the grid contains a small table with multiple columns and rows of text, possibly representing test parameters, results, or component specifications. The text is too small to read clearly but appears to be organized in a structured, tabular format.

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

.REM %

IDENTIFICATION

PRODUCT CODE:	AC-9125F-MC
PRODUCT NAME:	CZR6JFO UNIBUS RK6 DR-3
DATE:	JANUARY 1982
MAINTAINER:	STORAGE SYSTEMS SOFTWARE TEST APPLICATION
AUTHOR:	B. 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.

THE SOFTWARE DESCRIBED IN THIS DOCUMENT IS FURNISHED UNDER A LICENSE AND MAY ONLY BE USED OR COPIED IN ACCORDANCE WITH THE TERMS OF SUCH LICENSE.

DIGITAL EQUIPMENT CORPORATION ASSUMES NO RESPONSIBILITY FOR THE USE OR RELIABILITY OF ITS SOFTWARE ON EQUIPMENT THAT IS NOT SUPPLIED BY DIGITAL.

COPYRIGHT (C) 1976,1982 BY DIGITAL EQUIPMENT CORPORATION

42  
43  
44

REVISION

CHANGES

DATE

CZR6JF0

IMPLEMENTED XXDP LOAD MEDIA OPTION

JAN 82

45  
46  
47  
48  
49  
50  
51  
52  
53  
54  
55  
56  
57  
58  
59  
60  
61  
62  
63  
64  
65  
66  
67  
68  
69  
70  
71  
72  
73  
74  
75  
76  
77  
78  
79  
80  
81  
82  
83  
84  
85  
86  
87  
88  
89  
90

TABLE OF CONTENTS

1.0	ABSTRACT
2.0	REQUIREMENTS
2.1	HARDWARE
2.2	PRELIMINARY TESTING AND PROGRAMS
3.0	PROGRAM CONSIDERATIONS
3.1	PDP-11 FAMILY COMPATIBILITY
3.2	XXDP
3.3	ACT/APT
3.3.1	APT ETABLE DEFINITIONS
3.4	DUAL ACCESS
3.5	MEMORY MANAGEMENT
3.6	PARITY CHECK ENABLED
3.7	BAD SECTORS
3.8	EXECUTION TIME
3.9	FAULT ISOLATION
3.10	ERROR CORRECTION & FAILURE RATE ANALYSIS
3.11	DEFAULT UNIBUS ADDRESSES & VECTORS
4.0	OPERATING PROCEDURE & CONTROL FUNCTIONS
4.1	PROGRAM LOADING
4.2	STARTING LOCATIONS
4.3	CONSOLE SWITCH REGISTERS
4.4	SOFTWARE SWITCH REGISTER
4.5	INPUT DIALOGUE
4.6	PROGRAM EXAMPLE
4.7	HALTING THE PROGRAM
5.0	DRIVE DIAGNOSTIC FUNCTIONAL DESCRIPTION
5.1	GENERAL
5.2	TEST DESCRIPTIONS
6.0	ERROR REPORTING
6.1	ERROR INTERPRETATION
6.2	ERROR PRINTOUT EXAMPLE

91  
92  
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

1.0 ABSTRACT

THIS PROGRAM PERFORMS PART 3 OF THE DRIVE DIAGNOSTICS TO INSURE THAT THE DISK IS CAPABLE OF PROPERLY PERFORMING ALL OPERATOR INTERVENTION FUNCTIONS. ERROR DETECTION LOGIC IS CHECKED BY MANUAL & SOFTWARE ERROR FORCING.

AFTER A SUCCESSFUL RUN (WITH NO ERRORS) OF THIS PART, PRECEDED BY THE SUCCESSFUL RUN OF PARTS 1 & 2, IT CAN BE ASSERTED THAT THE RK06 DRIVE WILL WORK SUCCESSFULLY IN THE STAND-ALONE MODE. SYSTEMS INTERACTION, & ERROR RATE ANALYSIS ARE LEFT TO OTHER PROGRAMS.

TESTING IS BASED ON A HIERARCHY APPROACH STARTING WITH BASIC LOGIC TESTS AND PROCEEDING THRU DYNAMIC TESTING. THE TESTS WILL BE KEPT SMALL TO FACILITATE SCOPING LOOPS. THIS PROGRAM WILL TEST RK06 & RK07 WITHOUT NEED OF OPERATOR INPUTS.

\*\*\*\*\*CAUTION\*\*\*\*\*

HALTING THIS PROGRAM ANYWHERE BUT AT THE END OF A PASS, MAY LEAVE THE HEADERS IN THE DISK CARTRIDGE IN AN UNDETERMINED STATE.

2.0 REQUIREMENTS

2.1 HARDWARE

THE FOLLOWING HARDWARE IS REQUIRED TO RUN THE DISK DIAGNOSTIC:

PDP-11  
CONSOLE TELETYPE  
16K MEMORY  
KW11-L OR KW11-P CLOCK  
RK06 UNIBUS CONTROLLER (RK611)  
1 TO 8 RK06/RK07 DRIVES

- NOTES: 1. IF NEITHER KW11-L OR P CLOCK IS USED, ALL TIMING TESTS WILL BE BYPASSED. A MESSAGE AT THE BEGINNING OF THE TESTS WILL CONFIRM THIS.
2. THE PROGRAM CAN WORK OFF EITHER FORMATTED OR NON-FORMATTED PACKS.

2.2 PRELIMINARY TESTING & PROGRAMS

THE RK611 DISKLESS CONTROLLER DIAGNOSTICS (ALL PARTS) SHOULD FIRST RUN SUCCESSFU FOLLOWED BY THE RK06 DRIVE DIAGNOSTICS - PARTS 1 & 2.

3.0 PROGRAM CONSIDERATIONS

147  
148  
149  
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

3.1 PDP-11 FAMILY COMPATIBILITY

THIS PROGRAM CAN BE USED BY THE PDP-11/04,05,10,20,  
34,35,40,45,50, & 70.

IT IS COMPATABLE WITH THE LSI-11 INSTRUCTION SET AND CAN TEST  
THE RK06 ONLY IF THE DRIVE CONTROLLER FOR THE LSI-11 IS  
DESIGNED TO BE DIAGNOSTICALLY COMPATABLE WITH THE RK611.

3.2 XXDP

THIS PROGRAM SHOULD NOT BE CHAINED BY XXDP.

CHAIN MODE OPERATION (MONITOR)

BY DEFINITION, ANY PROGRAM THAT REQUIRES  
OPERATOR INTERVENTION SHOULD NOT BE CHAINED.

DUMP MODE OPERATION (MANUAL)

1. INPUT DIALOGUE IF STARTED FROM 220.
2. DRIVE 0 CAN BE TESTED, BUT THE OPERATOR IS FIRST GIVEN  
A MESSAGE TO REPLACE THE PACK IN DRO WITH A SCRATCH  
PACK & TYPE <CR> WHEN DONE.

3.3 ACT/APT

THIS PROGRAM IS ACT COMPATIBLE.  
HOWEVER, IT SHOULD NOT BE RUN IN THE AUTO MODE.

AUTOMATIC MODE (MONITOR)

BY DEFINITION, ANY PROGRAM THAT REQUIRES  
OPERATOR INTERVENTION SHOULD NOT BE  
RUN IN THE AUTO MODE.

DUMP MODE (MANUAL): INPUT DIALOGUE IF STARTED FROM 220.

3.3.1 APT ETABLE DEFINITIONS

THE FOLLOWING DEFINITIONS ARE VALID FOR SPECIFYING APT ENVIRONMENTAL  
TABLE (ETABLE) ENTRIES, VIA RUNNING THE APT UTILITY PROGRAM 'TSP':

1. SOFTWARE ENVIRONMENT:  
=1 IF APT SCRIPT MODE  
=0 IF STANDALONE MODE
2. ENVIRONMENT MODE:  
BIT 7 = 1 ETABLE DOES SIZING  
= 0 PROGRAM DOES SIZING  
BIT 6 = 1 SPOOL MESSAGES TO APT IF SCRIPT MODE  
= 0 DON'T SPOOL TO APT

203  
204  
205  
206  
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  
238  
239  
240  
241  
242  
243  
244  
245  
246  
247  
248  
249  
250  
251  
252  
253  
254  
255  
256  
257  
258

BIT 5 = 1 SUPPRESS CONSOLE OUTPUT  
= 0 ALLOW CONSOLE OUTPUT  
BITS 4-0 NOT USED

3. SWITCH i (SOFTWARE SWITCH REGISTER)  
IF ENVIRONMENT MODE BIT 7 (SIZING BIT) IS SET TO 1,  
THE SOFTWARE SWITCH REGISTER WILL BE USED, INSTEAD  
OF THE HARDWARE CONSOLE SWITCH REGISTER. REGARDLESS  
OF WHICH ONE IS USED, ALL BITS DEFINED IN SECTIONS  
4.3 & 4.4 (SWITCH REGISTER OPTIONS) MAY USED  
WHEN RUNNING IN STANDALONE MODE. IN APT SCRIPT MODE,  
HOWEVER, BIT 14 (LOOP ON TEST) MUST ALWAYS BE SET  
TO 0.

4. SWITCH 2 (USER SWITCH REGISTER)  
NOT USED

5. CPU OPTIONS:  
NOT USED

6. MEMORY TYPES 1-4 AND MAX MEMORY ADDRESSES  
NOT USED

7. INTERRUPT VECTOR 1:  
USED WHEN ENVIRONMENT MODE BIT 7 = 1. DEFAULT = 210

8. BUS PRIORITY 1:  
USED WHEN ENVIRONMENT MODE BIT 7 = 1. DEFAULT = 5

9. INTERRUPT VECTOR 2:  
NOT USED

10. BUS PRIORITY 2:  
NOT USED

11. BASE ADDRESS:  
USED WHEN ENVIRONMENT MODE BIT 7 = 1. DEFAULT = 177440

12. DEVICE MAP:  
USED WHEN ENVIRONMENT MODE BIT 7 = 1. EACH BIT  
SET TO 1 IN BITS 0-7 WILL SELECT THE CORRESPONDING  
DRIVE TO BE TESTED. BITS 8-15 ARE NOT USED.

13. CONTROLLER DESCRIPTOR WORDS:  
NOT USED

14. DEVICE DESCRIPTOR CODES (IN WORDS):  
NOT USED

### 3.4 DUAL ACCESS

THIS PROGRAM WILL NOT TEST OR SUPPORT DUAL-ACCESS. A DRIVE  
EQUIPED WITH DUAL ACCESS MUST BE SWITCHED TO THE PORT UNDER  
TEST TO PREVENT CONTENTION WITH THE OTHER PORT.

DUAL ACCESS TESTS WILL BE INCORPORATED IN A SEPARATE PROGRAM

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  
296  
297  
298  
299  
300  
301  
302  
303  
304  
305  
306  
307  
308  
309  
310  
311  
312  
313  
314

AT A LATER DATE.

3.5 MEMORY MANAGEMENT

MEMORY MANAGEMENT NOT USED

3.6 PARITY CHECK ENABLED

IF THE MEMORY PARITY CHECK OPTION IS AVAILABLE ON THE SYSTEM,  
THE PROGRAM WILL RUN WITH MEMORY CHECK ENABLED.

3.7 BAD SECTOR

THE PROGRAM WILL COMPARE DATA ERRORS WITH THE BAD SECTOR  
INFORMATION CONTAINED ON CYLINDER 410, HEAD 2. PRINTOUTS  
OF DATA ERRORS DUE TO BAD SECTORS/TRACKS WILL BE MASKED OUT.

3.8 EXECUTION TIME

TOTAL TIME: APPROX 5 MINUTES TO DO ALL THE TESTS  
(BASED ON THE PDP 11/50)

3.9 FAULT ISOLATION

TO BE DETERMINED.

3.10 ERROR CORRECTION AND FAILURE RATE ANALYSIS

THIS PROGRAM WILL NOT DO ERROR CORRECTION OR FAILURE RATE  
ANALYSIS.

3.11 DEFAULT UNIBUS ADDRESSES & VECTORS

THE FOLLOWING IS A LIST OF ALL DEFAULT ADDRESSES & VECTORS  
OF ALL HARDWARE TO BE USED & THEIR MEMORY ADDRESSES  
WHERE THEY CAN BE CHANGED.

	LOCATION	DEFAULT CONTENTS
RK06 BUS ADDRESS	1264	177440
CONTROLLER INTERRUPT VECTOR	1334	210
CONTROLLER PRIORITY	1336	240
P-CLOCK STATUS REG	1340	172540
P-CLOCK SET BUFFER	1342	172542
P-CLOCK READ BUFFER	1344	172544
L-CLOCK STATUS REG	1346	177546
L-CLOCK INTERRUPT VECTOR	1350	100
P-CLOCK INTERRUPT VECTOR	1352	104
TTY KB STATUS REG	1144	177560
TTY KB BUFFER	1146	177562



315 TTY PRINTER STATUS REG 1150 177564  
316 TTY PRINTER BUFFER 1152 177566  
317  
318  
319 4.0 OPERATING PROCEDURE & CONTROL FUNCTIONS  
320  
321  
322 4.1 PROGRAM LOADING  
323  
324 THE PROGRAM CAN BE LOADED FROM PAPER TAPE USING STANDARD  
325 PROCEDURE FOR ABSOLUTE LOADER TAPES; OR FROM ANY MEDIA  
326 SUPPORTED BY XXDP.  
327  
328  
329 4.1.1 LOAD THE STARTING ADDRESS (SEE SEC 4.2).  
330  
331  
332 4.1.2 SET SWITCH REGISTERS AS DESIRED (SEE SEC 4.3).  
333  
334  
335 4.1.3 SET DRIVES TO BE TESTED IN THE 'LOAD' CONDITION & WITH THE  
336 APPROPRIATE PORT SELECTED & WRITE LOCK DISABLED. DRIVES  
337 NOT TO BE TESTED MUST HAVE BOTH PORTS DESELECTED.  
338  
339 NOTE: THE DRIVE WILL NOT RESPOND TO THE 'START SPINDLE'  
340 COMMAND IF THE RUN/STOP SWITCH IS IN THE 'STOP'  
341 POSITION.  
342  
343  
344 4.1.4 PRESS 'START'  
345  
346 THE PROGRAM WILL IDENTIFY ITSELF AND WILL BEGIN A  
347 DIALOGUE WITH THE OPERATOR TO DETERMINE DRIVES TO BE TESTED  
348 (SEE SEC 4.5).  
349  
350 THE PROGRAM BEGINS TESTING ONLY THOSE DRIVES SPECIFIED BY  
351 THE INPUT DIALOGUE. IF A SPECIFIED DRIVE CANNOT BE FOUND BY  
352 THE PROGRAM IT WILL BE FLAGGED AS AN ERROR THAT THE DRIVE  
353 WAS NOT AVAILABLE. THEN BEGINNING WITH THE LOWEST NUMERICAL  
354 DRIVE AND PROCEEDING IN SEQUENTIAL ORDER, ALL VALID DRIVES  
355 WILL BE TESTED. ONE PASS THROUGH THE TEST SEQUENCE WILL BE  
356 PERFORMED ON EACH DRIVE BEFORE MOVING TO THE NEXT DRIVE  
357 IN SEQUENCE. THE DRIVE TO BE TESTED WILL BE TYPED AT THE  
358 BEGINNING OF EACH PASS. 'END OF PASS' WILL BE TYPED AFTER  
359 TESTING ALL DRIVES.  
360  
361  
362  
363  
364 4.2 STARTING LOCATIONS  
365  
366 LOCATION 200 - STARTING ADDRESS TO DEFAULT THE BUSS  
367 ADDRESS & THE CONTROLLER INTERRUPT VECTOR  
368 & TEST ALL DRIVES IN THE 'DRIVE PRESENT'  
369 CONDITION.  
370 NOTE: THE DRIVE PRESENT CONDITION IS:

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  
410  
411  
412  
413  
414  
415  
416  
417  
418  
419  
420  
421  
422  
423  
424  
425  
426

- A. HEADS MANUALLY LOADED
- B. CORRECT PORT SELECTED
- C. WRITE LOCK DISABLED
- D. DRIVE READY INDICATOR ON

LOCATION 220 - STARTING ADDRESS TO INPUT TESTING PARAMETERS VIA THE INPUT DIALOGUE. BUSS ADDRESS & CONT. INTERRUPT VECTOR INPUTTED ONLY ON 1ST PASS.

IMPORTANT: FOR VARIATIONS OF THE ABOVE, SEE XXDP, ACT/APT CONSIDERATIONS IN SECTIONS 3.2 & 3.3.

ALSO, THE PROGRAM WILL DETERMINE IF THE DRIVE IS AN RK06 OR RK07 WITHOUT OPERATOR INPUT.

### 4.3 SWITCH REGISTER

THE SWITCHES ARE USED TO PROVIDE CONTROL FUNCTIONS.

SWITCH	FUNCTION
15	HALT ON ERROR
14	LOOP ON TEST
13	INHIBIT ERROR TYPEOUT
12	BYPASS DRIVE AFTER 20 ERRORS
11	INHIBIT ITERATION
10	BELL ON ERROR
9	LOOP ON ERROR

#### 4.3.1 SW<15>

THE PROGRAM HALTS ON ENCOUNTERING AN ERROR, AFTER TYPING OUT THE ERROR MESSAGE AND PERTINENT INFORMATION. PRESSING "CONTINUE" CONTINUES NORMAL OPERATION OF THE PROGRAM.

#### 4.3.2 SW<14>

THE PROGRAM LOOPS ON THE TEST THAT IS BEING EXECUTED WHEN THE SWITCH IS PUT ON. THIS SWITCH IS NORMALLY USED ALONG WITH SW15.

#### 4.3.3 SW<13>

THIS SWITCH INHIBITS ALL ERROR MESSAGES. NORMALLY USED WHEN LOOPING ON TEST (SW14) OR LOOPING ON ERROR (SW9).

#### 4.3.4 SW<12>

427  
428  
429  
430  
431  
432  
433  
434  
435  
436  
437  
438  
439  
440  
441  
442  
443  
444  
445  
446  
447  
448  
449  
450  
451  
452  
453  
454  
455  
456  
457  
458  
459  
460  
461  
462  
463  
464  
465  
466  
467  
468  
469  
470  
471  
472  
473  
474  
475  
476  
477  
478  
479  
480  
481  
482

THIS SWITCH BYPASSES A GIVEN DRIVE AFTER 20 ERRORS HAVE BEEN DETECTED.

4.3.5 SW<11>

EACH TEST WILL BE EXECUTED ONLY ONCE. NORMALLY AFTER THE FIRST PASS, EACH SUBTEST IS ITERATED A NUMBER OF TIMES (USUALLY 50, 5 IN SOME CASES). SETTING THIS SWITCH INHIBITS ITERATIONS, SO THAT QUICK PASSES CAN BE MADE.

4.3.6 SW<10>

RINGS A BELL ON ERROR. USEFUL WHEN ERROR TYPEOUT IS INHIBITED.

4.3.7 SW<09>

THIS SWITCH PROVIDES THE TIGHTEST POSSIBLE SCOPE LOOP FOR ERRORS. IF THE PROGRAM DETECTS AN ERROR, IT WILL LOOP BACK TO THE BEGINNING OF TEST.

4.4 'SOFTWARE' SWITCH REGISTER

IF THE PROGRAM IS BEING RUN ON A SWITCHLESS PROCESSOR (I.E. AN 11/04 OR 11/34) THE PROGRAM WILL DETERMINE THAT THE HARDWARE SWITCH REGISTER IS NOT PRESENT AND WILL USE A 'SOFTWARE' SWITCH REGISTER. THE 'SOFTWARE' SWITCH REGISTER IS LOCATED AT LOCATION 176 (8). THE SETTINGS OF THE 'SOFTWARE' SWITCHES ARE CONTROLLED THROUGH A KEYBOARD ROUTINE WHICH IS CALLED BY TYPING A 'CONTROL G'. THE PROGRAM WILL RECOGNIZE THE 'CONTROL G' AT ANY TIME EXCEPT WHEN THE PROGRAM IS AT A HIGHER PRIORITY PROCESSING AN RK06 INTERRUPT. THE 'SOFTWARE' SWITCH VALUES ARE ENTERED AS AN OCTAL NUMBER IN RESPONSE TO THE PROMPT FROM THE SWITCH ENTRY ROUTINE:

SWR = NNNNNN NEW =

EACH TIME SWITCH SETTING ARE ENTERED, THE ENTIRE SWITCH REGISTER IMAGE MUST BE ENTERED. LEADING ZEROS ARE NOT REQUIRED. 'RUBOUT' AND 'CONTROL U' FUNCTIONS MAY BE USED TO CORRECT TYPING ERRORS DURING SWITCH ENTRY.

ON PROCESSORS WITH HARDWARE SWITCH REGISTERS, THE 'SOFTWARE' SWITCH REGISTER MAY BE USED. IF THE PROGRAM FINDS ALL 16 SWITCHES IN THE 'UP' POSITION, ALL SWITCH REGISTER REFERENCES WILL BE TO THE 'SOFTWARE' REGISTER AND THE PROCEDURES DESCRIBED ABOVE MUST BE FOLLOWED.

4.5 INPUT DIALOGUE

THE DIALOGUE WILL BE DONE INTERACTIVELY. THE PROGRAM WILL REQUEST A PARAMETER BY CONSOLE TYPEOUT. THE PARAMETER MAY THEN BE ENTERED AS SPECIFIED BELOW OR ALLOWED TO DEFAULT

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  
524  
525  
526  
527  
528  
529  
530  
531  
532  
533  
534  
535  
536  
537  
538

BY A CARRIAGE RETURN. UNRECOGNIZED OR ILLEGAL RESPONSES WILL BE ECHOED BACK FOLLOWED BY "?". THE PROPER RESPONSE MAY THEN BE ENTERED.

IMPORTANT: FOR VARIATIONS OF THE ABOVE, SEE XXDP, ACT/APT CONSIDERATIONS IN SECTIONS 3.2 & 3.4.

4.5.1 DRIVE SELECTION

THE REQUEST WILL BE:

DRIVES TO BE TESTED:

THE DEFAULT RESPONSE IS CARRIAGE RETURN TO TEST ALL DRIVES IN THE 'DRIVE PRESENT' CONDITION.

THE OPERATOR CAN ALSO TYPE IN THE SPECIFIC DRIVE NUMBERS TO BE TESTED, SEPARATED BY COMMAS & TERMINATED BY A CARRIAGE RETURN.

E.G. DRIVES TO BE TESTED: 1,2,4,6

IMPORTANT: FOR VARIATIONS OF THE ABOVE, SEE XXDP, ACT/APT CONSIDERATIONS IN SECTIONS 3.2 & 3.3.

4.5.2 BUS ADDRESS

THE REQUEST WILL BE:

TYPE IN BUSS ADDRESS IF NOT 177440

THE DEFAULT IS A CARRIAGE RETURN

4.5.3 CONTROLLER INTERRUPT VECTOR

THE REQUEST WILL BE:

TYPE IN CONTROLLER INTERRUPT VECTOR IF NOT 210

THE DEFAULT IS A CARRIAGE RETURN.

4.5.4 EXAMPLE OF PROGRAM DIALOGUE

THE EXAMPLE SHOWN IS FOR A PROGRAM STARTED AT ADDRESS 220. ALL OPERATOR RESPONSES ARE UNDERLINED.

CZR6JFO UNIBUSS RK6 DR-3

DRIVES TO BE TESTED: 1,3<CR>

TYPE IN BUSS ADDRESS IF NOT 177440 <CR>

539  
540  
541  
542  
543  
544  
545  
546  
547  
548  
549  
550  
551  
552  
553  
554  
555  
556  
557  
558  
559  
560  
561  
562  
563  
564  
565  
566  
567  
568  
569  
570  
571  
572  
573  
574  
575  
576  
577  
578  
579  
580  
581  
582  
583  
584  
585  
586  
587  
588  
589  
590  
591  
592  
593  
594

-----  
TYPE IN CONTROLLER INTERRUPT VECTOR IF NOT 210 <CR>  
-----

WILL TEST DRIVES:

1  
3

DRIVE 1

(THE REST IS IDENTICAL TO THE EXAMPLE SHOWN IN 4.6 BELOW)

#### 4.6 PROGRAM EXAMPLE

THE FOLLOWING IS AN EXAMPLE OF A PROGRAM STARTED AT THE  
DEFAULT ADDRESS (200) & WITH 2 DRIVES ON THE LINE.

CZR6JFO UNIBUSS RK6 DR-3

WILL TEST DRIVES:

0  
1

DRIVE 0

DRIVE SERIAL NO. AAA  
CARTRIDGE SERIAL NO. BBB

DRIVE 1

DRIVE SERIAL NO. CCC  
CARTRIDGE SERIAL NO. DDD

END PASS #1

WILL TEST DRIVES:

0  
1

DRIVE 0

DRIVE 1

END PASS # 2

(ETC)

THE ABOVE ASSUMES NO ERRORS DETECTED.  
THE NUMBER OF PASSES IS DETERMINED BY ACT/APT/XXDP

IMPORTANT: FOR VARIATIONS OF THE ABOVE, SEE XXDP, ACT/APT  
CONSIDERATIONS IN SECTIONS 3.2 & 3.3.

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  
646  
647  
648  
649  
650

#### 4.7 HALTING THE PROGRAM

THE PROGRAM PROVIDES A METHOD OF HALTING ITSELF SUCH THAT THE CARTRIDGE AND/OR DRIVE IS NOT LEFT IN ON UNDETERMINED STATE; IE: HEADS UNLOADED OR INVALID FORMAT.

TO PROPERLY HALT, TYPE CONTROL-C (^C) ON THE CONSOLE.

IF HEADS ARE LOADED & FORMATTING IS VALID, THE PROGRAM WILL:

1. ECHO ^C
2. TYPE "CPU HALTED"
3. HALT THE PROGRAM

IF HEADS ARE NOT LOADED AND/OR FORMATTING IS INVALID, THE PROGRAM WILL:

1. ECHO ^C
2. TYPE 'HALT PENDING, PLEASE WAIT'
3. DO THE TEST(S) THAT LOADS HEADS AND/OR FORMATS THE INVALID CYLINDERS
4. TYPE 'CPU HALTED'
5. HALT THE PROGRAM

#### NOTES:

1. THE ABOVE EXAMPLE IS FOR THE PROGRAM RUNNING IN DUMP MODE (MANUAL). IF THE PROGRAM IS RUNNING IN CHAIN/AUTO MODE VIA XXDP,ACT,APT; IT WILL FIRST LOAD HEADS AND/OR FORMAT CORRECTLY, IF REQ'D, THEN IT WILL JUMP ON TO THE MONITOR WHERE THE NEXT PROGRAM CAN BE CALLED IN.

THE TYPEOUTS WILL BE 'ABORT PENDING - PLEASE WAIT'  
& 'PROGRAM ABORTING'

2. OPERATING THE 'CONTINUE' SWITCH ON THE CPU CONSOLE WILL RETURN THE PROGRAM TO TEST 1 WHERE TESTING WILL BEGIN WITH THE 1'ST DRIVE AGAIN.

#### 5.0 DRIVE DIAGNOSTIC FUNCTIONAL DESCRIPTION

##### 5.1 GENERAL

#### OPERATOR INTERVENTION TESTS

THESE TESTS CHECK OUT ALL THE DRIVE INTERLOCKS, FRONT PANEL SWITCHES AND LIGHTS.

THE OPERATOR IS INSTRUCTED TO PERFORM A TEST AND TYPE A SPACE WHEN FINISHED.

OPERATOR INTERVENTION TESTS CAN BE INDIVIDUALLY BYPASSED BY TYPING A CONTROL-E <^E>. ONLY AT THE BEGINNING OF EACH TEST, AS INSTRUCTED BY THE TYPEOUT.

651  
652  
653  
654  
655  
656  
657  
658  
659  
660  
661  
662  
663  
664  
665  
666  
667  
668  
669  
670  
671  
672  
673  
674  
675  
676  
677  
678  
679  
680  
681  
682  
683  
684  
685  
686  
687  
688  
689  
690  
691  
692  
693  
694  
695  
696  
697  
698  
699  
700  
701  
702  
703  
704  
705  
706

IF THE PROGRAM DETERMINES IT WAS LOADED UNDER  
ACT, APT, OPERATOR INTERVENTION TESTS WILL BE  
BYPASSED UNLESS THE 'LOAD & DUMP MODE' IS BEING  
USED.

THEY WILL BE BYPASSED IN 'MONITOR MODE' AS OPERATOR  
INTERVENTION MAY NOT BE FEASIBLE IN OVER-NITE  
TESTING.

5.2 TEST DESCRIPTIONS

\*\*\*\*\*  
BASIC CONTROLLER TESTS, SIZING & SETUP  
\*\*\*\*\*

TEST 1 REFERENCE ALL CONTROLLER REGISTERS

THIS TEST VERIFIES THAT ALL THE CONTROLLER REGISTERS  
CAN BE ACCESSED. THE INABILITY TO BE ACCESSED WILL  
RESULT IN A TIMEOUT TRAP WITH AN ERROR MESSAGE. ANY  
ERROR IN THIS TEST WILL RESULT IN ABORTING ALL OTHER  
TESTS AND JUMPING TO 'END OF PASS'

TEST 2 SIZE THE BUSS

THIS TEST IS ENTERED ONLY IF 'DRIVE SELECTION' IS DEFAULTED  
EITHER BY RUNNING IN THE AUTO MODE OR A 200 START IN THE  
MANUAL MODE.  
EVERY DRIVE FROM 0 THRU 7 IS ADDRESSED.  
CONTROLLER ERROR (CERR) IS EXAMINED AND IF NOT SET, THE  
DRIVE WILL BE TESTED AS AN RK06. IF SET, THE PROGRAM WILL BYPASS  
TESTING THAT DRIVE ONLY IF THE ERROR WAS A RESULT OF  
MDS, UFE OR NED BEING SET; OR BOTH NED & DRA RESET IN-  
DICATING THE OTHER PORT IS ACCESSED.

IF CERR DUE TO DTYE, THE DRIVE WILL BE TESTED AS  
AN RK07.

TEST 3 VERIFY OPERATOR DRIVE SELECTIONS

THIS TEST IS ENTERED ONLY IF DRIVE SELECTION IS NOT  
DEFAULTED. EVERY DRIVE FROM 0 TO 7 IS ADDRESSED &  
CONTROLLER ERROR (CERR) IS EXAMINED. IF NOT SET, THE  
PROGRAM WILL ASSUME THE DRIVE IS PRESENT AS AN RK06.  
IF CERR WAS SET, THAT DRIVE WILL BE BYPASSED  
ONLY IF THE ERROR WAS A RESULT OF MDS OR UFE SET OR BOTH  
NED & DRA RESET (WRONG PORT). IF CERR IS A RESULT OF  
NED ONLY, IT IS CHECKED AGAINST THE INPUTTED INFOR TO  
VERIFY IT WAS NOT SPECIFIED.  
IF CERR DUE TO DTYE, THE DRIVE WILL BE TESTED

AS AN RK07.

707  
708  
709  
710  
711  
712  
713  
714  
715  
716  
717  
718  
719  
720  
721  
722  
723  
724  
725  
726  
727  
728  
729  
730  
731  
732  
733  
734  
735  
736  
737  
738  
739  
740  
741  
742  
743  
744  
745  
746  
747  
748  
749  
750  
751  
752  
753  
754  
755  
756  
757  
758  
759  
760  
761  
762

TEST 4 FIND NEXT DRIVE TO BE TESTED

THIS TEST FINDS THE NEXT DRIVE PRESENT & PUTS THAT ADDRESS IN 'DRVAD' & STMP4 IS SET TO CDT IF DRIVE IS RK07. THROUGHOUT THE FOLLOWING TESTS, THE DRIVE TESTED IS THE DRIVE WHOSE ADDRESS IS IN 'DRVAD'.

TEST 5 PRINT DRIVE SERIAL NUMBER

THIS TEST READS & PRINTS THE DRIVE SERIAL # FROM MSG A, WORD 11 IN DECIMAL & IS PERFORMED ON THE 1ST PASS ONLY

TEST 6 SET VV WITH PACK COMMAND

IF VV IS RESET, THE PACK COMMAND IS USED TO SET IT.

TEST 7 UNLOAD DRIVE TO BE TESTED

THIS TEST UNLOADS THE DRIVE TO BE TESTED NEXT, WAITS FOR ATTN & VERIFIES IT CAME FROM THE CORRECT DRIVE.

\*\*\*\*\*  
OPERATOR INTERVENTION TESTS  
\*\*\*\*\*

TEST 10 INTERLOCKS TESTS

THIS TEST VERIFIES THAT THE DOOR & CARTRIDGE STATUS BITS ARE OPERATING PROPERLY IN MESSAGE A0 & THAT THE REMOVAL OF THE CARTRIDGE CLEARS VOLUME VALID. IT FURTHER VERIFIES ALL REMAINING STATUS BITS OF MESSAGE A & B, WORDS 0 & 1. THIS TEST ALSO CHECKS THAT THE SPINDLE CANNOT BE STARTED WHEN THE DOOR IS OPEN OR THE CARTRIDGE IS REMOVED. IT ALSO VERIFIES THAT LOSS OF VOLUME VALID RESETS SACK WHICH ASSERTS NON EXISTENT DRIVE IN RKCS2

TEST 11 UNIT SELECT PLUG TEST

THIS TEST VERIFIES THAT WHEN THE UNIT SELECT PLUG IS PULLED OUT, THE QUAL LOGIC RESETS ATTN & VOLUME VALID, THAT THE DRIVE DE-SELECTS & NON EXISTENT DRIVE ASSERTS. FURTHER, THE OPERATOR IS ASKED TO INSERT ANY NUMBER OF UNIT SELECT PLUGS. THE PROGRAM WILL RESPOND BY TYPING THE PLUG CODE NUMBER AS SOON AS IT IS INSERTED. THIS PORTION OF THE TEST IS TERMINATED AT ANY TIME BY A CONTROL-C



763  
764  
765  
766  
767  
768  
769  
770  
771  
772  
773  
774  
775  
776  
777  
778  
779  
780  
781  
782  
783  
784  
785  
786  
787  
788  
789  
790  
791  
792  
793  
794  
795  
796  
797  
798  
799  
800  
801  
802  
803  
804  
805  
806  
807  
808  
809  
810  
811  
812  
813  
814  
815  
816  
817  
818

TEST 12 PORT SELECTION TESTS

THE OPERATOR IS ASKED TO SWITCH TO THE WRONG PORT  
& THEN DESELECT BOTH PORTS.  
IN BOTH CASES, NON EXISTENT DRIVE SHOULD ASSERT IN RKCS2

TEST 13 AC LOW DETECTION PART 1

A PRELIMINARY AC LOW TEST IS PERFORMED HERE WHILE HEADS ARE UNLOADED.  
BATTERY RETRACT WILL BE TESTED LATER.  
ACLO CANNOT BE TESTED DIRECTLY SINCE IT REQUIRES  
SECTOR PULSES, I.E. HEADS LOADED.  
THEREFORE THE INDICATOR CHECKED IS NON-EXISTENT  
DRIVE (NED) ASSERTING IN RKCS2 AS A RESULT OF DCLO.  
AFTER POWER UP, VOLUME VALID IS CHECKED TO BE CLEARED.

TEST 14 CHECK NXF LOGIC

THIS TEST VERIFIES NON-EXECUTABLE FUNCTION (NXF) IS DETECTED  
AS A RESULT OF DOING A SEEK WITH VOLUME VALID RESET.

TEST 15 READ & SAVE BAD SECTOR INFO & TYPE PACK SERIAL #

THIS TEST VERIFIES THAT CYL 632 (1456 FOR RK07), TRACK 2 CAN BE READ.  
THIS AREA CONTAINS BAD SECTOR INFOR WHICH IS WRITTEN BY THE  
FACTORY DURING MANF. ALL BAD SECTOR INFOR (BSE) WILL BE STORED  
AT THIS TIME TO MASK FUTURE READ HEADER OR DATA ERROR PRINTOUTS.  
IF BSE INFO CANNOT BE READ, OR IF AFTER READING THE BSE INFO  
IT IS DETERMINED THAT AN ALIGNMENT CARTRIDGE IS USED,  
A MESSAGE WILL BE TYPED INDICATING THAT ALL  
FUTURE FORMAT AND READ-WRITE TESTS WILL BE BYPASSED.  
THIS IS DONE SO AS NOT TO DESTROY BSE INFOR OR AN ALIGNMENT PACK BY WRIT  
THE PACK SERIAL # IS TYPED IN OCTAL & FOR THE FIRST PASS ONLY.

TEST 16 WRITE LOCK TEST

THIS TEST VERIFIES THAT DATA WRITTEN ON A SECTOR CANNOT BE  
ALTERED ONCE THE WRITE LOCK SWITCH HAS BEEN ENABLED.  
IT ALSO CHECKS THAT WRITE PROTECT TAKES EFFECT ONLY AT  
SECTOR BOUNDRIES WHEN DOING CONTINUOUS WRITING ON CYL 0, HEAD 0

TEST 17 AC LOW DETECTION PART 2

THIS TEST VERIFIES THAT WHEN AC POWER IS LOST, THAT WRITING CEASES  
AT SECTOR BOUNDRIES & THAT THE BATTERY RETRACT IS FUNCTIONAL.  
THERE IS APPROX 4 MS BETWEEN AC LOW ASSERTING AND NED ASSERTING  
WHEN THE INTERFACE SHUTS DOWN.

819  
820  
821  
822  
823  
824  
825  
826  
827  
828  
829  
830  
831  
832  
833  
834  
835  
836  
837  
838  
839  
840  
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

TEST 20 END OF PROGRAM

THIS IS NOT A TEST BUT A LINKAGE TO PERFORM ALL THE ABOVE TESTS FOR THE NEXT DRIVE PRESENT. THE NEXT TEST IS ENTERED ONLY AFTER ALL DRIVES PRESENT HAVE BEEN TESTED. DO NOT LOOP ON THIS 'TEST'.

TEST 21 MULTIPLE DRIVE DETECTION TEST

THIS TEST IS PERFORMED ONLY ONCE AT THE END OF PASS 1 AND IS BYPASSED IF ONLY ONE DRIVE IS PRESENT.

THE MULTIPLE DRIVE DETECTION LOGIC IS TESTED BY THE FOLLOWING METHOD:

- A. HEADS MUST BE LOADED (SECTOR PULSES REQ'D)
- B. THE OPERATOR IS INSTRUCTED TO INSERT THE SAME UNIT SELECT PLUG NUMBER (1 PAIR AT A TIME) ON ANY 2 DRIVES TO BE TESTED
- C. THE OPERATOR THEN DEPRESSES THE SPACE BAR TO CONTINUE THE TEST OR A CONTROL-C TO EXIT THE TEST

THE PROGRAM VERIFIES THAT MULTIPLE DRIVE SELECT & DRIVE UNSAFE BOTH SET & THAT THE DRIVE UNLOADS

THE OPERATOR IS ASKED TO VERIFY THAT HEADS UNLOAD FROM BOTH DRIVES

THE PROGRAM DOES NOT REQUIRE FORMATTED PACKS AS FORMATTING IS PERFORMED IN ANY CASE.

ANY TEST THAT MODIFIES STANDARD FORMATTING IS FOLLOWED BY A 'CLEAN UP' TEST TO PUT THOSE CYLINDERS BACK TO STANDARD FORMAT.

6.0 ERROR REPORTING

6.1 ERROR INTERPRETATION

WHENEVER AN ERROR MESSAGE IS PRINTED OUT, ALL REGISTERS AND OTHER DATA PERTAINING TO THE ERROR ARE ALSO GIVEN. MSG A(00), MSG B(01), RKER, RKBA...ETC, INDICATE THE CONTENTS OF THE CORRESPONDING REGISTERS AT THE TIME OF ERROR.

EVERY ERROR MESSAGE CONTAINS A PC. THIS PC INDICATES THE POSITION IN PROGRAM WHERE THE ERROR CALL IS LOCATED. THE

875  
876  
877  
878  
879  
880  
881  
882  
883  
884  
885  
886  
887  
888  
889  
890  
891  
892  
893  
894  
895  
896  
897  
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  
926  
927  
928  
929  
930

ERROR MESSAGE, BECAUSE OF PRACTICAL CONSIDERATIONS IS MADE SHORT AND MEANINGFUL. THE USER IS ADVISED TO LOOK UP THE PC IN THE PROGRAM LISTING, WHERE HE WILL FIND MORE INFORMATION ABOUT THE ERROR. IN MANY INSTANCES, A SINGLE FAULT WILL GIVE RISE TO MORE THAN ONE ERROR REPORT. A LITTLE DELIBERATION AND CAREFUL EXAMINATION OF THE DATA GIVEN WILL BE CERTAINLY VERY HELPFUL IN PINPOINTING THE FAULT. A BRIEF EXPLANATION OF WHAT IS BEING CHECKED IN THE TEST IS GIVEN AT THE BEGINNING OF EVERY TEST. ALL THE NUMBERS GIVEN WITH ERROR MESSAGES ARE IN OCTAL.

NOTE

NO ERROR LOGGING OR OPERATION HISTORY IS PROVIDED.

6.2 ERROR PRINTOUT EXAMPLE:

DEPRESS 'RUN-STOP' SWITCH TO 'RUN' WHILE DOOR OPEN  
VERIFY SPINDLE DOES NOT START & HEADS DO NOT LOAD

DEPRESS SPACE BAR WHEN FINISHED

SPINDLE ON SET IN RKMR2  
AFTER MANUALLY LOADING HEADS WITH DOOR OPEN

TEST NO.	PC					
000010	015700					
RKMR2	RKMR3	RKER	RKDS	RKCS1	RKCS2	RKASOF
050343	100000	000000	140301	040200	000103	004000

MESSAGE A0 ERROR  
AFTER MANUALLY LOADING HEADS WITH DOOR OPEN

TEST NO.	PC					
000010	015736					
	EXPECT					
A0	B0	A1	B1	A2	B2	B3
100143	100000	000543	000001			
	ACTUAL					
050343	100000	001723	000001			
RKCS1	RKCS2	RKASOF	RKER	RKDS	RKDC	
040200	000103	004000	000000	140301	000000	

THE ABOVE EXAMPLE SHOWS EXPECTED & ACTUAL DATA FOR MESSAGE REGISTERS A0, B0, A1 & B1.

MESSAGES A2, B2 & B3 WILL BE TYPED OUT ONLY AS REQUIRED IF THE CYLINDER DIFFERENCE/OFFSET, CYLINDER ADDRESS & HEAD & SECTOR INFORMATION IS A VARIABLE PARAMETER OF THE TEST.

[ END OF DOCUMENT ]

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

167400  
000001

```
*** PGM REV 036 ***  
.NLIST  CND,MC,MD  
.LIST   ME  
.ENABL  ABS,AMA  
  
;DEFINE SYSMAC MACROS  
  
$SWR= 167400 ;DEFINE SWITCHES 15,14,13,11,10,9,8  
$TN= 1 ;SET FIRST TEST NO. TO 1  
  
.TITLE  CZR6JFO UNIBUS RK6 DR-3  
;*COPYRIGHT (C) 1976,1982  
;*DIGITAL EQUIPMENT CORP.  
;*MAYNARD, MASS. 01754  
*  
;*PROGRAM BY GARY PAPAZIAN  
*  
;*THIS PROGRAM WAS ASSEMBLED USING THE PDP-11 MAINDEC SYSMAC  
;*PACKAGE (MAINDEC-11-DZQAC-C5), JAN, 1981.  
*  
.SBTTL  OPERATIONAL SWITCH SETTINGS  
*  
*      SWITCH          USE  
*      -----          -  
*      15             HALT ON ERROR  
*      14             LOOP ON TEST  
*      13             INHIBIT ERROR TYPEOUTS  
*      12             ABORT DRIVE AFTER 20 ERRORS  
*      11             INHIBIT ITERATIONS  
*      10             BELL ON ERROR  
*      9              LOOP ON ERROR  
*      8              LOOP ON TEST IN SWR<7:0>  
  
.SBTTL  SUMMARY OF STARTING LOCATIONS  
*  
*      200           DEFAULT PARAMETERS  
*      220           INPUT PARAMETERS  
*      240           ODT11  
*
```

```

976      .SBTTL  BASIC DEFINITIONS
977
978      ;*INITIAL ADDRESS OF THE STACK POINTER *** 1100 ***
979      001100  STACK= 1100
980      .EQUIV  EMT,ERROR      ;;BASIC DEFINITION OF ERROR CALL
981      .EQUIV  IOT,SCOPE      ;;BASIC DEFINITION OF SCOPE CALL
982
983      ;*MISCELLANEOUS DEFINITIONS
984      000011  HT= 11          ;;CODE FOR HORIZONTAL TAB
985      000012  LF= 12          ;;CODE FOR LINE FEED
986      000015  CR= 15          ;;CODE FOR CARRIAGE RETURN
987      000200  CRLF= 200       ;;CODE FOR CARRIAGE RETURN-LINE FEED
988      177776  PS= 177776     ;;PROCESSOR STATUS WORD
989      .EQUIV  PS,PSW
990      177774  STKLMT= 177774  ;;STACK LIMIT REGISTER
991      177772  PIRQ= 177772    ;;PROGRAM INTERRUPT REQUEST REGISTER
992      177570  DSWR= 177570    ;;HARDWARE SWITCH REGISTER
993      177570  DDISP= 177570  ;;HARDWARE DISPLAY REGISTER
994
995      ;*GENERAL PURPOSE REGISTER DEFINITIONS
996      000000  R0= %0          ;;GENERAL REGISTER
997      000001  R1= %1          ;;GENERAL REGISTER
998      000002  R2= %2          ;;GENERAL REGISTER
999      000003  R3= %3          ;;GENERAL REGISTER
1000     000004  R4= %4          ;;GENERAL REGISTER
1001     000005  R5= %5          ;;GENERAL REGISTER
1002     000006  R6= %6          ;;GENERAL REGISTER
1003     000007  R7= %7          ;;GENERAL REGISTER
1004     000006  SP= %6         ;;STACK POINTER
1005     000007  PC= %7         ;;PROGRAM COUNTER
1006
1007     ;*PRIORITY LEVEL DEFINITIONS
1008     000000  PR0= 0          ;;PRIORITY LEVEL 0
1009     000040  PR1= 40         ;;PRIORITY LEVEL 1
1010     000100  PR2= 100       ;;PRIORITY LEVEL 2
1011     000140  PR3= 140      ;;PRIORITY LEVEL 3
1012     000200  PR4= 200      ;;PRIORITY LEVEL 4
1013     000240  PR5= 240      ;;PRIORITY LEVEL 5
1014     000300  PR6= 300      ;;PRIORITY LEVEL 6
1015     000340  PR7= 340      ;;PRIORITY LEVEL 7
1016
1017     ;*'SWITCH REGISTER' SWITCH DEFINITIONS
1018     100000  SW15= 100000
1019     040000  SW14= 40000
1020     020000  SW13= 20000
1021     010000  SW12= 10000
1022     004000  SW11= 4000
1023     002000  SW10= 2000
1024     001000  SW09= 1000
1025     000400  SW08= 400
1026     000200  SW07= 200
1027     000100  SW06= 100
1028     000040  SW05= 40
1029     000020  SW04= 20
1030     000010  SW03= 10
1031     000004  SW02= 4

```

```

1032      000002      SW01= 2
1033      000001      SW00= 1
1034      .EQUIV     SW09,SW9
1035      .EQUIV     SW08,SW8
1036      .EQUIV     SW07,SW7
1037      .EQUIV     SW06,SW6
1038      .EQUIV     SW05,SW5
1039      .EQUIV     SW04,SW4
1040      .EQUIV     SW03,SW3
1041      .EQUIV     SW02,SW2
1042      .EQUIV     SW01,SW1
1043      .EQUIV     SW00,SW0
1044
1045      ;*DATA BIT DEFINITIONS (BIT00 TO BIT15)
1046      100000      BIT15= 100000
1047      040000      BIT14= 40000
1048      020000      BIT13= 20000
1049      010000      BIT12= 10000
1050      004000      BIT11= 4000
1051      002000      BIT10= 2000
1052      001000      BIT09= 1000
1053      000400      BIT08= 400
1054      000200      BIT07= 200
1055      000100      BIT06= 100
1056      000040      BIT05= 40
1057      000020      BIT04= 20
1058      000010      BIT03= 10
1059      000004      BIT02= 4
1060      000002      BIT01= 2
1061      000001      BIT00= 1
1062      .EQUIV     BIT09,BIT9
1063      .EQUIV     BIT08,BIT8
1064      .EQUIV     BIT07,BIT7
1065      .EQUIV     BIT06,BIT6
1066      .EQUIV     BIT05,BIT5
1067      .EQUIV     BIT04,BIT4
1068      .EQUIV     BIT03,BIT3
1069      .EQUIV     BIT02,BIT2
1070      .EQUIV     BIT01,BIT1
1071      .EQUIV     BIT00,BIT0
1072
1073      ;*BASIC "CPU" TRAP VECTOR ADDRESSES
1074      000004      ERRVEC= 4          ;;TIME OUT AND OTHER ERRORS
1075      000010      RESVEC= 10         ;;RESERVED AND ILLEGAL INSTRUCTIONS
1076      000014      TBITVEC=14         ;;"T" BIT
1077      000014      TRTVEC= 14         ;;TRACE TRAP
1078      000014      BPTVEC= 14         ;;BREAKPOINT TRAP (BPT)
1079      000020      IOTVEC= 20         ;;INPUT/OUTPUT TRAP (IOT) **SCOPE**
1080      000024      PWRVEC= 24         ;;POWER FAIL
1081      000030      EMTVEC= 30         ;;EMULATOR TRAP (EMT) **ERROR**
1082      000034      TRAPVEC=34         ;;"TRAP" TRAP
1083      000060      TKVEC= 60          ;;TTY KEYBOARD VECTOR
1084      000064      TPVEC= 64          ;;TTY PRINTER VECTOR
1085      000240      PIRQVEC=240        ;;PROGRAM INTERRUPT REQUEST VECTOR
1086
1087      .SBTTL      RK06 CONTROLLER REGISTER DEFINITION
    
```

```
1088
1089           :          $BASE=177440
1090
1091           000000      RKCS1=  0      ;CONTROL AND STATUS REGISTER 1
1092           000002      RKWC=  2      ;WORD COUNT REGISTER
1093           000004      RKBA=  4      ;BUS ADDRESS REGISTER
1094           000006      RKDA=  6      ;DESIRED TRACK SECTOR REGISTER
1095           000010      RKCS2= 10      ;CONTROL AND STATUS REGISTER 2
1096           000012      RKDS= 12      ;DRIVE STATUS REGISTER
1097           000014      RKER= 14      ;ERROR REGISTER
1098           000016      RKASOF= 16     ;ATTENTION SUMMARY AND OFFSET REGISTER
1099           000020      RKDC= 20      ;DESIRED CYLINDER REGISTER
1100           000024      RKDB= 24      ;DATA BUFFER
1101           000026      RKMR1= 26     ;MAINTENANCE REGISTER 1
1102           000034      RKMR2= 34     ;MAINTENANCE REGISTER 2 (MESSAGE LINE A)
1103           000036      RKMR3= 36     ;MAINTENANCE REGISTER 3 (MESSAGE LINE B)
1104           000030      RKECPS= 30    ;ECC POSITION INFORMATION
1105           000032      RKECPT= 32    ;ECC PATTERN INFORMATION
1106
1107           .SBTTL CONTROL AND STATUS REGISTER 1 BITS (RKCS1:0)
1108
1109           :          DRIVE COMMANDS
1110
1111           000001      SELDRV= 1      ;SELECT DRIVE (GET STATUS)
1112           000003      PACK=  3      ;PACK ACKNOWLEDGE
1113           000005      CLEAR=  5      ;DRIVE CLEAR
1114           000007      UNLOAD= 7      ;UNLOAD
1115           000011      SRTSPL= 11     ;START SPINDLE
1116           000013      RECAL= 13     ;RECALIBRATE
1117           000015      OFFSET= 15    ;OFFSET
1118           000017      SEEK= 17      ;SEEK
1119           000021      RDDATA= 21     ;READ DATA
1120           000023      WRDATA= 23    ;WRITE DATA
1121           000025      RDHEAD= 25    ;READ HEADER
1122           000027      WRHEAD= 27    ;WRITE HEADER AND DATA
1123           000031      WRTCHK= 31    ;WRITE CHECK
1124
1125           000001      GO=    BIT0     ;GO BIT
1126           000100      IE=    BIT6     ;INTERRUPT ENABLE
1127           000200      RDY=   BIT7     ;CONTROLLER READY
1128           000400      BA16=  BIT8     ;BUS ADDRESS BIT 16
1129           001000      BA17=  BIT9     ;BUS ADDRESS BIT 17
1130           002000      CDT=   BIT10    ;CONTROLLER DRIVE TYPE (0=RK06,1=RK07)
1131           004000      CTO=   BIT11    ;CONTROLLER TIMEOUT
1132           010000      CFMT=  BIT12    ;CONTROLLER DRIVE FORMAT (0=22 SECTOR, 1=20 SECTOR)
1133           020000      DCPAR= BIT13    ;SERCON PARITY ERROR DETECTED BY CONTROLLER
1134           040000      DI=    BIT14    ;DRIVE INTERRUPT
1135           100000      CERR=  BIT15    ;CONTROLLER ERROR
1136           100000      CCLR=  BIT15    ;CONTROLLER CLEAR
1137
1138           .SBTTL CONTROL AND STATUS REGISTER 2 BITS (RKCS2:10)
1139
1140           000007      DRVMSK= 7      ;MASK FOR DRIVE SELECTION CODE
1141           000010      RLS=   BIT3     ;DESELECT OR RELEASE DRIVE IN BITS 0-2
1142           000020      BAI=   BIT4     ;BUS ADDRESS INCREMENT INHIBIT
1143           000040      SCLR=  BIT5     ;SUBSYSTEM CLEAR CONTROLLER AND ALL DRIVES
```

1144	000100	IR=	BIT6	:INPUT READY
1145	000200	OR=	BIT7	:OUTPUT READY
1146	000400	UFE=	BIT8	:UNIT FIELD ERROR
1147	001000	MDS=	BIT9	:MULTIPLE DRIVE SELECT
1148	002000	PGE=	BIT10	:PROGRAMMING ERROR
1149	004000	NEM=	BIT11	:NON-EXISTENT MEMORY
1150	010000	NED=	BIT12	:NON-EXISTENT DRIVE
1151	020000	UPE=	BIT13	:UNIBUS PARITY ERROR
1152	040000	WCE=	BIT14	:WRITE CHECK ERROR
1153	100000	DLT=	BIT15	:DATA LATE ERROR
1154				
1155		.SBTTL	ERROR REGISTER BIT DEFINITION (RKER:14)	
1156				
1157	000001	ILF=	BIT0	:ILLEGAL FUNCTION CODE
1158	000002	SKI=	BIT1	:SEEK INCOMPLETE
1159	000004	NXF=	BIT2	:NON-EXECUTABLE FUNCTION
1160	000010	DRPAR=	BIT3	:DRIVE DETECTED SERCON PARITY ERROR
1161	000020	FMTE=	BIT4	:FORMAT ERROR
1162	000040	DTYPE=	BIT5	:DRIVE TYPE ERROR
1163	000100	ECH=	BIT6	:ECC HARD
1164	000200	BSE=	BIT7	:BAD SECTOR ERROR
1165	000400	HVRC=	BIT8	:HEADER VRC ERROR
1166	001000	COE=	BIT9	:CYLINDER ADDRESS OVERFLOW ERROR
1167	002000	IDAE=	BIT10	:INVALID DISK ADDRESS ERROR: HEAD/CYL
1168	004000	WLE=	BIT11	:WRITE LOCK ERROR
1169	010000	DTE=	BIT12	:DRIVE TIMING ERROR
1170	020000	OPI=	BIT13	:OPERATION (SEARCH) INCOMPLETE
1171	040000	UNS=	BIT14	:DRIVE UNSAFE



```
1172      100000      DCK=   BIT15      ;DATA CHECK
1173
1174      .SBTTL  STATUS REGISTER BIT DEFINITION (RKDS:12)
1175
1176      000001      DRA=   BIT0      ;DRIVE AVAILABLE (CONTROLLER IS SET IF
1177      ;          THIS BIT IS RESET)
1178      000004      OFST=  BIT2      ;DRIVE OFFSET
1179      000010      ACLO=  BIT3      ;AC LOW
1180      000020      DCLO=  BIT4      ;DC LOW
1181      000040      DROT=  BIT5      ;DRIVE OFF TRACK
1182      000100      VV=    BIT6      ;VOLUME VALID
1183      000200      DRDY=  BIT7      ;DRIVE READY
1184      000400      DDT=   BIT8      ;DRIVE TYPE (0=RK06,1=RK07)
1185      004000      WRL=   BIT11     ;WRITE LOCK
1186      020000      PIP=   BIT13     ;POSITIONING IN PROGRESS
1187      040000      DSC=   BIT14     ;DRIVE STATUS CHANGE
1188      100000      SVAL=  BIT15     ;STATUS VALID
1189
1190      .SBTTL  MAINTENANCE REGISTER 1 BIT DEFINITION (RKMR1:22)
1191
1192      000017      MESMSK= 17      ;MESSAGE MASK
1193      000020      PAT=   BIT4      ;FORCE EVEN PARITY ON SERCON MESSAGE LINES
1194      000040      DMD=   BIT5      ;DIAGNOSTIC MODE
1195      000100      MSP=   BIT6      ;MAINTENANCE SECTOR PULSE
1196      000200      MIND=  BIT7      ;MAINTENANCE INDEX
1197      000400      MCLK=  BIT8      ;MAINTENANCE CLOCK
1198      001000      MERD=  BIT9      ;MAINTENANCE ENCODED READ DATA
1199      002000      MEWD=  BIT10     ;MAINTENANCE ENCODED WRITE DATA
1200      004000      PCA=   BIT11     ;PRECOMPENSATION ADVANCE
1201      010000      PCD=   BIT12     ;PRECOMPENSATION DELAY
1202      020000      ECCW=  BIT13     ;ECC WORD IS BEING READ OR WRITTEN
1203      040000      WRTGAT= BIT14    ;WRITE GATE
1204      100000      RDGATE= BIT15    ;READ GATE
1205
1206      .SBTTL  DEFINITION OF DRIVE STATUS BYTE 00 MESSAGE A (RKMR2:34)
1207
1208      000040      D.DRA=  BIT5      ;DRIVE AVAILABLE
1209      000100      D.VV=   BIT6      ;VOLUME VALID
1210      000200      D.DRDY= BIT7      ;DRIVE READY
1211      000400      D.DDT=  BIT8      ;DRIVE TYPE (0=RK06,1=RK07)
1212      001000      D.FORM= BIT9      ;DRIVE FORMAT
1213      002000      D.OFF=  BIT10     ;OFFSET ON
1214      004000      D.WRL=  BIT11     ;WRITE LOCK
1215      010000      D.SPIN= BIT12     ;SPINDLE ON
1216      020000      D.PIP=  BIT13     ;POSITIONING IN PROGRESS
1217      040000      D.DSC=  BIT14     ;DRIVE STATUS CHANGE
1218
1219      .SBTTL  DEFINITION OF DRIVE STATUS BYTE 01 MESSAGE A (RKMR2:34)
1220
1221      000020      D.SSP=  BIT4      ;SERVO SIG PRESENT
1222      000040      D.HDMM= BIT5      ;HEADS HOME
1223      000100      D.BRMM= BIT6      ;BRUSHES HOME
1224      000200      D.DOOR= BIT7      ;DOOR INTERLOCKED
1225      000400      D.CART= BIT8      ;CARTRAGE INTERLOCK
1226      001000      D.SPOK= BIT9      ;SPEED OK
1227      002000      D.FWD=  BIT10     ;FORWARD
```

```
1228      004000      D.REV= BIT11      ;REVERSE
1229      010000      D.LOAD= BIT12     ;HEADS LOADING
1230      020000      D.RTZ= BIT13     ;RETURN TO ZERO
1231      040000      D.UNLD= BIT14     ;HEADS UNLOADING
1232
1233      .SBTTL  DEFINITION OF DRIVE STATUS BYTE 00 MESSAGE B (RKMR3:36)
1234
1235      000040      D.IDAE= BITS      ;INVALID DISK ADDRESS ERROR:HEAD/CYL
1236      000100      D.ACLO= BIT6      ;AC LOW
1237      000200      D.FLT= BIT7      ;DRIVE FAULT
1238      000400      D.NXF= BIT8      ;NON-EXECUTABLE FUNCTION CODE
1239      001000      D.PAR= BIT9      ;DRIVE DETECTED SERCON PARITY ERROR
1240      002000      D.SKI= BIT10     ;SEEK INCOMPLETE
1241      004000      D.WLE= BIT11     ;WRITE LOCK ERROR
1242      010000      D.SPLS= BIT12     ;SPEED LOSS
1243      020000      D.DROT= BIT13     ;DRIVE OFF TRACK
1244      040000      D.UNS= BIT14     ;R/W UNSAFE
1245
1246      .SBTTL  DEFINITION OF DRIVE STATUS BYTE 01 MESSAGE B (RKMR3:36)
1247
1248      000020      D.SECT= BIT4      ;SECTOR ERROR
1249      000040      D.WCUR= BITS      ;WRITE CURRENT AND NO WRITE GATE
1250      000100      D.WGAT= BIT6      ;WRITE GATE AND NO TRANSISTIONS
1251      000200      D.HDFL= BIT7      ;HEAD FAULT
1252      000400      D.MHD= BIT8      ;MULTIPLE HEAD SELECT
1253      001000      D.XERR= BIT9      ;INDEX ERROR
1254      002000      D.TIB= BIT10     ;TRIBIT ERROR
1255      004000      D.PLO= BIT11     ;PLO ERROR
1256      010000      D.NMOV= BIT12     ;SEEK AND NO MOTION
1257      020000      D.LIMD= BIT13     ;LIMIT DETECT ON SEEK
1258      040000      D.SUNS= BIT14     ;SERVO UNSAFE
1259
1260      .SBTTL  COMMON MASKS AND OTHER BITS: MESSAGE A (RKMR2:34)
1261
1262      000007      M.DRV= 7          ;DRIVE CODE, ALL BYTES
1263      077770      M.SER= 77770      ;DRIVE SERIAL #, BYTE 11
1264
1265      .SBTTL  COMMON MASKS AND OTHER BITS: MESSAGE B (RKMR3:36)
1266
1267      000003      M.ID= 3          ;BYTE ID, ALL BYTES
1268      040000      M.ALGN= BIT14     ;ALIGN SIGN, BYTE 10
1269      000760      M.SECT= 760       ;SECTOR COUNT, BYTE 11
1270      007000      M.HEAD= 7000     ;HEAD DECODE, BYTE 11
1271      100000      M.PAR= BIT15     ;PARITY, MESS A/B, ALL BYTES
```

```
1272  
1273  
1274  
1275          000000  
1276  
1277  
1278  
1279          000174  
1280 000174 000000  
1281 000176 000000  
1282  
1283 000200 000137 010012  
1284          000220  
1285 000220 000137 010002  
1286  
1287          000240  
1288 000240 000137 055504  
1289  
1290  
1291  
1292  
1293  
1294          000244  
1295          000046  
1296 000046 024022  
1297          000052  
1298 000052 120000  
1299          000244  
1300          001000  
1301  
1302  
1303  
1304  
1305  
1306          001000  
1307          000024  
1308 000024 000200  
1309          000044  
1310 000044 001000  
1311          001000  
1312  
1313  
1314  
1315  
1316 001000  
1317 001000 000000  
1318 001002 001210  
1319 001004 000454  
1320 001006 001130  
1321 001010 001130  
1322 001012 000052  
1323  
1324  
1325  
1326  
1327
```

```
      .SBTTL TRAP CATCHER  
      .=0  
      ;*ALL UNUSED LOCATIONS FROM 4 - 776 CONTAIN A ".+2,HALT"  
      ;*SEQUENCE TO CATCH ILLEGAL TRAPS AND INTERRUPTS  
      ;*LOCATION 0 CONTAINS 0 TO CATCH IMPROPERLY LOADED VECTORS  
      .=174  
DISPREG: .WORD 0          ;;SOFTWARE DISPLAY REGISTER  
SWREG:   .WORD 0          ;;SOFTWARE SWITCH REGISTER  
      .SBTTL STARTING ADDRESS(ES)  
      JMP @START ;;JUMP TO STARTING ADDRESS OF PROGRAM  
      .=220  
      JMP PARSRT          ;INPUT ALL PARAMETERS & START TESTING  
      .=240  
      JMP O.ODT           ;ENTER ODT11  
      .SBTTL ACT11 HOOKS  
      ;*****  
      ;HOOKS REQUIRED BY ACT11  
      $SVPC=.             ;SAVE PC  
      .=46  
      $ENDAD              ;;1)SET LOC.46 TO ADDRESS OF $ENDAD IN .SEOP  
      .=52  
      .WORD 120000        ;;2)SET LOC.52 TO 120000  
      .=$SVPC             ;; RESTORE PC  
      .=1000  
      .SBTTL APT PARAMETER BLOCK  
      ;*****  
      ;SET LOCATIONS 24 AND 44 AS REQUIRED FOR APT  
      ;*****  
      .SX=.              ;;SAVE CURRENT LOCATION  
      .=24                ;;SET POWER FAIL TO POINT TO START OF PROGRAM  
      200                 ;;FOR APT START UP  
      .=44                ;;POINT TO APT INDIRECT ADDRESS PNTR.  
      $APTHDR            ;;POINT TO APT HEADER BLOCK  
      .=$X                ;;RESET LOCATION COUNTER  
      ;*****  
      ;SETUP APT PARAMETER BLOCK AS DEFINED IN THE APT-PDP11 DIAGNOSTIC  
      ;INTERFACE SPEC.  
      $APTHD:  
      $HIBTS: .WORD 0      ;;TWO HIGH BITS OF 18 BIT MAILBOX ADDR.  
      $MADR:  .WORD $MAIL  ;;ADDRESS OF APT MAILBOX (BITS 0-15)  
      $STMT:  .WORD 300.   ;;RUN TIME OF LONGEST TEST  
      $PASTM: .WORD 600.   ;;RUN TIME IN SECS. OF 1ST PASS ON 1 UNIT (QUICK VERIFY)  
      $UNITM: .WORD 600.   ;;ADDITIONAL RUN TIME (SECS) OF A PASS FOR EACH ADDITIONAL UNIT  
      .WORD $ETEND-$MAIL/2 ;;LENGTH MAILBOX-ETABLE(WORDS)  
      .LIST MD  
      ;
```

1328  
1329  
1330  
1331  
1332  
1333  
1334  
1335  
1336  
1337  
1338  
1339  
1340  
1341  
1342  
1343  
1344  
1345  
1346  
1347  
1348  
1349  
1350  
1351  
1352  
1353  
1354  
1355  
1356  
1357  
1358  
1359  
1360  
1361  
1362  
1363  
1364  
1365  
1366  
1367  
1368  
1369  
1370  
1371  
1372  
1373  
1374  
1375  
1376  
1377  
1378  
1379  
1380  
1381  
1382  
1383

```
:USE LOOP X TO OMIT SUBCLR
:
:MACRO LOOP A
:   SCOPI
:   MOV #STACK,SP ;RESTORE STK PTR
:
:IF B A
:   JSR PC,SUBCLR
:   ERROR 24 ;CERR AFTER SCLR
:
:ENDC
:ENDM LOOP

:
:THIS MACRO FILLS EXPECTED MSG A0, B0, A1, B1, A2, B2 & B3 WITH STANDARD
:BITS SEE A=D.DSC AFTER ATTN OR 0 AFTER DRIVE CLEAR OR ANY IMPLIED SEEKS
:NOTE: A CAN BE ANY BIT COMBINATION DESIRED.
:
:MACRO F.EAB A
:
:   MOV #<A!D.SPIN!D.DRDY!D.VV!D.DRA>,E.A0 ;EXPECTED MSG A0
:   CLR E.B0 ;EXPECTED MSG B0
:   MOV #<D.SPOK!D.CART!D.DOOR!D.BRHM!D.SSP>,E.A1 ;EXPECTED A1
:   MOV #1,E.B1 ;MSG ID FOR EXPECTED MSG B1
:   CLR E.A2 ;EXPECTED MSG A2
:   MOV #2,E.B2 ;MSG ID FOR EXPECTED MSG B2
:   MOV #3,E.B3 ;MSG ID FOR EXPECTED MSG B3
:
:ENDM F.EAB

:
:THIS MACRO ASSUMES DRIVE MSG A0, B0, A1, B1 WILL ALWAYS BE TESTED
:USE A,C,D,E FOR MSG A0, B0, A1, B1 ERROR NUMBERS RESP.
:USE G=T.A2 TO READ MSG A2 & PUT INFO INTO 'CYLDIF'
:   H=T.B2 TO READ MSG B2 & PUT INFO INTO 'CYLADD'
:   I=T.B3 TO READ MSG B3 & PUT INFO INTO 'SECTOR' & 'HEAD'
:
:USE F=<ERROR DESCRIPTION>
:
:MACRO CHECK A,C,D,E,F,G,H,I
:
:   JSR PC,CHKMSG ;CHECK MSGS A0, B0, A1, B1
:   .WORD G!H!I ;& MSGS SPECIFIED HERE
:   ERROR A ;MSG A0 ERROR F
:   ERROR C ;MSG B0 ERROR
:   ERROR D ;MSG A1 ERROR
:   ERROR E ;MSG B1 ERROR
:
:ENDM CHECK

:
:A=CYL DIFF/OFFSET ERROR #
:B=CYL ADDR ERROR #
:C=<ERROR DESCRIPTION>
:
:MACRO CWD2 A,B,C,?D,?E
:
:   MOV #2,RKMR1(R5) ;SELECT WD 2
```

1384  
1385  
1386  
1387  
1388  
1389  
1390  
1391  
1392  
1393  
1394  
1395  
1396  
1397  
1398  
1399  
1400  
1401  
1402  
1403  
1404  
1405  
1406  
1407  
1408  
1409  
1410  
1411  
1412  
1413  
1414  
1415  
1416  
1417  
1418  
1419  
1420  
1421  
1422  
1423  
1424  
1425  
1426  
1427  
1428  
1429  
1430  
1431  
1432  
1433  
1434  
1435  
1436  
1437  
1438  
1439

```

      JSR      PC,GSTAT
      TST     CYLDIF      :SEE IF MSG A2=0
      BEQ     D           :BR IF YES
      ERROR   A           :MSG A2 NOT CLEARED C
D:     TST     CYLADD     :SEE IF MSG B2=0
      BEQ     E           :BR IF YES
      ERROR   B           :MSG B2 NOT CLEARED C
E:
.ENDM   CWD2

.MACRO  DRCLR  ?A
      MOV     #CCLR,RKCS1(R5)
      MOV     $UNIT,RKCS2(R5) :DRIVE#
      MOV     #CLEAR,HCS1
      JSR     PC,DOCMD      :DO DRIVE CLEAR CMD & GET CONTR RDY
      ERROR   151          :NO RDY AFTER DRIVE CLEAR CMD
      JSR     PC,TSTATN    :TEST FOR ATTN
      BR      A
      ERROR   154          :ATTN NOT CLEARED AFTER DRIVE CLEAR CMD
A:
.ENDM   DRCLR

.MACRO  CALIB  ?A
      MOV     #CCLR,RKCS1(R5)
      MOV     $UNIT,RKCS2(R5)
      MOV     #RECAL,HCS1
      JSR     PC,DOCMD      :DO RECAL CMD & GET CONTR RDY
      ERROR   124          :RDY NOT SET AFTER RECAL CMD
      MOV     #1,RKMR1(R5)  :SELECT WORD 1
      JSR     PC,GSTAT
      BIT     #D.RTZ,HMR2
      BNE    A
      ERROR   214          :RTZ NOT SET DURING RECAL CMD
A:     MOV     T10,TEMP2    :SETUP TIMEOUT
      JSR     PC,FATT1      :FIND ATTN
      ERROR   55           :NO ATTN AFTER RECAL CMD
      DRCLR
.ENDM   CALIB

:
: A=WRHEAD/<CFMT!WRHEAD>
: USE WRHDR <A>,X TO OMIT CHECKING A0, B0, A1, B1
:
.MACRO  WRHDR  A,C,?D
      MOV     #<A>,HCS1
      JSR     PC,DATCMD     :DO DATA X FOR CMD & GET CONTR RDY
      ERROR   200          :NO RDY AFTER WRITE HEADER CMD
      JSR     PC,GSTAT     :GET FRESH STATUS
      BIT     #CERR,HCS1
  
```

1440  
1441  
1442  
1443  
1444  
1445  
1446  
1447  
1448  
1449  
1450  
1451  
1452  
1453  
1454  
1455  
1456  
1457  
1458  
1459  
1460  
1461  
1462  
1463  
1464  
1465  
1466  
1467  
1468  
1469  
1470  
1471  
1472  
1473  
1474  
1475  
1476  
1477  
1478  
1479  
1480  
1481  
1482  
1483  
1484  
1485  
1486  
1487  
1488  
1489  
1490  
1491  
1492  
1493  
1494  
1495

```

      BEQ      D
      ERROR    201          ;CERR AFTER WRITE HEADER CMD
D:
  .IF B      C
      F.EAB    0
      CHECK    215,216,217,220,<AFTER WRITE HEADER CMD>,.0,0,0
  .ENDC
  .ENDM      WRHDR

:
: A=RDHEAD/<CFMT!RDHEAD>
: USE RDHDR <A>,X TO OMIT CHECKING A0, B0, A1, B1
:
: .MACRO RDHDR A,C,?D,?E
      MOV      #RHTAB,R0
      MOV      #<A>,HCS1
      JSR      PC,DATCMD          ;DO DATA X FOR CMD & GET CONTR RDY
      ERROR    171              ;NO RDY AFTER READ HEADER CMD
      BIT      #CERR,HCS1
      BEQ      D
      ERROR    174              ;CERR AFTER READ HEADER CMD
      TYPE     ,MSG26           ;ABORTING DATA TESTS TO DO TIMING TESTS
      JMP      TIMING
D:
      MOV      RKDB(R5),(R0)+    ;1'ST WORD FROM SILO TO RHTAB
      MOV      RKDB(R5),(R0)+    ;2'ND WORD
      MOV      RKDB(R5),(R0)+    ;3'RD WORD

      BIT      #DLT,RKCS2(R5)
      BEQ      E
      JSR      PC,GSTAT
      ERROR    173              ;DLT AFTER READ HEADER CMD
E:
  .IF B      B
      NOP
  .ENDC
  .ENDM      RDHDR

: .MACRO HDCHK3 ?A
      RDHDR    RDHEAD,X
      CMP      RHTAB,TOCYL      ;CHECK WORD 0 ONLY, CYL#
      BEQ      A                ;BR IF SAME
      ERROR    51                ;WRONG CYL# ON HEADER
A:
  .ENDM      HDCHK3

:
: A=TOCYL/FRCYL , B=HEAD#, C = 0 FOR 22 SECTOR, 1 FOR 20 SECTOR
:
```

1496  
1497  
1498  
1499  
1500  
1501  
1502  
1503  
1504  
1505  
1506  
1507  
1508  
1509  
1510  
1511  
1512  
1513  
1514  
1515  
1516  
1517  
1518  
1519  
1520  
1521  
1522  
1523  
1524  
1525  
1526  
1527  
1528  
1529  
1530  
1531  
1532  
1533  
1534  
1535  
1536  
1537  
1538  
1539  
1540  
1541  
1542  
1543  
1544  
1545  
1546  
1547  
1548  
1549  
1550  
1551

```
.MACRO HDTBL A,B,C
      MOV A,CALADD      ;SETUP
      MOV #B,HEAD      ;TO FILL
      MOV #C,FORMAT    ;HEADER
      JSR PC,FHDTAB    ;TABLE
.ENDM HDTBL

:QUICK SEEK. ENTER WITH CYL# IN RKDC
:
.MACRO QKSEEK ?A
      MOV #SEEK,HCS1
      JSR PC,DOCMD     ;DO SEEK CMD & GET CONTR READY
      ERROR 131        ;NO RDY AFTER SEEK CMD

      MOV T50000,TEMP1 ;SETUP TIMEOUT
      JSR PC,FATT2     ;FIND ATTN
      ERROR 132        ;NO ATTN AFTER SEEK CMD

      BIT #CERR,HCS1
      BEQ A
      ERROR 210        ;CERR AFTER SEEK CMD
A:
.ENDM QKSEEK

:
:A=WRDATA/<CFMT!WRDATA>
:C=ADDR TO JMP TO ATTEMPT TO WRITE ON ANOTHER SECTOR
:D=ADDR TO JMP TO BYPASS TEST
:E: IF BLANK WILL CHECK A0, B0, A1 & B1 AT THE END OF WRITING
:E: IF NON BLANK WILL OMIT CHECKING A0 THRU B1
:
.MACRO WDATA A,C,D,E,?F,?G,?H,?I
      MOV #<A>,HCS1
      JSR PC,DATCMD    ;DO DATA X FOR CMD & GET CONTR RDY
      ERROR 11        ;NO RDY AFTER WRITE DATA CMD
      JSR PC,GSTAT     ;GET FRESH STATUS
      BIT #CERR,HCS1
      BEQ I           ;BR IF NO ERRORS

      BIT #BSE,HER     ;SEE IF BAD SECTOR FLAG
      BEQ G           ;BR IF NO
      JSR PC,TRUERR    ;ELSE SEE IF SECTOR LISTED IN BSE TABLE
      BR H           ;RETURN HERE IF NO

      INC SECTOR
      CMP SECTOR,#10. ;RETURN HERE IF YES
      BNE F           ;ARE 10 CONSEC. SECTORS BAD
                       ;BR IF NO
```

```

1552          ERROR 40          ;ABORTING TEST DETECTED 10 BAD SECTORS
1553          JMP    D          ;BYPASS TEST
1554
1555 F:        MOV    #CCLR,RKCS1(R5) ;TRY ANOTHER SECTOR
1556          JMP    C
1557
1558 G:        ERROR 12          ;CERR WITH WRITE DATA CMD
1559          F.EAB 0
1560          CHECK 52,23,53,25,<AFTER WRITE DATA CMD>,T.A2,T.B2,0
1561          TYPE  ,MSG72      ;ABORTING BALANCE OF TESTS
1562          JMP    $EOP
1563 H:        ERROR 47          ;BAD SECTOR NOT LISTED IN TABLE
1564 I:
1565 .IF      B      E
1566          F.EAB 0
1567          CHECK 52,23,53,25,<AFTER WRITE DATA CMD>,T.A2,T.B2,0
1568 .ENDC
1569 .ENDM    WDATA
1570
1571
1572 ;
1573 ;A=RDDATA/<CFMT!RDDATA>
1574 ;USE RDATA <A>,X TO OMIT CKWD12
1575 ;
1576
1577 .MACRO   RDATA  A,C,?D,?E,?F,?G,?H
1578
1579          MOV    #<A>,HCS1
1580          JSR   PC,DATCMD      ;DO DATA X FOR CMD & GET CONTR RDY
1581          ERROR 13          ;NO RDY AFTER READ DATA CMD
1582          JSR   PC,GSTAT      ;GET FRESH STATUS
1583          BIT   #CERR,HCS1
1584          BEQ   G
1585          BIT   #BSE,HER      ;SEE IF BAD SECTOR
1586          BEQ   E
1587          ERROR 50          ;DETECTED BSE IN READ BUT NOT IN WRITE CMD.
1588          BR    H
1589 D:        TYPE  ,MSG72      ;ABORTING BALANCE OF TESTS
1590          JMP    $EOP
1591
1592 E:        BIT   #DCK,HER      ;SEE IF DATA CHECK ERROR
1593          BEQ   F
1594          ERROR 21          ;DATA CHECK ERROR AFTER READ CMD (ECC)
1595          BR    H
1596
1597 F:        ERROR 14          ;CERR AFTER READ DATA CMD.
1598
1599 H:        F.EAB 0
1600          CHECK 54,26,56,30,<AFTER READ DATA CMD>,T.A2,T.B2,0
1601          BR    D
1602
1603 G:
1604 .IF      B      C
1605          F.EAB 0
1606          CHECK 54,26,56,30,<AFTER READ DATA CMD>,T.A2,T.B2,0
1607 .ENDC
  
```



1608  
1609  
1610  
1611  
1612  
1613  
1614  
1615  
1616  
1617  
1618  
1619  
1620  
1621  
1622  
1623  
1624  
1625  
1626  
1627  
1628  
1629  
1630  
1631  
1632  
1633  
1634  
1635  
1636  
1637  
1638  
1639  
1640  
1641  
1642  
1643  
1644  
1645  
1646  
1647  
1648  
1649  
1650  
1651  
1652  
1653  
1654  
1655  
1656  
1657  
1658  
1659  
1660  
1661  
1662  
1663

```
.ENDM RDATA

;
;A=WRTCHK/<CFMT!WRTCHK>
;C=16 FOR STD ERROR MSG
;C=134/135/136/137 FOR ERROR MSG USED IN 'SBOUND' ROUTINE
;USE WRCHK <A>,C,X TO OMIT CKWD12
;

.MACRO WRCHK A,C,D,?E,?F

    MOV #<A>,HCS1
    JSR PC,DATCMD ;DO DATA X FOR CMD & GET CONTR RDY
    ERROR 15 ;NO RDY AFTER WRITE CHECK CMD
    JSR PC,GSTAT ;GET FRESH STATUS
    BIT #CERR,HCS1
    BEQ F
    BIT #WCE,HCS2 ;SEE IF WRITE CHECK ERROR
    BEQ E
    MOV RKDB(R5),WD1 ;ACTUAL WORD FOR PRINTOUT
    ERROR C ;WCE AFTER WRITE CMD
    BR F

E:    ERROR 22 ;CERR AFTER WRITE CHECK CMD
    F.EAB 0
    CHECK 57,31,60,32,<AFTER WRITE CHECK CMD>,T.A2,T.B2,0
    TYPE ,MSG72 ;ABORTING BALANCE OF TESTS
    JMP ENDRV

F:
    .IF B D
    F.EAB 0
    CHECK 57,31,60,32,<AFTER WRITE CHECK CMD>,T.A2,T.B2,0
    .ENDC
.ENDM WRCHK

;
;MACRO TO TEST THAT WRITE CHECK OCCURRED AT SECTOR BOUNDRY
;A&B=134,135 FOR WRITE PROTECT SW TEST
;A&B=136,137 FOR AC LOW TEST PART 2
;C=JUMP ADDR TO REPEAT TEST
;

.MACRO SBOUND A,B,C,?D,?E,?F,?G,?H,?I,?J,?K

    JSR PC,SUBCLR
    ERROR 24 ;CERR AFTER SCLR

    TST $TMP2 ;SEE IF TRK/SECTOR 0
    BEQ K ;REPEAT,NO NEW DATA XFER TOOK PLACE
    CMP $TMP2,#1023 ;SEE IF TRK 2,SECTOR 19
    BEQ K ;REPEAT,NO OLD DATA TO CHECK AGAINST
    BIT #BIT0,$TMP1
    BNE D ;BR IF WRITING 1'S WHEN WLE OCCURRED
    MOV #DATA1,RKBA(R5) ;WRITING 0'S:WLE SECTOR SHOULD HAVE ALL 1'S
    BR E
```

```

1664
1665
1666
1667
1668
1669
1670
1671
1672
1673
1674
1675
1676
1677
1678
1679
1680
1681
1682
1683
1684
1685
1686
1687
1688
1689
1690
1691
1692
1693
1694
1695
1696
1697
1698
1699
1700
1701
1702
1703
1704
1705
1706
1707
1708
1709
1710
1711
1712
1713
1714
1715
1716
1717
1718
1719

K:      JMP      C

D:      MOV      #DATA0,RKBA(R5) ;WRITING 1'S:WLE SECTOR SHOULD HAVE ALL 0'S
E:      BIS      #BAI,RKCS2(R5)
        MOV      #-256.,RKWC(R5)
        MOV      $TMP3,RKDA(R5) ;REFRESH RKDA
        MOV      @RKBA(R5),WD2 ;EXPECTED WORD FOR TRUERR TYPEOUT
        MOV      TOCYL,RKDC(R5)
        WRCHK   WRTCHK,A,X
        NOP
        NOP

        CMP      $TMP2,#400 ;SEE IF WRL AT TRK 1, SECTOR 0
        BNE     F ;BR IF NO
        MOV     #21.,RKDA(R5) ;ELSE SECTOR BEFORE WRL IS TRK 0, SEC 21
        BR      H

F:      CMP      $TMP2,#1000 ;SEE IF WRL AT TRK 2,SECTOR 0
        BNE     G ;BR IF NO
        MOV     #425,RKDA(R5) ;ELSE SECTOR BEFORE WRL IS TRK 1, SEC 21
        BR      H

G:      DEC      $TMP2 ;GET SECTOR BEFORE WRL
H:      MOV      $TMP2,RKDA(R5)
        MOV      RKDA(R5),$TMP3 ;FOR ERROR PRINTOUT
        BIT      #BI10,$TMP1
        BNE     I ;BR IF WRITING 1'S WHEN WLE OCCURRED
        MOV     #DATA0,RKBA(R5) ;WRITING 0'S:WLE SECTOR -1 SHOULD HAVE ALL 0'S
        BR      J

I:      MOV      #DATA1,RKBA(R5) ;WRITING 1'S:WLE SECTOR -1 SHOULD HAVE ALL 1'S
J:      BIS      #BAI,RKCS2(R5)
        MOV      #-256.,RKWC(R5)
        MOV      @RKBA(R5),WD2 ;EXPECTED WORD FOR TRUERR TYPEOUT
        MOV      TOCYL,RKDC(R5)
        WRCHK   WRTCHK,B,X

.ENDM   SBOUND

:
:QUICK START SPINDLE
:
:MACRO  QKSRT  ?A
        JSR     PC,SUBCLR ;CERR AFTER SCLR
        ERROR  24
        BIT     #D.SPIN,HMR2 ;SEE IF SPINDLE ALREADY ON
        BNE     A ;BR IF YES
        TYPE    ,MSG29 ;PLEASE WAIT, HEADS BEING LOADED

        MOV     #SRTSPL,HCS1
        JSR     PC,DOCMD ;DO START SPINDLE CMD & GET CONTR RDY
        ERROR  143 ;CONTR RDY NOT SET AFTER CMD

        MOV     T100,TEMP2
  
```

```
1720 JSR PC,FATT1 ;FIND ATTN
1721 ERROR 144 ;NO ATTN AFTER CMD
1722
1723 CLR UNLD
1724 A:
1725 .ENDM QKSRT
1726
1727 ;
1728 ;ROUTINE TO ISSUE PACK COMMAND
1729 ;
1730 .MACRO QKPACK ?A
1731 MOV #CCLR,RKCS1(R5)
1732 MOV $UNIT,RKCS2(R5) ;DRIVE #
1733 MOV #PACK,HCS1
1734 JSR PC,DOCMD ;DO PACK CMD & GET CONTR RDY
1735 ERROR 116 ;CONTR NOT RDY
1736
1737 BIT #D.VV,HMR2
1738 BNE A
1739 ERROR 27 ;VOLUME VALID NOT SET AFTER PACK CMD
1740 A:
1741 .ENDM QKPACK
1742
1743 ;
1744 ;QUICK UNLOAD
1745 ;
1746 .MACRO QKUNLD
1747
1748 JSR PC,SUBCLR
1749 ERROR 24 ;CERR AFTER SCLR
1750
1751 MOV #UNLOAD,HCS1
1752 JSR PC,DOCMD ;DO UNLOAD CMD & GET CONTR READY
1753 ERROR 17 ;NO RDY AFTER UNLD CMD
1754 JSR PC,TSTATN
1755 ERROR 20 ;NO ATTN AFTER UNLOAD CMD
1756 .ENDM QKUNLD
1757
1758
1759 .NLIST MD
1760
1761
1762
```

```
1763 .SBTTL COMMON TAGS
1764
1765 ::*****
1766 :*THIS TABLE CONTAINS VARIOUS COMMON STORAGE LOCATIONS
1767 :*USED IN THE PROGRAM.
1768
1769 001100 .=1100
1770 001100 000000 SCMTAG: .START OF COMMON TAGS
1771 001100 000000 .WORD 0
1772 001102 000 .STSTNM: .BYTE 0
1773 001103 000 .SERFLG: .BYTE 0
1774 001104 000000 .SICNT: .WORD 0
1775 001106 000000 .SLPADR: .WORD 0
1776 001110 000000 .SLPERR: .WORD 0
1777 001112 000000 .SERTTL: .WORD 0
1778 001114 000 .SITEMB: .BYTE 0
1779 001115 001 .SERMAX: .BYTE 1
1780 001116 000000 .SERRPC: .WORD 0
1781 001120 000000 .SGDADR: .WORD 0
1782 001122 000000 .SBDADR: .WORD 0
1783 001124 000000 .SGDDAT: .WORD 0
1784 001126 000000 .SBDDAT: .WORD 0
1785 001130 000000 .WORD 0
1786 001132 000000 .WORD 0
1787 001134 000 .SAUTOB: .BYTE 0
1788 001135 000 .SINTAG: .BYTE 0
1789 001136 000000 .WORD 0
1790 001140 177570 .SWR: .WORD DSWR
1791 001142 177570 .DISPLAY: .WORD DDISP
1792 001144 177560 .TKS: 177560
1793 001146 177562 .TKB: 177562
1794 001150 177564 .STPS: 177564
1795 001152 177566 .STPB: 177566
1796 001154 000 .SNULL: .BYTE 0
1797 001155 002 .SFILLS: .BYTE 2
1798 001156 012 .SFILLC: .BYTE 12
1799 001157 000 .STPFLG: .BYTE 0
1800 001160 000000 .STMP0: .WORD 0
1801 001162 000000 .STMP1: .WORD 0
1802 001164 000000 .STMP2: .WORD 0
1803 001166 000000 .STMP3: .WORD 0
1804 001170 000000 .STMP4: .WORD 0
1805 001172 000000 .STMP5: .WORD 0
1806 001174 000000 .STIMES: 0
1807 001176 000000 .SESCAPE: 0
1808 001200 177607 000377 .SBELL: .ASCIZ <207><377><377>
1809 001204 077 .SQUES: .ASCII /?/
1810 001205 015 .SCRFL: .ASCII <15>
1811 001206 000012 .SLF: .ASCIZ <12>
1812 ::*****
1813 .SBTTL APT MAILBOX-ETABLE
1814
1815 ::*****
1816 .EVEN
1817 001210 .MAIL: .APT MAILBOX
1818 001210 000000 .MSGTY: .WORD MSGTY ;;MESSAGE TYPE CODE
```

1819	001212	000000	\$FATAL: .WORD	AFATAL	:: FATAL ERROR NUMBER
1820	001214	000000	\$TESTN: .WORD	ATESTN	:: TEST NUMBER
1821	001216	000000	\$PASS: .WORD	APASS	:: PASS COUNT
1822	001220	000000	\$DEVCT: .WORD	ADEVCT	:: DEVICE COUNT
1823	001222	000000	\$UNIT: .WORD	AUNIT	:: I/O UNIT NUMBER
1824	001224	000000	\$MSGAD: .WORD	AMSGAD	:: MESSAGE ADDRESS
1825	001226	000000	\$MSGLG: .WORD	AMSGLG	:: MESSAGE LENGTH
1826	001230		\$ETABLE:		:: APT ENVIRONMENT TABLE
1827	001230	000	\$ENV: .BYTE	AENV	:: ENVIRONMENT BYTE
1828	001231	000	\$ENVM: .BYTE	AENVM	:: ENVIRONMENT MODE BITS
1829	001232	000000	\$SWREG: .WORD	ASWREG	:: APT SWITCH REGISTER
1830	001234	000000	\$USWR: .WORD	AUSWR	:: USER SWITCHES
1831	001236	000000	\$CPUOP: .WORD	ACPUOP	:: CPU TYPE, OPTIONS
1832			*		BITS 15-11=CPU TYPE
1833			*		11/04=01, 11/05=02, 11/20=03, 11/40=04, 11/45=05
1834			*		11/70=06, PDQ=07, Q=10
1835			*		BIT 10=REAL TIME CLOCK
1836			*		BIT 9=FLOATING POINT PROCESSOR
1837			*		BIT 8=MEMORY MANAGEMENT
1838	001240	000	\$MAMS1: .BYTE	AMAMS1	:: HIGH ADDRESS, M.S. BYTE
1839	001241	000	\$MTYP1: .BYTE	AMTYP1	:: MEM. TYPE, BLK#1
1840			*		MEM. TYPE BYTE -- (HIGH BYTE)
1841			*		900 NSEC CORE=001
1842			*		300 NSEC BIPOLAR=002
1843			*		500 NSEC MOS=003
1844	001242	000000	\$MADR1: .WORD	AMADR1	:: HIGH ADDRESS, BLK#1
1845			*		MEM. LAST ADDR.=3 BYTES, THIS WORD AND LOW OF 'TYPE' ABOVE
1846	001244	000	\$MAMS2: .BYTE	AMAMS2	:: HIGH ADDRESS, M.S. BYTE
1847	001245	000	\$MTYP2: .BYTE	AMTYP2	:: MEM. TYPE, BLK#2
1848	001246	000000	\$MADR2: .WORD	AMADR2	:: MEM. LAST ADDRESS, BLK#2
1849	001250	000	\$MAMS3: .BYTE	AMAMS3	:: HIGH ADDRESS, M.S. BYTE
1850	001251	000	\$MTYP3: .BYTE	AMTYP3	:: MEM. TYPE, BLK#3
1851	001252	000000	\$MADR3: .WORD	AMADR3	:: MEM. LAST ADDRESS, BLK#3
1852	001254	000	\$MAMS4: .BYTE	AMAMS4	:: HIGH ADDRESS, M.S. BYTE
1853	001255	000	\$MTYP4: .BYTE	AMTYP4	:: MEM. TYPE, BLK#4
1854	001256	000000	\$MADR4: .WORD	AMADR4	:: MEM. LAST ADDRESS, BLK#4
1855	001260	000000	\$VECT1: .WORD	AVECT1	:: INTERRUPT VECTOR#1, BUS PRIORITY#1
1856	001262	000000	\$VECT2: .WORD	AVECT2	:: INTERRUPT VECTOR#2, BUS PRIORITY#2
1857	001264	177440	\$BASE: .WORD	ABASE	:: BASE ADDRESS OF EQUIPMENT UNDER TEST
1858	001266	000000	\$DEVN: .WORD	ADEVN	:: DEVICE MAP
1859	001270	000000	\$CDW1: .WORD	ACDW1	:: CONTROLLER DESCRIPTION WORD#1
1860	001272	000000	\$CDW2: .WORD	ACDW2	:: CONTROLLER DESCRIPTION WORD#2
1861	001274	000000	\$DDW0: .WORD	ADDW0	:: DEVICE DESCRIPTOR WORD#0
1862	001276	000000	\$DDW1: .WORD	ADDW1	:: DEVICE DESCRIPTOR WORD#1
1863	001300	000000	\$DDW2: .WORD	ADDW2	:: DEVICE DESCRIPTOR WORD#2
1864	001302	000000	\$DDW3: .WORD	ADDW3	:: DEVICE DESCRIPTOR WORD#3
1865	001304	000000	\$DDW4: .WORD	ADDW4	:: DEVICE DESCRIPTOR WORD#4
1866	001306	000000	\$DDW5: .WORD	ADDW5	:: DEVICE DESCRIPTOR WORD#5
1867	001310	000000	\$DDW6: .WORD	ADDW6	:: DEVICE DESCRIPTOR WORD#6
1868	001312	000000	\$DDW7: .WORD	ADDW7	:: DEVICE DESCRIPTOR WORD#7
1869	001314	000000	\$DDW8: .WORD	ADDW8	:: DEVICE DESCRIPTOR WORD#8
1870	001316	000000	\$DDW9: .WORD	ADDW9	:: DEVICE DESCRIPTOR WORD#9
1871	001320	000000	\$DDW10: .WORD	ADDW10	:: DEVICE DESCRIPTOR WORD#10
1872	001322	000000	\$DDW11: .WORD	ADDW11	:: DEVICE DESCRIPTOR WORD#11
1873	001324	000000	\$DDW12: .WORD	ADDW12	:: DEVICE DESCRIPTOR WORD#12
1874	001326	000000	\$DDW13: .WORD	ADDW13	:: DEVICE DESCRIPTOR WORD#13

1875	001330	000000	SDDW14: .WORD	ADDW14	::DEVICE DESCRIPTOR WORD#14
1876	001332	000000	SDDW15: .WORD	ADDW15	::DEVICE DESCRIPTOR WORD#15
1877					
1878					
1879	001334		SETEND:		
1880					
1881		177440	ABASE=	177440	:DEFAULT BUSS ADDRESS
1882	001334	000210	RKVEC:	210	:DEFAULT CONTROLLER INTERRUPT VECTOR
1883	001336	000240	RKPRI:	PR5	:PRIORITY
1884	001340	172540	PKS:	172540	:P-CLOCK STATUS REG
1885	001342	172542	PKSB:	172542	:P-CLOCK SET BUFFER
1886	001344	172544	PKRB:	172544	:P-CLOCK READ BUFFER
1887	001346	177546	LKS:	177546	:L-CLOCK STATUS REG.
1888					
1889	001350	000100	LCVEC:	100	:L-CLOCK INTERRUPT VECTOR
1890	001352	000104	PCVEC:	104	:P-CLOCK INTERRUPT VECTOR.
1891					
1892		000114	MEMVEC=	114	:MEMORY PARITY VECTOR
1893		172100	MEMBAS=	172100	:MEMORY PARITY OPTION CSR START ADDR
1894					
1895	001354	000000	TRAPPC:	0	:PC FOR MEMORY PARITY ERROR TRAP
1896					
1897	001356	000000	PARAM:	0	:1 FOR 220 START, NO DEFAULT
1898	001360	000000	FTITLE:	0	:FLAG FOR PRINTING OUT 1ST PROGRAM TITLE
1899					
1900	001362	000000	DRVPTR:	0	:CONTAINS THE POINTER TO THE DRIVE FLAG
1901					: (DRIVO-DRIV7) OF THE DRIVE TO BE CHECKED NEXT.
1902					
1903		000040	SPBAR=	40	:SPACE BAR
1904	001364	000000	FRCYL:	0	:FROM CYLINDER
1905	001366	000000	TOCYL:	0	:TO CYLINDER
1906	001370	000000	CCYL:	0	:CURRENT CYL, USED IN N SQUARE TEST
1907	001372	000000	PCYL:	0	:PREV CYL., USED IN N SQUARE TEST
1908	001374	000000	CALDIF:	0	:CALC CYL DIFF USED IN N SQUARE TEST
1909	001376	000000	CYLDIF:	0	:CYL DIFF, RIGHT JUSTIFIED FROM RKMR3
1910	001400	000000	CYLADD:	0	:CYL ADDR, RIGHT JUSTIFIED FROM RKMR3
1911	001402	000000	CALADD:	0	:CYL ADDR USED IN FHDTAB ROUTINE
1912					
1913	001404	000074	HZ:	60.	:60 FOR 60 CPS
1914					:50 FOR 50 CPS
1915	001406	000000	COUNT:	0	:LOADED TO 50 OR 60 TO COUNT TO 1 SEC
1916					:OR ANY OTHER NUMBER TO COUNT OFF FRACTIONAL SECOND
1917	001410	000000	SEC:	0	:SECOND COUNTER
1918	001412	000000	TIMUP:	0	:FLAG TO INDICATE TIME IS UP
1919	001414	000000	SECNT:	0	:SECTOR COUNT
1920	001416	000000	PSEC:	0	:PREVIOUS SECTOR
1921	001420	000000	ESEC:	0	:EXPECTED SECTOR
1922	001422	000000	SECTOR:	0	:SECTOR COUNT, RIGHT JUSTIFIED FROM RKMR3
1923					
1924	001424	000012	T10:	10.	:TIMEOUT CONSTANTS
1925	001426	000144	T100:	100.	
1926	001430	011610	T5000:	5000.	
1927	001432	141520	T50000:	50000.	
1928					
1929					
1930	001434	000000	WD1:	0	:ACTUAL HEADER/DATA WORD



1987	005340	000000	HCS1:	0		:HOLD RKCS1
1988	005342	000000	HCS2:	0		:HOLD RKCS2
1989	005344	000000	HWC:	0		:HOLD RKWC
1990	005346	000000	HBA:	0		:ETC.
1991	005350	000000	HDA:	0		
1992	005352	000000	HDS:	0		
1993	005354	000000	HER:	0		
1994	005356	000000	HASOF:	0		
1995	005360	000000	HDC:	0		
1996	005362	000000	HDB:	0		
1997	005364	000000	HMR1:	0		
1998	005366	000000	HMR2:	0		
1999	005370	000000	HMR3:	0		
2000	005372	000000	HPOS:	0		
2001	005374	000000	HPAT:	0		
2002						
2003	005376	000000	TEMP1:	0		:TEMPORARY STORAGE.
2004	005400	000000	TEMP2:	0		
2005	005402	000000	TEMP3:	0		
2006	005404	000000	TEMP4:	0		
2007	005406	000000	TEMP5:	0		
2008						
2009						:THE FOLLOWING ARE HOLDING REGISTERS FOR MSGA(0-3) & MSGB(0-3).
2010						
2011	005410	000000	H.A0:	0		
2012	005412	000000	H.B0:	0		
2013	005414	000000	H.A1:	0		
2014	005416	000000	H.B1:	0		
2015	005420	000000	H.A2:	0		
2016	005422	000000	H.B2:	0		
2017	005424	000000	H.A3:	0		
2018	005426	000000	H.B3:	0		
2019						
2020						:THE FOLLOWING ARE 'EXPECTED' REGISTER FOR THE ABOVE.
2021						
2022	005430	000000	E.A0:	0		
2023	005432	000000	E.B0:	0		
2024	005434	000000	E.A1:	0		
2025	005436	000000	E.B1:	0		
2026	005440	000000	E.A2:	0		
2027	005442	000000	E.B2:	0		
2028	005444	000000	E.A3:	0		
2029	005446	000000	E.B3:	0		
2030						
2031						:THE FOLLOWING ARE IDENTIFIERS FOR DRIVE MSG WORDS TO BE TESTED.
2032						
2033		000001	T.A2=BIT0			:TEST MSG A2 IF SET
2034		000002	T.B2=BIT1			
2035		000004	T.B3=BIT2			
2036						
2037						
2038						:ALL THE FLAGS BELOW ARE CLEARED INITIALLY BY THE CLRFLG ROUTINE.
2039						
2040						
2041	005450	000000	DDUMP:	0		:FLAG - SET WHEN IN DDP DUMP MODE
2042	005452	000000	DDPCH:	0		:FLAG - SET WHEN IN DDP CHAIN MODE



2043	005454	000000	ACT11: 0	:FLAG - SET WHEN IN ACT11 MODE OF OPERATION
2044	005456	000000	PPTP: 0	:FLAG - SET WHEN PROGRAM LOADED BY PAPER TAPE
2045	005460	000000	DRIVS: 0	:CONTAINS THE NUMBER OF DRIVES PRESENT
2046				
2047				
2048			:THE FLAGS BELOW ARE SET TO 1 TO INDICATE THAT A PARTICULAR DRIVE	
2049			:IS PRESENT AND IS TO BE TESTED.	
2050	005462	000000	DRIV0: 0	:FLAG SET TO 1 WHEN DRIVE 0 PRESENT
2051	005464	000000	DRIV1: 0	:FOR DRIVE 1
2052	005466	000000	DRIV2: 0	:FOR DRIVE 2
2053	005470	000000	DRIV3: 0	:FOR DRIVE 3
2054	005472	000000	DRIV4: 0	:FOR DRIVE 4
2055	005474	000000	DRIV5: 0	:FOR DRIVE 5
2056	005476	000000	DRIV6: 0	:FOR DRIVE 6
2057	005500	000000	DRIV7: 0	:FOR DRIVE 7
2058				
2059	005502	000000	LCLKF: 0	:L-CLOCK FLAG PRESENT FLAG
2060	005504	000000	PCLKF: 0	:P-CLOCK FLAG PRESENT FLAG
2061	005506	000000	DOTIM: 0	:SET IF EITHER CLOCK PRESENT FOR TIMING TESTS.
2062	005510	000000	SIZFLG: 0	:SET IF DEFAULT DO SIZING IN TEST 1

2063  
2064  
2065  
2066  
2067  
2068  
2069  
2070  
2071  
2072  
2073  
2074  
2075  
2076  
2077  
2078  
2079  
2080  
2081  
2082  
2083  
2084  
2085  
2086  
2087  
2088  
2089  
2090  
2091  
2092  
2093  
2094  
2095  
2096  
2097  
2098  
2099  
2100  
2101  
2102  
2103  
2104  
2105  
2106  
2107  
2108  
2109  
2110  
2111  
2112  
2113  
2114  
2115  
2116  
2117  
2118

005512

005512 044456  
005514 051007  
005516 054126  
005520 054550

005522 044675  
005524 051007  
005526 054126  
005530 054550

005532 044716  
005534 051007  
005536 054126  
005540 054550

005542 044737  
005544 051007  
005546 054126  
005550 054550

005552 000000  
005554 000000  
005556 000000  
005560 000000

005562 045102  
005564 051007  
005566 054126  
005570 054550

005572 045156  
005574 051007  
005576 054126  
005600 054550

.SBTTL ERROR POINTER TABLE

;\*THIS TABLE CONTAINS THE INFORMATION FOR EACH ERROR THAT CAN OCCUR.  
;\*THE INFORMATION IS OBTAINED BY USING THE INDEX NUMBER FOUND IN  
;\*LOCATION \$ITEMB. THIS NUMBER INDICATES WHICH ITEM IN THE TABLE IS PERTINENT.  
;\*NOTE1: IF \$ITEMB IS 0 THE ONLY PERTINENT DATA IS (\$ERRPC).  
;\*NOTE2: EACH ITEM IN THE TABLE CONTAINS 4 POINTERS EXPLAINED AS FOLLOWS:

;\* EM ;;POINTS TO THE ERROR MESSAGE  
;\* DH ;;POINTS TO THE DATA HEADER  
;\* DT ;;POINTS TO THE DATA  
;\* DF ;;POINTS TO THE DATA FORMAT

\$ERRTB:

:ERROR 1  
EM2 ;DR # IN RKCS2 CANNOT BE READ BACK CORRECTLY IN RKMR2  
DH1  
DT1  
DF1

:ERROR 2  
EM5 ;DETECTED MDS  
DH1  
DT1  
DF1

:ERROR 3  
EM6 ;DETECTED UFE  
DH1  
DT1  
DF1

:ERROR 4  
EM7 ;DETECTED DRA & NED RESET (WRONG PORT SELECTED?)  
DH1  
DT1  
DF1

:ERROR 5  
0  
0  
0  
0

:ERROR 6  
EM9 ;DR NOT PRESENT BUT SPECIFIED BY OPERATOR  
DH1  
DT1  
DF1

:ERROR 7  
EM10 ;ABORT TEST, COULD NOT REFERENCE CONTROLLER REGISTER  
DH1  
DT1  
DF1

2119				
2120			:ERROR 10	
2121	005602	045241	EM11	:DRA & NED BOTH SET
2122	005604	051007	DH1	
2123	005606	054126	DT1	
2124	005610	054550	DF1	
2125			:ERR 11	
2126	005612	045305	EM12	:NO RDY
2127	005614	052074	DH27	:AFTER WRITE DATA CMD
2128	005616	054126	DT1	
2129	005620	054654	DF10	
2130			:ERR 12	
2131	005622	045575	EM21	:CERR SET
2132	005624	052074	DH27	
2133	005626	054126	DT1	
2134	005630	054654	DF10	
2135			:ERR 13	
2136	005632	045305	EM12	:NO RDY
2137	005634	052044	DH26	:AFTER READ DATA CMD
2138	005636	054126	DT1	
2139	005640	054654	DF10	
2140			:ERR 14	
2141	005642	045575	EM21	:CERR SET
2142	005644	052044	DH26	
2143	005646	054126	DT1	
2144	005650	054654	DF10	
2145			:ERR 15	
2146	005652	045305	EM12	:NO RDY
2147	005654	052303	DH32	:AFTER WRITE CHECK CMD
2148	005656	054126	DT1	
2149	005660	054654	DF10	
2150			:ERR 16	
2151	005662	050650	EM80	:WRITE CHECK ERROR SET
2152	005664	052303	DH32	:AFTER WRITE CHECK CMD
2153	005666	054126	DT1	
2154	005670	054654	DF10	
2155			:ERR 17	
2156	005672	045305	EM12	:CONTR NOT RDY
2157	005674	051604	DH18	:AFTER UNLD CMD
2158	005676	054126	DT1	
2159	005700	054654	DF10	
2160			:ERR 20	
2161	005702	045343	EM13	:NO ATTN
2162	005704	051604	DH18	
2163	005706	054126	DT1	
2164	005710	054654	DF10	
2165			:ERR 21	
2166	005712	050707	EM83	:DATA CHECK ERROR
2167	005714	052044	DH26	:AFTER READ DATA CMD
2168	005716	054126	DT1	
2169	005720	054654	DF10	
2170			:ERR 22	
2171	005722	045575	EM21	:CERR SET
2172	005724	052303	DH32	:AFTER WRITE CHECK CMD
2173	005726	054126	DT1	
2174	005730	054654	DF10	

2175					
2176	005732	045512			
2177	005734	052074			
2178	005736	054304			
2179	005740	055010			
2180					
2181	005742	045575			
2182	005744	051707			
2183	005746	054126			
2184	005750	054654			
2185					
2186	005752	045554			
2187					
2188					
2189					
2190					
2191					
2192	005754	052074			
2193	005756	054304			
2194	005760	055010			
2195					
2196	005762	045512			
2197	005764	052044			
2198	005766	054304			
2199	005770	055010			
2200					
2201					
2202	005772	045710			
2203	005774	051631			
2204	005776	054126			
2205	006000	054654			
2206					
2207	006002	045554			
2208	006004	052044			
2209	006006	054304			
2210	006010	055010			
2211					
2212	006012	045512			
2213	006014	052303			
2214	006016	054304			
2215	006020	055010			
2216					
2217	006022	045554			
2218	006024	052303			
2219	006026	054304			
2220	006030	055010			
2221					
2222	006032	046535			
2223	006034	051145			
2224	006036	054126			
2225	006040	054654			
2226					
2227	006042	050545			
2228	006044	051007			
2229	006046	054126			
2230	006050	054550			

2231			:ERR 35		
2232	006052	045471		EM17	:MSG A0 ERROR
2233	006054	051213		DH5	:AFTER AC SW OFF
2234	006056	054304		DT13	
2235	006060	055010		DF21	
2236			:ERR 36		
2237	006062	045533		EM19	:MSG A1 ERROR
2238	006064	051213		DH5	
2239	006066	054304		DT13	
2240	006070	055010		DF21	
2241			:ERR 37		
2242	006072	045554		EM20	:MSG A1 ERROR
2243	006074	051213		DH5	:AFTER OFFSET CMD
2244	006076	054304		DT13	
2245	006100	055010		DF21	
2246			:ERR 40		
2247	006102	050176		EM71	:DETECTED 10 BAD SECTORS
2248	006104	052074		DH27	:AFTER WRITE DATA CMD
2249	006106	054126		DT1	
2250	006110	054654		DF10	
2251			:ERR 41		
2252	006112	045365		EM14	:WRONG ATTN
2253	006114	051604		DH18	:AFTER UNLOAD CMD
2254					
2255	006116	054126		DT1	
2256	006120	054654		DF10	
2257			:ERR 42		
2258	006122	045412		EM15	:DRDY NOT CLEARED
2259	006124	051604		DH18	
2260					
2261	006126	054126		DT1	
2262	006130	054654		DF10	
2263			:ERR 43		
2264	006132	045444		EM16	:DSC NOT SET
2265	006134	051604		DH18	
2266	006136	054126		DT1	
2267	006140	054654		DF10	
2268			:ERR 44		
2269	006142	045617		EM22	:DOOR NOT CLEARED
2270	006144	051241		DH8	:AFTER DRIVE UNLOADED & DOOR OPENED
2271	006146	054126		DT1	
2272	006150	054654		DF10	
2273			:ERR 45		
2274	006152	045471		EM17	:MSG A0 ERROR
2275	006154	051241		DH8	
2276	006156	054304		DT13	
2277	006160	055010		DF21	
2278			:ERR 46		
2279	006162	045512		EM18	:MSG B0 ERROR
2280	006164	051241		DH8	
2281	006166	054304		DT13	
2282	006170	055010		DF21	
2283			:ERR 47		
2284	006172	050246		EM72	:BSE ERROR IN WRITE CMD NOT ON BSE TABLE
2285	006174	052074		DH27	:AFTER WRITE DATA CMD
2286	006176	054126		DT1	

2287	006200	054654	DF10	
2288				
2289			:ERR 50	
2290	006202	050325	EM73	:DETECTED BSE IN READ BUT NOT IN WRITE CMD.
2291	006204	051007	DH1	
2292	006206	054126	DT1	
2293	006210	054550	DF1	
2294			:ERR 51	
2295	006212	050744	EM93	:WRONG CYL# IN HEADER WORD
2296	006214	052021	DH25	:AFTER SEEK CMD
2297	006216	054240	DT9	
2298	006220	054764	DF20	
2299			:ERR 52	
2300	006222	045471	EM17	:MSG A0 ERROR
2301	006224	052074	DH27	:AFTER WRITE DATA CMD
2302	006226	054304	DT13	
2303	006230	055010	DF21	
2304			:ERR 53	
2305	006232	045533	EM19	:MSG A1 ERROR

2306	006234	052074		DH27	
2307	006236	054304		DT13	
2308	006240	055010		DF21	
2309			:ERR 54		
2310	006242	045471		EM17	:MSG A0 ERROR
2311	006244	052044		DH26	:AFTER READ DATA CMD
2312	006246	054304		DT13	
2313	006250	055010		DF21	
2314			:ERROR 55		
2315	006252	045343		EM13	:NO ATTN
2316	006254	051560		DH17	:AFTER RECAL CMD
2317	006256	054126		DT1	
2318	006260	054654		DF10	
2319			:ERR 56		
2320	006262	045533		EM19	:MSG A1 ERROR
2321	006264	052044		DH26	
2322	006266	054304		DT13	
2323	006270	055010		DF21	
2324			:ERR 57		
2325	006272	045471		EM17	:MSG A0 ERROR
2326	006274	052303		DH32	:AFTER WRITE CHECK CMD
2327	006276	054304		DT13	
2328	006300	055010		DF21	
2329			:ERR 60		
2330	006302	045533		EM19	:MSG A1 ERROR
2331	006304	052303		DH32	
2332	006306	054304		DT13	
2333	006310	055010		DF21	
2334			:ERR 61		
2335	006312	050007		EM69	:NO DRIVES PRESENT
2336	006314	051007		DH1	
2337	006316	054126		DT1	
2338	006320	054550		DF1	
2339			:ERR 62		
2340	006322	050471		EM75	:FOUND 10 BAD CYL
2341	006324	052074		DH27	:AFTER WRITE DATA CMD
2342	006326	054126		DT1	
2343	006330	054654		DF10	
2344			:ERR 63		
2345	006332	045533		EM19	:MSG A1 ERROR
2346	006334	051241		DH8	:AFTER DRIVE UNLOADED & DOOR OPENED
2347	006336	054304		DT13	
2348	006340	055010		DF21	
2349			:ERR 64		
2350	006342	045554		EM20	:MSG B1 ERROR
2351	006344	051241		DH8	
2352	006346	054304		DT13	
2353	006350	055010		DF21	
2354			:ERR 65		
2355	006352	045660		EM23	:SPIN SET
2356	006354	051304		DH11	:AFTER LOADING HEADS WITH DOOR OPEN
2357	006356	054126		DT1	
2358	006360	054654		DF10	
2359			:ERR 66		
2360	006362	045471		EM17	:MSG A0 ERROR
2361	006364	051304		DH11	

2362	006366	054304		DT13	
2363	006370	055010		DF21	
2364			:ERR 67		
2365	006372	045512		EM18	:MSG B0 ERROR
2366	006374	051304		DH11	
2367	006376	054304		DT13	
2368	006400	055010		DF21	
2369			:ERR 70		
2370	006402	045533		EM19	:MSG A1 ERROR
2371	006404	051304		DH11	
2372	006406	054304		DT13	
2373	006410	055010		DF21	
2374			:ERR 71		
2375	006412	045554		EM20	:MSG B1 ERROR
2376	006414	051304		DH11	
2377	006416	054304		DT13	
2378	006420	055010		DF21	
2379			:ERR 72		
2380	006422	045746		EM25	:CARTRIDGE NOT CLEARED
2381	006424	051360		DH12	:AFTER DISK PACK REMOVED
2382	006426	054126		DT1	
2383	006430	054654		DF10	
2384			:ERR 73		
2385	006432	000000		0	
2386	006434	000000		0	
2387	006436	000000		0	
2388	006440	000000		0	
2389			:ERR 74		
2390	006442	045471		EM17	:MSG A0 ERROR
2391	006444	051145		DH4	:AFTER PACK RE-INSERTED & HDS LOADED
2392	006446	054304		DT13	
2393	006450	055010		DF21	
2394			:ERR 75		
2395	006452	045512		EM18	:MSG B0 ERROR
2396	006454	051145		DH4	
2397	006456	054304		DT13	
2398	006460	055010		DF21	
2399			:ERR 76		
2400	006462	045533		EM19	:MSG A1 ERROR
2401	006464	051145		DH4	
2402	006466	054304		DT13	
2403	006470	055010		DF21	
2404			:ERR 77		
2405	006472	045554		EM20	:MSG B1 ERROR
2406	006474	051145		DH4	
2407	006476	054304		DT13	
2408	006500	055010		DF21	
2409			:ERR 100		
2410	006502	045660		EM23	:SPIN SET
2411	006504	051410		DH13	:AFTER LOADING HEADS WITH CART. OUT
2412	006506	054126		DT1	
2413	006510	054654		DF10	
2414			:ERR 101		
2415	006512	000000		0	
2416	006514	000000		0	
2417	006516	000000		0	



2418	006520	000000		
2419			:ERR 102	0
2420	006522	000000		0
2421	006524	000000		0
2422	006526	000000		0
2423	006530	000000		0
2424			:ERR 103	0
2425	006532	000000		0
2426	006534	000000		0
2427	006536	000000		0
2428	006540	000000		0
2429			:ERR 104	0
2430	006542	000000		0
2431	006544	000000		0
2432	006546	000000		0
2433	006550	000000		0
2434			:ERR 105	
2435	006552	047204		EM52
2436	006554	053535		DH64
2437	006556	054126		DT1
2438	006560	054654		DF10
2439			:ERR 106	
2440	006562	046103		EM28
2441	006564	051474		DH15
2442	006566	054126		DT1
2443	006570	054654		DF10
2444			:ERR 107	
2445	006572	046135		EM29
2446	006574	053211		DH59
2447	006576	054126		DT1
2448	006600	054654		DF10
2449			:ERR 110	
2450	006602	046773		EM48
2451	006604	053016		DH48
2452	006606	054126		DT1
2453	006610	054654		DF10
2454			:ERR 111	
2455	006612	046162		EM30
2456	006614	051521		DH16
2457	006616	054126		DT1
2458	006620	054654		DF10
2459			:ERR 112	
2460	006622	046056		EM27
2461	006624	051521		DH16
2462	006626	054126		DT1
2463	006630	054654		DF10
2464			:ERR 113	
2465	006632	045343		EM13
2466	006634	053211		DH59
2467	006636	054126		DT1
2468	006640	054654		DF10
2469			:ERROR 114	
2470	006642	047456		EM58
2471	006644	053211		DH59
2472	006646	054126		DT1
2473	006650	054654		DF10

:UNS NOT SET  
:AFTER MDS FOUND

:VV SET  
:WITHOUT PACK CMD

:DSC NOT SET  
:AFTER EVEN PARITY ISSUED

:WRL NOT CLEARED  
:AFTER WRITE LOCK SWITCH DISABLED

:ATTN NOT CLEARED  
:AFTER UNIT SELECT PLUG REMOVED

:NED NOT SET

:ATTN NOT SET  
:AFTER EVEN PARITY ISSUED

:PARITY NOT SET

2474			:ERR 115		
2475	006652	047033		EM49	:WRL NOT SET
2476	006654	053057		DH49	:AFTER WRITE LOCK SW ENABLED
2477	006656	054126		DT1	
2478	006660	054654		DF10	
2479			:ERROR 116		
2480	006662	045305		EM12	:CONT NOT RLY
2481	006664	051631		DH19	:AFTER PACK CMD
2482	006666	054126		DT1	
2483	006670	054654		DF10	
2484			:ERROR 117		
2485	006672	045305		EM12	:CONT NOT RDY
2486	006674	051654		DH20	:AFTER SEL DR CML
2487	006676	054126		DT1	
2488	006700	054654		DF10	
2489			:ERROR 120		
2490	006702	045305		EM12	
2491	006704	051707		DH21	:AFTER SUBSYS CLEAR
2492	006706	054126		DT1	
2493	006710	054654		DF10	
2494			:ERR 121		
2495	006712	047067		EM50	:WLE NOT SET
2496	006714	053117		DH50	:AFTER WRITING WITH WRITE LOCK SET
2497	006716	054126		DT1	
2498	006720	054654		DF10	
2499			:ERR 122		
2500	006722	046056		EM27	:NED NOT SET
2501	006724	051767		DH23	:AFTER WRONG PORT SELECTED
2502	006726	054126		DT1	
2503	006730	054654		DF10	
2504			:ERR 123		
2505	006732	045471		EM17	:MSG A0 ERROR
2506	006734	053117		DH50	:AFTER WRITING WITH WRITE LOCK ENABLED
2507	006736	054304		DT13	
2508	006740	055010		DF21	
2509			:ERROR 124		
2510	006742	045305		EM12	
2511	006744	051560		DH17	:AFTER RECAL CMD
2512	006746	054126		DT1	
2513	006750	054654		DF10	
2514			:ERR 125		
2515	006752	045512		EM18	:MSG B0 ERROR
2516	006754	053117		DH50	:AFTER WITING WITH WRL ENABLED
2517	006756	054304		DT13	
2518	006760	055010		DF21	
2519			:ERR 126		
2520	006762	045533		EM19	:MSG A1 ERROR
2521	006764	053117		DH50	
2522	006766	054304		DT13	
2523	006770	055010		DF21	
2524			:ERR 127		
2525	006772	045554		EM20	:MSG B1 ERROR
2526	006774	053117		DH50	
2527	006776	054304		DT13	
2528	007000	055010		DF21	
2529			:ERR 130		

2530	007002	046213	EM31	:NED NOT CLEARED
2531	007004	052161	DH29	:AFTER CORRECT PORT SELECTED
2532	007006	054126	DT1	
2533	007010	054654	DF10	
2534			:ERR 131	
2535	007012	045305	EM12	:NO RDY
2536	007014	052021	DH25	:AFTER SEEK CMD
2537	007016	054126	DT1	
2538	007020	054654	DF10	
2539			:ERR 132	
2540	007022	045343	EM13	:NO ATTN
2541	007024	052021	DH25	
2542	007026	054126	DT1	
2543	007030	054654	DF10	
2544			:ERR 133	
2545	007032	046056	EM27	:NED NOT SET
2546	007034	052125	DH28	:AFTER BOTH PORTS DESELECTED
2547	007036	054126	DT1	
2548	007040	054654	DF10	
2549			:ERR 134	
2550	007042	047131	EM51	:WRITE LOCK NOT SET SECTOR BOUNDRY
2551	007044	053117	DH50	:AFTER WRITING WITH WRL ENABLED
2552	007046	054166	DT3	
2553	007050	054564	DF3	
2554			:ERR 135	
2555	007052	047131	EM51	
2556	007054	053265	DH60	:AFTER WRITE LOCK ENABLED WHILE WRITING
2557	007056	054166	DT3	
2558	007060	054610	DF4	
2559			:ERR 136	
2560	007062	047131	EM51	
2561	007064	053464	DH63	:AFTER WRITE LOCK ENABLED FROM AC OFF
2562	007066	054166	DT3	
2563	007070	054564	DF3	
2564			:ERR 137	
2565	007072	047131	EM51	
2566	007074	053464	DH63	
2567	007076	054166	DT3	
2568	007100	054610	DF4	
2569			:ERR 140	
2570	007102	046244	EM32	:SPINDLE ON NOT SET
2571	007104	052247	DH31	:AFTER DRIVE MANUALLY LOADED
2572	007106	054126	DT1	
2573	007110	054654	DF10	
2574			:ERR 141	
2575	007112	046300	EM33	:DRIVE NOT READY
2576	007114	052456	DH38	:AFTER AC POWERED UP
2577	007116	054126	DT1	
2578	007120	054654	DF10	
2579			:ERR 142	
2580	007122	046331	EM34	
2581	007124	052247	DH31	
2582	007126	054126	DT1	
2583	007130	054654	DF10	
2584			:ERR 143	
2585	007132	045305	EM12	:CONT NOT READY

2586	007134	052335	DH34	:AFTER ST SPIN. CMD
2587	007136	054126	DT1	
2588	007140	054654	DF10	
2589			:ERR 144	
2590	007142	045343	EM13	:NO ATTN
2591	007144	052335	DH34	
2592	007146	054126	DT1	
2593	007150	054654	DF10	
2594			:ERR 145	
2595	007152	046372	EM35	:HEADS NOT HOME
2596	007154	052371	DH35	:AFTER MANUAL UNLOAD
2597	007156	054126	DT1	
2598	007160	054654	DF10	
2599			:ERR 146	
2600	007162	047430	EM57	:CERR NOT SET
2601	007164	052422	DH37	:AFTER TIMEOUT TO POWER DOWN
2602	007166	054126	DT1	
2603	007170	054654	DF10	
2604			:ERR 147	
2605	007172	046426	EM37	:AC LOW NOT SET
2606	007174	052422	DH37	
2607	007176	054126	DT1	
2608	007200	054654	DF10	
2609			:ERR 150	
2610	007202	046456	EM42	:NED NOT SET
2611	007204	052422	DH37	
2612	007206	054126	DT1	
2613	007210	054654	DF10	
2614			:ERROR 151	
2615	007212	045305	EM12	:NO RDY
2616	007214	051735	DH22	:AFTER CLEAR CMD
2617	007216	054126	DT1	
2618	007220	054654	DF10	
2619			:ERR 152	
2620	007222	046503	EM43	:AC LO NOT CLEARED
2621	007224	052456	DH38	:AFTER AC POWERED UP
2622	007226	054126	DT1	
2623	007230	054654	DF10	
2624			:ERR 153	
2625	007232	046535	EM44	:VV NOT CLEARED
2626	007234	052456	DH38	
2627	007236	054126	DT1	
2628	007240	054654	DF10	
2629			:ERROR 154	
2630	007242	047331	EM55	:ATTN NOT CLEARED
2631	007244	051735	DH22	
2632	007246	054126	DT1	
2633	007250	054654	DF10	
2634			:ERR 155	
2635	007252	046577	EM45	:VV SET AFTER HDS LOADED
2636	007254	051474	DH15	:WITHOUT 'PACK' CMD
2637	007256	054126	DT1	
2638	007260	054654	DF10	
2639			:ERR 156	
2640	007262	046654	EM46	:NXF=0
2641	007264	052712	DH45	:AFTER SEEK WITH VV=0

2642	007266	054126	DT1	
2643	007270	054654	DF10	
2644				
2645	007272	046733	EM47	:CYL ADDR CHANGED FROM 0
2646	007274	052712	DH45	
2647	007276	054364	DT14	
2648	007300	055044	DF22	
2649				
2650	007302	045471	EM17	:MSG A0 ERROR
2651	007304	052712	DH45	
2652	007306	054304	DT13	
2653	007310	055010	DF21	
2654				
2655	007312	045512	EM18	:MSG B0 ERROR
2656	007314	052712	DH45	
2657	007316	054304	DT13	
2658	007320	055010	DF21	
2659				
2660	007322	045533	EM19	:MSG A1 ERROR
2661	007324	052712	DH45	
2662	007326	054304	DT13	
2663	007330	055010	DF21	
2664				
2665	007332	045554	EM20	:MSG B1 ERROR
2666	007334	052712	DH45	
2667	007336	054304	DT13	
2668	007340	055010	DF21	
2669				
2670	007342	046654	EM46	:NXF NOT SET
2671	007344	052751	DH46	:AFTER WRITE DATA WITH VV=0
2672	007346	054126	DT1	
2673	007350	054654	DF10	
2674				
2675	007352	047744	EM68	:CANNOT READ BSE INFO
2676	007354	052562	DH42	:ON SECTORS 0, 2, 4, 6, 8
2677	007356	054126	DT1	
2678	007360	054740	DF17	
2679				
2680	007362	000000	0	
2681	007364	000000	0	
2682	007366	000000	0	
2683	007370	000000	0	
2684				
2685	007372	047744	EM68	:ON SEC 10, 12....20
2686	007374	054041	DH74	
2687	007376	054126	DT1	
2688	007400	054740	DF17	
2689				
2690	007402	000000	0	
2691	007404	000000	0	
2692	007406	000000	0	
2693	007410	000000	0	
2694				
2695	007412	045305	EM12	:NO RDY
2696	007414	052215	DH30	:AFTER READ HEADER CMD
2697	007416	054126	DT1	

2698	007420	054654		DF10	
2699			:ERROR 172	EM61	:NXF D:ID NOT SET FAULT
2700	007422	047527		DH45	:AFTER SEEK WITH VV=0
2701	007424	052712		DT1	
2702	007426	054126		DF10	
2703	007430	054654		DF10	
2704			:ERROR 173	EM63	:DLT SET
2705	007432	047555		DH30	
2706	007434	052215		DT1	
2707	007436	054126		DF15	
2708	007440	054720		DF15	
2709			:ERROR 174	EM21	:CERR SET
2710	007442	045575		DH30	
2711	007444	052215		DT1	
2712	007446	054126		DF15	
2713	007450	054720		DF15	
2714			:ERR 175	EM53	:UNLD NOT SET
2715	007452	047231		DH64	:AFTER MDS FOUND
2716	007454	053535		DT1	
2717	007456	054126		DF10	
2718	007460	054654		DF10	
2719			:ERR 176	EM54	:CANNOT FIND MDS
2720	007462	047261		DH65	:AFTER SEARCHING ALL DRIVES
2721	007464	053571		DT1	
2722	007466	054126		DF10	
2723	007470	054654		DF10	
2724			:ERROR 177	EM26	:VV NOT CLEARED
2725	007472	046014		DH16	:AFTER UNIT SEL PLUG REMOVED
2726	007474	051521		DT1	
2727	007476	054126		DF10	
2728	007500	054654		DF10	
2729			:ERROR 200	EM12	:NO RDY
2730	007502	045305		DH39	:AFTER WRITE HEADER CMD
2731	007504	052502		DT1	
2732	007506	054126		DF15	
2733	007510	054720		DF15	
2734			:ERROR 201	EM21	:CERR SET
2735	007512	045575		DH39	
2736	007514	052502		DT1	
2737	007516	054126		DF15	
2738	007520	054720		DF15	
2739			:ERROR 202	EM56	:UNEXP MEMORY PARITY ERROR
2740	007522	047364		DH66	:TEST #,PRAP PC
2741	007524	053634		DT6	
2742	007526	054234		DF5	
2743	007530	054634		DF5	
2744			:ERROR 203	EM18	:MSG B0 ERROR
2745	007532	045512		DH5	:AFTER AC SWITCHED OFF
2746	007534	051213		DT13	
2747	007536	054304		DF21	
2748	007540	055010		DF21	
2749			:ERR 204	EM57	:CERR NOT SET
2750	007542	047430		DH67	:AFTER TIMEOUT TO ENABLE WRL
2751	007544	053655		DT1	
2752	007546	054126		DF10	
2753	007550	054654		DF10	

2754			:ERR 205	
2755	007552	047067	EM50	:WRL NOT SET
2756	007554	053655	DH67	
2757	007556	054126	DT1	
2758	007560	054654	DF10	
2759			:ERROR 206	
2760	007562	047576	EM64	:WCE AT CYL 411,TRK 2, SEC 21
2761	007564	051007	DH1	
2762	007566	054126	DT1	
2763	007570	054640	DF7	
2764			:ERROR 207	
2765	007572	047430	EM57	:CERR NOT SET
2766	007574	053117	DH50	:AFTER WRITING WITH WRL ENABLED
2767	007576	054126	DT1	
2768	007600	054654	DF10	
2769			:ERROR 210	
2770	007602	045575	EM21	:CERR SET
2771	007604	052021	DH25	
2772	007606	054126	DT1	
2773	007610	054654	DF10	
2774			:ERR 211	
2775	007612	047506	EM59	:CTO SET
2776	007614	050105	EM70	:WHILE WAITING FOR OR REC'D CONTR RDY MSG A & B BAD
2777	007616	054126	DT1	
2778	007620	054674	DF12	
2779			:ERR 212	
2780	007622	047723	EM67	:NED SET
2781	007624	050105	EM70	
2782	007626	054126	DT1	
2783	007630	054674	DF12	
2784			:ERR 213	
2785	007632	044675	EM5	:MDS SET
2786	007634	050105	EM70	
2787	007636	054126	DT1	
2788	007640	054674	DF12	
2789			:ERR 214	
2790	007642	050444	EM74	:RTZ NOT SET
2791	007644	052535	DH41	:DURING RECD CMD
2792	007646	054126	DT1	
2793	007650	054654	DF10	
2794			:ERR 215	
2795	007652	045471	EM17	:MSG A0 ERROR
2796	007654	052502	DH39	:AFTER WRITE HEADER CMD.
2797	007656	054304	DT13	
2798	007660	055010	DF21	
2799			:ERR 216	
2800	007662	045512	EM18	:B0 ERROR
2801	007664	052502	DH39	
2802	007666	054304	DT13	
2803	007670	055010	DF21	
2804			:ERR 217	
2805	007672	045533	EM19	:A1 ERROR
2806	007674	052502	DH39	
2807	007676	054304	DT13	
2808	007700	055010	DF21	
2809			:ERR 220	

2810	007702	045554	EM20	;B1 ERROR
2811	007704	052502	DH39	
2812	007706	054304	DT13	
2813	007710	055010	DF21	
2814			;ERROR 221	
2815	007712	000000	0	
2816	007714	000000	0	
2817	007716	000000	0	
2818	007720	000000	0	
2819			;ERROR 222	
2820	007722	000000	0	
2821	007724	000000	0	
2822	007726	000000	0	
2823	007730	000000	0	
2824			;ERROR 223	
2825	007732	000000	0	
2826	007734	000000	0	
2827	007736	000000	0	
2828	007740	000000	0	
2829			;ERROR 224	
2830	007742	000000	0	
2831	007744	000000	0	
2832	007746	000000	0	
2833	007750	000000	0	
2834			;ERROR 225	
2835	007752	000000	0	
2836	007754	000000	0	
2837	007756	000000	0	
2838	007760	000000	0	
2839			;ERROR 226	
2840	007762	045305	EM12	;NO RDY
2841	007764	052044	DH26	;AFTER READ DATA CMD
2842	007766	054126	DT1	
2843	007770	054654	DF10	
2844			;ERROR 227	
2845	007772	045575	EM21	;CERR SET
2846	007774	052044	DH26	
2847	007776	054126	DT1	
2848	010000	054720	DF15	



2849  
2850  
2851  
2852  
2853  
2854  
2855  
2856  
2857  
2858  
2859  
2860  
2861  
2862  
2863  
2864  
2865  
2866  
2867  
2868  
2869  
2870  
2871  
2872  
2873  
2874  
2875  
2876  
2877  
2878  
2879  
2880  
2881  
2882  
2883  
2884  
2885  
2886  
2887  
2888  
2889  
2890  
2891  
2892  
2893  
2894  
2895  
2896  
2897  
2898  
2899  
2900  
2901  
2902  
2903  
2904

```

.SBTTL PROGRAM SETUP
PARSRT: MOV #1,PARAM ;SET FLAG FOR 220 START
        BR PRGSRT ;START PROGRAM

START: CLR PARAM ;CLEAR FOR 200 START
PRGSRT: RESET ;CLEAR ALL INT ENABLE & INIT
        MOV #STACK,SP ;SETUP STACK POINTER
        MOV #PRO,-(SP) ;PSW LOADED TO BE
        MOV #1$,-(SP) ;LSI-11 COMPATABLE
        RTI ;ENABLE ALL INTERRUPTS

1$: JSR PC,$TKINT ;SETUP KB VECTOR ADDR, PRIORITY 4
    ;& TURN ON KB INTERRUPT

;*** CPU PRIORITY LEVEL NOW AT 0 ***
;*** ANY DEVICE WHICH SETS ITS ***
;*** INTERRUPT ENABLE BIT WILL ***
;*** SERVICED. ***

;CLOCK INTERRUPTS WILL CHANGE CPU PRIORITY TO LEVEL 6 (IN 'ST5')
;RK06 CONTROLLER INTERRUPTS WILL CHANGE CPU PRIORITY TO LEVEL 5 IN 'SETINT')
;KEYBOARD INTERRUPTS WILL CHANGE CPU PRIORITY TO LEVEL 4 (SEE ABOVE)

;ALL 'SYSMAC' TRAPS WILL CHANGE CPU PRIORITY TO LEVEL 7 (SEE BELOW)

;SYSMAC 'SETUP'
.SBTTL INITIALIZE THE COMMON TAGS
;;CLEAR THE COMMON TAGS ($CMTAG) AREA
MOV #CMTAG,R6 ;:FIRST LOCATION TO BE CLEARED
CLR (R6)+ ;:CLEAR MEMORY LOCATION
CMP #SWR,R6 ;:DONE?
BNE -6 ;:LOOP BACK IF NO
MOV #STACK,SP ;:SETUP THE STACK POINTER
;;INITIALIZE A FEW VECTORS
MOV #SCOPE,@IOTVEC ;:IOT VECTOR FOR SCOPE ROUTINE
MOV #340,@IOTVEC+2 ;:LEVEL 7
MOV #ERROR,@EMTVEC ;:EMT VECTOR FOR ERROR ROUTINE
MOV #340,@EMTVEC+2 ;:LEVEL 7
MOV #STRAP,@TRAPVEC ;:TRAP VECTOR FOR TRAP CALLS
MOV #340,@TRAPVEC+2 ;:LEVEL 7
MOV #SPURDN,@PURVEC ;:POWER FAILURE VECTOR
MOV #340,@PURVEC+2 ;:LEVEL 7
MOV SENDCT,$EOPCT ;:SETUP END-OF-PROGRAM COUNTER
CLR $TIMES ;:INITIALIZE NUMBER OF ITERATIONS
CLR $ESCAPE ;:CLEAR THE ESCAPE ON ERROR ADDRESS
MOV #1,$ERMAX ;:ALLOW ONE ERROR PER TEST
MOV #,$SLPADR ;:INITIALIZE THE LOOP ADDRESS FOR SCOPE
MOV #,$SLPERR ;:SETUP THE ERROR LOOP ADDRESS
;;SIZE FOR A HARDWARE SWITCH REGISTER. IF NOT FOUND OR IT IS
;;EQUAL TO A "-1", SETUP FOR A SOFTWARE SWITCH REGISTER.
MOV @ERRVEC,-(SP) ;:SAVE ERROR VECTOR
MOV #64,$@ERRVEC ;:SET UP ERROR VECTOR
MOV #DSWR,$SWR ;:SETUP FOR A HARDWARE SWICH REGISTER
    
```

```

2905 010222 012737 177570 001142      MOV    #DDISP,DISPLAY    ;;AND A HARDWARE DISPLAY REGISTER
2906 010230 022777 177777 170702      CMP    #-1,@SWR         ;;TRY TO REFERENCE HARDWARE SWR
2907 010236 001012                BNE    66$              ;;BRANCH IF NO TIMEOUT TRAP OCCURRED
2908                                ;;AND THE HARDWARE SWR IS NOT = -1
2909 010240 000403                BR     65$              ;;BRANCH IF NO TIMEOUT
2910 010242 012716 010250      64$:  MOV    #65$, (SP)      ;;SET UP FOR TRAP RETURN
2911 010246 000002                RTI
2912 010250 012737 000176 001140      65$:  MOV    #SWREG,SWR     ;;POINT TO SOFTWARE SWR
2913 010256 012737 000174 001142      MOV    #DISPREG,DISPLAY
2914 010264 012637 000004      66$:  MOV    (SP)+,@ERRVEC   ;;RESTORE ERROR VECTOR
2915
2916 010270 005037 001216                CLR    $PASS           ;;CLEAR PASS COUNT
2917 010274 132737 000200 001231      BITB  #APTSIZE,$ENVM   ;;TEST USER SIZE UNDER APT
2918 010302 001403                BEQ    67$             ;;YES,USE NON-APT SWITCH
2919 010304 012737 001232 001140      MOV    #SSWREG,SWR    ;;NO,USE APT SWITCH REGISTER
2920 010312                67$:
2921
2922 010312 012737 010356 000004      MEMPAR: MOV #1$,ERRVEC      ;TIMEOUT VECTOR
2923 010320 012737 000340 000006      MOV    #PR7,ERRVEC+2
2924
2925 010326 012701 172100                MOV    #MEMBAS,R1     ;ADDR OF MEM CSR
2926 010332 005011                CLR    (R1)           ;SEE IF CAN REFERENCE
2927 010334 012711 000001                MOV    #1,(R1)        ;SET ENABLE BIT IF YES
2928 010340 012737 031110 000114      MOV    #MEMERR,MEMVEC ;LD MEMORY CHK VECTOR IF DONT TIMEOUT
2929 010346 012737 000340 000116      MOV    #PR7,MEMVEC+2
2930 010354 000401                BR     2$
2931
2932 010356 022626                1$:  CMP    (SP)+,(SP)+    ;ADJ STACK
2933 010360 062701 000002      2$:  ADD    #2,R1         ;TRY NEXT CSR
2934 010364 020127 172140                CMP    R1,#MEMBAS+40 ;ALL TRIED?
2935 010370 001360                BNE    3$             ;BR IN NO
2936 010372 012737 000006 000004      MOV    #ERRVEC+2,ERRVEC ;RESTORE TRAP CATCHER
2937 010400 005037 000006                CLR    ERRVEC+2
2938
2939 010404 004737 024110                JSR    PC,CLRFLG      ;CLEAR DDUMP THRU SIZFLG
2940 010410 005037 001220                CLR    $DEVCT
2941 010414 005037 001222                CLR    $UNIT
2942
2943
2944                ;FIND OUT IF XXDP, ACT, APT; CHAIN OR DUMP MODE
2945                ;
2946
2947 010420 005737 000042      START1: TST 42
2948 010424 001015                BNE 1$                ;BR IF AUTO
2949 010426 004737 024130                JSR PC,TITLE          ;MANUAL, TYPE PROG ID
2950 010432 123727 000041 000013      CMPB 41,#13          ;13=LOADED BY XXDP
2951 010440 001011                BNE 2$
2952 010442 005237 005450                INC DDUMP             ;SET RK06 DUMP MODE FLAG
2953 010446 104401 036575                TYPE ,MSG2           ;REPLACE DRO PACK W/SCRATCH & DO<CR>
2954 010452 000000                HALT                 ;HALT
2955 010454 000137 010470                JMP ST2
2956 010460 000137 010534      1$:  JMP ST3
2957 010464 005237 005456      2$:  INC PPTP            ;SET ACT/APT/PTP DUMP MODE FLAG
2958
2959                ;
2960                ;CHECK IF ALL PARAMETERS DEFAULT'ED. IF NOT, BEGIN INPUT DIALOGUE

```

```
2961 ;WITH OPERATOR. THE REPLY TO 'DRIVES TO BE TESTED' SHOULD BE
2962 ;DRIVE NOS. SEPERATED BY COMMAS & TERMINATED BY <CR>
2963 ; EX: DRIVES TO BE TESTED: 1,2,4<CR>
2964 ;
2965 ;
2966 010470 005737 001356 ST2: TST PARAM
2967 010474 001002 BNE 1$ ;BR IF 220 START
2968 010476 000137 010566 JMP ST4 ;200 START, DEFAULT & SIZE THE BUSS
2969 010502 104401 036656 1$: TYPE ,MSG3 ;DRIVES TO BE TESTED
2970 010506 004737 024210 JSR PC,GDRVS ;GET DR NOS.
2971 010512 104401 036710 TYPE ,MSG4 ;BUSS ADDR
2972 010516 004737 024350 JSR PC,GBA ;GET BA
2973 010522 104401 036755 TYPE ,MSG5 ;CONT INT VECTOR
2974 010526 004737 024376 JSR PC,GINT ;GET INT VECTOR
2975 010532 000427 BR ST5
2976 ;
2977 ;
2978 ;AUTO MODE
2979 ;CHECK IF LOADED BY XXDP OR OTHER. SET FLAGS & NO INPIIT DIALOGUE.
2980 ;DEFAULT ALL PARAMETERS. TEST ONLY THOSE DRIVES THAT ARE READY
2981 ;ON THE BUSS
2982 ;
2983 ;
2984 010534 123727 000041 000013 ST3: CMPB 41,#13 ;13=LOADED BY XXDP
2985 010542 001007 BNE 1$
2986 010544 005237 005452 INC DDPCH ;SET RK06 CHAIN MODE FLAG
2987 010550 004737 024130 JSR PC,TITLE
2988 010554 104401 037072 TYPE ,MSG7 ;DRO NOT TSTD
2989 010560 000402 BR ST4
2990 010562 005237 005454 1$: INC ACT11 ;SET ACT AUTO FLAG.
2991 ;
2992 010566 012737 177440 001264 ST4: MOV #177440,$BASE ;DEFAULT VALUE
2993 010574 012737 000210 001334 MOV #210,RKVEC ;DEFAULT VALUE
2994 010602 004737 024430 JSR PC,SETINT
2995 010606 005237 005510 INC SIZFLG ;DO 'SIZE THE BUSS' TEST
2996 ;
2997 010612 005037 005322 ST5: CLR UNLD ;INITIALIZE FLAGS
2998 010616 005037 005324 CLR BADHDR ;USED IN 'STOP' ROUTINE
2999 010622 005037 005326 CLR HPEND ;FOR VALID PROGRAM HALTS
3000 010626 005037 001176 CLR $ESCAPE
3001 010632 005037 001172 CLR $TMP5 ;CLEAR RK07 FLAG
3002 010636 012737 005462 001362 MOV #DRIVO,DRVPTR ;SETUP
3003 010644 005037 001220 CLR $DEVCT ;NO. OF DRVS DONE
3004 010650 005037 001222 CLR $UNIT ;CURRENT DRV UNDER TEST
3005 010654 012737 010722 000004 MOV #1$,ERRVEC ;SETUP TIMEOUT ERROR VECTOR
3006 010662 005777 170460 TST @LKS ;SEE IF L-CLOCK THERE
3007 010666 005237 005502 INC LCLKF ;PRESENT, SET FLAG.
3008 010672 013700 001350 MOV LCVEC,R0 ;VECTOR ADDR
3009 010676 012737 010764 000004 MOV #2$,ERRVEC
3010 010704 005777 170430 TST @PKS ;SEE IF P-CLOCK THERE
3011 010710 005237 005504 INC PCLKF ;PRESENT, SET FLAG
3012 010714 013700 001352 MOV PCVEC,R0 ;VECTOR ADDR
3013 010720 000412 BR 3$
3014 ;
3015 010722 022626 1$: CMP (SP)+,(SP)+ ;L-CLOCK NOT THERE, CLEAR STACK
3016 010724 012737 010770 000004 MOV #4$,ERRVEC
```

```
3017 010732 005777 170402          TST    @PKS          ;SEE IF P-CLOCK THERE
3018 010736 005237 005504          INC    PCLKF        ;PRESENT, SET FLAG
3019 010742 013700 001352          MOV    PCVEC,RO     ;VECTOR ADDR
3020 010746 005237 005506          3$:   INC    DOTIM   ;INDICATES TIMING TESTS CAN BE DONE
3021 010752 012720 030230          MOV    #CLOCK,(RO)+ ;SERVICE ROUTINE FOR CLOCKS
3022 010756 012710 000300          MOV    #PR6,(RO)
3023 010762 000407                   BR     TST1         ;;GO TO NEXT TEST
3024
3025 010764 022626                   2$:   CMP    (SP)+,(SP)+ ;P-CLOCK NOT THERE, CLEAR STACK
3026 010766 000767                   BR
3027
3028 010770 022626                   4$:   CMP    (SP)+,(SP)+ ;NEITHER CLOCK THERE, CLEAR STACK
3029 010772 005037 005506          CLR    DOTIM        ;TIMING TESTS CANNOT BE DONE.
3030 010776 104401 037301          TYPE  ,MSG13       ;ALL TIMING TESTS BYPASSED
3031
3032
```

.SBTTL BASIC CONTROLLER TESTS, SIZING & SETUP

3033  
3034  
3035  
3036  
3037  
3038  
3039  
3040  
3041  
3042  
3043  
3044  
3045  
3046  
3047  
3048  
3049  
3050  
3051  
3052  
3053  
3054  
3055  
3056  
3057  
3058  
3059  
3060  
3061  
3062  
3063  
3064  
3065  
3066  
3067  
3068  
3069  
3070  
3071  
3072  
3073  
3074  
3075  
3076  
3077  
3078  
3079  
3080  
3081  
3082  
3083  
3084  
3085  
3086  
3087  
3088

\*\*\*\*\*  
\*TEST 1 REFERENCE ALL CONTROLLER REGISTERS

\* THIS TEST VERIFIES THAT ALL THE CONTROLLER REGISTERS  
\* CAN BE ACCESSED. THE INABILITY TO BE ACCESSED WILL  
\* RESULT IN A TIMEOUT TRAP WITH AN ERROR MESSAGE. ANY  
\* ERROR IN THIS TEST WILL RESULT IN ABORTING ALL OTHER  
\* TESTS AND JUMPING TO 'END OF PASS'

\*\*\*\*\*

```
TST1:  SCOPE
      MOV  #1,$TIMES      ;;DO 1 ITERATION
      MOV  #STACK,$SP    ;;RESTORE STK PTR
      MOV  #PRO,-(SP)    ;RESET PSW TO PRIORITY 0
      MOV  #SS,-(SP)    ;& MAKE IT LSI COMPATABLE
      RTI

SS:

      MOV  #1$,ERRVEC    ;SETUP TIMEOUT ERROR VECTOR
      MOV  $BASE,R5     ;SETUP INDEX REG.
      TST  RKCS1(R5)    ;REFERENCE ALL THE
      TST  RKCS2(R5)    ;CONTROLLER REGISTERS
      TST  RKWC(R5)
      TST  RKBA(R5)
      TST  RKDA(R5)
      TST  RKDS(R5)    ;TIMEOUTS IN THIS SECTION
      TST  RKER(R5)    ;INDICATE THAT THE CONTROLLER
      TST  RKASOF(R5)  ;REGISTERS CANNOT BE READ.
      TST  RKDC(R5)    ;TESTING SHOULD NOT PROCEED
      TST  RKDB(R5)    ;UNTIL THIS IS REMEDIED.
      TST  RKMR1(R5)
      TST  RKMR2(R5)
      TST  RKMR3(R5)
      TST  RKECPS(R5)
      TST  RKECPT(R5)

      MOV  #BADTMO,ERRVEC ;SETUP TIMEOUT HANDLER
      MOV  #PR7,ERRVEC+2
      BR   TST2         ;;GO TO NEXT TEST

1$:   CMP  (SP)+,(SP)+  ;RESTORE STACK POINTER
      ERROR 7          ;ABORT-COULD NOT REFERENCE CONTROLLER REGISTER
      JMP  $EOP
```

\*\*\*\*\*  
\*TEST 2 SIZE THE BUSS

\* THIS TEST IS ENTERED ONLY IF 'DRIVE SELECTION' IS DEFAULTED  
\* EITHER BY RUNNING IN THE AUTO MODE OR A 200 START IN THE  
\* MANUAL MODE.  
\* EVERY DRIVE FROM 0 THRU 7 IS ADDRESSED.  
\* CONTROLLER ERROR (CERR) IS EXAMINED AND IF NOT SET, THE

```

3089          : * DRIVE WILL BE TESTED AS AN RK06.  IF SET, THE PROGRAM WILL BYPASS
3090          : * TESTING THAT DRIVE ONLY IF THE ERROR WAS A RESULT OF
3091          : * MDS, UFE OR NED BEING SET; OR BOTH NED & DRA RESET IN-
3092          : * DICATING THE OTHER PORT IS ACCESSED.
3093          : * IF CERR DUE TO DTYE, DRIVE WILL BE TESTED AS AN RK07.
3094          : *
3095          : *****
3096 011164 000004          TST2: SCOPE
3097 011166 012737 000001 001174  MOV     #1,$TIMES      ;;DO 1 ITERATION
3098 011174 012706 001100          MOV     #STACK,SP     ;RESTORE STK PTR
3099
3100 011200 005237 001502          INC     BYPCERR       ;DO NOT TEST CERR IN FRDY
3101
3102 011204 132737 000200 001231  BITB   #BIT7,$ENVM   ;SEE IF USE APT SELECTED DRIVES
3103 011212 001002          BNE    14$           ;BR IF YES
3104 011214 000137 011334          JMP    12$           ;ELSE DO NORM SIZING OR VERIFY
3105
3106 011220 104401 037205          14$:  TYPE   ,MSG10    ;WILL TEST DRIVES
3107 011224 005037 005460          CLR    DRIVS        ;# OF DRIVES PRESENT
3108 011230 005000          CLR    R0           ;DRV ADDR
3109 011232 012701 005462          MOV    #DRIVO,R1    ;DRV FLAG
3110 011236 013702 001266          MOV    $DEVN,R2     ;APT DEVICE MAP
3111
3112 011242 032702 000001          15$:  BIT    #BIT0,R2  ;SEE IF DRV IN DEVICE MAP
3113 011246 001410          BEQ   16$           ;BR IF NO
3114 011250 005237 005460          INC    DRIVS        ;ELSE INCR DRIVE COUNT
3115 011254 005211          INC    (R1)         ;& SET DRIVE PRESENT FLAG
3116 011256 104401 001205          TYPE  ,$CRLF
3117 011262 010046          MOV    R0,-(SP)     ;;SAVE R0 FOR TYPEOUT
3118          ;;TYPE DRIVE #
3119 011264 104403          TYPOS
3120 011266 001          .BYTE 1            ;;GO TYPE--OCTAL ASCII
3121 011267 000          .BYTE 0            ;;TYPE 1 DIGIT(S)
3122          ;;SUPPRESS LEADING ZEROS
3123 011270 005721          16$:  TST    (R1)+    ;ADV POINTER TO NEXT FLAG
3124 011272 005200          INC    R0           ;INC DRIVE #
3125 011274 022700 000010          CMP    #8.,R0      ;ALL 8 TESTED?
3126 011300 001402          BEQ   17$           ;BR IF YES
3127
3128 011302 006002          ROR    R2           ;ELSE GET NEXT BIT OFF DEVICE MAP
3129 011304 000756          BR    15$          ;& TRY AGAIN
3130
3131 011306 005737 005460          17$:  TST    DRIVS    ;SEE IF MORE DRIVES PRESENT
3132 011312 001402          BEQ   18$           ;BR IF NO
3133 011314 000137 012010          JMP    VERIFY       ;ELSE EXIT TEST & SETUP FOR RK07'S.
3134
3135 011320 104034          18$:  ERROR 34       ;NO DRIVES FOUND IN $DEVN
3136 011322 000000          HALT          ;SETUP CORRECTLY & PRESS 'CONTINUE'
3137 011324 000137 010612          JMP    ST5         ;TO TRY AGAIN
3138 011330 000137 012010          20$:  JMP    VERIFY   ;DO NOT SIZE, GO THE NEXT TEST.
3139
3140 011334 012765 000040 000010  12$:  MOV    #SCLR,RKCS2(R5) ;SUBSYSTEM CLEAR
3141
3142
3143
3144
    
```

```

3145
3146 011342 013737 001424 005376      MOV    T10,TEMP1      ;SET TIMEOUT
3147 011350 004737 024542                JSR    PC,FRDY        ;FIND RDY
3148 011354 104120                ERROR  120           ;RDY NOT SET BY END OF SCLR
3149
3150 011356 005737 005510                TST    SIZFLG        ;SIZE BUS?
3151 011362 001762                BEQ    20$           ;BR IF NO
3152 011364 104401 037205                TYPE  ,MSG10        ;WILL TEST DRIVES
3153 011370 005037 005460                CLR    DRIVS        ;# OF DRIVES PRESENT
3154 011374 005000                CLR    R0           ;DRV ADDR
3155 011376 012701 005462                MOV    #DRIVO,R1    ;DRV FLAG
3156 011402
3157 011402 104415      1$:      SCOP1
3158 011404 012706 001100                MOV    #STACK,SP    ;RESTORE STK PTR
3159
3160 011410 012765 000040 000010      MOV    #SCLR,RKCS2(R5) ;SUBSYSTEM CLEAR
3161 011416 013737 001424 005376      MOV    T10,TEMP1    ;SET TIMEOUT
3162 011424 004737 024542                JSR    PC,FRDY        ;FIND RDY
3163 011430 104120                ERROR  120           ;RDY NOT SET BY END OF SCLR
3164 011432 010065 000010      MOV    R0,RKCS2(R5)  ;SELECT THE DRIVE ADDR
3165 011436 012737 000001 005340      MOV    #SELDRV,HCS1
3166 011444 053737 001170 005340      BIS    $TMP4,HCS1   ;ADD CDT IF RK07
3167 011452 013765 005340 000000      MOV    HCS1,RKCS1(R5) ;GET STATUS
3168 011460 013737 001432 005376      MOV    T50000,TEMP1
3169 011466 004737 025232                JSR    PC,DLY        ;DO DELAY TO CATCH MDS
3170 011472 013737 001424 005376      MOV    T10,TEMP1
3171 011500 004737 024542                JSR    PC,FRDY        ;FIND RDY
3172 011504 104117                ERROR  117           ;NO RDY AFTER SELECT DRIVE CMD.
3173 011506 032737 100000 005340      BIT    #CERR,HCS1
3174 011514 001052                BNE   2$
3175 011516 013737 005366 005376      MOV    HMR2,TEMP1
3176 011524 042737 177770 005376      BIC    #^C<DRVMSK>,TEMP1
3177 011532 020037 005376                CMP    R0,TEMP1     ;S/B SAME
3178 011536 001020                BNE   3$
3179 011540 005700                TST   R0
3180 011542 001003                BNE   4$
3181 011544 005737 005452                TST   DDPCH         ;SEE IF XXDP CHAIN MODE
3182 011550 001016                BNE   5$
3183 011552 005237 005460      4$:      INC    DRIVS        ;INC DRIVE COUNT.
3184 011556 005211                INC    (R1)         ;SET DRIVE PRESENT FLAG
3185 011560 053711 001170                BIS    $TMP4,(R1)   ;ADD CDT IF RK07
3186 011564 104401 001205                TYPE  ,$CRLF
3187 011570 010046                MOV    R0,-(SP)     ;;SAVE R0 FOR TYPEOUT
3188
3189 011572 104403                TYPOS
3190 011574 001                .BYTE 1            ;;TYPE DR #
3191 011575 000                .BYTE 0            ;;TYPE 1 DIGIT(S)
3192 011576 000403                BR    5$            ;;SUPPRESS LEADING ZEROS
3193
3194 011600 004737 025250      3$:      JSR    PC,BYP        ;TYPE BYPASS DR #
3195 011604 104001                ERROR  1            ;WRITTEN DR # DOES NOT MATCH RKMR2 DR #
3196
3197 011606 005721      5$:      TST   (R1)+        ;SHIFT PTR TO NEXT DR. FLAG
3198 011610 005200                INC   R0           ;INC DR #
3199 011612 005037 001170                CLR   $TMP4        ;CLEAR RK07 FLAG FOR NEXT DRIVE
3200 011616 022700 000010                CMP   #8.,R0

```

```

3201 011622 001267          BNE      1$          ;MORE LEFT.
3202 011624 005737 005460   TST      DRIVS
3203 011630 001065          BNE      10$
3204 011632 104061          ERROR   61          ;NO DRIVES SEEN
3205 011634 000000          HALT
3206 011636 000137 010612   JMP      ST5        ;SETUP & PRESS 'CONTINUE'
3207
3208 011642 032737 000040 005354 2$:   BIT      #DTYE,HER
3209 011650 001405          BEQ     13$
3210 011652 012737 002000 001170   MOV     #CDT,$TMP4  ;ADD CDT
3211 011660 000137 011402          JMP     1$          ;TRY AGAIN
3212 011664 032737 001000 005342 13$:  BIT      #MDS,HCS2
3213 011672 001015          BNE     6$
3214 011674 032737 000400 005342   BIT      #UFE,HCS2
3215 011702 001015          BNE     7$
3216 011704 032737 000001 005352   BIT      #DRA,HDS
3217 011712 001015          BNE     8$
3218 011714 032737 010000 005342   BIT      #NED,HCS2
3219 011722 001424          BEQ     9$
3220 011724 000730          BR      5$
3221
3222 011726 004737 025250          6$:   JSR     PC,BYP      ;TYPE BYP DR #
3223 011732 104002          ERROR   2$          ;MDS DETECTED
3224 011734 000724          BR      5$
3225
3226 011736 004737 025250          7$:   JSR     PC,BYP
3227 011742 104003          ERROR   3$          ;UFE DETECTED
3228 011744 000720          BR      5$
3229
3230 011746 032737 010000 005342 8$:   BIT      #NED,HCS2
3231 011754 001676          BEQ     4$
3232 011756 104401 037412   TYPE   ,MSG15      ;DRV#
3233 011762 010046          MOV     R0,-(SP)   ;:SAVE R0 FOR TYPEOUT
3234
3235 011764 104403          TYPOS
3236 011766 001          .BYTE  1          ;:TYPE DR#
3237 011767 000          .BYTE  0          ;:GO TYPE--OCTAL ASCII
3238 011770 104010          ERROR  10         ;:TYPE 1 DIGIT(S)
3239 011772 000705          BR      5$        ;:SUPPRESS LEADING ZEROS
3240
3241 011774 004737 025250          9$:   JSR     PC,BYP
3242 012000 104004          ERROR   4$          ;NO DRA & NO NED = OTHER PORT SELECTED
3243 012002 000701          BR      5$
3244 012004 000137 012402 10$:  JMP     NUDRV
3245
3246 012010          VERIFY:
3247
3248
3249
3250
3251
3252
3253
3254
3255
3256
    
```

```

:*****
:*TEST 3          VERIFY OPERATOR DRIVE SELECTIONS
:*
:* THIS TEST IS ENTERED ONLY IF DRIVE SELECTION IS NOT
:* DEFAULTED. EVERY DRIVE FROM 0 TO 7 IS ADDRESSED &
:* CONTROLLER ERROR (CERR) IS EXAMINED. IF NOT SET, THE
:* PROGRAM WILL ASSUME THE DRIVE IS PRESENT AS AN RK06.
:* IF CERR WAS SET, THAT DRIVE WILL BE BYPASSED
    
```

CZ  
CZ



3257  
3258  
3259  
3260  
3261  
3262  
3263  
3264  
3265  
3266  
3267  
3268  
3269  
3270  
3271  
3272  
3273  
3274  
3275  
3276  
3277  
3278  
3279  
3280  
3281  
3282  
3283  
3284  
3285  
3286  
3287  
3288  
3289  
3290  
3291  
3292  
3293  
3294  
3295  
3296  
3297  
3298  
3299  
3300  
3301  
3302  
3303  
3304  
3305  
3306  
3307  
3308  
3309  
3310  
3311  
3312

012010 000004  
012012 012737 000001 001174  
012020 012706 001100  
012024 005000  
012026 012701 005462  
012032  
012032 104415  
012034 012706 001100  
012040 012765 000040 000010  
012046 013737 001424 005376  
012054 004737 024542  
012060 104120  
012062 010065 000010  
012066 012737 000001 005340  
012074 053737 001170 005340  
012102 013765 005340 000000  
012110 013737 001432 005376  
012116 004737 025232  
012122 013737 001424 005376  
012130 004737 024542  
012134 104117  
012136 032737 100000 005340  
012144 001036  
012146 013737 005366 005376  
012154 042737 177770 005376  
012162 020037 005376  
012166 001014  
012170 005711  
012172 001402  
012174 053711 001170  
012200 005721  
012202 005200  
012204 005037 001170  
012210 022700 000010  
012214 001306  
012216 000475  
012220 004737 025250  
012224 104001  
012226 005711  
012230 001763  
012232 005337 005460  
012236 005011  
012240 000757  
012242 032737 000040 005354  
012250 001405  
012252 012737 002000 001170  
012260 000137 012032

```

: * ONLY IF THE ERROR WAS A RESULT OF MDS OR UFE SET OR BOTH
: * NED & DRA RESET (WRONG PORT). IF CERR IS A RESULT OF
: * NED ONLY, IT IS CHECKED AGAINST THE INPUTTED INFOR TO
: * VERIFY IT WAS NOT SPECIFIED.
: * IF CERR DUE TO DTYE, DRIVE WILL BE TESTED AS RK07.
: *
: *****
TST3: SCOPE
MOV #1,STIMES ;:DO 1 ITERATION
MOV #STACK,SP ;:RESTORE STK PTR
CLR R0 ;:DRIVE ADDR
MOV #DRIVO,R1 ;:DRIVE FLAG

1$: SCOPI
MOV #STACK,SP ;:RESTORE STK PTR

MOV #SCLR,RKCS2(R5) ;:SUBSYSTEM CLEAR
MOV T10,TEMP1 ;:SET TIME OUT
JSR PC,FRDY ;:FIND RDY
ERROR 120 ;:NO RDY AFTER SCLR
MOV R0,RKCS2(R5) ;:DRV ADDR
MOV #SELDRV,HCS1
BIS $TMP4,HCS1 ;:ADD CDT IF RK07
MOV HCS1,RKCS1(R5) ;:GET STATUS
MOV T50000,TEMP1
JSR PC,DLY ;:DO DELAY TO CATCH MDS
MOV T10,TEMP1
JSR PC,FRDY ;:FIND RDY
ERROR 117 ;:NO RDY AFTER SELECT DRIVE CMD.
BIT #CERR,HCS1
BNE 2$
MOV HMR2,TEMP1
BIC #^C<DRVMSK>,TEMP1
CMP R0,TEMP1 ;:S/B SAME
BNE 3$

11$: TST (R1)
BEQ 4$
BIS $TMP4,(R1) ;:ADD CDT IF RK07
4$: TST (R1)+ ;:SHIFT PTR TO NEXT DR FLAG
INC R0 ;:INC DR#
CLR $TMP4 ;:CLEAR RK07 FLAG FOR NEXT DRIVE
CMP #8.,R0
BNE 1$ ;:MORE LEFT
BR TST4 ;:GO TO NEXT TEST

3$: JSR PC,BYP ;:TRY BYPASS DRIVE#
ERROR 1 ;:WRITTEN DR# DOES NOT MATCH RKMR2 DR#
TST (R1)
BEQ 4$ ;:BRANCH IF NOT SPEC BY INPUT
12$: DEC DRIVS ;:DECREMENT TOTAL DRIVS
CLR (R1) ;:CLEAR DRIVE FLAG
BR 4$

2$: BIT #DTYE,HER
BEQ 13$
MOV #CDT,$TMP4 ;:ADD CDT
JMP 1$ ;:TRY AGAIN
```

```

3313
3314
3315 012264 032737 001000 005342 13$: BIT #MDS,HCS2
3316 012272 001027 BNE 6$
3317 012274 032737 000400 005342 BIT #UFE,HCS2
3318 012302 001027 BNE 7$
3319 012304 032737 000001 005352 BIT #DRA,HDS
3320 012312 001005 BNE 8$
3321 012314 032737 010000 005342 BIT #NED,HCS2
3322 012322 001423 BEQ 9$
3323 012324 000404 BR 10$
3324 012326 032737 010000 005342 8$: BIT #NED,HCS2
3325 012334 001715 BEQ 11$
3326 012336 005711 10$: TST (R1)
3327 012340 001717 BEQ 4$
3328
3329 012342 004737 025250 JSR PC,BYP ;TYPE BYPASS DRIVE#
3330 012346 104006 ERROR 6
3331 012350 000730 BR 12$
3332
3333 012352 004737 025250 6$: JSR PC,BYP ;TYPE BYPASS DRIVE#
3334 012356 104002 ERROR 2 ;MDS DETECTED
3335 012360 000724 BR 12$
3336
3337 012362 004737 025250 7$: JSR PC,BYP
3338 012366 104003 ERROR 3 ;UFE DETECTED
3339 012370 000720 BR 12$
3340
3341 012372 004737 025250 9$: JSR PC,BYP
3342 012376 104004 ERROR 4 ;DRA & NED RESET - OTHER PORT SELECTED
3343 012400 000714 BR 12$
3344
3345
3346
3347
3348 ; THIS PART OF THE PROGRAM WILL BE REPEATED FOR EACH
3349 ; DRIVE PRESENT
3350
3351 ; 'SUNIT' CONTAINS THE ADDRESS OF THE DRIVE CURRENTLY
3352 ; UNDER TEST
3353
3354 012402 005037 001502 NUDRV: CLR BYPCERR ;ENTER HERE FROM LAST TEST
3355 ;TEST CERR IN 'FRDY'
3356 012406 005037 001170 CLR $TMP4 ;RK07 FLAG
3357

```

```
3358
3359
3360
3361
3362
3363
3364
3365
3366
3367 012412 000004
3368 012414 012737 000001 001174
3369 012422 012706 001100
3370 012426 012737 000004 001214
3371 012434 012737 000004 001102
3372
3373 012442 005737 005460
3374 012446 001004
3375 012450 104401 037772
3376 012454 000137 023734
3377
3378 012460 013701 001362 4$: MOV DRVPTR,R1 ;ADDR OF NEXT DRIVE FLAG
3379 012464 005737 001220 TST $DEVCT ;IS FIRST DRIVE BEING CHECKED
3380 012470 001402 BEQ 2$ ;YES, BRANCH
3381 012472 005237 001222 1$: INC $UNIT ;INCR DRIVE ADDR TO NEXT DRIVE
3382 012476 005711 2$: TST (R1) ;IS DRIVE PRESENT?
3383 012500 001002 BNE 5$ ;BR IF YES
3384 012502 005721 TST (R1)+ ;ELSE FIND NEXT DRIVE
3385 012504 000772 BR 1$
3386 012506 005737 005452 5$: TST DDPCH ;DDP CHAIN MODE?
3387 012512 001405 BEQ 3$ ;BR IF NO
3388 012514 005737 001222 TST $UNIT ;ELSE SEE IF DRV 0
3389 012520 001002 BNE 3$ ;BR IF NO
3390 012522 005721 TST (R1)+ ;ELSE FIND NEXT DRIVE PRESENT
3391 012524 000762 BR 1$
3392
3393 012526 032721 002000 3$: BIT #CDT,(R1)+ ;SEE IF DRIVE UNDER TEST IS RK07
3394 012532 001403 BEQ 6$ ;BR IF NO
3395 012534 012737 002000 001170 MOV #CDT,$TMP4 ;ELSE SET RK07 FLAG
3396 012542 010137 001362 6$: MOV R1,DRVPTR ;STORE POINTER TO NEXT DR FLAG
3397 012546 104401 037412 TYPE ,MSG15 ;"DRIVE"
3398 012552 013700 001222 MOV $UNIT,R0
3399 012556 010046 MOV R0,-(SP) ;SAVE R0 FOR TYPEOUT
3400 ; ;DRIVE #
3401 012560 104403 TYPOS ;GO TYPE--OCTAL ASCII
3402 012562 001 .BYTE 1 ;TYPE 1 DIGIT(S)
3403 012563 000 .BYTE 0 ;SUPPRESS LEADING ZEROS
3404
3405 ; TYPE ,$CRLF
3406
3407 012564 005737 001170 TST $TMP4 ;SEE IF RK07 UNDER TEST
3408 012570 001014 BNE 7$ ;BR IF YES
3409 012572 012737 000632 012654 MOV #632,LC ;ELSE LOAD RK06 PARAMETERS
3410 012600 005037 012662 CLR E,DDT
3411 012604 012737 000777 012656 MOV #777,MASK
3412 012612 012737 160017 012660 MOV #160017,MASK1
3413 012620 000421 BR TST5 ;GOTO NEXT TEST
```

```
3414  
3415 012622 012737 001456 012654 78: MOV #1456,LC ;LOAD RK07 PARAMETERS  
3416 012630 012737 000400 012662 MOV #D.DDT,E.DDT  
3417 012636 012737 001777 012656 MOV #1777,MASK  
3418 012644 012737 140017 012660 MOV #140017,MASK1  
3419 012652 000404 BR TST5 ;;GOTO NEXT TEST  
3420  
3421 012654 000000 LC: 0 ;LAST CYL  
3422 012656 000000 MASK: 0  
3423 012660 000000 MASK1: 0  
3424 012662 000000 E.DDT: 0 ;EXPECTED DRIVE TYPE TO E.A0
```

```

3425
3426 012664 PFSRT: ;ENTER HERE FOR POWER FAIL RESTART
3427 :*****
3428 :*TEST 5 PRINT DRIVE SERIAL NUMBER
3429 :*
3430 :* THIS TEST READS & PRINTS THE DRIVE SERIAL # FROM MSG A, WORD 11
3431 :* IN BCD & IS PERFORMED ON THE 1ST PASS ONLY
3432 :*
3433 :*****
3434 012664 000004 TST5: SCOPE
3435 012666 012737 000001 001174 MOV #1,$TIMES ;:DO 1 ITERATION
3436 012674 012706 001100 MOV #STACK,SP ;:RESTORE STK PTR
3437
3438 012700 005737 001216 TST $PASS
3439 012704 001042 BNE TST6 ;:GO TO NEXT IF NOT FIRST PASS
3440 012706 004737 026436 JSR PC,SUBCLR ;:DO SUBSYS CLEAR
3441 012712 104024 ERROR 24 ;:CERR AFTER SCLR
3442
3443 012714 104401 037424 TYPE ,MSG16 ;:DRIVE SERIAL NO.
3444 012720 012765 000003 000026 MOV #3,RKMR1(R5) ;:SELECT BYTE 3
3445 012726 004737 026106 JSR PC,GSTAT ;:GET STATUS
3446 012732 013701 005366 MOV HMR2,R1 ;:GET SERIAL #
3447 012736 012704 035132 MOV #SOCTLV,R4 ;:GET ADDR CHAR BUFF
3448 012742 010446 MOV R4,-(SP) ;:STORE ON STACK FOR $SUPRS
3449 012744 012703 000003 MOV #3,R3 ;:SETUP CHAR COOUNT
3450 012750 006101 ROL R1 ;:INITIALIZE BIT POSITIONS
3451 012752 006101 ROL R1
3452 012754 006101 1$: ROL R1 ;:GET NEXT 4 BITS
3453 012756 006101 ROL R1
3454 012760 006101 ROL R1
3455 012762 006101 ROL R1
3456 012764 010100 MOV R1,R0 ;:GET WORKING COPY
3457 012766 042700 177760 BIC #177760,R0 ;:CLEAR ALL BUT LOW 4 BITS
3458 012772 052700 000060 BIS #60,R0 ;:CONVERT TO ASCII DIGIT
3459 012776 110024 MOVB R0,(R4)+ ;:PUT ASCII DIGIT INTO CHAR BUFF
3460 013000 005303 DEC R3
3461 013002 001364 BNE 1$ ;:BR IF ALL 3 CHARS NOT DONE
3462 013004 105014 CLR B (R4) ;:ELSE INSERT NULL TERMINATOR
3463
3464 013006 004737 035400 JSR PC,$SUPRS ;:TYPE
3465 : TYPE ,SCLF
3466 : TYPE ,SCLF
3467
3468 :*****
3469 :*TEST 6 SET VV WITH PACK COMMAND
3470 :*
3471 :* IF VV IS RESET, THE PACK COMMAND IS USED TO SET IT.
3472 :*
3473 :*****
3474 013012 000004 TST6: SCOPE
3475 013014 012737 000001 001174 MOV #1,$TIMES ;:DO 1 ITERATION
3476 013022 012706 001100 MOV #STACK,SP ;:RESTORE STK PTR
3477
3478 013026 004737 026436 JSR PC,SUBCLR
3479 013032 104024 ERROR 24 ;:CERR AFTER SCLR
3480
    
```

3481 013034 032737 000100 005366 BIT #D.VV,HMR2  
3482 013042 001021 BNE TST7 ;:GO TO NEXT TEST IF VV SET  
3483 013044 012765 100000 000000 MOV #CCLR,RKCS1(R5)  
3484 013052 013765 001222 000010 MOV \$UNIT,RKCS2(R5);DRIVE #  
3485 013060 012737 000003 005340 MOV #PACK,HCS1  
3486 013066 004737 024446 JSR PC,DOCMD ;DO PACK CMD & GET CONTR RDY  
3487 013072 104116 ERROR 116 ;CONTR NOT RDY  
3488  
3489 013074 032737 000100 005366 BIT #D.VV,HMR2  
3490 013102 001001 BNE 64\$  
3491 013104 104027 ERROR 27 ;VOLUME VALID NOT SET AFTER PACK CMD  
3492 013106  
3493  
3494  
3495  
3496  
3497  
3498  
3499  
3500

64\$:

\*\*\*\*\*  
\*TEST 7 UNLOAD DRIVE TO BE TESTED  
\*  
\* THIS TEST UNLOADS THE DRIVE TO BE TESTED NEXT,  
\* WAITS FOR ATTN & VERIFIES IT CAME FROM THE CORRECT DRIVE.  
\*  
\*\*\*\*\*

3501  
3502 013106 000004 TST7: SCOPE  
3503 013110 012737 000001 001174 MOV #1,\$TIMES ;:DO 1 ITERATION  
3504 013116 012706 001100 MOV #STACK,SP  
3505  
3506 013122 005237 005322 INC UNLD ;USED TO CHECK FOR VALID HALT  
3507  
3508 013126 004737 026436 JSR PC,SUBCLR  
3509 013132 104024 ERROR 24 ;CERR AFTER SCLR  
3510  
3511 013134 012737 000007 005340 MOV #UNLOAD,HCS1  
3512 013142 004737 024446 JSR PC,DOCMD ;DO UNLOAD CMD & GET CONTR READY  
3513 013146 104017 ERROR 17 ;NO RDY AFTER UNLD CMD  
3514 013150 004737 025024 JSR PC,TSTATN  
3515 013154 104020 ERROR 20 ;NO ATTN AFTER UNLOAD CMD  
3516  
3517  
3518  
3519

.SBTTL OPERATOR INTERVENTION TESTS

\*\*\*\*\*  
\*TEST 10 INTERLOCKS TESTS  
\*  
\* THIS TEST VERIFIES THAT THE DOOR & CARTRIDGE STATUS BITS  
\* ARE OPERATING PROPERLY IN MESSAGE A0 & THAT THE REMOVAL  
\* OF THE CARTRIDGE CLEARS VOLUME VALID.  
\* IT FURTHER VERIFIES ALL REMAINING STATUS BITS OF  
\* MESSAGE A & B, WORDS 0 & 1.  
\* THIS TEST ALSO CHECKS THAT THE SPINDLE CANNOT BE STARTED  
\* WHEN THE DOOR IS OPEN OR THE CARTRIDGE IS REMOVED.  
\* IT ALSO VERIFIES THAT LOSS OF VOLUME VALID RESETS SACK WHICH  
\* ASSERTS NON EXISTENT DRIVE IN RKCS2  
\*  
\*\*\*\*\*

3520  
3521  
3522  
3523  
3524  
3525  
3526  
3527  
3528  
3529  
3530  
3531  
3532  
3533  
3534 013156 000004 TST10: SCOPE  
3535 013160 012737 000001 001174 MOV #1,\$TIMES ;:DO 1 ITERATION  
3536 013166 012706 001100 MOV #STACK,SP ;RESET STACK PTR

```

3537
3538 013172 004737 026436 JSR PC,SUBCLR
3539 013176 104024 ERROR 24 ;CERR AFTER SCLR
3540
3541 013200 005237 001502 INC BYPCERR ;DONT DO CKCERR ROUTINE
3542
3543 013204 104401 037127 TYPE ,MSG8 ;INTERLOCKS TEST
3544 013210 104401 042017 TYPE ,MSG52 ;CONT-E TO EXIT OR SPACE TO CONTINUE
3545 013214 004737 030412 JSR PC,CCSP ;INPUT CONT-E OR SPACE
3546 013220 000137 014020 JMP 8$ ;RET HERE FOR CONT-E
3547
3548 013224 104401 041473 TYPE ,MSG47 ;VERIFY DOOR CANNOT BE OPENED
3549 013230 104401 040625 TYPE ,MSG37 ;DEPRESS SPACE WHEN DONE
3550 013234 004737 030452 JSR PC,GETSP ;GET SPACE
3551
3552 013240 104401 043464 TYPE ,MSG65 ;DEPRESS RUN/STOP SW TO 'STOP'
3553 013244 104401 040130 TYPE ,MSG30 ;OPEN DOOR & LEAVE IT OPEN
3554 013250 104401 040625 TYPE ,MSG37 ;DEPRESS SPACE BAR WHEN DONE
3555 013254 004737 030452 JSR PC,GETSP ;INPUT SPACE
3556 013260 012765 000001 000026 MOV #1,RKMR1(R5) ;SELECT WORD 1
3557 013266 004737 026106 JSR PC,GSTAT
3558 013272 032737 000200 005366 BIT #D.DOOR,HMR2
3559 013300 001401 BEQ 1$
3560 013302 104044 ERROR 44 ;DOOR STATUS BIT NOT CLEARED
3561
3562 013304 012737 000140 005430 1$: MOV #<D.DRA!D.VV>,E.A0 ;EXPECTED A0
3563 013312 005037 005432 CLR E.B0
3564 013316 012737 000540 005434 MOV #<D.HDHM!D.BRHM!D.CART>,E.A1
3565 013324 012737 000001 005436 MOV #1,E.B1
3566
3567 013332 004737 025264 JSR PC,CHKMSG ;CHECK MSGS A0, B0, A1, B1
3568 013336 000000 .WORD 0!0!0 ;# MSGS SPECIFIED HERE
3569 013340 104045 ERROR 45 ;MSG A0 ERROR AFTER DRIVE UNLOADED & DOOR OPENED
3570 013342 104046 ERROR 46 ;MSH B0 ERROR
3571 013344 104063 ERROR 63 ;MSG A1 ERROR
3572 013346 104064 ERROR 64 ;MSG B1 ERROR
3573
3574 013350 004737 026436 JSR PC,SUBCLR
3575 013354 104024 ERROR 24 ;CERR AFTER SCLR
3576
3577 013356 104401 040260 TYPE ,MSG33 ;PRESS 'RUN-STOP' TO 'RUN'
3578 013362 104401 040345 TYPE ,MSG34 ;VERIFY DOES NOT START
3579 013366 104401 040625 TYPE ,MSG37
3580 013372 004737 030452 JSR PC,GETSP ;GET SPACE FROM TTY
3581
3582 013376 004737 026106 JSR PC,GSTAT
3583 013402 032737 010000 005366 BIT #D.SPIN,HMR2
3584 013410 001401 BEQ 2$
3585 013412 104065 ERROR 65 ;SPIN SET IN MSGA0
3586
3587 013414 012737 000140 005430 2$: MOV #<D.DRA!D.VV>,E.A0 ;EXPECTED MSG A0
3588 013422 005037 005432 CLR E.B0
3589 013426 012737 000540 005434 MOV #<D.HDHM!D.BRHM!D.CART>,E.A1
3590 013434 012737 000001 005436 MOV #1,E.B1
3591
3592 013442 004737 025264 JSR PC,CHKMSG ;CHECK MSGS A0, B0, A1, B1
    
```

3593	013446	000000			.WORD	0!0!0		:& MSGS SPECIFIED HERE
3594	013450	104066			ERROR	66		:MSG A0 ERROR AFTER ATTEMPT TO START SPIN & DOOR OPEN
3595	013452	104067			ERROR	67		:MSG B0 ERROR
3596	013454	104070			ERROR	70		:MSG A1 ERROR
3597	013456	104071			ERROR	71		:MSG B1 ERROR
3598								
3599	013460	004737	026436		JSR	PC,SUBCLR		
3600	013464	104024			ERROR	24		:CERR AFTER SCLR
3601								
3602	013466	104401	043464		TYPE	,MSG65		:DEPRESS 'RUN-STOP' SW TO STOP
3603	013472	104401	040220		TYPE	,MSG32		:REMOVE PACK & CLOSE DOOR
3604	013476	104401	040625		TYPE	,MSG37		
3605	013502	004737	030452		JSR	PC,GETSP		:GET SPACE FROM TTY
3606								
3607	013506	012765	000001	000026	MOV	#1,RKMR1(R5)		:SELECT WORD 1
3608	013514	004737	026106		JSR	PC,GSTAT		
3609	013520	032737	000400	005366	BIT	#D.CART,HMR2		
3610	013526	001401			BEQ	3\$		
3611	013530	104072			ERROR	72		:CARTRIDGE STATUS BIT NOT RESET
3612								
3613								
3614	013532	004737	026436		3\$: JSR	PC,SUBCLR		
3615	013536	104024			ERROR	24		:CERR AFTER SCLR
3616								
3617	013540	104401	040433		TYPE	,MSG35		:PRESS 'RUN-STOP' TO 'RUN'
3618	013544	104401	040345		TYPE	,MSG34		:VERIFY DOES NOT START
3619	013550	104401	040625		TYPE	,MSG37		
3620	013554	004737	030452		JSR	PC,GETSP		:GET SPACE FROM TTY
3621								
3622	013560	004737	026106		JSR	PC,GSTAT		
3623	013564	032737	010000	005366	BIT	#D.SPIN,HMR2		
3624	013572	001401			BEQ	5\$		
3625	013574	104100			ERROR	100		:SPIN SET IN MSG A0
3626								
3627	013576	104401	043464		5\$: TYPE	,MSG65		:PRESS 'RUN-STOP' TO 'STOP'
3628	013602	104401	040530		TYPE	,MSG36		:INSERT CARTRIDGE, SET 'RUN' SW. & CLOSE DOOR
3629	013606	104401	043724		TYPE	,MSG70		:VERIFY HEADS LOAD
3630	013612	104401	043647		TYPE	,MSG69		:DEPRESS SPACE WHEN READY GOES ON
3631	013616	004737	030452		JSR	PC,GETSP		:GET SPACE FROM TTY
3632	013622	005037	005322		CLR	UNLD		:FOR VALID HALT
3633								
3634	013626	012765	100000	000000	MOV	#CCLR,RKCS1(R5)		
3635	013634	005065	000026		CLR	RKMR1(R5)		:SELECT WORD 0
3636	013640	004737	026106		JSR	PC,GSTAT		
3637	013644	032737	000100	005366	BIT	#D.VV,HMR2		
3638	013652	001401			BEQ	7\$		
3639	013654	104033			ERROR	33		:VV NOT CLEARED AFTER PACK RE-INSERTED
3640								
3641	013656	012737	050240	005430	7\$: MOV	#<D.DSC!D.SPIN!D.DRDY!D.DRA>,E.A0		:EXPECTED A0
3642	013664	005037	005432		CLR	E.B0		
3643	013670	012737	001720	005434	MOV	#<D.SPOK!D.CART!D.SSP!D.BRHM!D.DOOR>,E.A1		
3644	013676	012737	000001	005436	MOV	#1,E.B1		
3645								
3646	013704	004737	025264		JSR	PC,CHKMSG		:CHECK MSGS A0, B0, A1, B1
3647	013710	000000			.WORD	0!0!0		:& MSGS SPECIFIED HERE
3648	013712	104074			ERROR	74		:MSG A0 ERROR AFTER PACK REINSERTED & HEADS LOADED



```

3649 013714 104075          ERROR 75          :MSH B0 ERROR
3650 013716 104076          ERROR 76          :MSG A1 ERROR
3651 013720 104077          ERROR 77          :MSG B1 ERROR
3652
3653 013722 012765 100000 000000      MOV    #CCLR,RKCS1(R5)
3654 013730 013765 001222 000010      MOV    $UNIT,RKCS2(R5) ;DRIVE #
3655 013736 012737 000003 005340      MOV    #PACK,HCS1
3656 013744 004737 024446          JSR    PC,DOCMD      ;DO PACK CMD & GET CONTR RDY
3657 013750 104116          ERROR 116         ;CONTR NOT RDY
3658
3659 013752 032737 000100 005366      BIT    #D.VV,HMR2
3660 013760 001001          BNE    64$
3661 013762 104027          ERROR 27          ;VOLUME VALID NOT SET AFTER PACK CMD
3662 013764          64$:
3663 013764 005237 005322          INC    UNLD        ;FOR VALID HALT
3664
3665 013770 004737 026436          JSR    PC,SUBCLR
3666 013774 104024          ERROR 24          ;CERR AFTER SCLR
3667
3668 013776 012737 000007 005340      MOV    #UNLOAD,HCS1
3669 014004 004737 024446          JSR    PC,DOCMD      ;DO UNLOAD CMD & GET CONTR READY
3670 014010 104017          ERROR 17          ;NO RDY AFTER UNLD CMD
3671 014012 004737 025024          JSR    PC,TSTATN
3672 014016 104020          ERROR 20          ;NO ATTN AFTER UNLOAD CMD
3673 014020          8$:
3674
3675          :*****
3676          :*TEST 11          UNIT SELECT PLUG TEST
3677          :*
3678          :*          THIS TEST VERIFIES THAT WHEN THE UNIT SELECT PLUG IS PULLED
3679          :*          OUT, THE QUAL LOGIC RESETS ATTN & VOLUME VALID, THAT
3680          :*          THE DRIVE DE-SELECTS & NON EXISTENT DRIVE ASSERTS.
3681          :*          FURTHER, THE OPERATOR IS ASKED TO INSERT ANY NUMBER OF
3682          :*          UNIT SELECT PLUGS. THE PROGRAM WILL RESPOND BY TYPING
3683          :*          THE PLUG CODE NUMBER AS SOON AS IT IS INSERTED.
3684          :*          THIS PORTION OF THE TEST IS TERMINATED AT ANY TIME BY A CONT-E
3685          :*
3686          :*****
3687          :TST11: SCOPE
3688          014020 000004          MOV    #1,$TIMES      ;;DO 1 ITERATION
3689          014022 012737 000001 001174      MOV    #STACK,SP     ;RESTORE STACK PTR
3690          014030 012706 001100
3691          014034 004737 026436          JSR    PC,SUBCLR
3692          014040 104024          ERROR 24          ;CERR AFTER SCLR
3693
3694          014042 104401 037500          TYPE   ,MSG18        ;UNIT SELECT PLUG TEST
3695          014046 104401 042017          TYPE   ,MSG52        ;CONT-E TO EXIT OR SPACE TO CONTINUE
3696          014052 004737 030412          JSR    PC,CCSP       ;INPUT CONT-E OR SPACE
3697          014056 000137 014652          JMP    12$          ;RETURN HERE FOR CONT-E
3698
3699          014062 012765 000020 000026      MOV    #PAT,RKMR1(R5) ;EVEN PARITY TO GET DSC & ATTN
3700          014070 004737 026106          JSR    PC,GSTAT
3701          014074 032737 001000 005370      BIT    #D.PAR,HMR3   ;SEE IF PARITY SET
3702          014102 001001          BNE    2$          ;BR IF YES
3703          014104 104114          ERROR 114         ;PARITY BIT NOT SET AFTER SEL DRIVE CMD
3704          ;WITH EVEN PARITY ISSUED.

```

```

3705 014106 032737 040000 005366 2$: BIT #D.DSC,HMR2 ;SEE IF DSC SET
3706 014114 001001 BNE 1$
3707 014116 104107 ERROR 107 ;DSC NOT SET AFTER SEL DRV WITH EVEN PARITY
3708
3709 014120 012765 100000 000000 1$: MOV #CCLR,RKCS1(R5)
3710 014126 004737 026106 JSR PC,GSTAT
3711 014132 004737 025024 JSR PC,TSTATN
3712 014136 104113 ERROR 113 ;NO ATTN AFTER SEL DRV WITH EVEN PARITY
3713
3714 014140 104401 037153 TYPE ,MSG9 ;REMOVE UNIT SELECT PLUG
3715 014144 104401 040625 TYPE ,MSG37 ;DEPRESS SPACE WHEN DONE
3716 014150 004737 030452 JSR PC,GETSP ;GET SPACE
3717
3718 014154 012765 100000 000000 MOV #CCLR,RKCS1(R5)
3719 014162 004737 026106 JSR PC,GSTAT
3720 014166 004737 025024 JSR PC,TSTATN ;RETURN HERE FOR SPACE
3721 014172 000401 BR 3$ ;NO ATTN
3722 014174 104111 ERROR 111 ;REMOVING UNIT SEL PLUG, DID NOT
3723 ;DISABLE ATTN.
3724 014176 012765 100000 000000 3$: MOV #CCLR,RKCS1(R5)
3725 014204 004737 026106 JSR PC,GSTAT
3726 014210 032765 010000 000010 BIT #NED,RKCS2(R5)
3727 014216 001001 BNE 4$
3728 014220 104112 ERROR 112 ;REMOVING UNIT SEL PLUG DID NOT
3729 ;ASSERT NON-EXISTENT DRIVE
3730
3731 014222 104401 043532 4$: TYPE ,MSG67 ;DESELECT ALL OTHER PORTS
3732 014226 104401 040625 TYPE ,MSG37 ;TYPE SPACE WHEN DONE
3733 014232 004737 030452 JSR PC,GETSP ;GET SPACE
3734
3735 014236 104401 040671 TYPE ,MSG38 ;INSERT UNIT SELECT PLUGS
3736 014242 104401 041027 TYPE ,MSG39 ;DEPRESS CONTROL-E WHEN FINISHED
3737 014246 104401 042300 TYPE ,MSG55 ;EXIT WITH CORRECT UNIT SELECT PLUG #
3738 014252 013746 001222 MOV $UNIT,-(SP) ;:SAVE $UNIT FOR TYPEOUT
3739 ;:TYPE CORRECT#
3740 014256 104403 TYPOS ;:GO TYPE--OCTAL ASCII
3741 014260 001 .BYTE 1 ;:TYPE 1 DIGIT(S)
3742 014261 000 .BYTE 0 ;:SUPPRESS LEADING ZEROS
3743 014262 104401 001205 TYPE ,$CRLF
3744
3745 014266 105037 033414 CLRB $TKQSRT ;CLEAR PREVIOUS INFO
3746 014272 005037 001160 CLR $TMP0
3747
3748 014276 004737 033416 5$: JSR PC,$TKINT
3749 014302 113737 033414 001160 MOV $TKQSRT,$TMP0 ;GET CHAR IF THERE
3750 014310 042737 177600 001160 BIC #^C<177>,$TMP0 ;GET RID OF JUNK, IF ANY
3751 014316 005737 001160 TST $TMP0
3752 014322 001422 BEQ 6$ ;BR IF NOTHING TYPED YET
3753 014324 023727 001160 000005 CMP $TMP0,#5 ;SEE IF CONT-E
3754 014332 001470 BEQ 9$ ;BR IF YES
3755 014334 023727 001160 000003 CMP $TMP0,#3 ;SEE IF CONT-C
3756 014342 001004 BNE 13$ ;BR IF NO
3757 014344 004737 033416 JSR PC,$TKINT ;ENABLE KB INT
3758 014350 000137 030502 JMP STOP1 ;ELSE DO VALID HALT
3759 014354 104401 040212 13$: TYPE ,MSG31 ;?
3760 014360 105037 033414 CLRB $TKQSRT

```

```

3761 014364 004737 033416          JSR    PC,$TKINT
3762
3763 014370 005000                    6$:   CLR    R0
3764 014372 012765 100000 000000 7$:   MOV    #CCLR,RKCS1(R5)
3765 014400 010065 000010                    MOV    R0,RKCS2(R5)      ;DRIVE NO.
3766 014404 012737 000001 005340      MOV    #SELDRV,HCS1
3767 014412 004737 024446          JSR    PC,DOCMD          ;DO SELDRV (STATUS) CMD & GET CONTR RDY
3768 014416 104117          ERROR  117              ;CONTR RDY NOT SET
3769
3770 014420 032765 010000 000010      BIT    #NED,RKCS2(R5)   ;SEE IF UNIT SELECT PLUG INSERTED
3771 014426 001405          BEQ    8$                ;& IF MATCH DRIVE NO IN CS2. BR IF YES
3772 014430 005200          INC    R0
3773 014432 020027 000010          CMP    R0,#8.          ;ALL 8 DRIVE NOS TRIED?
3774 014436 001355          BNE    7$                ;BR IF NO
3775 014440 000716          BR     5$                ;ELSE RETURN
3776
3777 014442 032737 000100 005366 8$:   BIT    #D.VV,HMR2
3778 014450 001312          BNE    5$                ;TYPE SAME UNIT SELECT PLUG RDY ONCE
3779 014452 012765 100000 000000      MOV    #CCLR,RKCS1(R5)
3780 014460 010065 000010                    MOV    R0,RKCS2(R5)      ;DRIVE ADDR
3781 014464 012737 000003 005340      MOV    #PACK,HCS1
3782 014472 004737 024446          JSR    PC,DOCMD          ;DO PACK CMD & GET CONTR RDY
3783 014476 104116          ERROR  116              ;CONTR NOT RDY
3784 014500 010046          MOV    R0,-(SP)         ;:SAVE R0 FOR TYPEOUT
3785
3786 014502 104403          TYPOS
3787 014504 001                .BYTE  1                ;:GO TYPE--OCTAL ASCII
3788 014505 000                .BYTE  0                ;:TYPE 1 DIGIT(S)
3789 014506 104401 001205          TYPE  ,SCLF             ;:SUPPRESS LEADING ZEROS
3790 014512 000671          BR     5$
3791
3792 014514 004737 033416          9$:   JSR    PC,$TKINT        ;ENABLE KB INT
3793 014520 012765 100000 000000      MOV    #CCLR,RKCS1(R5)
3794 014526 013765 001222 000010      MOV    $UNIT,RKCS2(R5)
3795 014534 012737 000001 005340      MOV    #SELDRV,HCS1
3796 014542 004737 024446          JSR    PC,DOCMD          ;DO SELDRV (STATUS) CMD & GET CONTR RDY
3797 014546 104117          ERROR  117              ;CONT NOT RDY AFTER SEL DRV CMD
3798
3799 014550 032765 010000 000010      BIT    #NED,RKCS2(R5)   ;SEE IF CORRECT PLUG FOR EXIT
3800 014556 001411          BEQ    10$               ;BR IF YES
3801 014560 104401 042300          TYPE  ,MSG55            ;EXIT WITH CORRECT UNIT SELECT PLUG NO.
3802 014564 013746 001222          MOV    $UNIT,-(SP)     ;:SAVE $UNIT FOR TYPEOUT
3803
3804 014570 104403          TYPOS
3805 014572 001                .BYTE  1                ;:GO TYPE--OCTAL ASCII
3806 014573 000                .BYTE  0                ;:TYPE 1 DIGIT(S)
3807 014574 104401 041027          TYPE  ,MSG39            ;:SUPPRESS LEADING ZEROS
3808 014600 000636          BR     5$                ;DEPRESS CONT-E WHEN DONE
3809
3810 014602 004737 026436          10$:  JSR    PC,SUBCLR
3811 014606 104024          ERROR  24                ;CERR AFTER SCLR
3812
3813 014610 004737 026106          JSR    PC,GSTAT
3814 014614 032737 000100 005366      BIT    #D.VV,HMR2
3815 014622 001005          BNE    11$
3816 014624 104401 041072          TYPE  ,MSG40            ;VV NOT SET, INSERT UNIT SELECT PLUG

```

```

3817 014630 104401 041027          TYPE  ,MSG39          ;DEPRESS CONT-E TO EXIT TEST
3818 014630 000620                    BR      5$
3819
3820 014636 104401 043752          11$:  TYPE  ,MSG71          ;SELECT CORRECT PORT ON OTHER DRIVES
3821 014642 104401 040625          TYPE  ,MSG37          ;TYPE SPACE WHEN DONE
3822 014646 004737 030452          JSR    PC,GETSP       ;GET SPACE
3823 014652
3824
3825
3826
3827
3828
3829
3830
3831
3832
3833
3834 014652 000004                    TST12: SCOPE
3835 014654 012737 000001 001174  MOV    #1,$TIMES      ;;DO 1 ITERATION
3836 014662 012706 001100          MOV    #STACK,SP     ;RESTORE STACK PTR
3837
3838 014666 004737 026436          JSR    PC,SUBCLR
3839 014672 104024                    ERROR  24             ;CERR AFTER SCLR
3840
3841 014674 104401 037532          TYPE  ,MSG19          ;PORT SELECTION TESTS
3842 014700 104401 042017          TYPE  ,MSG52          ;DEPRESS CONT-E TO EXIT OR SPACE BAR TO CONTINUE
3843 014704 004737 030412          JSR    PC,CCSP
3844 014710 000137 015070          JMP    4$             ;RETURN HERE FOR CONT-E
3845
3846 014714 104401 041204          TYPE  ,MSG41          ;SWITCH TO OTHER PORT
3847 014720 104401 040625          TYPE  ,MSG37          ;DEPRESS SPACE WHEN DONE
3848 014724 004737 030452          JSR    PC,GETSP       ;GET SPACE
3849
3850 014730 004737 026106          JSR    PC,GSTAT
3851 014734 032765 010000 000010  BIT    #NED,RKCS2(R5)
3852 014742 001001                    BNE    1$
3853 014744 104122                    ERROR  122           ;NED NOT SET AFTER WRONG PORT SELECTED
3854
3855
3856 014746 104401 041264          1$:  TYPE  ,MSG42          ;SELECT CORRECT PORT
3857 014752 104401 040625          TYPE  ,MSG37          ;DEPRESS SPACE BAR WHEN DONE
3858 014756 004737 030452          JSR    PC,GETSP       ;GET SPACE
3859
3860 014762 004737 026436          JSR    PC,SUBCLR
3861 014766 104024                    ERROR  24             ;CERR AFTER SCLR
3862
3863 014770 032765 010000 000010  BIT    #NED,RKCS2(R5)
3864 014776 001401                    BEG    2$
3865 015000 104130                    ERROR  130           ;NED NOT CLEARED AFTER CORRECT PORT SELECTED
3866
3867 015002 104401 041342          2$:  TYPE  ,MSG43          ;DESELECT BOTH PORTS
3868 015006 104401 040625          TYPE  ,MSG37          ;DEPRESS SPACE BAR WHEN DONE
3869 015012 004737 030452          JSR    PC,GETSP
3870 015016 004737 026106          JSR    PC,GSTAT
3871 015022 032765 010000 000010  BIT    #NED,RKCS2(R5)
3872 015030 001001                    BNE    3$
  
```

```

3873 015032 104133          ERROR 133          ;NED NOT SET AFTER BOTH PORTS DESELECTED
3874
3875
3876 015034 104401 041372    3$:  TYPE      ,MSG44          ;SELECT CORRECT PORT
3877 015040 104401 040625    TYPE      ,MSG37          ;DEPRESS SPACE BAR WHEN DONE
3878 015044 004737 030452    JSR      PC,GETSP
3879
3880 015050 004737 026436    JSR      PC,SUBCLR
3881 015054 104024          ERROR 24          ;CERR AFTER SCLR
3882
3883 015056 032765 010000 000010  BIT      #NED,RKCS2(R5)
3884 015064 001421          BEQ      TST13          ;;GOTO NEXT TEST
3885 015066 104130          ERROR 130          ;NED NOT CLEARED AFTER CORRECT PORT SELECTED
3886
3887 015070 012765 100000 000000 4$:  MOV      #CCLR,RKCS1(R5)
3888 015076 004737 026106    JSR      PC,GSTAT
3889 015102 032765 010000 000010  BIT      #NED,RKCS2(R5)
3890 015110 001407          BEQ      TST13          ;;GOTO NEXT TEST
3891 015112 104401 041422    TYPE      ,MSG45          ;CORRECT PORT NOT SELECTED, TRY AGAIN
3892 015116 104401 040625    TYPE      ,MSG37          ;DEPRESS SPACE BAR WHEN DONE
3893 015122 004737 030452    JSR      PC,GETSP
3894 015126 000760          BR       4$
3895
3896
3897
3898
3899
3900
3901
3902
3903
3904
3905
3906
3907
3908

```

```

:*****
:TEST 13          AC LOW DETECTION PART 1
:

```

```

: A PRELIMINARY AC LOW TEST IS PERFORMED HERE WHILE HEADS ARE UNLOADED.
: BATTERY RETRACT WILL BE TESTED LATER.
: ACLO CANNOT BE TESTED DIRECTLY SINCE IT REQUIRES SECTOR PULSES
: I.E. HEADS LOADED.
: THEREFORE THE INDICATOR CHECKED IS NON-EXISTENT DRIVE (NED)
: ASSERTING IN RKCS2 AS A RESULT OF DCLO.
: AFTER POWER UP, VOLUME VALID IS CHECKED TO BE CLEARED.
:

```

```

3909 015130 000004          TST13: SCOPE
3910 015132 012737 000001 001174  MOV      #1,$TIMES          ;;DO 1 ITERATION
3911 015140 012706 001100    MOV      #STACK,SP          ;RESTORE STK PTR
3912
3913 015144 004737 026436    JSR      PC,SUBCLR
3914 015150 104024          ERROR 24          ;CERR AFTER SCLR
3915
3916 015152 104401 037563    TYPE      ,MSG21          ;AC LOW TEST
3917 015156 104401 042017    TYPE      ,MSG52          ;CONT-E TO BYPASS TEST
3918                                ;OR SPACE TO CONTINUE
3919 015162 004737 030412    JSR      PC,CCSP          ;INPUT CONT-E OR SPACE
3920 015166 000137 015442    JMP      10$          ;RETURN HERE IF CONT-E
3921 015172 104401 041553    TYPE      ,MSG49          ;TURN OFF AC(RET HERE IF SPACE)
3922
3923 015176 005037 005400          CLR      TEMP2
3924 015202 012737 177777 005376 12$:  MOV      #-1,TEMP1          ;SETUP TIMEOUT
3925 015210 004737 026106    JSR      PC,GSTAT
3926 015214 032737 100000 005340 1$:  BIT      #CERR,HCS1
3927 015222 001012          BNE     3$
3928 015224 005337 005376          DEC     TEMP1

```

```

3929 015230 001367          BNE      1$
3930 015232 005237 005400    INC      TEMP2
3931 015236 023727 005400 000002    CMP      TEMP2,#2      ;SEE IF GONE THRU 2 TIMES
3932 015244 001356          BNE      12$           ;BR IF NO
3933 015246 104146          ERROR    146           ;CERR NOT DETECTED BEFORE TIMEOUT
3934 015250 013737 001432 005376 3$:    MOV      T50000,TEMP1
3935 015256 012765 100000 000000 4$:    MOV      #CCLR,RKCS1(R5)
3936 015264 004737 026106          JSR      PC,GSTAT
3937 015270 032765 010000 000010    BIT      #NED,RKCS2(R5)
3938 015276 001004          BNE      5$
3939 015300 005337 005376          DEC      TEMP1
3940 015304 001364          BNE      4$
3941 015306 104150          ERROR    150           ;NED NOT SET BEFORE TIMEOUT
3942
3943 015310 104401 041626          5$:    TYPE     ,MSG50        ;SWITCH AC BACK ON
3944 015314 104401 043724          TYPE     ,MSG70        ;VERIFY HEADS LOAD
3945 015320 104401 043647          TYPE     ,MSG69        ;PRESS SPACE AFTER READY ON
3946 015324 004737 030452          JSR      PC,GETSP      ;GET SPACE
3947
3948 015330 004737 026436          JSR      PC,SUBCLR
3949 015334 104024          ERROR    24           ;CERR AFTER SCLR
3950
3951 015336 032737 000200 005366    BIT      #D.DRDY,HMR2  ;SEE IF LOADED
3952 015344 001001          BNE      6$           ;BR IN YES
3953 015346 104141          ERROR    141          ;DRV NOT RDY AFTER AC UP
3954
3955 015350 005037 005322          6$:    CLR      UNLD         ;FOR VALID HALT
3956 015354 032737 000100 005370    BIT      #D.ACLO,HMR3
3957 015362 001401          BEQ      7$
3958 015364 104152          ERROR    152          ;ACLO NOT RESET AFTER POWER UP
3959
3960 015366 032737 000100 005366    7$:    BIT      #D.VV,HMR2
3961 015374 001401          BEQ      8$
3962 015376 104153          ERROR    153          ;VV NOT RESET AFTER POWER UP
3963
3964 015400          8$:
3965 015400 012765 100000 000000    MOV      #CCLR,RKCS1(R5)
3966 015406 013765 001222 000010    MOV      $UNIT,RKCS2(R5) ;DRIVE #
3967 015414 012737 000003 005340    MOV      #PACK,HCS1
3968 015422 004737 024446          JSR      PC,DOCMD      ;DO PACK CMD & GET CONTR RDY
3969 015426 104116          ERROR    116          ;CONTR NOT RDY
3970
3971 015430 032737 000100 005366    BIT      #D.VV,HMR2
3972 015436 001001          BNE      64$
3973 015440 104027          ERROR    27           ;VOLUME VALID NGT SET AFTER PACK CMD
3974
3975          64$:
3976          10$:
3977          ;*****
3978          ;*TEST 14      CHECK NXF LOGIC
3979          ;*
3980          ;*      THIS TEST VERIFIES NON-EXECUTABLE FUNCTION (NXF) IS DETECTED
3981          ;*      AS A RESULT OF DOING A SEEK WITH VOLUME VALID RESET.
3982          ;*
3983          ;*****
3984 015442 000004          TST14: SCOPE

```

```

3985 015444 012737 000001 001174      MOV      #1,STIMES      ;;DO 1 ITERATION
3986 015452 012706 001100              MOV      #STACK,SP      ;RESTORE STACK PTR
3987
3988 015456 004737 026436              JSR      PC,SUBCLR      ;
3989 015462 104024              ERROR    24              ;CERR AFTER SCLR
3990
3991 015464 032737 010000 005366      BIT      #D.SPIN,HMR2   ;SEE IF SPINDLE ALREADY ON
3992 015472 001020              BNE     64$              ;BR IF YES
3993 015474 104401 040064              TYPE    ,MSG29          ;PLEASE WAIT, HEADS BEING LOADED
3994
3995 015500 012737 000011 005340      MOV      #SRTSPL,HCS1   ;
3996 015506 004737 024446              JSR      PC,DOCMD       ;DO START SPINDLE CMD & GET CONTR RDY
3997 015512 104143              ERROR    143            ;CONTR RDY NOT SET AFTER CMD
3998
3999 015514 013737 001426 005400      MOV      T100,TEMP2     ;
4000 015522 004737 025056              JSR      PC,FATT1       ;FIND ATTN
4001 015526 104144              ERROR    144            ;NO ATTN AFTER CMD
4002
4003 015530 005037 005322              CLR      UNLD            ;
4004 015534              64$: CLR      UNLD            ;FOR VALID HALT
4005 015534 005037 005322              CLR      UNLD            ;
4006
4007 015540 004737 026436              JSR      PC,SUBCLR      ;
4008 015544 104024              ERROR    24              ;CERR AFTER SCLR
4009
4010 015546 104401 037624              TYPE    ,MSG22          ;NXF TEST
4011 015552 104401 042017              TYPE    ,MSG52          ;PRESS CONT-E TO EXIT OR SPACE TO CONTINUE
4012 015556 004737 030412              JSR      PC,CCSP        ;INPUT CONT-E OR SPACE
4013 015562 000137 016070              JMP      7$              ;RETURN HERE FOR CONT-E
4014
4015 015566 104401 041676              TYPE    ,MSG51          ;REMOVE UNIT SELECT PLUG TO CLEAR VV
4016 015572 104401 040625              TYPE    ,MSG37          ;PRESS SPACE WHEN DONE
4017 015576 004737 030452              JSR      PC,GETSP       ;GET SPACE
4018
4019 015602 004737 026106              JSR      PC,GSTAT       ;
4020 015606 032737 000100 005366      BIT      #D.VV,HMR2     ;
4021 015614 001403              BEQ     1$              ;
4022 015616 104177              ERROR    177            ;VV NOT CLEARED AFTER UNIT SEL PLUG
4023 015620 000137 016070              JMP      7$              ;EXIT TEST
4024
4025 015624 032737 000100 005366 1$: BIT      #D.VV,HMR2     ;
4026 015632 001406              BEQ     2$              ;
4027 015634 104155              ERROR    155            ;VV SET AFTER HEADS LOADED
4028 015636 000137 016070              JMP      7$              ;EXIT TEST
4029
4030 015642 004737 026436              JSR      PC,SUBCLR      ;
4031 015646 104024              ERROR    24              ;CERR AFTER SCLR
4032
4033 015650 012765 000001 000020 2$: MOV      #1,RKDC(R5)     ;
4034 015656 012737 000017 005340      MOV      #SEEK,HCS1     ;
4035 015664 004737 024446              JSR      PC,DOCMD       ;DO SEEK CMD & GET CONTR READY
4036 015670 104131              ERROR    131            ;NO RDY AFTER SEEK CMD
4037
4038 015672 013737 001432 005376      MOV      T5000,TEMP1    ;
4039 015700 004737 025152              JSR      PC,FATT2       ;FIND ATTN
4040 015704 104132              ERROR    132            ;NO ATTN AFTER SEEK CMD
  
```

```

4041
4042 015706 032737 000400 005370      BIT      #D.NXF,HMR3
4043 015714 001003                      BNE      3$
4044 015716 104156                      ERROR   156      ;NXF NOT SET AFTER SEEK WITH VV=0
4045 015720 000137 016070      JMP      7$      ;EXIT TEST
4046
4047 015724 032737 000200 005370 3$:   BIT      #D.FLT,HMR3
4048 015732 001003                      BNE      4$
4049 015734 104172                      ERROR   172      ;NXF DID NOT SET FAULT
4050 015736 000137 016070      JMP      7$      ;EXIT TEST
4051
4052 015742 012737 050240 005430 4$:   MOV      #<D.DSC!D.SPIN!D.DRDY!D.DRA>,E.A0 ;EXPECTED A0
4053 015750 012737 000600 005432      MOV      #<D.NXF!D.FLT>,E.B0
4054 015756 012737 001720 005434      MOV      #<D.SPOK!D.CART!D.DOOR!D.BRHM!D.SSP>,E.A1
4055 015764 012737 000001 005436      MOV      #1,E.B1
4056
4057 015772 004737 025264      JSR      PC,CHKMSG      ;CHECK MSGS A0, B0, A1, B1
4058 015776 000003                      .WORD   T.A2!T.B2!0      ;& MSGS SPECIFIED HERE
4059 016000 104160                      ERROR   160      ;MSG A0 ERROR AFTER SEEK WITH VV=0
4060 016002 104161                      ERROR   161      ;MSH B0 ERROR
4061 016004 104162                      ERROR   162      ;MSG A1 ERROR
4062 016006 104163                      ERROR   163      ;MSG B1 ERROR
4063 016010 005737 001400      TST      CYLADD
4064 016014 001401                      BEQ     5$
4065 016016 104157                      ERROR   157      ;HEADS MOVED WITH SEEK & DXF
4066
4067 016020 004737 026436      5$:   JSR      PC,SUBCLR
4068 016024 104024                      ERROR   24      ;CERR AFTER SCLR
4069
4070 016026 012765 100000 000000      MOV      #CCLR,RKCS1(R5)
4071 016034 013765 001222 000010      MOV      $UNIT,RKCS2(R5) ;DRIVE #
4072 016042 012737 000003 005340      MOV      #PACK,HCS1
4073 016050 004737 024446      JSR      PC,DOCMD      ;DO PACK CMD & GET CONTR RDY
4074 016054 104116                      ERROR   116      ;CONTR NOT RDY
4075
4076 016056 032737 000100 005366      BIT      #D.VV,HMR2
4077 016064 001001                      BNE     65$
4078 016066 104027                      ERROR   27      ;VOLUME VALID NOT SET AFTER PACK CMD
4079 016070      65$:
4080 016070      7$:

```

```

4081
4082 :*****
4083 :*TEST 15      READ & SAVE BAD SECTOR INFO & TYPE PACK SERIAL #
4084 :*
4085 :*      THIS TEST VERIFIES THAT CYL 632 (1456 FOR RK07), TRACK 2 CAN BE READ.
4086 :*      THIS AREA CONTAINS BAD SECTOR INFOR WHICH IS WRITTEN BY THE
4087 :*      FACTORY DURING MANF. ALL BAD SECTOR INFOR (BSE) WILL BE STORED
4088 :*      AT THIS TIME TO MASK FUTURE READ HEADER OR DATA ERROR PRINTOUTS.
4089 :*
4090 :*      SECTORS 0,2,4,6,8 CONTAIN IDENTICAL INFO FOR 22 SECTOR HARDWARE DETECTED BAD SEC
4091 :*      SECTORS 10,12,14,16,18,20 CONTAIN IDENTICAL INFO FOR 22 SECTOR SOFTWARE DETECTED
4092 :*
4093 :*      IF BSE INFO CANNOT BE READ, OR IF AFTER READING THE BSE INFO
4094 :*      IT IS DETERMINED THAT AN ALIGNMENT CARTRIDGE IS USED,
4095 :*      A MESSAGE WILL BE TYPED INDICATING THAT ALL
4096 :*      FUTURE FORMAT AND READ-WRITE TESTS WILL BE BYPASSED.

```

CZ  
CZ



```
4097 :* THIS IS DONE SO AS NOT TO DESTROY BSE INFOR OR AN ALIGNMENT PACK BY WRITING
4098 :*
4099 :* THE PACK SERIAL # IS TYPED IN OCTAL & FOR THE FIRST PASS ONLY.
4100 :*
4101 :*****
4102 016070 000004 TST15: SCOPE
4103 016072 012737 000001 001174 MOV #1,$TIMES ;;DO 1 ITERATION
4104 016100 012706 001100 MOV #STACK,SP ;RESTORE STK PTR
4105
4106 016104 004737 026436 JSR PC,SUBCLR
4107 016110 104024 ERROR 24 ;CERR AFTER SCLR
4108 016112 005037 005400 CLR TEMP2 ;SECTOR CTR
4109 016116 005037 005402 CLR TEMP3 ;0=22 SECTOR HARDWARE DETECTED TABLE
4110 ;1=22 SECTOR SOFTWARE DETECTED TABLE
4111 ;2=DONE
4112 016122 012737 002322 005404 MOV #BSE22H,TEMP4 ;STORE 22 SECTOR HARDWARE BSE ADDR.
4113 016130 013765 005404 000004 MOV TEMP4,RKBA(R5)
4114 016136 012737 001000 005406 MOV #1000,TEMP5 ;TRACK 2, SECTOR 0
4115 016144 013765 005406 000006 MOV TEMP5,RKDA(R5)
4116
4117 016152 013765 012654 000020 1$: MOV LC,RKDC(R5) ;LAST CYL
4118 016160 012765 177400 000002 MOV #-256,RKWC(R5) ;LOAD WORD CT
4119 016166 012737 000021 005340 MOV #RDATA,HCS1
4120 016174 004737 024504 JSR PC,DATCMD ;DO READ DATA CMD & GET CONTR RDY
4121 016200 104226 ERROR 226 ;NO RDY AFTER READ DATA CMD
4122 016202 004737 026106 JSR PC,GSTAT ;GET FRESH DATA
4123 016206 032737 100000 005340 BIT #CERR,HCS1
4124 016214 001470 BEQ 8$
4125 016216 104227 ERROR 227 ;CERR AFTER READ DATA CMD
4126
4127 016220 012737 010340 005430 MOV #<0!D.SPIN!D.DRDY!D.VV!D.DRA>,E.A0 ;EXPECTED MSG A0
4128 016226 005037 005432 CLR E.B0 ;EXPECTED MSG B0
4129 016232 012737 001720 005434 MOV #<D.SPOK!D.CART!D.DOOR!D.BRH!D.SSP>,E.A1 ;EXPECTED A1
4130 016240 012737 000001 005436 MOV #1,E.B1 ;MSG ID FOR EXPECTED MSG B1
4131 016246 005037 005440 CLR E.A2 ;EXPECTED MSG A2
4132 016252 012737 000002 005442 MOV #2,E.B2 ;MSG ID FOR EXPECTED MSG B2
4133 016260 012737 000003 005446 MOV #3,E.B3 ;MSG ID FOR EXPECTED MSG B3
4134
4135 016266 004737 025264 JSR PC,CHKMSG ;CHECK MSGS A0, B0, A1, B1
4136 016272 000000 .WORD 0!0!0 ;& MSGS SPECIFIED HERE
4137 016274 104054 ERROR 54 ;MSG A0 ERROR AFTER READ DATA CMD
4138 016276 104026 ERROR 26 ;MSH B0 ERROR
4139 016300 104056 ERROR 56 ;MSG A1 ERROR
4140 016302 104030 ERROR 30 ;MSG B1 ERROR
4141 016304 004737 026436 JSR PC,SUBCLR
4142 016310 104024 ERROR 24 ;CERR AFTER SUBCLR
4143 016312 005237 005400 INC TEMP2
4144 016316 023727 005400 000005 CMP TEMP2,#5 ;READ ALL 5 SECTORS?
4145 016324 001007 BNE 5$
4146 016326 005737 005402 TST TEMP3
4147 016332 001002 BNE 2$
4148 016334 104165 ERROR 165 ;CANT READ SECTORS 0,2,4,6,8
4149 016336 000414 BR 3$
4150 016340 104167 2$: ERROR 167 ;CANT READ SECTORS 10,12,14,16,18,20
4151 016342 000412 BR 3$
4152
```

```

4153 016344 013765 005404 000004 5$: MOV TEMP4,RKBA(R5) ;RESTORE TABLE ADDR
4154 016352 062737 000002 005406 ADD #2,TEMP5 ;SETUP TO READ 2 SECTORS FROM LAST
4155 016360 013765 005406 000006 MOV TEMP5,RKDA(R5)
4156 016366 000671 BR 1$
4157
4158 016370 005237 001474 3$: INC BSERR ;SET BSE FLAG
4159 016374 000501 BR TST16 ;GO TO NEXT TEST
4160 016376 005737 002330 8$: TST BSE22H+6 ;TEST CARTRIDGE TYPE
4161 016402 001405 BEQ 9$ ;BRANCH IF DATA CARTRIDGE
4162 016404 104401 042360 TYPE ,MSG56 ;ALIGNMENT CARTRIDGE USED
4163 016410 005237 001474 INC BSERR ;SET BSE ERROR FLAG
4164 016414 000426 BR 10$
4165
4166 016416 005237 005402 9$: INC TEMP3
4167 016422 023727 005402 000001 CMP TEMP3,#1
4168 016430 001020 BNE 10$
4169 016432 005037 005400 CLR TEMP2
4170 016436 012737 003322 005404 MOV #BSE22S,TEMP4 ;STORE 22 SECTOR SOFTWARE BSE ADDR
4171 016444 013765 005404 000004 MOV TEMP4,RKBA(R5)
4172 016452 012737 001012 005406 MOV #1012,TEMP5 ;TRACK 2, SECTOR 12
4173 016460 013765 005406 000006 MOV TEMP5,RKDA(R5)
4174 016466 000137 016152 JMP 1$ ;REPEAT
4175
4176 016472 005737 001216 10$: TST $PASS
4177
4178
4179 016476 001014 BNE 13$ ;TYPE CART # ONLY ON 1'ST PASS
4180 016500 104401 037450 TYPE ,MSG17 ;CART SERIAL #
4181 016504 012746 002322 MOV #BSE22H,-(SP)
4182 016510 004737 035330 JSR PC,$DB20 ;CONVERT DBL BINARY WORD TO OCTAL
4183 016514 004737 035400 JSR PC,$SUPRS ;TYPE SERIAL #
4184 016520 104401 001205 TYPE ,SCLRF
4185 016524 104401 001205 TYPE ,SCLRF
4186
4187 016530 004737 026436 13$: JSR PC,SUBCLR
4188 016534 104024 ERROR 24 ;CERR AFTER SCLR
4189 ;RETURN TO CYL 0
4190
4191 016536 012737 000017 005340 MOV #SEEK,HCS1
4192 016544 004737 024446 JSR PC,DOCMD ;DO SEEK CMD & GET CONTR READY
4193 016550 104131 ERROR 131 ;NO RDY AFTER SEEK CMD
4194
4195 016552 013737 001432 005376 MOV T50000,TEMP1 ;SETUP TIMEOUT
4196 016560 004737 025152 JSR PC,FATT2 ;FIND ATTN
4197 016564 104132 ERROR 132 ;NO ATTN AFTER SEEK CMD
4198
4199 016566 032737 100000 005340 BIT #CERR,HCS1
4200 016574 001401 BEQ 64$
4201 016576 104210 ERROR 210 ;CERR AFTER SEEK CMD
4202
4203 016600 64$:
4204
4205
4206 *****
4207 *TEST 16 WRITE LOCK TEST
4208 *

```

CZ  
CZ

```

4209          : *      THIS TEST VERIFIES THAT DATA WRITTEN ON A SECTOR CANNOT BE
4210          : *      ALTERED ONCE THE WRITE LOCK SWITCH HAS BEEN ENABLED.
4211          : *      IT ALSO CHECKS THAT WRITE PROTECT TAKES EFFECT ONLY AT
4212          : *      SECTOR BOUNDRIES WHEN DOING CONTINUOUS WRITING ON CYL 0, HEAD 0
4213          : *
4214          : *
4215          : *****
4215 016600 000004 TST16: SCOPE
4216 016602 012737 000001 001174 MOV #1,$TIMES ;:DO 1 ITERATION
4217 016610 012706 001100 MOV #STACK,SP ;:RESTORE STACK PTR
4218 016614 005737 001474 TST BSERR ;:SEE IF ALIGN CART
4219 016620 001402 BEQ 15$ ;:BR IF NO
4220 016622 000137 021450 JMP 12$ ;:ELSE EXIT TEST
4221
4222 016626 004737 026436 15$: JSR PC,SUBCLR
4223 016632 104024 ERROR 24 ;:CERR AFTER SCLR
4224
4225 016634 104401 037705 TYPE ,MSG24 ;:WRITE LOCK TEST
4226 016640 104401 042017 TYPE ,MSG52 ;:PRESS CONT-E TO EXIT OR SPACE TO CONTINUE
4227 016644 004737 030412 JSR PC,CCSP ;:INPUT CONT-E OR SPACE
4228 016650 000137 021450 JMP 12$ ;:RETURN HERE IF CONT-E
4229
4230 016654 004737 026106 JSR PC,GSTAT
4231 016660 032737 004000 005366 BIT #D.WRL,HMR2 ;:SEE IF WRITE LOCK IS ON
4232 016666 001416 BEQ 2$ ;:BR IF NO
4233 016670 104401 042120 1$: TYPE ,MSG53 ;:DISABLE WRITE LOCK
4234 016674 104401 040625 TYPE ,MSG37 ;:TYPE SPACE WHEN DONE
4235 016700 004737 030452 JSR PC,GETSP ;:GET SPACE
4236
4237 016704 004737 026106 JSR PC,GSTAT
4238 016710 032737 004000 005366 BIT #D.WRL,HMR2 ;:SEE IF WRITE LOCK IS OFF
4239 016716 001402 BEQ 2$
4240 016720 104110 ERROR 110 ;:WRITE LOCK NOT DISABLED
4241 016722 000762 BR 1$
4242
4243 016724 005037 001366 2$: CLR TOCYL ;:SETUP
4244 016730 005037 001402 CLR CALADD ;:FOR
4245 016734 005037 001442 CLR HEAD ;:FILL HEADER TABLE
4246 016740 005037 001450 CLR FORMAT ;:ROUTINE
4247
4248 016744 004737 027420 16$: JSR PC,FHDTAB ;:BUILD STD 22 SECTOR HEADER TABLE
4249
4250 016750 012765 001506 000004 MOV #HDTAB,RKBA(R5)
4251 016756 012765 177676 000002 MOV #-66.,RKWC(R5)
4252 016764 000337 001442 SWAB HEAD
4253 016770 013765 001442 000006 MOV HEAD,RKDA(R5) ;:HEAD ADDR
4254 016776 000337 001442 SWAB HEAD
4255
4256
4257 017002 012737 000027 005340 MOV #<WRHEAD>,HCS1
4258 017010 004737 024504 JSR PC,DATCMD ;:DO DATA X FOR CMD & GET CONTR RDY
4259 017014 104200 ERROR 200 ;:NO RDY AFTER WRITE HEADER CMD
4260 017016 004737 026106 JSR PC,GSTAT ;:GET FRESH STATUS
4261 017022 032737 100000 005340 BIT #CERR,HCS1
4262 017030 001401 BEQ 64$
4263 017032 104201 ERROR 201 ;:CERR AFTER WRITE HEADER CMD
4264 017034 64$:

```

```

4265
4266
4267 017034 005237 001442          INC    HEAD
4268 017040 023727 001442 000003    CMP    HEAD,#3      ;SEE IF ALL HEADS DONE
4269 017046 001336                    BNE    16$          ;BR IF NO
4270
4271 017050 004737 026436          JSR    PC,SUBCLR
4272 017054 104024                    ERROR  24          ;CERR AFTER SCLR
4273
4274 017056 005037 001422          CLR    SECTOR
4275 017062 013765 001422 000006 14$:  MOV    SECTOR,RKDA(R5)
4276
4277 017070 012765 001456 000004    MOV    #DATA0,RKBA(R5) ;WRITE ALL 0'S
4278 017076 052765 000020 000010    BIS    #BAI,RKCS2(R5)
4279 017104 012765 177400 000002    MOV    #-256.,RKWC(R5) ;CYL 0, TRK 0, SECTOR 0
4280
4281 017112 012737 000023 005340    MOV    #<WRDATA>,HCS1
4282 017120 004737 024504          JSR    PC,DATCMD    ;DO DATA X FOR CMD & GET CONTR RDY
4283 017124 104011                    ERROR  11          ;NO RDY AFTER WRITE DATA CMD
4284 017126 004737 026106          JSR    PC,GSTAT    ;GET FRESH STATUS
4285 017132 032737 100000 005340    BIT    #CERR,HCS1
4286 017140 001465                    BEQ    68$          ;BR IF NO ERRORS
4287
4288 017142 032737 000200 005354    BIT    #BSE,HER    ;SEE IF BAD SECTOR FLAG
4289 017150 001421                    BEQ    66$          ;BR IF NO
4290 017152 004737 030056          JSR    PC,TRUERR   ;ELSE SEE IF SECTOR LISTED IN BSE TABLE
4291 017156 000455                    BR     67$          ;RETURN HERE IF NO
4292
4293 017160 005237 001422          INC    SECTOR      ;RETURN HERE IF YES
4294 017164 023727 001422 000012    CMP    SECTOR,#10. ;ARE 10 CONSEC. SECTORS BAD
4295 017172 001003                    BNE    65$          ;BR IF NO
4296 017174 104040                    ERROR  40          ;ABORTING TEST DETECTED 10 BAD SECTORS
4297 017176 000137 021450          JMP    12$         ;BYPASS TEST
4298
4299 017202 012765 100000 000000 65$:  MOV    #CCLR,RKCS1(R5) ;TRY ANOTHER SECTOR
4300 017210 000137 017062          JMP    14$
4301
4302 017214 104012                    ERROR  12          ;CERR WITH WRITE DATA CMD
4303
4304 017216 012737 010340 005430    MOV    #<0!D.SPIN!D.DRDY!D.VV!D.DRA>,E.A0 ;EXPECTED MSG A0
4305 017224 005037 005432          CLR    E.B0        ;EXPECTED MSG B0
4306 017230 012737 001720 005434    MOV    #<D.SPOK!D.CART!D.DOOR!D.BRHM!D.SSP>,E.A1 ;EXPECTED A1
4307 017236 012737 000001 005436    MOV    #1,E.B1     ;MSG ID FOR EXPECTED MSG B1
4308 017244 005037 005440          CLR    E.A2        ;EXPECTED MSG A2
4309 017250 012737 000002 005442    MOV    #2,E.B2     ;MSG ID FOR EXPECTED MSG B2
4310 017256 012737 000003 005446    MOV    #3,E.B3     ;MSG ID FOR EXPECTED MSG B3
4311
4312 017264 004737 025264          JSR    PC,CHKMSG   ;CHECK MSGS A0, B0, A1, B1
4313 017270 000003                    .WORD T.A2!T.B2!0 ;& MSGS SPECIFIED HERE
4314 017272 104052                    ERROR  52          ;MSG A0 ERROR AFTER WRITE DATA CMD
4315 017274 104023                    ERROR  23          ;MSG B0 ERROR
4316 017276 104053                    ERROR  53          ;MSG A1 ERROR
4317 017300 104025                    ERROR  25          ;MSG B1 ERROR
4318 017302 104401 044035          TYPE  #MSG72
4319 017306 000137 023734          JMP    $EOP
4320 017312 104047                    ERROR  47          ;BAD SECTOR NOT LISTED IN TABLE
  
```

```

4321 017314 68$:
4322
4323 017314 004737 026436 JSR PC,SUBCLR
4324 017320 104024 ERROR 24 ;CERR AFTER SCLR
4325
4326 017322 012765 001456 000004 MOV #DATA0,RKBA(R5)
4327 017330 052765 000020 000010 BIS #BAI,RKCS2(R5)
4328 017336 012765 177400 000002 MOV #-256.,RKWC(R5)
4329 017344 013737 001456 001436 MOV DATA0,WD2 ;EXPECTED WORD FOR TRUERR TYPEOUT
4330 017352 013765 001422 000006 MOV SECTOR,RKDA(R5)
4331
4332 017360 012737 000031 005340 MOV #<WRTCHK>,HCS1
4333 017366 004737 024504 JSR PC,DATCMD ;DO DATA X FOR CMD & GET CONTR RDY
4334 017372 104015 ERROR 15 ;NO RDY AFTER WRITE CHECK CMD
4335 017374 004737 026106 JSR PC,GSTAT ;GET FRESH STATUS
4336 017400 032737 100000 005340 BIT #CERR,HCS1
4337 017406 001450 BEQ 70$
4338 017410 032737 040000 005342 BIT #WCE,HCS2 ;SEE IF WRITE CHECK ERROR
4339 017416 001405 BEQ 69$
4340 017420 016537 000024 001434 MOV RKDB(R5),WD1 ;ACTUAL WORD FOR PRINTOUT
4341 017426 104016 ERROR 16 ;WCE AFTER WRITE CMD
4342 017430 000437 BR 70$
4343
4344 017432 104022 69$: ERROR 22 ;CERR AFTER WRITE CHECK CMD
4345
4346 017434 012737 010340 005430 MOV #<O!D.SPIN!D.DRDY!D.VV!D.DRA>,E.A0 ;EXPECTED MSG A0
4347 017442 005037 005432 CLR E.B0 ;EXPECTED MSG B0
4348 017446 012737 001720 005434 MOV #<D.SPOK!D.CART!D.DOOR!D.BRHM!D.SSP>,E.A1 ;EXPECTED A1
4349 017454 012737 000001 005436 MOV #1,E.B1 ;MSG ID FOR EXPECTED MSG B1
4350 017462 005037 005440 CLR E.A2 ;EXPECTED MSG A2
4351 017466 012737 000002 005442 MOV #2,E.B2 ;MSG ID FOR EXPECTED MSG B2
4352 017474 012737 000003 005446 MOV #3,E.B3 ;MSG ID FOR EXPECTED MSG B3
4353
4354 017502 004737 025264 JSR PC,CHKMSG ;CHECK MSGS A0, B0, A1, B1
4355 017506 000003 .WORD T.A2!T.B2!0 ;& MSGS SPECIFIED HERE
4356 017510 104057 ERROR 57 ;MSG A0 ERROR AFTER WRITE CHECK CMD
4357 017512 104031 ERROR 31 ;MSG B0 ERROR
4358 017514 104060 ERROR 60 ;MSG A1 ERROR
4359 017516 104032 ERROR 32 ;MSG B1 ERROR
4360 017520 104401 044035 TYPE ,MSG72 ;ABORTING BALANCE OF TESTS
4361 017524 000137 023374 JMP ENDRV
4362
4363 017530 70$:
4364
4365 017530 004737 026436 3$: JSR PC,SUBCLR
4366 017534 104024 ERROR 24 ;CERR AFTER SCLR
4367
4368 017536 104401 042211 TYPE ,MSG54 ;ENABLE WRITE LOCK
4369 017542 104401 040625 TYPE ,MSG37 ;PRESS SPACE WHEN DONE
4370 017546 004737 030452 JSR PC,GETSP ;GET SPACE
4371
4372 017552 012765 100000 000000 MOV #CLR,RKCS1(R5)
4373 017560 004737 026106 JSR PC,GSTAT
4374 017564 032737 004000 005366 BIT #D.WRL,HMR2 ;SEE IF WRITE LOCK IS ON
4375 017572 001002 BNE 4$
4376 017574 104115 ERROR 11$ ;WRITE LOCK NOT ENABLED

```

```

4377 017576 000754 BR 3$
4378
4379 017600 012765 001462 000004 4$: MOV #DATA1,RKBA(R5) ;ATTEMPT TO WRITE ALL 1'S WITH WRITE LOCK SET
4380 017606 052765 000020 000010 BIS #BAI,RKCS2(R5)
4381 017614 012765 177400 000002 MOV #-256.,RKWC(R5) ; CYLINDER 0, HEAD 0
4382 017622 013765 001422 000006 MOV SECTOR,RKDA(R5)
4383 017630 012737 000023 005340 MOV #WRDATA,HCS1
4384 017636 004737 024504 JSR PC,DATCMD ;DO WRITE DATA CMD & GET CONTR RDY
4385 017642 104116 ERROR 116 ;CONTR NOT RDY
4386
4387 017644 004737 026106 JSR PC,GSTAT ;GET FRESH STATUS
4388 017650 032737 100000 005340 BIT #CERR,HCS1
4389 017656 001001 BNE 17$
4390 017660 104207 ERROR 207 ;CERR NOT SET AFTER WRITE WITH WRL SET
4391 017662 032737 004000 005370 17$: BIT #D.WLE,HMR3 ;CHECK WRITE LOCK ERROR SET
4392 017670 001001 BNE 5$ ;BR IF SET
4393 017672 104121 ERROR 121 ;WRITE LOCK ERROR NOT SET
4394 ;AFTER WRITE DATA WITH WRITE LOCK SET
4395 017674 012737 054340 005430 5$: MOV #<D.DSC!D.SPIN!D.WRL!D.DRDY!D.VV!D.DRA>,E.A0 ;EXP A0
4396 017702 012737 004200 005432 MOV #<D.WLE!D.FLT>,E.B0
4397 017710 012737 001720 005434 MOV #<D.SPOK!D.CART!D.DOOR!D.BRHM!D.SSP>,E.A1
4398 017716 012737 000001 005436 MOV #1,E.B1
4399
4400 017724 004737 025264 JSR PC,CHKMSG ;CHECK MSGS A0, B0, A1, B1
4401 017730 000000 .WORD 0!0!0 ;# MSGS SPECIFIED HERE
4402 017732 104123 ERROR 123 ;MSG A0 ERROR AFTER WRITE WITH WRITE LOCK SET
4403 017734 104125 ERROR 125 ;MSH B0 ERROR
4404 017736 104126 ERROR 126 ;MSG A1 ERROR
4405 017740 104127 ERROR 127 ;MSG B1 ERROR
4406
4407 017742 004737 026436 JSR PC,SUBCLR
4408 017746 104024 ERROR 24 ;CERR AFTER SCLR
4409
4410 017750 012765 001456 000004 MOV #DATA0,RKBA(R5) ;CHECK THAT NONE OF ORIG DATA
4411 017756 052765 000020 000010 BIS #BAI,RKCS2(R5) ;HAS CHANGED
4412 017764 012765 177400 000002 MOV #-256.,RKWC(R5) ;AS RESULT OF WRITE WITH WRITE LOCK SET
4413 017772 013737 001456 001436 MOV DATA0,WD2 ;EXPECTED WORD FOR TRUERR TYPEOUT
4414 020000 013765 001422 000006 MOV SECTOR,RKDA(R5)
4415
4416 020006 012737 000031 005340 MOV #<WRTCHK>,HCS1
4417 020014 004737 024504 JSR PC,DATCMD ;DO DATA X FOR CMD & GET CONTR RDY
4418 020020 104015 ERROR 15 ;NO RDY AFTER WRITE CHECK CMD
4419 020022 004737 026106 JSR PC,GSTAT ;GET FRESH STATUS
4420 020026 032737 100000 005340 BIT #CERR,HCS1
4421 020034 001450 BEQ 72$
4422 020036 032737 040000 005342 BIT #WCE,HCS2 ;SEE IF WRITE CHECK ERROR
4423 020044 001405 BEQ 71$
4424 020046 016537 000024 001434 MOV RKDB(R5),WD1 ;ACTUAL WORD FOR PRINTOUT
4425 020054 104016 ERROR 16 ;WCE AFTER WRITE CMD
4426 020056 000437 BR 72$
4427
4428 020060 104022 71$: ERROR 22 ;CERR AFTER WRITE CHECK CMD
4429
4430 020062 012737 010340 005430 MOV #<0!D.SPIN!D.DRDY!D.VV!D.DRA>,E.A0 ;EXPECTED MSG A0
4431 020070 005037 005432 CLR E.B0 ;EXPECTED MSG B0
4432 020074 012737 001720 005434 MOV #<D.SPOK!D.CART!D.DOOR!D.BRHM!D.SSP>,E.A1 ;EXPECTED A1

```

```

4433 020102 012737 000001 005436      MOV    #1,E.B1      ;MSG ID FOR EXPECTED MSG B1
4434 020110 005037 005440              CLR    E.A2        ;EXPECTED MSG A2
4435 020114 012737 000002 005442      MOV    #2,E.B2      ;MSG ID FOR EXPECTED MSG B2
4436 020122 012737 000003 005446      MOV    #3,E.B3      ;MSG ID FOR EXPECTED MSG B3
4437
4438 020130 004737 025264              JSR    PC,CHKMSG    ;CHECK MSGS A0, B0, A1, B1
4439 020134 000003                      .WORD  T.A2!T.B2!0 ;& MSGS SPECIFIED HERE
4440 020136 104057                      ERROR  57          ;MSG A0 ERROR AFTER WRITE CHECK CMD
4441 020140 104031                      ERROR  31          ;MSH B0 ERROR
4442 020142 104060                      ERROR  60          ;MSG A1 ERROR
4443 020144 104032                      ERROR  32          ;MSG B1 ERROR
4444 020146 104401 044035              TYPE  ,MSG72      ;ABORTING BALANCE OF TESTS
4445 020152 000137 023374              JMP    ENDRV
4446
4447 020156                                72$:
4448
4449 020156 004737 026436              6$:  JSR    PC,SUBCLR
4450 020162 104024                      ERROR  24          ;CERR AFTER SCLR
4451
4452 020164 104401 042120              TYPE  ,MSG53      ;DISABLE WRITE LOCK
4453 020170 104401 040625              TYPE  ,MSG37      ;SPACE BAR WHEN DONE
4454 020174 004737 030452              JSR    PC,GETSP    ;GET SPACE
4455 020200 004737 026106              JSR    PC,GSTAT
4456 020204 032737 004000 005366      BIT    #D.WRL,HMR2 ;SEE IF WRITE LOCK OFF
4457 020212 001361                      BNE   6$
4458
4459                                ;TEST FOR WRITE LOCK AT SECTOR BOUNDRIES
4460 020214 104401 042211              TYPE  ,MSG54      ;FOR CONTINUOUS WRITING ON CYL 0, TRACK 0,1,2
4461 020220 013737 001430 001160      MOV    I5000,$TMP0 ;SET WRITE LOCK
4462 020226 005037 001162                      CLR    $TMP1
4463 020232 005037 001366                      CLR    TOCYL      ;BIT0=0:WRITE 0'S; BIT0=1;WRITE 1'S
4464
4465 020236 004737 026436              7$:  JSR    PC,SUBCLR
4466 020242 104024                      ERROR  24          ;CERR AFTER SCLR
4467
4468 020244 032737 000001 001162      BIT    #BIT0,$TMP1
4469 020252 001004                      BNE   8$          ;BR IF WRITING 1'S
4470 020254 012765 001456 000004      MOV    #DATA0,RKBA(R5) ;SETUP ALL 0'S
4471 020262 000403                      BR    9$
4472
4473 020264 012765 001462 000004      8$:  MOV    #DATA1,RKBA(R5) ;SETUP ALL 1'S
4474 020272 052765 000020 000010      9$:  BIS    #BA1,RKCS2(R5)
4475 020300 012765 140400 000002      MOV    #-63.*256.,RKWC(R5) ;WRITE TRACK 0,1,2 63 SECTORS
4476 020306 005065 000006                      CLR    RKDA(R5)   ;BEGIN AT TRACK AND SEC 0
4477 020312 013765 001366 000020      MOV    TOCYL,RKDC(R5)
4478 020320 012737 000023 005340      MOV    #WRDATA,HCS1
4479 020326 004737 024504              JSR    PC,DATCMD  ;DO WRITE DATA CMD & GET CONTR RDY
4480 020332 104011                      ERROR  11          ;CONTR NOT RDY
4481
4482 020334 032737 100000 005340      BIT    #CERR,HCS1
4483 020342 001017                      BNE   13$
4484 020344 005337 001160                      DEC    $TMP0
4485 020350 001011                      BNE   10$
4486 020352 104204                      ERROR  204        ;CERR NOT SET BY TIMEOUT
4487 020354 104401 042120              TYPE  ,MSG53      ;DISABLE WRITE LOCK
4488 020360 104401 040625              TYPE  ,MSG37      ;PRESS SPACE WHEN DONE
  
```

4489	020364	004737	030452			JSR	PC,GETSP	:INPUT SPACE
4490	020370	000137	021450			JMP	12\$	:EXIT TEST
4491								
4492	020374	005237	001162		10\$:	INC	\$TMP1	
4493	020400	000716				BR	7\$	:GO WRITE OPPOSITE DATA
4494	020402	032737	000200	005354	13\$:	BIT	#BSE,HER	:SEE IF BAD SECTOR
4495	020410	001411				BEQ	21\$	:BR IF NO
4496	020412	005237	001366			INC	TOCYL	:ELSE TRY ANOTHER CYL
4497	020416	023727	001366	000012		CMP	TOCYL,#10.	:SEE IF 10 CONSEC CYL BAD
4498	020424	001304				BNE	7\$	:BR IF NO & DO AGAIN
4499	020426	104062				ERROR	62	:10 BAD CONSEC CYLINDERS
4500	020430	000137	021450			JMP	12\$	:EXIT TEST
4501								
4502	020434	032737	004000	005370	21\$:	BIT	#D.WLE,HMR3	
4503	020442	001001				BNE	11\$	
4504	020444	104205				ERROR	205	:NO WRL BY TIMEOUT
4505								
4506	020446	104401	042120		11\$:	TYPE	,MSG53	:DISABLE WRITE LOCK
4507	020452	104401	040625			TYPE	,MSG37	:TYPE SPACE WHEN DONE
4508	020456	004737	030452			JSR	PC,GETSP	
4509								
4510	020462	013737	005350	001164		MOV	HDA,\$TMP2	:STORE RKDA OF ERROR
4511	020470	013737	005350	001166		MOV	HDA,\$TMP3	:STORE FOR ERROR TYPEOUT
4512								
4513	020476	023727	001164	000001		CMP	\$TMP2,#1	:SEE IF WRL ON TRK 0, SEC 1
4514	020504	001003				BNE	22\$	:BR IF NO
4515	020506	104401	044233			TYPE	,MSG100	:ELSE WRL OCCURRED ON TRK 2, SEC 21
4516	020512	000621				BR	6\$	:& NO NEW DATA XFER TOOK PLACE
4517								
4518	020514	005737	001164		22\$:	TST	\$TMP2	:SEE IF WRL ON TRK 0, SEC 0
4519	020520	001004				BNE	23\$	:BR IN NO
4520	020522	104401	044233			TYPE	,MSG100	:ELSE WRL ON TRK 2, SEC 20
4521	020526	000137	020156			JMP	6\$	:& NO NEW DATA XFER TOOK PLACE
4522								
4523	020532	023727	001164	001025	23\$:	CMP	\$TMP2,#1025	:SEE IF WRL ON TRK 2, SEC 21
4524	020540	001004				BNE	24\$	:BE IF NO
4525	020542	104401	044233			TYPE	,MSG100	:ELSE WRL ON TRK 2, SEC 19
4526	020546	000137	020156			JMP	6\$	:& NO NEW DATA XFER TOOK PLACE
4527								
4528	020552	023727	001164	001024	24\$:	CMP	\$TMP2,#1024	:SEE IF WRL ON TRK 2, SEC 20
4529	020560	001004				BNE	25\$	:BR IF NO
4530	020562	104401	044233			TYPE	,MSG100	:ELSE WRL ON TRK 2, SEC 18
4531	020566	000137	020156			JMP	6\$	:& NO OLD DATA TO CHECK AGAINST
4532								
4533	020572	023727	001164	000400	25\$:	CMP	\$TMP2,#400	:SEE IF WRL AT TRK 1, SEC 0
4534	020600	001004				BNE	18\$	:BR IF NO
4535	020602	012765	000025	000006		MOV	#21.,RKDA(R5)	:ELSE SECTOR AT WRL IS TRK 0, SEC 21
4536	020610	000415				BR	20\$	
4537								
4538	020612	023727	001164	001000	18\$:	CMP	\$TMP2,#1000	:SEE IF WRL AT TRK 2, SEC 0
4539	020620	001004				BNE	19\$	:BR IF NO
4540	020622	012765	000425	000006		MOV	#425,RKDA(R5)	:ELSE SECTOR AT WRL IS TRK 1, SEC 21
4541	020630	000405				BR	20\$	
4542								
4543	020632	005337	001164		19\$:	DEC	\$TMP2	:GET SECTOR AT WRL
4544	020636	013765	001164	000006		MOV	\$TMP2,RKDA(R5)	



```

4545
4546 020644 016537 000006 001166 20$: MOV RKDA(R5),$TMP3 ;FOR ERROR PRINTOUT
4547
4548
4549
4550 020652 004737 026436 JSR PC,SUBCLR
4551 020656 104024 ERROR 24 ;CERR AFTER SCLR
4552
4553 020660 005737 001164 TST $TMP2 ;SEE IF TRK/SECTOR 0
4554 020664 001414 BEQ 80$ ;REPEAT,NO NEW DATA XFER TOOK PLACE
4555 020666 023727 001164 001023 CMP $TMP2,#1023 ;SEE IF TRK 2,SECTOR 19
4556 020674 001410 BEQ 80$ ;REPEAT,NO OLD DATA TO CHECK AGAINST
4557 020676 032737 000001 001162 BIT #BIT0,$TMP1
4558 020704 001006 BNE 73$ ;BR IF WRITING 1'S WHEN WLE OCCURRED
4559 020706 012765 001462 000004 MOV #DATA1,RKBA(R5) ;WRITING 0'S:WLE SECTOR SHOULD HAVE ALL 1'S
4560 020714 000405 BR 74$
4561
4562 020716 000137 020156 80$: JMP 6$
4563
4564 020722 012765 001456 000004 73$: MOV #DATA0,RKBA(R5) ;WRITING 1'S:WLE SECTOR SHOULD HAVE ALL 0'S
4565 020730 052765 000020 000010 74$: BIS #BAI,RKCS2(R5)
4566 020736 012765 177400 000002 MOV #-256.,RKWC(R5)
4567 020744 013765 001166 000006 MOV $TMP3,RKDA(R5) ;REFRESH RKDA
4568 020752 017537 000004 001436 MOV @RKBA(R5),WD2 ;EXPECTED WORD FOR TRUERR TYPEOUT
4569 020760 013765 001366 000020 MOV TOCYL,RKDC(R5)
4570
4571 020766 012737 000031 005340 MOV #<WRTCHK>,HCS1
4572 020774 004737 024504 JSR PC,DATCMD ;DO DATA X FOR CMD & GET CONTR RDY
4573 021000 104015 ERROR 15 ;NO RDY AFTER WRITE CHECK CMD
4574 021002 004737 026106 JSR PC,GSTAT ;GET FRESH STATUS
4575 021006 032737 100000 005340 BIT #CERR,HCS1
4576 021014 001450 BEQ 82$
4577 021016 032737 040000 005342 BIT #WCE,HCS2 ;SEE IF WRITE CHECK ERROR
4578 021024 001405 BEQ 81$
4579 021026 016537 000024 001434 MOV RKDB(R5),WD1 ;ACTUAL WORD FOR PRINTOUT
4580 021034 104134 ERROR 134 ;WCE AFTER WRITE CMD
4581 021036 000437 BR 82$
4582
4583 021040 104022 81$: ERROR 22 ;CERR AFTER WRITE CHECK CMD
4584
4585 021042 012737 010340 005430 MOV #<0!D.SPIN!D.DRDY!D.VV!D.DRA>,E.A0 ;EXPECTED MSG A0
4586 021050 005037 005432 CLR E.B0 ;EXPECTED MSG B0
4587 021054 012737 001720 005434 MOV #<D.SPOK!D.CART!D.DOOR!D.BRM!D.SSP>,E.A1 ;EXPECTED A1
4588 021062 012737 000001 005436 MOV #1,E.B1 ;MSG ID FOR EXPECTED MSG B1
4589 021070 005037 005440 CLR E.A2 ;EXPECTED MSG A2
4590 021074 012737 000002 005442 MOV #2,E.B2 ;MSG ID FOR EXPECTED MSG B2
4591 021102 012737 000003 005446 MOV #3,E.B3 ;MSG ID FOR EXPECTED MSG B3
4592
4593 021110 004737 025264 JSR PC,CHKMSG ;CHECK MSGS A0, B0, A1, B1
4594 021114 000003 .WORD T.A2!T.B2!0 ;& MSGS SPECIFIED HERE
4595 021116 104057 ERROR 57 ;MSG A0 ERROR AFTER WRITE CHECK CMD
4596 021120 104031 ERROR 31 ;MSG B0 ERROR
4597 021122 104060 ERROR 60 ;MSG A1 ERROR
4598 021124 104032 ERROR 32 ;MSG B1 ERROR
4599 021126 104401 044035 TYPE ,MSG72 ;ABORTING BALANCE OF TESTS
4600 021132 000137 023374 JMP ENDRV
    
```

4601									
4602	021136				82\$:				
4603	021136	000240				NOP			
4604	021140	000240				NOP			
4605									
4606	021142	023727	001164	000400		CMP	\$TMP2,#400	:SEE IF WRL AT TRK 1, SECTOR 0	
4607	021150	001004				BNE	75\$	:BR IF NO	
4608	021152	012765	000025	000006		MOV	#21.,RKDA(R5)	:ELSE SECTOR BEFORE WRL IS TRK 0, SEC 21	
4609	021160	000415				BR	77\$		
4610	021162	023727	001164	001000	75\$:	CMP	\$TMP2,#1000	:SEE IF WRL AT TRK 2,SECTOR 0	
4611	021170	001004				BNE	76\$	:BR IF NO	
4612	021172	012765	000425	000006		MOV	#425,RKDA(R5)	:ELSE SECTOR BEFORE WRL IS TRK 1, SEC 21	
4613	021200	000405				BR	77\$		
4614									
4615	021202	005337	001164		76\$:	DEC	\$TMP2	:GET SECTOR BEFORE WRL	
4616	021206	013765	001164	000006		MOV	\$TMP2,RKDA(R5)		
4617	021214	016537	000006	001166	77\$:	MOV	RKDA(R5),\$TMP3	:FOR ERROR PRINTOUT	
4618	021222	032737	000001	001162		BIT	#BIT0,\$TMP1		
4619	021230	001004				BNF	78\$	:BR IF WRITING 1'S WHEN WLE OCCURRED	
4620	021232	012765	001456	000004		MOV	#DATA0,RKBA(R5)	:WRITING 0'S:WLE SECTOR -1 SHOULD HAVE ALL 0'S	
4621	021240	000403				BR	79\$		
4622									
4623	021242	012765	001462	000004	78\$:	MOV	#DATA1,RKBA(R5)	:WRITING 1'S:WLE SECTOR -1 SHOULD HAVE ALL 1'S	
4624	021250	052765	000020	000010	79\$:	BIS	#BA1,RKCS2(R5)		
4625	021256	012765	177400	000002		MOV	#-256.,RKWC(R5)		
4626	021264	017537	000004	001436		MOV	@RKBA(R5),WD2	:EXPECTED WORD FOR TRUERR TYPEOUT	
4627	021272	013765	001366	000020		MOV	TOCYL,RKDC(R5)		
4628									
4629	021300	012737	000031	005340		MOV	#<WRTCHK>,HCS1		
4630	021306	004737	024504			JSR	PC,DATCMD	:DO DATA X FOR CMD & GET CONTR RDY	
4631	021312	104015				ERROR	15	:NO RDY AFTER WRITE CHECK CMD	
4632	021314	004737	026106			JSR	PC,GSTAT	:GET FRESH STATUS	
4633	021320	032737	100000	005340		BIT	#CERR,HCS1		
4634	021326	001450				BEQ	84\$		
4635	021330	032737	040000	005342		BIT	#WCE,HCS2	:SEE IF WRITE CHECK ERROR	
4636	021336	001405				BEQ	83\$		
4637	021340	016537	000024	001434		MOV	RKDB(R5),WD1	:ACTUAL WORD FOR PRINTOUT	
4638	021346	104135				ERROR	135	:WCE AFTER WRITE CMD	
4639	021350	000437				BR	84\$		
4640									
4641	021352	104022			83\$:	ERROR	22	:CERR AFTER WRITE CHECK CMD	
4642									
4643	021354	012737	010340	005430		MOV	#<0!D.SPIN!D.DRDY!D.VV!D.DRA>,E.A0	:EXPECTED MSG A0	
4644	021362	005037	005432			CLR	E.B0	:EXPECTED MSG B0	
4645	021366	012737	001720	005434		MOV	#<D.SPOK!D.CART!D.DOOR!D.BRHM!D.SSP>,E.A1	:EXPECTED A1	
4646	021374	012737	000001	005436		MOV	#1,E.B1	:MSG ID FOR EXPECTED MSG B1	
4647	021402	005037	005440			CLR	E.A2	:EXPECTED MSG A2	
4648	021406	012737	000002	005442		MOV	#2,E.B2	:MSG ID FOR EXPECTED MSG B2	
4649	021414	012737	000003	005446		MOV	#3,E.B3	:MSG ID FOR EXPECTED MSG B3	
4650									
4651	021422	004737	025264			JSR	PC,CHKMSG	:CHECK MSGS A0, B0, A1, B1	
4652	021426	000003				.WORD	T.A2!T.B2!0	:& MSGS SPECIFIED HERE	
4653	021430	104057				ERROR	57	:MSG A0 ERROR AFTER WRITE CHECK CMD	
4654	021432	104031				ERROR	31	:MSG B0 ERROR	
4655	021434	104060				ERROR	60	:MSG A1 ERROR	
4656	021436	104032				ERROR	32	:MSG B1 ERROR	

```

4657 021440 104401 044035          TYPE      ,MSG72          :ABORTING BALANCE OF TESTS
4658 021444 000137 023374          JMP        ENDRV
4659
4660 021450          84$:
4661
4662 021450          12$:
4663
4664          :*****
4665          :*TEST 17          AC LOW DETECTION PART 2
4666          :*
4667          :*          THIS TEST VERIFIES THAT WHEN AC POWER IS LOST, THAT WRITING CEASES
4668          :*          AT SECTOR BOUNDRIES & THAT THE BATTERY RETRACT IS FUNCTIONAL.
4669          :*          THERE IS APPROX 4 MS BETWEEN AC LOW ASSERTING AND NED ASSERTING
4670          :*          WHEN THE INTERFACE SHUTS DOWN.
4671          :*
4672          :*****
4673 021450 000004          TST17: SCOPE
4674 021452 012737 000001 001174  MOV      #1, $TIMES          ;;DO 1 ITERATION
4675 021460 012706 001100          MOV      #STACK, SP          ;RESTORE STK PTR
4676
4677 021464 004737 026436          JSR      PC, SUBCLR
4678 021470 104024          ERROR   24          ;CERR AFTER SCLR
4679
4680 021472 104401 037731          TYPE      ,MSG26          ;AC LOW-PART 2
4681 021476 104401 042017          TYPE      ,MSG52          ;CONT-E TO BYPASS TEST
4682          ;OR SPACE TO CONINUE
4683 021502 004737 030412          JSR      PC, CCSP
4684 021506 000137 023374          JMP      12$          ;INPUT CONT-E OR SPACE
4685          ;RETURN HERE FOR CONT-E
4686 021512 005737 001474          TST      BSERR          ;RETURN HERE FOR SPACE
4687 021516 001402          BEQ     1$          ;TEST FOR ALIGN CARTRIDGE
4688 021520 000137 022142          JMP      2$          ;BR IF NOT ALIGN CART.
4689
4690 021524 004737 026106          JSR      PC, GSTAT
4691 021530 032737 004000 005366  BIT      #D.WRL, HMR2          ;SEE IF WRITE LOCK
4692 021536 001417          BEQ     11$          ;BR IF NO
4693
4694 021540 104401 042120          TYPE      ,MSG53          ;DISABLE WRITE LOCK
4695 021544 104401 040625          TYPE      ,MSG37          ;PRESS SPACE WHEN DONE
4696 021550 004737 030452          JSR      PC, GETSP          ;GET SPACE
4697
4698 021554 004737 026106          JSR      PC, GSiAT
4699 021560 032737 004000 005366  BIT      #D.WRL, HMR2          ;SEE IF STILL WRITE LOCK
4700 021566 001403          BEQ     11$          ;BR IF NO
4701 021570 104110          ERROR   110          ;WRITE LOCK NOT DISABLED
4702 021572 000137 023374          JMP      12$          ;EXIT TEST
4703
4704 021576 005037 001366          11$:  CLR     TOCYL          ;SETUP
4705 021602 005037 001402          CLR     CALADD          ;FOR
4706 021606 005037 001442          CLR     HEAD           ;FILL HEADER TABLE
4707 021612 005037 001450          CLR     FORMAT          ;ROUTINE
4708
4709 021616 004737 027420          13$:  JSR      PC, FHDTAB          ;BUILD STD 22 SECTOR HEADER TABLE
4710
4711 021622 012765 001506 000004  MOV      #HDTAB, RKBA(R5)
4712 021630 012765 177676 000002  MOV      #-66., RKWC(R5)
  
```

4713	021636	000337	001442		SWAB	HEAD	
4714	021642	013765	001442	000006	MOV	HEAD,RKDA(R5)	;HEAD ADDR
4715	021650	000337	001442		SWAB	HEAD	
4716							
4717	021654	012737	000027	005340	MOV	#<WRHEAD>,HCS1	
4718	021662	004737	024504		JSR	PC,DATCMD	;DO DATA X FOR CMD & GET CONTR RDY
4719	021666	104200			ERROR	200	;NO RDY AFTER WRITE HEADER CMD
4720	021670	004737	026106		JSR	PC,GSTAT	;GET FRESH STATUS
4721	021674	032737	100000	005340	BIT	#CERR,HCS1	
4722	021702	001401			BEQ	64\$	
4723	021704	104201			ERROR	201	;CERR AFTER WRITE HEADER CMD
4724	021706						
4725							64\$:
4726	021706	005237	001442		INC	HEAD	
4727	021712	023727	001442	000003	CMP	HEAD,#3	;SEE IF ALL HEADS DONE
4728	021720	001336			BNE	13\$	;BR IF NO
4729	021722	005037	001366		CLR	TOCYL	
4730							
4731	021726	004737	026436		JSR	PC,SUBCLR	
4732	021732	104024			ERROR	24	;CERR AFTER SCLR
4733							
4734	021734	012765	001462	000004	MOV	#DATA1,RKBA(R5)	;RETURN HERE FOR SPACE
4735	021742	052765	000020	000010	BIS	#BAI,RKCS2(R5)	;WRITE INITIAL BACKGROUND OF 1'S
4736	021750	012765	137000	000002	MOV	#-66.*256.,RKWC(R5)	;TRACK 0,1,2 ALL SECTORS
4737	021756	013765	001366	000020	MOV	TOCYL,RKDC(R5)	
4738	021764	012737	000023	005340	MOV	#WRDATA,HCS1	
4739	021772	004737	024504		JSR	PC,DATCMD	;DO WRITE DATA CMD & GET CONTR RDY
4740	021776	104011			ERROR	11	;NO RDY AFTER WRITE DATA CMD
4741	022000	004737	026106		JSR	PC,GSTAT	;GET FRESH STATUS
4742	022004	032737	100000	005340	BIT	#CERR,HCS1	
4743	022012	001453			BEQ	2\$	
4744	022014	032737	000200	005354	BIT	#BSE,HER	;SEE IF BAD SECTOR
4745	022022	001034			BNE	17\$	
4746							
4747	022024	012737	010340	005430	MOV	#<0!D.SPIN!D.DRDY!D.VV!D.DRA>,E.A0	;EXPECTED MSG A0
4748	022032	005037	005432		CLR	E.B0	;EXPECTED MSG B0
4749	022036	012737	001720	005434	MOV	#<D.SPOK!D.CART!D.DOOR!D.BRM!D.SSP>,E.A1	;EXPECTED A1
4750	022044	012737	000001	005436	MOV	#1,E.B1	;MSG ID FOR EXPECTED MSG B1
4751	022052	005037	005440		CLR	E.A2	;EXPECTED MSG A2
4752	022056	012737	000002	005442	MOV	#2,E.B2	;MSG ID FOR EXPECTED MSG B2
4753	022064	012737	000003	005446	MOV	#3,E.B3	;MSG ID FOR EXPECTED MSG B3
4754							
4755	022072	004737	025264		JSR	PC,CHKMSG	;CHECK MSGS A0, B0, A1, B1
4756	022076	000003			.WORD	T.A2!T.B2!0	; & MSGS SPECIFIED HERE
4757	022100	104052			ERROR	52	;MSG A0 ERROR AFTER WRITE DATA CMD
4758	022102	104023			ERROR	23	;MSH B0 ERROR
4759	022104	104053			ERROR	53	;MSG A1 ERROR
4760	022106	104025			ERROR	25	;MSG B1 ERROR
4761	022110	000137	023374		JMP	12\$	
4762							
4763	022114	005237	001366		INC	TOCYL	
4764	022120	023727	001366	000012	CMP	TOCYL,#10.	;TRIED 10 CYLINDERS?
4765	022126	001003			BNE	18\$	;BR IF NO
4766	022130	104062			ERROR	62	;CANNOT WRITE ON 10 CONSEC CYL
4767	022132	000137	023374		JMP	12\$	
4768							

```

4769 022136 000137 021726      18$:  JMP      16$
4770
4771 022142 004737 026436      2$:  JSR      PC,SUBCLR
4772 022146 104024                ERROR    24      ;CERR AFTER SCLR
4773
4774 022150 104401 041553                TYPE    ,MSG49      ;TURN OFF AC
4775 022154 104401 042564                TYPE    ,MSG57      ;VERIFY BATTERY RETRACT FUNCTIONAL
4776
4777 022160 005737 001474                TST     BSERR      ;SEE IF ALIGN CART USED
4778 022164 001405                BEQ     15$        ;BR IF NO
4779 022166 104401 040625                TYPE    ,MSG37      ;PRESS SPACE WHEN DONE
4780 022172 004737 030452                JSR     PC,GETSP    ;GET SPACE
4781 022176 000523                BR      8$        ;SKIP ALL WRITING
4782
4783 022200 013737 001432 001160 15$:  MOV     T50000,$TMP0
4784 022206 005037 001162                CLR     $TMP1      ;BIT0=0;WRITE 0'S; BIT0=1:WRITE 1'S
4785
4786 022212 004737 026436      4$:  JSR     PC,SUBCLR
4787 022216 104024                ERROR    24      ;CERR AFTER SCLR
4788
4789 022220 032737 000001 001162                BIT     #BIT0,$TMP1
4790 022226 001004                BNE     5$        ;BR IF WRITING 1'S
4791 022230 012765 001456 000004                MOV     #DATA0,RKBA(R5) ;SETUP ALL 0'S
4792 022236 000403                BR      6$
4793
4794 022240 012765 001462 000004 5$:  MOV     #DATA1,RKBA(R5) ;SETUP ALL 1'S
4795 022246 052765 000020 000010 6$:  BIS     #BAI,RKCS2(R5)
4796 022254 012765 140400 000002                MOV     #-63.*256.,RKWC(R5) ;TRACK 0,1,2 63 SECTORS
4797 022262 005065 000006                CLR     RKDA(R5)   ;BEGIN AT TRK AND SECTOR 0
4798 022266 013765 001366 000020                MOV     TOCYL,RKDC(R5)
4799 022274 012737 000023 005340                MOV     #WRDATA,HCS1
4800 022302 004737 024504                JSR     PC,DATCMD   ;DO WRITE DATA CMD & GET CONTR RDY
4801 022306 104011                ERROR    11      ;CONTR NOT RDY
4802
4803 022310 004737 026106                JSR     PC,GSTAT
4804 022314 032737 100000 005340                BIT     #CERR,HCS1
4805 022322 001017                BNE     3$
4806 022324 005337 001160                DEC     $TMP0
4807 022330 001011                BNE     7$
4808 022332 104146                ERROR    146     ;CERR NOT SET BY TIMEOUT
4809 022334 104401 041626                TYPE    ,MSG50     ;TURN AC BACK ON
4810 022340 104401 043647                TYPE    ,MSG69     ;DEPRESS SPACE AFTER 'READY' LIGHT ON
4811 022344 004737 030452                JSR     PC,GETSP    ;GET SPACE
4812 022350 000137 023374                JMP     12$        ;EXIT TEST
4813
4814 022354 005237 001162      7$:  INC     $TMP1
4815 022360 000714                BR      4$        ;GO WRITE OPPOSITE DATA
4816

```

4817	022362	032737	000100	005370	3\$:	BIT	#D.ACLO,HMR3	
4818	022370	001001				BNE	10\$	
4819	022372	104147				ERROR	147	;AC LOW NOT SET
4820								
4821	022374	005237	005322		10\$:	INC	UNLD	;FOR VALID HALT
4822	022400	012737	070140	005430		MOV	#<D.DSC!D.PIP!D.SPIN!D.VV!D.DRA>,E.A0	;EXPECTED MSG A0
4823	022406	012737	010300	005432		MOV	#<D.SPLS!D.ACLO!D.FLT>,E.B0	
4824	022414	012737	044720	005434		MOV	#<D.UNLD!D.REV!D.CART!D.DOOR!D.BRHM!D.SSP>,E.A1	
4825	022422	012737	000001	005436		MOV	#1,E.B1	
4826								
4827	022430	004737	025264			JSR	PC,CHKMSG	;CHECK MSGS A0, B0, A1, B1
4828	022434	000000				.WORD	0!0!0	;# MSGS SPECIFIED HERE
4829	022436	104035				ERROR	35	;MSG A0 ERROR AFTER AC SWITCH OFF FROM HEADS LOADED
4830	022440	104203				ERROR	203	;MSH B0 ERROR
4831	022442	104036				ERROR	36	;MSG A1 ERROR
4832	022444	104037				ERROR	37	;MSG B1 ERROR
4833								
4834	022446	013737	005350	001164	8\$:	MOV	HDA,\$TMP2	;SAVE RKDA
4835	022454	013737	005350	001166		MOV	HDA,\$TMP3	;SAVE FOR TYPEOUT
4836								
4837	022462	104401	041626			TYPE	,MSG50	;TURN AC BACK ON
4838	022466	104401	043647			TYPE	,MSG69	;DEPRESS SPACE AFTER 'READY' LIGHT ON
4839	022472	004737	030452			JSR	PC,GETSP	;GET SPACE
4840								
4841	022476	004737	026436			JSR	PC,SUBCLR	
4842	022502	104024				ERROR	24	;CERR AFTER SCLR
4843								
4844	022504	032737	000100	005370		BIT	#D.ACLO,HMR3	
4845	022512	001401				BEQ	9\$	
4846	022514	104152				ERROR	152	;ACLO NOT RESET AFTER POWER UP
4847								
4848	022516	005037	005322		9\$:	CLR	UNLD	;FOR VALID HALT
4849	022522	012765	100000	000000		MOV	#CCLR,RKCS1(R5)	
4850	022530	013765	001222	000010		MOV	\$UNIT,RKCS2(R5)	;DRIVE #
4851	022536	012737	000003	005340		MOV	#PACK,HCS1	
4852	022544	004737	024446			JSR	PC,DOCMD	;DO PACK CMD & GET CONTR RDY
4853	022550	104116				ERROR	116	;CONTR NOT RDY
4854								
4855	022552	032737	000100	005366		BIT	#D.VV,HMR2	
4856	022560	001001				BNE	65\$	
4857	022562	104027				ERROR	27	;VOLUME VALID NOT SET AFTER PACK CMD
4858	022564				65\$:			
4859	022564	005737	001474			TST	BSERR	;SEE IF ALIGN CART USED
4860	022570	001402				BEQ	14\$	;BR IF NO
4861	022572	000137	023374			JMP	12\$	;ELSE EXIT TEST
4862								
4863	022576				14\$:			
4864								
4865	022576	004737	026436			JSR	PC,SUBCLR	
4866	022602	104024				ERROR	24	;CERR AFTER SCLR
4867								
4868	022604	005737	001164			TST	\$TMP2	;SEE IF TRK/SECTOR 0
4869	022610	001414				BEQ	73\$	;REPEAT,NO NEW DATA XFER TOOK PLACE
4870	022612	023727	001164	001023		CMP	\$TMP2,#1023	;SEE IF TRK 2,SECTOR 19
4871	022620	001410				BEQ	73\$	;REPEAT,NO OLD DATA TO CHECK AGAINST
4872	022622	032737	000001	001162		BIT	#BIT0,\$TMP1	

4873	022630	001006			BNE	66\$	:BR IF WRITING 1'S WHEN WLE OCCURRED
4874	022632	012765	001462	000004	MOV	#DATA1,RKBA(R5)	:WRITING 0'S:WLE SECTOR SHOULD HAVE ALL 1'S
4875	022640	000405			BR	67\$	
4876							
4877	022642	000137	022142		73\$: JMP	2\$	
4878							
4879	022646	012765	001456	000004	66\$: MOV	#DATA0,RKBA(R5)	:WRITING 1'S:WLE SECTOR SHOULD HAVE ALL 0'S
4880	022654	052765	000020	000010	67\$: BIS	#BA1,RKCS2(R5)	
4881	022662	012765	177400	000002	MOV	#-256.,RKWC(R5)	
4882	022670	013765	001166	000006	MOV	\$TMP3,RKDA(R5)	:REFRESH RKDA
4883	022676	017537	000004	001436	MOV	@RKBA(R5),WD2	:EXPECTED WORD FOR TRUERR TYPEOUT
4884	022704	013765	001366	000020	MOV	TOCYL,RKDC(R5)	
4885							
4886	022712	012737	000031	005340	MOV	#<WRTCHK>,HCS1	
4887	022720	004737	024504		JSR	PC,DATCMD	:DO DATA X FOR CMD & GET CONTR RDY
4888	022724	104015			ERROR	15	:NO RDY AFTER WRITE CHECK CMD
4889	022726	004737	026106		JSR	PC,GSTAT	:GET FRESH STATUS
4890	022732	032737	100000	005340	BIT	#CERR,HCS1	
4891	022740	001450			BEQ	75\$	
4892	022742	032737	040000	005342	BIT	#WCE,HCS2	:SEE IF WRITE CHECK ERROR
4893	022750	001405			BEQ	74\$	
4894	022752	016537	000024	001434	MOV	RKDB(R5),WD1	:ACTUAL WORD FOR PRINTOUT
4895	022760	104136			ERROR	136	:WCE AFTER WRITE CMD
4896	022762	000437			BR	75\$	
4897							
4898	022764	104022			74\$: ERROR	22	:CERR AFTER WRITE CHECK CMD
4899							
4900	022766	012737	010340	005430	MOV	#<0!D.SPIN!D.DRDY!D.VV!D.DRA>,E.A0	:EXPECTED MSG A0
4901	022774	005037	005432		CLR	E.B0	:EXPECTED MSG B0
4902	023000	012737	001720	005434	MOV	#<D.SPOK!D.CART!D.DOOR!D.BRHM!D.SSP>,E.A1	:EXPECTED A1
4903	023006	012737	000001	005436	MOV	#1,E.B1	:MSG ID FOR EXPECTED MSG B1
4904	023014	005037	005440		CLR	E.A2	:EXPECTED MSG A2
4905	023020	012737	000002	005442	MOV	#2,E.B2	:MSG ID FOR EXPECTED MSG B2
4906	023026	012737	000003	005446	MOV	#3,E.B3	:MSG ID FOR EXPECTED MSG B3
4907							
4908	023034	004737	025264		JSR	PC,CHKMSG	:CHECK MSGS A0, B0, A1, B1
4909	023040	000003			.WORD	T.A2!T.B2!0	:& MSGS SPECIFIED HERE
4910	023042	104057			ERROR	57	:MSG A0 ERROR AFTER WRITE CHECK CMD
4911	023044	104031			ERROR	31	:MSG B0 ERROR
4912	023046	104060			ERROR	60	:MSG A1 ERROR
4913	023050	104032			ERROR	32	:MSG B1 ERROR
4914	023052	104401	044035		TYPE	,MSG72	:ABORTING BALANCE OF TESTS
4915	023056	000137	023374		JMP	ENDRV	
4916							
4917	023062				75\$: NOP		
4918	023062	000240			NOP		
4919	023064	000240			NOP		
4920							
4921	023066	023727	001164	000400	CMP	\$TMP2,#400	:SEE IF WRL AT TRK 1, SECTOR 0
4922	023074	001004			BNE	68\$	:BR IF NO
4923	023076	012765	000025	000006	MOV	#21.,RKDA(R5)	:ELSE SECTOR BEFORE WRL IS TRK 0, SEC 21
4924	023104	000415			BR	70\$	
4925	023106	023727	001164	001000	68\$: CMP	\$TMP2,#1000	:SEE IF WRL AT TRK 2,SECTOR 0
4926	023114	001004			BNE	69\$	:BR IF NO
4927	023116	012765	000425	000006	MOV	#425,RKDA(R5)	:ELSE SECTOR BEFORE WRL IS TRK 1, SEC 21
4928	023124	000405			BR	70\$	

```
4929
4930 023126 005337 001164 69$: DEC $TMP2 ;GET SECTOR BEFORE WRL
4931 023132 013765 001164 000006 MOV $TMP2,RKDA(R5)
4932 023140 016537 000006 001166 70$: MOV RKDA(R5),$TMP3 ;FOR ERROR PRINTOUT
4933 023146 032737 000001 001162 BIT #BIT0,$TMP1
4934 023154 001004 BNE 71$ ;BR IF WRITING 1'S WHEN WLE OCCURRED
4935 023156 012765 001456 000004 MOV #DATA0,RKBA(R5) ;WRITING 0'S:WLE SECTOR -1 SHOULD HAVE ALL 0'S
4936 023164 000403 BR 72$
4937
4938 023166 012765 001462 000004 71$: MOV #DATA1,RKBA(R5) ;WRITING 1'S:WLE SECTOR -1 SHOULD HAVE ALL 1'S
4939 023174 052765 000020 000010 72$: BIS #BAI,RKCS2(R5)
4940 023202 012765 177400 000002 MOV #-256.,RKWC(R5)
4941 023210 017537 000004 001436 MOV @RKBA(R5),WD2 ;EXPECTED WORD FOR TRUERR TYPEOUT
4942 023216 013765 001366 000020 MOV TOCYL,RKDC(R5)
4943
4944 023224 012737 000031 005340 MOV #<WRTCHK>,HCS1
4945 023232 004737 024504 JSR PC,DATCMD ;DO DATA X FOR CMD & GET CONTR RDY
4946 023236 104015 ERROR 15 ;NO RDY AFTER WRITE CHECK CMD
4947 023240 004737 026106 JSR PC,GSTAT ;GET FRESH STATUS
4948 023244 032737 100000 005340 BIT #CERR,HCS1
4949 023252 001450 BEQ 77$
4950 023254 032737 040000 005342 BIT #WCE,HCS2 ;SEE IF WRITE CHECK ERROR
4951 023262 001405 BEQ 76$
4952 023264 016537 000024 001434 MOV RKDB(R5),WD1 ;ACTUAL WORD FOR PRINTOUT
4953 023272 104137 ERROR 137 ;WCE AFTER WRITE CMD
4954 023274 000437 BR 77$
4955
4956 023276 104022 76$: ERROR 22 ;CERR AFTER WRITE CHECK CMD
4957
4958 023300 012737 010340 005430 MOV #<0!D.SPIN!D.DRDY!D.VV!D.DRA>,E.A0 ;EXPECTED MSG A0
4959 023306 005037 005432 CLR E.B0 ;EXPECTED MSG B0
4960 023312 012737 001720 005434 MOV #<D.SPOK!D.CART!D.DOOR!D.BRM!D.SSP>,E.A1 ;EXPECTED A1
4961 023320 012737 000001 005436 MOV #1,E.B1 ;MSG ID FOR EXPECTED MSG B1
4962 023326 005037 005440 CLR E.A2 ;EXPECTED MSG A2
4963 023332 012737 000002 005442 MOV #2,E.B2 ;MSG ID FOR EXPECTED MSG B2
4964 023340 012737 000003 005446 MOV #3,E.B3 ;MSG ID FOR EXPECTED MSG B3
4965
4966 023346 004737 025264 JSR PC,CHKMSG ;CHECK MSGS A0, B0, A1, B1
4967 023352 000003 .WORD T.A2!T.B2!0 ;& MSGS SPECIFIED HERE
4968 023354 104057 ERROR 57 ;MSG A0 ERROR AFTER WRITE CHECK CMD
4969 023356 104031 ERROR 31 ;MSG B0 ERROR
4970 023360 104060 ERROR 60 ;MSG A1 ERROR
4971 023362 104032 ERROR 32 ;MSG B1 ERROR
4972 023364 104401 044035 TYPE ,MSG72 ;ABORTING BALANCE OF TESTS
4973 023370 000137 023374 JMP ENDRV
4974
4975 023374 77$:
4976
4977 023374 12$:
4978
4979 023374 ENDRV:
4980
4981 :*****
4982 :*TEST 20 END OF PROGRAM
4983 :*
4984 :* THIS IS NOT A TEST BUT A LINKAGE TO PERFORM ALL THE
```



```
4985 :* ABOVE TESTS FOR THE NEXT DRIVE PRESENT.  
4986 :* THE NEXT TEST IS ENTERED ONLY AFTER ALL DRIVES PRESENT  
4987 :* HAVE BEEN TESTED.  
4988 :* DO NOT LOOP ON THIS 'TEST'.  
4989 :*  
4990 :*****  
4991 023374 000004 TST20: SCOPE  
4992 023376 012737 000001 001174 MOV #1,$TIMES ;;DO 1 ITERATION  
4993 023404 012706 001100 MOV #STACK,SP ;RESTORE STK PTR  
4994  
4995 023410 005237 001220 INC $DEVCT ;INCR COUNT FOR # OF DRIVES THAT ARE CHECKED  
4996 023414 023737 005460 001220 CMP DRIVS,$DEVCT ;ARE ALL DRIVES PRESENT TESTED?  
4997 023422 001402 BEQ TST21 ;GO TO NEXT TEST IF YES  
4998 023424 000137 012402 JMP NUDRV ;IF NOT, GO BACK & TEST NEXT DRIVE PRESENT.  
4999
```

```
5000 :*****  
5001 :*TEST 21 MULTIPLE DRIVE DETECTION TEST  
5002 :*  
5003 :* THIS TEST IS PERFORMED ONLY ONCE AT THE END OF PASS 1  
5004 :* AND IS BYPASSED IF ONLY ONE DRIVE IS PRESENT.  
5005 :*  
5006 :* THE MULTIPLE DRIVE DETECTION LOGIC IS TESTED BY THE FOLLOWING METHOD:  
5007 :*  
5008 :* A. HEADS MUST BE LOADED (SECTOR PULSES REQ'D)  
5009 :* B. THE OPERATOR IS INSTRUCTED TO INSERT THE SAME UNIT SELECT  
5010 :* PLUG NUMBER (1 PAIR AT A TIME) ON ANY 2 DRIVES  
5011 :* TO BE TESTED  
5012 :* C. THE OPERATOR THEN DEPRESSES THE SPACE BAR TO CONTINUE THE TEST  
5013 :* OR A CONT-E TO EXIT THE TEST  
5014 :*  
5015 :* THE PROGRAM VERIFIES THAT MULTIPLE DRIVE SELECT & DRIVE UNSAFE  
5016 :* BOTH SET & THAT THE DRIVE UNLOADS  
5017 :*  
5018 :* THE OPERATOR IS ASKED TO VERIFY THAT HEADS UNLOAD FROM BOTH DRIVES  
5019 :*  
5020 :*****
```

```
5021 023430 000004 TST21: SCOPE  
5022 023432 012737 000001 001174 MOV #1,$TIMES ;;DO 1 ITERATION  
5023 023440 012706 001100 MOV #STACK,SP ;RESTORE STACK PTR  
5024  
5025 023444 005737 001216 TST $PASS  
5026 023450 001402 BEQ 1$ ;DO TEST ONLY IN 1ST PASS  
5027 023452 000137 023722 JMP 11$ ;ELSE EXIT TEST  
5028  
5029 023456 023727 005460 000001 1$: CMP DRIVS,#1  
5030 023464 001004 BNE 2$ ;BR IF MORE THAN 1 DRIVE PRESENT  
5031 023466 104401 043175 TYPE ,MSG62 ;BYPASS TEST, ONLY 1 DRIVE PRESENT  
5032 023472 000137 023722 JMP 11$ ;ELSE EXIT TEST  
5033  
5034 023476 104401 040022 2$: TYPE ,MSG28 ;MULT DRV DETECTION TEST  
5035 023502 104401 042017 TYPE ,MSG52 ;PRESS CONT-E TO EXIT OR SPACE TO CONTINUE  
5036 023506 004737 030412 JSR PC,CCSP ;INPUT CONT-E OR SPACE  
5037 023512 000137 023722 JMP 11$ ;RETURN HERE FOR CONT-E  
5038  
5039 023516 104401 042632 TYPE ,MSG58 ;LOAD HEADS ON ALL DRIVES  
5040 023522 104401 040625 TYPE ,MSG37 ;PRESS SPACE WHEN SONE
```

```

5041 023526 004737 030452          JSR    PC,GETSP          ;GET SPACE
5042
5043 023532 004737 026436          3$:   JSR    PC,SUBCLR
5044 023536 104024                    ERROR  24                ;CERR AFTER SCLR
5045
5046 023540 104401 042714          TYPE  ,MSG59            ;INSERT SAME UNIT SEL PLUG # IN 2 DRIVES
5047 023544 104401 040625          TYPE  ,MSG37            ;DEPRESS SPACE BAR WHEN DONE
5048 023550 004737 030452          JSR    PC,GETSP          ;GET SPACE
5049
5050 023554 005000                    CLR    R0                ;DRIVE # COUNTER
5051
5052 023556 012765 100000 000000 6$:   MOV    #CCLR,RKCS1(R5)
5053 023564 010065 000010                    MOV    R0,RKCS2(R5)    ;DRIVE #
5054 023570 012737 000001 005340    MOV    #SELDRV,HCS1
5055 023576 053737 001170 005340    BIS    $TMP4,HCS1      ;ADD CDT IF RK07
5056 023604 013765 005340 000000    MOV    HCS1,RKCS1(R5) ;GET STATUS
5057 023612 013737 001432 005376    MOV    T5000,TEMP1
5058 023620 004737 025232          JSR    PC,DLY            ;DO DELAY TO CATCH MDS
5059
5060 023624 032765 001000 000010    BIT    #MDS,RKCS2(R5) ;SEE IF THAT DRIVE HAS MDS
5061 023632 001006                    BNE    7$                ;BR IF YES
5062 023634 005200                    INC    R0                ;ELSE TRY ANOTHER DRIVE
5063 023636 020027 000010    CMP    R0,#8.           ;SEE IF ALL DRIVES TESTED
5064 023642 001345                    BNE    6$                ;BR IF NO
5065 023644 104176                    ERROR  176              ;CANNOT FIND MDS
5066 023646 000411                    BR     10$              ;TRY AGAIN
5067
5068 023650 104401 043127          7$:   TYPE  ,MSG61
5069 023654 010046                    MOV    R0,-(SP)        ;MULT DRIVES FOUND ON DRIVE #
5070
5071 023656 104403                    TYPOS
5072 023660 001                    .BYTE 1                ;;SAVE R0 FOR TYPEOUT
5073 023661 000                    .BYTE 0                ;;TYPE UNIT NO
5074 023662 104401 043607          TYPE  ,MSG68            ;;GO TYPE--OCTAL ASCII
5075 023666 005237 005322          INC    UNLD             ;;TYPE 1 DIGIT(S)
5076 023672 104401 043017          10$:  TYPE  ,MSG60            ;;SUPPRESS LEADING ZEROS
5077 023676 104401 043271          TYPE  ,MSG63            ;VERIFY BOTH DRIVES UNLOADED
5078 023702 004737 030412          JSR    PC,CCSP          ;FOR VALID HALT
5079 023706 000137 023722          JMP    11$              ;INSERT CORRECT UNIT SEL PLUG & LOAD HEADS
5080 023712 005037 005322          CLR    UNLD             ;DEPRESS CONT-E OR SPACE BAR WHEN DONE
5081 023716 000137 023532          JMP    3$                ;INPUT CONT-E OR SPACE
5082 023722 005037 005322          11$:  CLR    UNLD             ;RETURN HERE FOR CONTROL-E (EXIT)
5083
5084 023726 004737 026436          JSR    PC,SUBCLR
5085 023732 104024                    ERROR  24                ;CERR AFTER SCLR
    
```

```

5086 .SBTTL END OF PASS ROUTINE
5087
5088 *****
5089 :*INCREMENT THE PASS NUMBER ($PASS)
5090 :*TYPE 'END PASS #XXXXX' (WHERE XXXXX IS A DECIMAL NUMBER)
5091 :*IF THERES A MONITOR GO TO IT
5092 :*IF THERE ISN'T JUMP TO ST5XY
5093
5094 $EOP:
5095 SCOPE
5096 CLR $STNM ;;ZERO THE TEST NUMBER
5097 CLR $TIMES ;;ZERO THE NUMBER OF ITERATIONS
5098 INC $PASS ;;INCREMENT THE PASS NUMBER
5099 BIC #10000,$PASS ;;DON'T ALLOW A NEG. NUMBER
5100 DEC (PC)+ ;;LOOP?
5101 $EOPCT: .WORD 1
5102 BGT $DOAGN ;;YES
5103 MOV (PC)+,@(PC)+ ;;RESTORE COUNTER
5104 $ENDCT: .WORD 1
5105 $EOPCT
5106 TYPE $SENDMG ;;TYPE 'END PASS #'
5107 MOV $PASS,-(SP) ;;SAVE $PASS FOR TYPEOUT
5108 TYPDS ;;GO TYPE--DECIMAL ASCII WITH SIGN
5109 TYPE $ENULL ;;TYPE A NULL CHARACTER
5110 $GET42: MOV @#42,R0 ;;GET MONITOR ADDRESS
5111 BEQ $DOAGN ;;BRANCH IF NO MONITOR
5112 RESET ;;CLEAR THE WORLD
5113 $ENDAD: JSR PC,(R0) ;;GO TO MONITOR
5114 NOP ;;SAVE ROOM
5115 NOP ;;FOR
5116 NOP ;;ACT11
5117 $DOAGN:
5118 JMP @(PC)+ ;;RETURN
5119 $RTNAD: .WORD ST5XY
5120 $ENULL: .BYTE -1,-1,0 ;;NULL CHARACTER STRING
5121 $ENDMG: .ASCIZ <15><12>/END PASS #/
5122
5123 ST5XY: CMPB #APTENV,$ENV ;;RUN UNDER APT ?
5124 BNE 2$ ;;BRANCH IF NOT
5125 CMP #2,$PASS ;;TWO PASSES DONE ?
5126 BHI 2$ ;;BRANCH IF NOT
5127 INC $STNM ;;CHANGE THE TEST NUMBER
5128 BR 1$ ;;LOOPING
5129 JMP ST5 ;;EXIT
5130

```

```

5131      .SBTTL  SUBROUTINES
5132
5133      ;SUBROUTINE TO CLEAR ALL FLAGS FROM DDUMP THRU DOTIM
5134      ;
5135
5136      024110  012700  005450  CLRFLG: MOV      #DDUMP,R0
5137      024114  012701  177757      MOV      #-17.,R1
5138      024120  005020      1$:      CLR      (R0)+
5139      024122  005201          INC      R1
5140      024124  001375          BNE     1$
5141      024126  000207          RTS     PC
5142
5143
5144      ;TYPE PROGRAM ID IF FTITLE=0
5145      ;
5146
5147      024130  005737  001360  TITLE:  TST      FTITLE
5148      024134  001024          BNE     1$
5149      024136  005237  001360      INC      FTITLE
5150      024142  104401  035736      TYPE     ,MSG1      ;PROGRAM ID
5151      .SBTTL  GET VALUE FOR SOFTWARE SWITCH REGISTER
5152      024146  005737  000042      TST     @#42      ;;ARE WE RUNNING UNDER XXDP/ACT?
5153      024152  001012          BNE     64$      ;;BRANCH IF YES
5154      024154  123727  001230  000001  CMPB    $ENV,#1    ;;ARE WE RUNNING UNDER APT?
5155      024162  001406          BEQ     64$      ;;BRANCH IF YES
5156      024164  023727  001140  000176      CMP     SWR,#SWREG ;;SOFTWARE SWITCH REG SELECTED?
5157      024172  001005          BNE     65$      ;;BRANCH IF NO
5158      024174  104406          GTSWR          ;;GET SOFT-SWR SETTINGS
5159      024176  000403          BR      65$
5160      024200  112737  000001  001134  64$:   MOVB    #1,$AUTOB  ;;SET AUTO-MODE INDICATOR
5161      024206          65$:
5162      024206  000207      1$:   RTS     PC
5163
5164
5165      ;ROUTINE TO INPUT DRIVE NOS. TYPED IN & SET
5166      ;DRIVS, DRIVO-DRIV7 REGISTERS APPROPRIATELY
5167      ;
5168
5169      024210  104411  GDRVS:  RDLIN
5170      024212  012600      MOV     (SP)+,R0      ;GET STARTING ADDR OF ASCII STRING
5171      024214  012701  177770      MOV     #-8.,R1      ;SET UP COUNT
5172      024220  112002      1$:   MOVB    (R0)+,R2      ;GET ASCII CHAR
5173      024222  042702  177400      BIC    #177400,R2    ;MASK HI BYTE
5174      024226  012703  005462      MOV     #DRIVO,R3    ;DRIVE FLAG ADDR
5175      024232  012704  000060      MOV     #60,R4
5176
5177      024236  020402      2$:   CMP     R4,R2      ;WAS TYPED CHAR 0 THRU 7?
5178      024240  001415          BEQ     3$      ;BRANCH IF YES
5179      024242  005723          TST     (R3)+    ;NO, INCREMENT DR FLAG ADDR
5180      024244  005204          INC     R4
5181      024246  020427  000070      CMP     R4,#70
5182      024252  001371          BNE     2$      ;S/B 0-7 OR TERMINATOR
5183      024254  005702          TST     R2
5184      024256  001022          BNE     4$
5185      024260  020127  177770      CMP     R1,#-8.
5186      024264  001426          BEQ     6$      ;DEFAULT ALL DRIVES

```

```

5187 024266 005037 005510 7$: CLR SIZFLG ;BYPASS TEST 1 (SIZING)
5188 024272 000207 RTS PC ;FOUND TERMINATOR, EXIT
5189
5190 024274 005213 3$: INC @R3 ;SET UP FLAG FOR THE DRIVE
5191 024276 005237 005460 INC DRIVS ;INCREMENT TOTAL # DRIVES TO BE TESTED
5192 024302 112002 MOV#B (R0)+,R2 ;GET NEXT ASCII CHAR.
5193 024304 042702 177400 BIC #177400,R2 ;MASK
5194 024310 022702 000054 CMP #54,R2 ;IS IT A COMMA?
5195 024314 001407 BEQ 5$ ;YES, GO TO NEXT WORD.
5196 024316 005702 TST R2 ;NO, IS IT A TERMINATOR?
5197 024320 001001 BNE 4$ ;IF NOT, SOMETHING WRONG.
5198 024322 000761 BR 7$ ;FOUND TERMINATOR, EXIT
5199
5200 024324 104401 044402 4$: TYPE ,EM1 ;ONLY 0-7 ALLOWED.
5201 024330 000137 010016 JMP PRGSRT ;START ALL OVER
5202
5203 024334 005201 5$: INC R1 ;S/B NO MORE THAN 8 DIFF
5204 024336 001330 BNE 1$ ;DRIVES TYPED IN.
5205 024340 000771 BR 4$ ;IF MORE, HAVE ERROR.
5206
5207 024342 005237 005510 6$: INC SIZFLG ;DO TEST 1 (SIZING)
5208 024346 000207 RTS PC ;EXIT.
5209
5210 ;
5211 ;ROUTINE TO INPUT RKBAS OR DEFAULT.
5212 ;
5213 ;
5214 024350 104412 GBA: RDOCT
5215 024352 012600 MOV (SP)+,R0 ;GET LOW ORDER FROM STACK
5216 024354 005700 TST R0
5217 024356 001403 BEQ 1$ ;BRANCH IF DEFAULT.
5218 024360 010037 001264 MOV R0,$BASE
5219 024364 000207 RTS PC
5220 024366 012737 177440 001264 1$: MOV #177440,$BASE ;DEFAULT VALUE
5221 024374 000207 RTS PC
5222
5223 ;
5224 ;ROUTINE TO INPUT RKVEC OR DEFAULT
5225 ;
5226 ;
5227 024376 104412 GINT: RDOCT
5228 024400 012600 MOV (SP)+,R0 ;GET LOW ORDER FROM STACK
5229 024402 005700 TST R0
5230 024404 001405 BEQ 1$ ;BRANCH IF DEFAULT
5231 024406 010037 001334 MOV R0,RKVEC
5232 024412 004737 024430 2$: JSR PC,SETINT
5233 024416 000207 RTS PC
5234 024420 012737 000210 001334 1$: MOV #210,RKVEC ;DEFAULT VALUE
5235 024426 000771 BR 2$
5236
5237 ;
5238 ;ROUTINE TO SETUP INTERRUPT VECTOR & PRIORITY
5239 ;
5240 ;
5241 024430 013700 001334 SETINT: MOV RKVEC,R0
5242 024434 012720 031152 MOV #INTER,(R0)+ ;INTER ADDR TO RKVEC

```

```

5243 024440 013710 001336          MOV    RKPRI,(R0)      ;PR5 TO RKVEC+2
5244 024444 000207          RTS    PC
5245
5246
5247
5248          ; THIS ROUTINE SETS CDT IN RKCS1 IF DRIVE UNDER TEST IS AN RK07.
5249          ; ENTER WITH COMMAND IN HCS1
5250
5251 024446 053737 001170 005340 DOCMD: BIS    $TMP4,HCS1      ;ADD CDT IF RK07
5252 024454 013765 005340 000000      MOV    HCS1,RKCS1(R5) ;DO COMMAND
5253 024462 013737 001424 005376      MOV    T10,TEMP1
5254 024470 004737 024542          JSR    PC,FRDY        ;FIND CONTR READY
5255 024474 000207          RTS    PC             ;SET HERE IF NOT RDY
5256 024476 062716 000002          ADD    #2,(SP)       ;ELSE SKIP OVER ERROR
5257 024502 000207          RTS    PC
5258
5259          ; THIS ROUTINE IS SIMILAR TO THE ABOVE BUT IS USED FOR DATA TRANSFERS
5260          ; & REQUIRES A LONGER TIMEOUT
5261
5262 024504 053737 001170 005340 DATCMD: BIS    $TMP4,HCS1      ;ADD CDT IF RK07
5263 024512 013765 005340 000000      MOV    HCS1,RKCS1(R5) ;DO CMD
5264 024520 013737 001432 005376      MOV    T5000,TEMP1
5265 024526 004737 024542          JSR    PC,FRDY        ;FIND CONTR RDY
5266 024532 000207          RTS    PC
5267 024534 062716 000002          ADD    #2,(SP)
5268 024540 000207          RTS    PC
5269
5270          ;
5271          ; ROUTINE TO FIND CONTROLLER READY (RDY) DURING A DELAY
5272          ; ENTER WITH A COUNT IN TEMP1
5273          ; RETURN IF RDY NOT PRESENT (ERROR CONDITION)
5274          ; RETURN +2 IF RDY PRESENT (SKIP OVER ERROR)
5275          ; STATUS IS OBTAINED BEFORE THE RETURN FOR EITHER CASE
5276
5277 024542 032765 000200 000000 FRDY:  BIT    #RDY,RKCS1(R5)
5278 024550 001010          BNE    1$
5279 024552 005337 005376          DEC    TEMP1
5280 024556 001371          BNE    FRDY
5281 024560 004737 024676          JSR    PC,HOLD        ;STORE ALL RK611 REGS IN HOLDING REGS.
5282 024564 004737 026024          JSR    PC,CKCERR      ;CHECK FOR SPECIAL CERR
5283 024570 000207          RTS    PC             ;NO RDY, EXIT
5284 024572 062716 000002 1$:    ADD    #2,(SP)       ;SKIP OVER ERROR
5285 024576 004737 024676          JSR    PC,HOLD
5286 024602 004737 026024          JSR    PC,CKCERR      ;CHECK FOR SPECIAL CERR
5287 024606 000207          RTS    PC
5288
5289          ;
5290          ; ROUTINE TO FIND CONTROLLER READY AND STORE DRIVE REGS ONLY
5291
5291 024610 032765 000200 000000 FRDY1: BIT    #RDY,RKCS1(R5)
5292 024616 001014          BNE    1$
5293 024620 005337 005376          DEC    TEMP1
5294 024624 001371          BNE    FRDY1
5295 024626 016537 000034 005366      MOV    RKMR2(R5),HMR2
5296 024634 016537 000036 005370      MOV    RKMR3(R5),HMR3
5297 024642 004737 026024          JSR    PC,CKCERR      ;CHECK FOR SPECIAL CERR CONDITIONS
5298 024646 000207          RTS    PC             ;NO RDY, EXIT

```

```
5299 024650 062716 000002 1$: ADD #2,(SP) ;SKIP OVER ERROR
5300 024654 016537 000034 005366 MOV RKMR2(R5),HMR2
5301 024662 016537 000036 005370 MOV RKMR3(R5),HMR3
5302 024670 004737 026024 JSR PC,CKCERR ;CHECK FOR SPECIAL CERR CONDITIONS
5303 024674 000207 RTS PC
5304
5305
5306 ;STORE ALL RK611 REGISTERS IN HOLDING REGS
5307
5308
5309 024676 016537 000000 005340 HOLD: MOV RKCS1(R5),HCS1
5310 024704 016537 000010 005342 MOV RKCS2(R5),HCS2
5311 024712 016537 000002 005344 MOV RKWC(R5),HWC
5312 024720 016537 000004 005346 MOV RKBA(R5),HBA
5313 024726 016537 000006 005350 MOV RKDA(R5),HDA
5314 024734 016537 000012 005352 MOV RKDS(R5),HDS
5315 024742 016537 000014 005354 MOV RKER(R5),HER
5316 024750 016537 000016 005356 MOV RKASOF(R5),HASOF
5317 024756 016537 000020 005360 MOV RKDC(R5),HDC
5318 024764 016537 000026 005364 MOV RKMR1(R5),HMR1
5319 024772 016537 000034 005366 MOV RKMR2(R5),HMR2
5320 025000 016537 000036 005370 MOV RKMR3(R5),HMR3
5321 025006 016537 000030 005372 MOV RKECPS(R5),HPOS
5322 025014 016537 000032 005374 MOV RKECPT(R5),HPAT
5323 025022 000207 RTS PC
5324
5325
5326 ;ROUTINE TO CHECK FOR CORRECT ATTN
5327 ;RETURN IF ATTN NOT PRESENT (ERROR CONDITION)
5328 ;RETURN +2 IF ATTN PRESENT (SKIP OVER ERROR)
5329
5330 025024 010446 TSTATN: MOV R4,-(SP) ;SAV R4
5331 025026 013704 001222 MOV $UNIT,R4
5332 025032 136437 005330 005357 BITB ATTN(R4),HASOF+1
5333 025040 001404 BEQ 1$ ;BRANCH IF ATTN NOT PRESENT
5334 025042 012604 MOV (SP)+,R4 ;RESTOR R4
5335 025044 062716 000002 ADD #2,(SP) ;INCR RET ADDR TO JUMP OVER ERROR.
5336 025050 000207 RTS PC
5337 025052 012604 1$: MOV (SP)+,R4 ;RESTOR R4
5338 025054 000207 RTS PC
5339
5340
5341
5342 ;ROUTINE TO FIND ATTN WITHIN TIMES GREATER THAN 1 SEC
5343 ;ENTER WITH TIME IN SECONDS IN TEMP2
5344 ;RETURN IF NO ATTN (ERROR CONDITION)
5345 ;RETURN +2 IF ATTN FOUND
5346 ;STATUS IS OBTAINED BEFORE THE RETURN FOR EITHER CASE
5347
5348
5349 025056 010446 FATT1: MOV R4,-(SP) ;SAV R4
5350 025060 012737 177777 005376 3$: MOV #-1,TEMP1
5351 025066 013704 001222 MOV $UNIT,R4
5352 025072 136465 005330 000017 1$: BITB ATTN(R4),RKASOF+1(R5) ;FIND CORRECT ATTN
5353 025100 001014 BNE 2$
5354 025102 005337 005376 DEC TEMP1
```

```

5355 025106 001371          BNE      1$
5356 025110 005337 005400    DEC      TEMP2
5357 025114 001361          BNE      3$
5358 025116 005065 000026    CLR      RKMR1(R5)      ;SELECT WORD 0
5359 025122 004737 026106    JSR      PC,GSTAT      ;GET LATEST STATUS
5360 025126 012604          MOV      (SP)+,R4      ;RESTOR R4
5361 025130 000207          RTS      PC
5362
5363 025132 005065 000026    2$:    CLR      RKMR1(R5)
5364 025136 004737 026106    JSR      PC,GSTAT      ;GET STATUS AFTER ATTN SEEN
5365 025142 012604          MOV      (SP)+,R4      ;RESTOR R4
5366 025144 062716 000002    ADD      #2,(SP)      ;SKIP OVER ERROR
5367 025150 000207          RTS      PC
5368
5369
5370          ;
5371          ;ROUTINE TO FIND ATTN WITHIN 1 SEC
5372          ;ENTER WITH COUNT IN TEMP1
5373          ;RETURN IF NO ATTN (ERROR)
5374          ;RETURN +2 IF ATTN FOUND
5375          ;STATUS IS OBTAINED BEFORE THE RETURN FOR EITHER CASE
5376          ;
5377          ;
5377 025152 010446          FATT2:  MOV      R4,-(SP)      ;SAV R4
5378 025154 013704 001222    2$:    MOV      $UNIT,R4
5379 025160 136465 005330 000017  BITB     ATTN(R4),RKASOF+1(R5) ;FIND CORRECT ATTN
5380 025166 001011          BNE      1$
5381 025170 005337 005376    DEC      TEMP1
5382 025174 001367          BNE      2$
5383 025176 005065 000026    CLR      RKMR1(R5)      ;SELECT WORD 0
5384 025202 004737 026106    JSR      PC,GSTAT      ;GET LATEST STATUS.
5385 025206 012604          MOV      (SP)+,R4      ;RESTOR R4
5386 025210 000207          RTS      PC
5387 025212 005065 000026    1$:    CLR      RKMR1(R5)
5388 025216 004737 026106    JSR      PC,GSTAT
5389 025222 012604          MOV      (SP)+,R4      ;RESTOR R4
5390 025224 062716 000002    ADD      #2,(SP)      ;SKIP OVER ERROR
5391 025230 000207          RTS      PC
5392
5393          ;
5394          ;ENTER WITH A COUNT IN TEMP1
5395          ;THE DELAY IS APPROX 17 US/ITERATION + 12 US TO EXIT
5396          ;WHEN COUNT IS 0. BASED ON AN 11/05
5397          ;
5397 025232 005737 005376    DLY:    TST      TEMP1          ;5.6 US
5398 025236 001403          BEQ      1$            ;2.5 US
5399 025240 005337 005376    DEC      TEMP1          ;6.8 US
5400 025244 000772          BR       DLY           ;2.5 US
5401 025246 000207          1$:    RTS      PC          ;3.8 US
5402
5403          ;
5404          ;THIS ROUTINE TYPES BYPASSED DRIVE#. ENTER WITH DRIVE# IN R0
5405          ;
5406 025250 104401 037367    BYP:    TYPE     ,MSG14      ;BYPASS DRIVE
5407 025254 010046          MOV      R0,-(SP)      ;;SAVE R0 FOR TYPEOUT
5408          ;TYPE DR#
5409 025256 104403          TYPOS   1              ;;GO TYPE--OCTAL ASCII
5410 025260 001          .BYTE   1              ;;TYPE 1 DIGIT(S)

```



```

5411 025261 000 .BYTE 0 ;:SUPPRESS LEADING ZEROS
5412 025262 000207 RTS PC
5413
5414 ;:THIS ROUTINE READS ALL MSG A & B WORDS & CHECKS THEM AS REQ'D.
5415
5416 025264 017637 000000 001504 CHKMSG: MOV @ (SP),CHKFLG ;:PASS MSGS TO BE TESTED
5417 025272 062716 000002 ADD #2,(SP) ;:BUMP RETURN ADDR TO 1ST ERROR
5418 025276 004737 026142 JSR PC,GSTAT1 ;:GET ALL ACTUAL DRIVE & CONTR STATUS
5419
5420 025302 053737 001222 005430 BIS $UNIT,E.A0 ;:SET UNIT #
5421 025310 053737 001222 005434 BIS $UNIT,E.A1
5422 025316 053737 001222 005440 BIS $UNIT,E.A2
5423 025324 053737 001222 005444 BIS $UNIT,E.A3
5424
5425 025332 053737 012662 005430 BIS E.DDT,E.A0 ;:ADD DRIVE TYPE
5426
5427 025340 013746 005376 MOV TEMP1,-(SP) ;:SAVE TEMP1
5428
5429 025344 013737 005430 005376 MOV E.A0,TEMP1
5430 025352 004737 030306 JSR PC,SBPAR ;:GET PARITY FOR MSG A0
5431 025356 013737 005376 005430 MOV TEMP1,E.A0
5432
5433 025364 013737 005434 005376 MOV E.A1,TEMP1
5434 025372 004737 030306 JSR PC,SBPAR ;:GET PARITY FOR MSG A1
5435 025376 013737 005376 005434 MOV TEMP1,E.A1
5436
5437 025404 013737 005440 005376 MOV E.A2,TEMP1
5438 025412 004737 030306 JSR PC,SBPAR ;:GET PARITY FOR MSG A2
5439 025416 013737 005376 005440 MOV TEMP1,E.A2
5440
5441 025424 013737 005432 005376 MOV E.B0,TEMP1
5442 025432 004737 030306 JSR PC,SBPAR ;:GET PARITY FOR MSG B0
5443 025436 013737 005376 005432 MOV TEMP1,E.B0
5444
5445 025444 013737 005436 005376 MOV E.B1,TEMP1
5446 025452 004737 030306 JSR PC,SBPAR ;:GET PARITY FOR MSG B1
5447 025456 013737 005376 005436 MOV TEMP1,E.B1
5448
5449 025464 013737 005442 005376 MOV E.B2,TEMP1
5450 025472 004737 030306 JSR PC,SBPAR ;:GET PARITY FOR MSG B2
5451 025476 013737 005376 005442 MOV TEMP1,E.B2
5452
5453 025504 013737 005446 005376 MOV E.B3,TEMP1
5454 025512 004737 030306 JSR PC,SBPAR ;:GET PARITY FOR MSG B3
5455 025516 013737 005376 005446 MOV TEMP1,E.B3
5456
5457 025524 012637 005376 MOV (SP)+,TEMP1 ;:RESTORE TEMP1
5458 025530 013737 001176 001172 MOV $ESCAPE,$TMP5 ;:SAVE ESCAPE
5459
5460 025536 023737 005410 005430 CMP H.A0,E.A0 ;:TEST MSG A0
5461 025544 001411 BEQ 2$ ;:BR IF OK
5462 025546 012737 025560 001176 MOV #1$, $ESCAPE ;:ELSE SETUP ESCAPE
5463 025554 011646 MOV (SP),-(SP) ;:COPY RET ADDR.
5464 025556 000207 RTS PC ;:& RETURN TO MAINLINE ERROR
5465
5466 025560 032777 001000 153352 1$: BIT #SW9,@SWR ;:RET HERE FROM MAINLINE ERROR
  
```

```

5467 025566 001107          BNE      20$          :& BR IF LOOP ON ERROR
5468 025570 062716 000002 2$:      ADD      #2,(SP)      :BUMP RET ADDR TO NEXT ERROR
5469
5470 025574 023737 005412 005432          CMP      H.B0,E.B0      :TEST MSG B0
5471 025602 001411          BEQ      5$            :BR IF OK
5472 025604 012737 025616 001176          MOV      #4$, $ESCAPE  :ELSE SETUP ESCAPE
5473 025612 011646          MOV      (SP),-(SP)    :COPY RET ADDR
5474 025614 000207          RTS      PC           :& RETURN TO MAINLINE ERROR
5475
5476 025616 032777 001000 153314 4$:      BIT      #SW9,@SWR     :RETURN HERE FROM MAINLINE ERROR
5477 025624 001070          BNE      20$          :& BR IF LOOP ON ERROR
5478 025626 062716 000002 5$:      ADD      #2,(SP)      :BUMP RET ADDR TO NEXT ERROR
5479
5480 025632 023737 005414 005434          CMP      H.A1,E.A1      :TEST MSG A1
5481 025640 001411          BEQ      8$            :BR IF OK
5482 025642 012737 025654 001176          MOV      #7$, $ESCAPE
5483 025650 011646          MOV      (SP),-(SP)
5484 025652 000207          RTS      PC
5485
5486 025654 032777 001000 153256 7$:      BIT      #SW9,@SWR
5487 025662 001051          BNE      20$
5488 025664 062716 000002 8$:      ADD      #2,(SP)
5489
5490 025670 023737 005416 005436          CMP      H.B1,E.B1      :TEST MSG B1
5491 025676 001411          BEQ      11$           :BR IF OK
5492 025700 012737 025712 001176          MOV      #10$, $ESCAPE
5493 025706 011646          MOV      (SP),-(SP)
5494 025710 000207          RTS      PC
5495
5496 025712 032777 001000 153220 10$:     BIT      #SW9,@SWR
5497 025720 001032          BNE      20$
5498 025722 062716 000002 11$:     ADD      #2,(SP)
5499
5500 025726 032737 000001 001504 12$:     BIT      #T.A2,CHKFLG   :TEST MSG A2?
5501 025734 001402          BEQ      13$           :BR IF NO
5502 025736 004737 026722          JSR      PC,RCYLD      :PUT INFO CYLDIF, DO NOT CHECK
5503 025742 032737 000002 001504 13$:     BIT      #T.B2,CHKFLG   :TEST MSG B2?
5504 025750 001402          BEQ      14$           :BR IF NO
5505 025752 004737 026774          JSR      PC,RCYLA      :PUT INFO IN CYLADD, DO NOT CHECK
5506
5507 025756 032737 000004 001504 14$:     BIT      #T.B3,CHKFLG   :TEST MSG B3?
5508 025764 001404          BEQ      15$
5509 025766 004737 027032          JSR      PC,RSEC      :PUT INFO IN SECTOR, DO NOT CHECK
5510 025772 004737 027070          JSR      PC,RHEAD     :PUT INFO IN HEADA, DO NOT CHECK
5511
5512 025776 013737 001172 001176 15$:     MOV      $TMP5,$ESCAPE :RESTORE ESCAPE
5513 026004 000207          RTS      PC
5514
5515 026006 012706 001100          MOV      #STACK,SP     :RESET STACK PTR
5516 026012 013737 001172 001176 20$:     MOV      $TMP5,$ESCAPE :RESTORE ESCAPE
5517 026020 000177 153064          JMP      @SLPERR
5518
5519          :THIS ROUTINE CHECKS FOR CERTAIN ERROR CONDITIONS ONLY
5520          :I.E.: IF NED, CTO OR MDS SET MESSAGE A & B ARE INVALID
5521          :
5522 026024 005737 001502          CKCERR: TST      BYPCERR
  
```

```

5523 026030 001025          BNE      4$
5524 026032 032737 100000 005340  BIT      #CERR,HCS1
5525 026040 001001          BNE      1$          ;BR IF CERR
5526 026042 000207          RTS      PC
5527
5528 026044 032737 004000 005340 1$:  BIT      #CTO,HCS1
5529 026052 001402          BEQ      2$          ;BR IF NOT CTO
5530 026054 104211          ERROR    211        ;CTO ERROR, MSG A & B INVALID
5531 026056 000207          RTS      PC
5532
5533 026060 032737 010000 005342 2$:  BIT      #NED,HCS2
5534 026066 001401          BEQ      3$          ;BR IF NOT NED
5535 026070 104212          ERROR    212        ;NED ERROR, MSG A & B INVALID
5536
5537 026072 032737 001000 005342 3$:  BIT      #MDS,HCS2
5538 026100 001401          BEQ      4$
5539 026102 104213          ERROR    213        ;MDS ERROR, MSG A & B INVALID
5540
5541 026104 000207          4$:  RTS      PC
5542
5543
5544      ; THIS ROUTINE DOES THE SELECT DRIVE COMMAND TO GET STATUS
5545      ; IT THEN WAITS FOR CONTROLLER READY
5546
5547      ; IF RDY NOT RECEIVED BY THE TIMEOUT, AN ERROR IS FLAGGED
5548
5549
5550 026106 013746 005376          GSTAT:  MOV      TEMP1,-(SP)      ;SAVE TEMP1
5551 026112 013765 001222 000010  MOV      $UNIT,RKCS2(R5)      ;CURRENT DRIVE #
5552 026120 012737 000001 005340  MOV      #SELDRV,HCS1
5553 026126 004737 024446          JSR      PC,DOCMD            ;DO SELDRV (STATUS) CMD & GET CONTR RDY
5554 026132 104117          ERROR    117              ;RDY NOT SET BY END OF SELECT DRIVE CMD
5555 026134 012637 005376          MOV      (SP)+,TEMP1        ;RESTOR TEMP1.
5556 026140 000207          RTS      PC
5557
5558      ; THIS ROUTINE GETS STATUS OF ALL DRIVE REGISTERS (MSG A0-A3, B0-B3)
5559      ; & ALL CONTROLLER REGISTERS.
5560
5561 026142 013746 005376          GSTAT1: MOV      TEMP1,-(SP)      ;SAVE TEMP1
5562 026146 004737 024676          JSR      PC,HOLD            ;GET ALL CONTR REG
5563 026152 012765 100000 000000  MOV      #CCLR,RKCS1(R5)      ;CLEAR CONTR
5564 026160 013765 001222 000010  MOV      $UNIT,RKCS2(R5)      ;CURRENT DRIVE #
5565 026166 012765 000003 000026  MOV      #3,RKMR1(R5)        ;SELECT WORD 3
5566 026174 004737 026372          JSR      PC,GSTAT2
5567 026200 104117          ERROR    117              ;RDY NOT SET BY END OF SELECT DRV CMD
5568 026202 013737 005366 005424  MOV      HMR2,H.A3          ;STORE MSG A3
5569 026210 013737 005370 005426  MOV      HMR3,H.B3          ;STORE MSG B3
5570
5571 026216 012765 100000 000000  MOV      #CCLR,RKCS1(R5)
5572 026224 013765 001222 000010  MOV      $UNIT,RKCS2(R5)
5573 026232 012765 000002 000026  MOV      #2,RKMR1(R5)        ;SELECT WORD 2
5574 026240 004737 026372          JSR      PC,GSTAT2
5575 026244 104117          ERROR    117              ;RDY NOT SET BY END OF SELECT DRV CMD
5576 026246 013737 005366 005420  MOV      HMR2,H.A2          ;STORE MSG A2
5577 026254 013737 005370 005422  MOV      HMR3,H.B2          ;STORE MSG B2
5578

```

```

5579 026262 012765 100000 000000      MOV      #CCLR,RKCS1(R5)
5580 026270 013765 001222 000010      MOV      $UNIT,RKCS2(R5)
5581 026276 012765 000001 000026      MOV      #1,RKMR1(R5)      ;SELECT WORD 1
5582 026304 004737 026372      JSR      PC,GSTAT2
5583 026310 104117      ERROR    117              ;RDY NOT SET BY END OF SELECT DRV CMD
5584 026312 013737 005366 005414      MOV      HMR2,H.A1        ;STORE MSG A1
5585 026320 013737 005370 005416      MOV      HMR3,H.B1        ;STORE MSG B1
5586
5587 026326 012765 100000 000000      MOV      #CCLR,RKCS1(R5)
5588 026334 013765 001222 000010      MOV      $UNIT,RKCS2(R5)
5589 026342 004737 026372      JSR      PC,GSTAT2
5590 026346 104117      ERROR    117              ;RDY NOT SET BY END OF SEL DRV CMD
5591 026350 013737 005366 005410      MOV      HMR2,H.A0        ;STORE MSG A0
5592 026356 013737 005370 005412      MOV      HMR3,H.B0        ;STORE MSG B0
5593
5594 026364 012637 005376      MOV      (SP)+,TEMP1      ;RESTORE TEMP1
5595 026370 000207      RTS      PC
5596
5597
5598 026372 012737 000001 005340  GSTAT2: MOV      #SELDIV,HCS1
5599 026400 053737 001170 005340      BIS      $TMP4,HCS1      ;ADD CDT IF RK07
5600 026406 013765 005340 000000      MOV      HCS1,RKCS1(R5)  ;GET STATUS
5601 026414 013737 001424 005376      MOV      T10,TEMP1
5602 026422 004737 024610      JSR      PC,FRDY1        ;FIND CONTR RDY & STORE DRIVE REGS ONLY
5603 026426 000207      RTS      PC              ;RET HERE IF NOT RDY
5604 026430 062716 000002      ADD      #2,(SP)         ;RET HERE IF OK
5605 026434 000207      RTS      PC
5606
5607      ; THIS ROUTINE DOES A SUBSYSTEM CLEAR & WAITS FOR CONTROLLER READY
5608      ; IF RDY IS NOT RECEIVED BY THE END OF THE TIMEOUT, AN ERROR IS FLAGGED.
5609      ; THE ROUTINE THEN GETS CURRENT STATUS & CHECKS FOR CONTROLLER ERROR (CERR)
5610      ; RETURN IF CERR SET
5611      ; RETURN +2 IF CERR CLEAR
5612
5613 026436 012765 000040 000010  SUBCLR: MOV      #SCLR,RKCS2(R5) ;SUBSYS CLEAR
5614 026444 013737 001424 005376      MOV      T10,TEMP1
5615 026452 004737 024542      JSR      PC,FRDY        ;FIND RDY
5616 026456 104120      ERROR    120              ;RDY NOT SET BY END OF SCLR
5617 026460 013765 001222 000010      MOV      $UNIT,RKCS2(R5) ;CURRENT DRIVE #
5618 026466 005065 000026      CLR      RKMR1(R5)       ;SELECT WORD 0
5619 026472 004737 026106      JSR      PC,GSTAT        ;GET STATUS
5620 026476 032737 100000 005340      BIT      #CERR,HCS1     ;CHECK FOR CONT ERROR
5621 026504 001401      BEQ     1$
5622 026506 000207      RTS      PC
5623 026510 062716 000002  1$:  ADD      #2,(SP)         ;SKIP OVER ERROR
5624 026514 000207      RTS      PC
5625
5626
5627      ; READ THE SECTOR COUNT IN RKMR3, RIGHT JUSTIFY IT & STORE IT IN 'SECTOR'
5628
5629 026516 012765 000003 000026  RDSEC: MOV      #3,RKMR1(R5)   ;WORD 3
5630 026524 004737 026106      JSR      PC,GSTAT
5631 026530 013737 005370 001422      MOV      HMR3,SECTOR
5632 026536 042737 177017 001422      BIC      #<M.SECT>,SECTOR
5633 026544 006237 001422      ASR      SECTOR          ;RIGHT JUSTIFY
5634 026550 006237 001422      ASR      SECTOR          ;SECTOR

```

```
5635 026554 006237 001422          ASR      SECTOR      ·INFO
5636 026560 006237 001422          ASR      SECTOR
5637 026564 000207                    RTS      PC
5638
5639          ;READ THE CYL DIFF/OFFSET IN RKMR2, RIGHT JUSTIFY IT & STORE IT IN 'CYLDIF'
5640
5641 026566 012765 000002 000026 RDCYLD: MOV      #2,RKMR1(R5) ;WORD 2
5642 026574 004737 026106          JSR      PC,GSTAT
5643 026600 013737 005366 001376          MOV      HMR2,CYLDIF
5644
5645 026606 043737 012660 001376          BIC      MASK1,CYLDIF
5646 026614 006237 001376          ASR      CYLDIF          ;RIGHT JUSTIFY
5647 026620 006237 001376          ASR      CYLDIF          ;CYL DIFF/OFFSET
5648 026624 006237 001376          ASR      CYLDIF          ;INFO
5649 026630 006237 001376          ASR      CYLDIF
5650 026634 023737 001376 012656          CMP      CYLDIF,MASK    ;CHK TO SEE IF RET IN COMPL. FORM
5651 026642 001002                    BNE      1$              ;BR IF NOT
5652 026644 005037 001376          CLR      CYLDIF          ;CLR IF YES
5653 026650 000207          1$:      RTS      PC
5654
5655          ;READ THE CYL ADDR IN RKMR3, RIGHT JUSTIFY IT & STORE IT IN 'CYLADD'
5656
5657 026652 012765 000002 000026 RDCYLA: MOV      #2,RKMR1(R5) ;WORD 2
5658 026660 004737 026106          JSR      PC,GSTAT
5659 026664 013737 005370 001400          MOV      HMR3,CYLADD
5660 026672 043737 012660 001400          BIC      MASK1,CYLADD
5661 026700 006237 001400          ASR      CYLADD          ;RIGHT JUSTIFY
5662 026704 006237 001400          ASR      CYLADD          ;CYL ADDR
5663 026710 006237 001400          ASR      CYLADD          ;INFO
5664 026714 006237 001400          ASR      CYLADD
5665 026720 000207          RTS      PC
5666
5667          ;READ THE CYL DIFF/OFFSET IN H.A2, RIGHT JUSTIFY IT & STORE IT IN 'CYLDIF'
5668
5669 026722 013737 005420 001376 RCYLD:  MOV      H.A2,CYLDIF
5670 026730 043737 012660 001376          BIC      MASK1,CYLDIF ;CLEAR UNWANTED INFO
5671 026736 006237 001376          ASR      CYLDIF ;RIGHT JUSTIFY
5672 026742 006237 001376          ASR      CYLDIF
5673 026746 006237 001376          ASR      CYLDIF
5674 026752 006237 001376          ASR      CYLDIF
5675 026756 023737 001376 012656          CMP      CYLDIF,MASK    ;CHK TO SEE IF RET IN COMPL. FORM
5676 026764 001002                    BNE      1$              ;BR IF NO
5677 026766 005037 001376          CLR      CYLDIF          ;ELSE CLEAR
5678 026772 000207          1$:      RTS      PC
5679
5680          ;READ THE CYL ADDR IN H.B2, RIGHT JUSTIFY IT & STORE IT IN 'CYLADD'
5681
5682 026774 013737 005422 001400 RCYLA:  MOV      H.B2,CYLADD
5683 027002 043737 012660 001400          BIC      MASK1,CYLADD ;CLEAR UNWANTED INFO
5684 027010 006237 001400          ASR      CYLADD          ;RIGHT JUSTIFY
5685 027014 006237 001400          ASR      CYLADD
5686 027020 006237 001400          ASR      CYLADD
5687 027024 006237 001400          ASR      CYLADD
5688 027030 000207          RTS      PC
5689
5690          ;READ THE SECTOR COUNT IN H.B3, RIGHT JUSTIFY IT & STORE IT IN 'SECTOR'
```

```

5691
5692 027032 013737 005426 001422 RSEC:  MOV    H.B3,SECTOR
5693 027040 042737 177017 001422      BIC    #^C<M.SECT>,SECTOR ;CLEAR UNWANTED INFO
5694 027046 006237 001422      ASR    SECTOR             ;RIGHT JUSTIFY
5695 027052 006237 001422      ASR    SECTOR
5696 027056 006237 001422      ASR    SECTOR
5697 027062 006237 001422      ASR    SECTOR
5698 027066 000207                RTS    PC
5699
5700      ;:READ THE HEAD ADDR IN H.B3, RIGHT IT & STORE IT IN 'HEAD'
5701
5702 027070 013737 005426 001444 RHEAD: MOV    H.B3,HEAD
5703 027076 042737 170777 001444      BIC    #^C<M.HEAD>,HEAD  ;CLEAR UNWANTED INFO
5704 027104 006237 001444      ASR    HEAD              ;RIGHT JUSTIFY IT
5705 027110 000337 001444      SWAB  HEAD
5706 027114 000207                RTS    PC
5707
5708      ;:FIND SECTOR 17
5709      ;:RETURN IF NOT FOUND
5710      ;:RETURN +4 IF FOUND
5711
5712 027116 013737 001430 005376 FSEC17: MOV   T5000,TEMP1 ;SETUP TIMEOUT
5713 027124 004737 026516          1$:  JSR   PC,RDSEC ;READ SECTOR
5714 027130 023727 001422 000021      CMP   SECTOR,#17. ;TEST FOR SECTOR 17
5715 027136 001014          BNE   2$ ;BR IF NOT 17
5716
5717          JSR   PC,RDSEC
5718 027144 023727 001422 000021      CMP   SECTOR,#17.
5719 027152 001412          BEQ   3$ ;BR IF READ SAME TWICE
5720 027154 004737 026516          JSR   PC,RDSEC ;ELSE TRY 1 MORE TIME
5721 027160 023727 001422 000021      CMP   SECTOR,#17.
5722 027166 001404          BEQ   3$ ;BR IF 17
5723
5724 027170 005337 005376          2$:  DEC   TEMP1
5725 027174 001353          BNE   1$ ;TRY AGAIN
5726 027176 000207                RTS    PC
5727
5728 027200 062716 000004          3$:  ADD   #4,(SP) ;SKIP OVER ERROR
5729 027204 000207                RTS    PC
5730
5731      ;:FIND DESIRED CYL DIFF
5732      ;:RETURN IF NOT FOUND
5733      ;:RETURN+6 IF FOUND
5734
5735
5736 027206 013737 001430 005376 FCYL:  MOV   T5000,TEMP1 ;SETUP TIMEOUT
5737 027214 004737 026566          1$:  JSR   PC,RDCYLD
5738 027220 023737 001376 005400      CMP   CYLDIF,TEMP2 ;TEST FOR CYL DIFF
5739 027226 001014          BNE   2$ ;BR IF NOT FOUND
5740
5741          JSR   PC,RDCYLD
5742 027234 023737 001376 005400      CMP   CYLDIF,TEMP2
5743 027242 001412          BEQ   3$ ;BR IF READ SAME TWICE
5744 027244 004737 026566          JSR   PC,RDCYLD ;ELSE TRY 1 MORE TIME
5745 027250 023737 001376 005400      CMP   CYLDIF,TEMP2
5746 027256 001404          BEQ   3$
  
```

```

5747
5748 027260 005337 005376      2$:  DEC    TEMP1
5749 027264 001353              BNE    1$      ;TRY AGAIN
5750 027266 000207              RTS    PC
5751
5752 027270 062716 000006      3$:  ADD    #6,(SP)  ;SKIP OVER ERROR
5753 027274 000207              RTS    PC
5754
5755
5756      ;ROUTINE TO FIND HEADS HOME IN RKMR2 WORD 1 BEFORE SPECIFIED DELAY
5757      ;ENTER WITH TIME IN SECONDS IN TEMP2
5758      ;RETURN IF NOT FOUND
5759      ;RETURN+2 IF FOUND - SKIP OVER ERROR
5760
5761 027276 012737 177777 005376  FHDHM: MOV    #-1,TEMP1      ;ALL 1'S
5762 027304 012765 000001 000026      MOV    #1,RKMR1(R5)  ;WORD 1
5763 027312 004737 026106      1$:  JSR    PC,GSTAT
5764 027316 032737 000040 005366      BIT    #D.HDHM,HMR2
5765 027324 001007              BNE    2$
5766 027326 005337 005376      DEC    TEMP1
5767 027332 001367              BNE    1$
5768
5769 027334 005337 005400      DEC    TEMP2
5770 027340 001356              BNE    FHDHM
5771 027342 000207              RTS    PC
5772 027344 062716 000002      2$:  ADD    #2,(SP)  ;SKIP OVER ERROR
5773 027350 000207              RTS    PC
5774
5775      ;ROUTINE TO FIND LOAD HEADS IN RKMR2 WORD 1 BEFORE SMS
5776      ;RETURN IF NOT FOUND
5777      ;RETURN+2 IF FOUND: SKIP OVER ERROR
5778
5779 027352 012737 000372 005376  FLOAD: MOV    #250.,TEMP1
5780 027360 012765 000001 000026      MOV    #1,RKMR1(R5)  ;WORD 1
5781 027366 004737 026106      1$:  JSR    PC,GSTAT
5782 027372 032737 010000 005366      BIT    #D.LOAD,HMR2
5783 027400 001004              BNE    2$
5784 027402 005337 005376      DEC    TEMP1
5785 027406 001367              BNE    1$
5786 027410 000207              RTS    PC
5787 027412 062716 000002      2$:  ADD    #2,(SP)  ;SKIP OVER ERROR
5788 027416 000207              RTS    PC
5789
5790      ;FILL HEADER TABLE WITH 66 WORDS OF VALID HEADERS
5791      ;ENTER WITH CYL # IN 'CALADD'
5792      ;ENTER WITH HEAD # IN 'HEAD'
5793      ;ENTER WITH FORMAT IN 'FORMAT'
5794
5795 027420 010046      FHDTAB: MOV    R0,-(SP)      ;SAV R0
5796 027422 010146      MOV    R1,-(SP)      ;SAV R1
5797 027424 012700 001506      MOV    #HDTAB,R0     ;HEADER WORD TABLE ADDR
5798 027430 005001              CLR    R1            ;SECTOR COUNTER
5799 027432 013737 001442 001446      MOV    HEAD,HD1
5800 027440 006337 001446      ASL   HD1
5801 027444 006337 001446      ASL   HD1
5802 027450 006337 001446      ASL   HD1
  
```

```

5803 027454 006337 001446      ASL    HD1
5804 027460 006337 001446      ASL    HD1          ;SETUP HEAD # FOR WORD 2 OF HEADER
5805 027464 013737 001450 001452  MOV    FORMAT,FMT1
5806 027472 000337 001452      SWAB   FMT1
5807 027476 006337 001452      ASL    FMT1          ;SETUP FORMAT FOR WORD 2 OF HEADER
5808
5809 027502 013720 001402 1$:    MOV    CALADD,(R0)+ ;HEADER WORD 1-CYL ADDR
5810 027506 010110      MOV    R1,(R0)      ;HEADER WORD 2-SECTOR NO
5811 027510 053710 001446      BIS    HD1,(R0)     ;
5812 027514 053710 001452      BIS    FMT1,(R0)   ;
5813 027520 004737 027600      JSR    PC,SECFLG   ;GET SECTOR FLAGS
5814
5815 027524 013737 001402 005376  MOV    CALADD,TEMP1
5816 027532 011037 005400      MOV    (R0),TEMP2
5817 027536 043737 001402 005400  BIC    CALADD,TEMP2
5818 027544 042037 005376      BIC    (R0)+,TEMP1
5819 027550 053737 005376 005400  BIS    TEMP1,TEMP2
5820 027556 013720 005400      MOV    TEMP2,(R0)+ ;HEADER WORD 3-HEADER CHECK
5821
5822 027562 005201      INC    R1          ;SECTOR CTR
5823 027564 020127 000026      CMP    R1,#22.    ;ALL 22 SECTORS DONE? (66 WORDS)
5824 027570 001344      BNE    1$         ;BR IF NO
5825
5826 027572 012601      MOV    (SP)+,R1   ;RESTOR R1
5827 027574 012600      MOV    (SP)+,R0   ;RESTOR R0
5828 027576 00C207      RTS    PC
5829
5830      ; THIS ROUTINE GETS INFORMATION FROM THE BAD SECTOR TABLE FILLED BY A PREVIOUS
5831      ; TEST & SETS BITS 14 & 15 APPROPRIATLY.
5832
5833 027600 010246  SECFLG: MOV    R2,-(SP)   ;SAVE R2
5834 027602 005737 001450      TST    FORMAT
5835 027606 001016      BNE    1$         ;BR IF 20 SECTOR FORMAT
5836 027610 012702 002332      MOV    #BSE22H+8.,R2
5837 027614 004737 027650      JSR    PC,FLGTST  ;GET HARDWARE DETECTED FLAG
5838 027620 052710 100000      BIS    #BIT15,(R0) ;RETURN HERE IF GOOD SECTOR
5839
5840 027624 012702 003332      MOV    #BSE22S+8.,R2 ;ELSE RETURN HERE
5841 027630 004737 027650      JSR    PC,FLGTST  ;GET SOFTWARE DETECTED FLAG
5842 027634 052710 040000      BIS    #BIT14,(R0) ;RETURN HERE IF GOOD SECTOR
5843
5844 027640 012602      MOV    (SP)+,R2   ;ELSE RETURN HERE
5845 027642 000207      RTS    PC
5846
5847 027644 012602 1$:    MOV    (SP)+,R2   ;RESTORE R2
5848 027646 000207      RTS    PC
5849
5850      ;
5851      ; THIS ROUTINE DOES THE ACTUAL SCANNING OF THE BAD SECTOR TABLES.
5852      ; ENTER WITH THE ADDRESS OF TABLE (BSE22H, BSE22S, ETC.) IN TEMP1
5853      ; RETURN IF NO COMPARE
5854      ; RETURN+4 .F COMPARE
5855      ;
5856 027650 010346  FLGTST: MOV    R3,-(SP)   ;SAVE R3
5857
5858 027652 021227 177777 1$:    CMP    (R2),#-1   ;SEE IF ALL 1'S
    
```



```

5859 027656 001002      BNE      2$      ;BR IF NO
5860 027660 012603      MOV      (SP)+,R3 ;RESTORE R3
5861 027662 000207      RTS      PC
5862
5863 027664 022237 001402  2$:      CMP      (R2)+,CALADD ;SEE IF=CYL # & ADR PTR TO TRK/SECTOR WORD
5864 027670 001403      BEQ      3$
5865 027672 062702 000002  ADD      #2,R2      ;GO TO NEXT CYL WORD IN TABLE
5866 027676 000765      BR       1$
5867
5868 027700 013703 001442  3$:      MOV      HEAD,R3      ;GET HEAD # FROM FHDTAB ROUTINE
5869 027704 000303      SWAB    R3
5870 027706 050103      BIS     R1,R3      ;ADD SECTOR # FROM FHDTAB ROUTINE
5871 027710 022203      CMP     (R2)+,R3    ;SEE IF SECTOR/HEAD COMPARE
5872                                ;& INCR PTR TO NEXT CYL WORD
5873 027712 001401      BEQ     4$      ;BR IF COMPARE
5874 027714 000756      BR     1$      ;ELSE TRY NEXT CYL
5875
5876 027716 012603 000004  4$:      MOV      (SP)+,R3 ;RESTORE R3
5877 027720 062716      ADD     #4,(SP)    ;INCREMENT RET ADDR
5878 027724 000207      RTS     PC
5879
5880                                ;
5881                                ; THIS ROUTINE SORTS THE RHTAB TABLE FROM WHATEVER SECTOR IT BEGINS
5882                                ; WITH AND RE-WRITES THE INFO IN SRTTAB TABLE TO BEGIN WITH SECTOR 0
5883                                ;
5884                                ; SORT:
5884 027726 010046      MOV     R0,-(SP)    ;SAVE R0
5885 027730 010146      MOV     R1,-(SP)    ;SAVE R1
5886 027732 004737 026516  JSR     PC,RDSEC
5887 027736 062737 000001 001422  ADD     #1,SECTOR
5888 027744 004737 030034  JSR     PC,MULT6      ;MULT SECTOR BY 6
5889
5890 027750 012700 000204  MOV     #132,R0
5891 027754 163700 001422  SUB     SECTOR,R0    ;R0-SECTOR TO R0 = INDEX
5892 027760 010037 001422  MOV     R0,SECTOR
5893 027764 062737 001712 001422  ADD     #RHTAB,SECTOR ;SAVE INDEX
5894
5895 027772 062700 001712  ADD     #RHTAB,R0    ;INDEX TO BOT HALF OF RHTAB
5896 027776 012701 002116  MOV     #SRTTAB,R1   ;INDEX TO TOP HALF OF SRTTAB
5897
5898
5899 030002 012021 002116  1$:      MOV     (R0)+,(R1)+ ;PUT BOTTOM OF RHTAB TO TOP OF SRTTAB
5900 030004 020027      CMP     R0,#RHTAB+132.
5901 030010 001374      BNE     1$
5902
5903 030012 012700 001712  2$:      MOV     #RHTAB,R0    ;PUT TOP OF RHTAB TO BOT OF SRTTAB
5904 030016 012021      MOV     (R0)+,(R1)+
5905 030020 020037 001422  CMP     R0,SECTOR
5906 030024 001374      BNE     2$
5907
5908 030026 012601      MOV     (SP)+,R1    ;RESTOR R1
5909 030030 012600      MOV     (SP)+,R0    ;RESTOR R0
5910 030032 000207      RTS     PC
5911
5912                                ;
5913                                ; MULT BY 6. ENTER WITH DESIRED # IN 'SECTOR'
5914                                ;

```

```

5915 030034 006337 001422 MULT6: ASL SECTOR ;2 X SECTOR
5916 030040 013746 001422 MOV SECTOR,-(SP)
5917 030044 006337 001422 ASL SECTOR ;4 X SECTOR
5918 030050 062637 001422 ADD (SP)+,SECTOR ;(4 X S)+(2 X S) = 6 X SECTOR
5919 030054 000207 RTS PC
  
```

```

5920
5921
5922 ;THIS ROUTINE IS ENTERED ONLY IF THERE IS A BSE ERROR AFTER A WRITE DATA
5923 ;CMD. IT VERIFIES THAT THE BAD SECTOR IS LISTED IN THE BSE INFORMATION
5924 ;CYLINDER AT CYL 410, TRACK 2.
5925
5926 ;RETURN IF SECTOR NOT LISTED IN BSE TABLE, ERROR CONDITION.
5927 ;RETURN+2 IF LISTED, SKIP OVER ERROR
  
```

```

5928
5929 030056 010446 TRUERR: MOV R4,-(SP) ;SAVE R4
5930
5931 030060 032737 010000 005340 BIT #CFMT,HCS1 ;CHECK FORMAT
5932 030066 091014 BNE 3$ ;BR FOR 20 SECTOR FORMAT
5933 ;NOTE, 20 SECTOR FORMAT NOT
5934 ;DONE IN THIS PROGRAM
  
```

```

5935
5936 030070 012704 002332 MOV #BSE22H+8.,R4
5937 030074 004737 030130 JSR PC,TERR1 ;SEE IF ON HARDWARE DETECTED TABLE
5938 030100 000407 BR 3$ ;RETURN HERE IF YES
  
```

```

5939
5940 030102 012704 003332 MOV #BSE22S+8.,R4 ;ELSE RETURN HERE
5941 030106 004737 030130 JSR PC,TERR1 ;& SEE IF ON SOFTWARE DETECTED TABLE
5942 030112 000402 BR 3$ ;RETURN HERE IF YES
  
```

```

5943
5944 030114 012604 1$: MOV (SP)+,R4 ;RESTORE R4
5945 030116 000207 RTS PC ;RETURN WITHOUT JUMPING OVER ERROR
  
```

```

5946
5947
5948 030120 012604 3$: MOV (SP)+,R4 ;RESTORE R4
5949 030122 062716 000002 ADD #2,(SP) ;SKIP OVER ERROR ON RETURN
5950 030126 000207 RTS PC
  
```

```

5951
5952
5953 ;THIS ROUTINE DOES THE ACTUAL COMPARING OF CYLINDER, HEAD & TRACK AGAINST
5954 ;THE BSE TABLE FOR THE ABOVE SUBROUTINE.
5955 ;RETURN IF FOUND ON TABLE
5956 ;RETURN+2 IF NOT FOUND
  
```

```

5957
5958 030130 021427 177777 TERR1: CMP (R4),#-1 ;SEE IF ALL 1'S
5959 030134 001405 BEQ 1$ ;BR IF YES, NOT ON TABLE
5960 030136 022437 005360 CMP (R4)+,HDC ;SEE IF CYL MATCH
5961 030142 001405 BEQ 2$ ;BR IF YES
5962 030144 005724 TST (R4)+ ;ELSE ADV TO NEXT CYL WORD
5963 030146 000770 BR TERR1 ;& TRY AGAIN.
  
```

```

5964
5965 030150 062716 000002 1$: ADD #2,(SP)
5966 030154 000207 RTS PC
  
```

```

5967
5968 030156 022437 005350 2$: CMP (R4)+,HDA ;SEE IF SECTOR & TRACK MATCH
5969 030162 001401 BEQ 3$ ;BR IF YES
5970 030164 000761 BR TERR1 ;OR TRY AGAIN
  
```

```

5971
5972 030166 000207          3$:   RTS   PC
5973
5974
5975
5976
5977
5978
5979          :
5980          :ROUTINE TO TURN L OR P CLOCK INTERRUPT ON
5981          :
5981 030170 005037 001412    CLKON: CLR   TIMUP
5982 030174 005737 005504      TST   PCLKF
5983 030200 001004          BNE   1$           ;BRANCH IF P-CLOCK PRESENT
5984 030202 012777 000100 151136  MOV   #100,@LKS   ;L-CLOCK, ENABLE INT
5985 030210 000207          RTS   PC
5986 030212 012777 177777 151122 1$:   MOV   #-1,@PKSB  ;P-CLOCK, ALL 1'S
5987 030220 012777 000135 151112  MOV   #135,@PKS  ;ENABLE INT, CT UP, REP INT
5988 030226 000207          RTS   PC           ;LINE FREQ & RUN
5989
5990          :
5991          :KW11-L & KW11-P INTERRUPT HANDLER
5992          :
5992 030230 005037 001412    CLOCK: CLR   TIMUP
5993 030234 005337 001406      DEC   COUNT
5994 030240 001010          BNE   1$
5995 030242 013737 001404 001406  MOV   HZ,COUNT
5996 030250 005337 001410      DEC   SEC
5997 030254 001002          BNE   1$
5998 030256 005237 001412      INC   TIMUP       ;SORRY, TIME IS UP
5999 030262 000002          1$:   RTI
6000
6001          :
6002          :ROUTINE TO TURN L OR P CLOCK INTERRUPT OFF
6003          :
6003 030264 005737 005504    CLKOF: TST   PCLKF
6004 030270 001003          BNE   1$           ;BRACH IF P-CLOCK PRESENT
6005 030272 005077 151050      CLR   @LKS       ;L-CLOCK, CLEAR INTERRUPT
6006 030276 000207          RTS   PC
6007 030300 005077 151034      1$:   CLR   @PKS       ;P-CLOCK, CLEAR INTERRUPT
    
```

```

6008 030304 000207          RTS    PC
6009
6010
6011      ; THIS ROUTINE GENERATES PARITY FOR THE EXPECTED MESSAGES
6012      ; ENTER WITH THE EXPECTED WORD IN TEMP1
6013      ; TEMP1 IS ROTATED LEFT 17 TIMES. EACH TIME THE CARRY BIT IS SET,
6014      ; R1 IS INCREMENTED. AT THE END OF 17 ROTATES ( TEMP1 BACK TO ORIG),
6015      ; R1 BIT 0 IS EXAMINED. IF IT IS SET, INDICATING AN ODD # OF 1'S,
6016      ; THE PARITY BIT IS NOT SET IN B .
6017      ; IF IT IS NOT SET, INDICATING AN EVEN # OF 1'S ,THE PARITY BIT IS
6018      ; SET IN TEMP1
6019
6020 030306 010046          SBPAR:  MOV    R0,-(SP)      ;SAVE R0
6021 030310 010146          MOV    R1,-(SP)      ;SAVE R1
6022 030312 012700 000021  MOV    #17,R0      ;SHIFT COUNTER
6023 030316 005001          CLR    R1            ;COUNT # OF 1'S IN TEMP1
6024 030320 000241          CLC                    ;CLEAR CARRY
6025
6026 030322 006137 005376  1$:    ROL    TEMP1
6027 030326 103001          BCC    2$            ;BR IF CARRY CLEAR
6028 030330 005201          INC    R1            ;COUNT # OF 1'S
6029 030332 005300          2$:    DEC    R0            ;SHIFT COUNTER
6030 030334 001372          BNE    1$
6031
6032 030336 032701 000001  BIT    #BIT0,R1
6033 030342 001003          BNE    3$            ;BR IF ODD # IN R0
3$:
  
```

```

6034 030344 052737 100000 005376      BIS      #M.PAR,TEMP1      ;SET PARITY BIT
6035 030352 012601      3$:     MOV      (SP)+,R1      ;RESTORE R1
6036 030354 012600      MOV      (SP)+,R0      ;RESTORE R0
6037 030356 000207      RTS      PC
6038
6039
6040      ;ROUTINE TO ENABLE LOOPING ON INTERMITTANT ERRORS
6041      ;WHEN $LPERR SET BY OTHER THAN SCOPE ROUTINE
6042      ; IE: MY LOOP MACRO
6043
6044 030360 032777 001000 150552  SCOP1$: BIT      #SW9,@SWR      ;LOOP ON ERROR?
6045 030366 001406      BEQ      1$              ;BR IF NO
6046 030370 105737 001103      TSTB     $ERFLG          ;HAD ERROR?
6047 030374 001403      BEQ      1$              ;BR IF NO
6048 030376 013716 001110      MOV      $LPERR,(SP)
6049 030402 000002      RTI
6050
6051 030404 011637 001110      1$:     MOV      (SP),$LPERR  ;SET LOOP ADDR FOR TIGHT SCOPE LOOP
6052 030410 000002      RTI
6053
6054
6055      ;ROUTINE TO INPUT A 'SPACE' OR 'CONTROL-E' FROM TTY
6056      ;RETURN IF CONTROL-E
6057      ;RETURN +4 IF SPACE
6058
6059
6060
6061 030412 005777 150530  CCSP:   TST      @STKB          ;CLEAR DONE FLAG
6062 030416 104410      RDCHR           ;READ CHAR FROM TTY
6063 030420 012600      MOV      (SP)+,R0      ;GET CHAR OFF STACK
6064 030422 020027 000040      CMP      R0,#SPBAR     ;SEE IF SPACE
6065 030426 001406      BEQ      1$              ;BR IF YES.
6066
6067      CMP      R0,#5          ;SEE IF CONTROL-E
6068 030434 001405      BEQ      2$              ;BR IF YES
6069 030436 104401 040212      TYPE     ,MSG31         ;?
6070 030442 000763      BR       CCSP           ;TRY AGAIN
6071
6072 030444 062716 000004      1$:     ADD      #4,(SP)
6073 030450 000207      2$:     RTS      PC
6074
6075
6076      ;ROUTINE TO INPUT A 'SPACE' FROM TTY
6077
6078 030452 005777 150470  GETSP:  TST      @STKB          ;CLEAR DONE FLAG
6079 030456 104410      RDCHR           ;READ CHAR OFF TTY
6080 030460 012600      MOV      (SP)+,R0      ;GET CHAR OFF STACK
6081 030462 020027 000040      CMP      R0,#SPBAR     ;SEE IF SPACE
6082 030466 001403      BEQ      1$              ;EXIT IF YES
6083 030470 104401 040212      TYPE     ,MSG31         ;?
6084 030474 000766      BR       GETSP          ;TRY AGAIN
6085 030476 000207      1$:     RTS      PC
6086
6087
6088      ;THIS ROUTINE IS ENTERED BY TYPING A CONTROL-C.
6089      ;IT IS USED TO ALLOW THE OPERATOR TO HALT THE CPU WHILE INSURING

```

```

6090 ;THAT HEADS ARE LOADED & FORMATTING IS VALID BEFORE ACTUALLY HALTING
6091 ;THE CPU.
6092 .
6093 030500 022626 STOP: CMP (SP)+,(SP)+ ;RESTORE STACK FROM INTERRUPT
6094
6095 030502 004737 026436 STOP1: JSR PC,SUBCLR
6096 030506 104024 ERROR 24 ;CERR AFTER
6097
6098 030510 005737 005322 TST UNLD ;SEE IF HEADS UNLOADED
6099 030514 001437 BEQ 3$ ;BR IF NO
6100 030516 005737 000042 TST 42 ;SEE IF MANUAL OR AUTO MODE
6101 030522 001403 BEQ 1$ ;BR IF MANUAL MODE
6102 030524 104401 044071 TYPE ,MSG74 ;PGM ABORT PENDING
6103 030530 000402 BR 2$
6104 030532 104401 044137 1$: TYPE ,MSG75 ;HALT PENDING
6105 030536 2$:
6106
6107 030536 004737 026436 JSR PC,SUBCLR
6108 030542 104024 ERROR 24 ;CERR AFTER SCLR
6109
6110 030544 032737 010000 005366 BIT #D.SPIN,HMR2 ;SEE IF SPINDLE ALREADY ON
6111 030552 001020 BNE 64$ ;BR IF YES
6112 030554 104401 040064 TYPE ,MSG29 ;PLEASE WAIT, HEADS BEING LOADED
6113
6114 030560 012737 000011 005340 MOV #SRTSPL,HCS1
6115 030566 004737 024446 JSR PC,DOCMD ;DO START SPINDLE CMD & GET CONTR RDY
6116 030572 104143 ERROR 143 ;CONTR RDY NOT SET AFTER CMD
6117
6118 030574 013737 001426 005400 MOV T100,TEMP2
6119 030602 004737 025056 JSR PC,FATT1 ;FIND ATTN
6120 030606 104144 ERROR 144 ;NO ATTN AFTER CMD
6121
6122 030610 005037 005322 CLR UNLD
6123 030614 64$:
6124
6125 030614 005737 005324 3$: TST BADHDR ;SEE IF HEADERS VALID
6126 030620 001460 BEQ 4$ ;BR IF YES
6127 030622 005237 005326 INC HPEND
6128 030626 012765 100000 000000 MOV #CCLR,RKCS1(R5)
6129 030634 013765 001222 000010 MOV $UNIT,RKCS2(R5)
6130 030642 012737 000013 005340 MOV #RECAL,HCS1
6131 030650 004737 024446 JSR PC,DOCMD ;DO RECAL CMD & GET CONTR RDY
6132 030654 104124 ERROR 124 ;RDY NOT SET AFTER RECAL CMD
6133
6134 030656 012765 000001 000026 MOV #1,RKMR1(R5) ;SELECT WORD 1
6135 030664 004737 026106 JSR PC,GSTAT
6136 030670 032737 020000 005366 BIT #D.RTZ,HMR2
6137 030676 001001 BNE 65$
6138 030700 104214 ERROR 214 ;RTZ NOT SET DURING RECAL CMD
6139 030702 013737 001424 005400 65$: MOV T10,TEMP2 ;SETUP TIMEOUT
6140 030710 004737 025056 JSR PC,FATT1 ;FIND ATTN
6141 030714 104055 ERROR 55 ;NO ATTN AFTER RECAL CMD
6142
6143 030716 012765 100000 000000 MOV #CCLR,RKCS1(R5)
6144 030724 013765 001222 000010 MOV $UNIT,RKCS2(R5) ;DRIVE#
6145 030732 012737 000005 005340 MOV #CLEAR,HCS1

```

```

6146 030740 004737 024446      JSR    PC,DOCMD      ;DO DRIVE CLEAR CMD & GET CONTR RDY
6147 030744 104151              ERROR  151          ;NO RDY AFTER DRIVE CLEAR CMD
6148 030746 004737 025024      JSR    PC,TSTATN    ;TEST FOR ATTN
6149 030752 000401              BR     66$          ;
6150 030754 104154              ERROR  154          ;ATTN NOT CLEARED AFTER DRIVE CLEAR CMD
6151 030756 66$:
6152
6153
6154 030756 000137 031022      JMP    FORM          ;WRITE VALID FORMATS
6155
6156 030762 005737 000042      4$:   TST    42          ;SEE IF MANUAL OR AUTO MODE
6157 030766 001410              BEQ    5$            ;BR IF MANUAL MODE
6158 030770 104401 044174      TYPE  ,MSG76        ;PGM ABORTED
6159 030774 005037 023762      CLR    $EOPCT       ;SET UP EOP TO EXIT TO MONITOR
6160 031000 005037 001176      CLR    $ESCAPE
6161 031004 000137 023734      JMP    $EOP          ;ABORT PROGRAM
6162
6163 031010 104401 044216      5$:   TYPE  ,MSG77        ;CPU HALTED
6164 031014 000000              HALT
6165 031016 000137 010612      JMP    ST5           ;START OVER IF CONTINUE PRESSED
6166
6167 031022      FORM:
6168      .SBTTL UNEXPECTED TIMEOUT HANDLER
6169
6170
6171      ; THIS ROUTINE IS ENTERED IF THERE IS
6172      ; A. NON EXISTANT MEMORY (NO SSYN)
6173      ; B. BOUNDRY ERROR
6174      ; C. STACK OVERFLOW
6175
6176
6177 031022 011600      BADTMO: MOV    (SP),R0      ;SAVE PC WHERE TIMEOUT OCCURRED.
6178 031024 005740      TST    -(R0)          ;GET PC BEFORE UPDATE
6179 031026 032777 020000 150104  BIT    #SW13,@SWR     ;INHIBIT ERR TYPOUT?
6180 031034 001005      BNE    1$            ;YES, DON'T TYPE
6181 031036 104401 044546      TYPE  ,EM3           ;ABORT TESTS,UNEXP T.O. @ PC=
6182 031042 010046      MOV    R0,-(SP)      ;SAVE R0 FOR TYPEOUT
6183
6184 031044 104403      TYPOS
6185 031046 006          .BYTE  6            ;GO TYPE--OCTAL ASCII
6186 031047 000          .BYTE  0            ;TYPE 6 DIGIT(S)
6187 031050 032777 001000 150062 1$:  BIT    #SW9,@SWR     ;SUPPRESS LEADING ZEROS
6188 031056 001403      BEQ    2$            ;LOOP ON ERROR?
6189 031060 022626      CMP    (SP)+,(SP)+   ;NO, BRANCH
6190 031062 000177 150020      JMP    @SLPADR       ;YES, RESTORE STACK
6191
6192 031066 032777 040000 150044 2$:  BIT    #SW14,@SWR    ;GO TO STARTING ADDR OF TEST
6193 031074 001401      BEQ    3$            ;THAT GAVE BAD TIMEOUT
6194 031076 000002      RTI                   ;LOOP ON TEST?
6195
6196 031100 000000      3$:   HALT              ;NO BRANCH
6197
6198      ;UNEXPECTED TIME OUT OCCURRED
6199      ;AS INDICATED, YOU CAN LOOP ON
6200      ;ERROR, LOOP ON TEST OR INHIBIT
6201      ;ERROR TYPEOUT BY SETTING THOSE
        ;SWITCHES.

```

```

6202 031102 022626          CMP      (SP)+,(SP)+      ;RESTORE STACK
6203 031104 000137 023734  JMP      $EOP             ;ABORT TESTS
6204
6205          .SBTTL MEMORY CHECK ENABLE TRAP
6206
6207 031110 012737 031124 001176 MEMERR: MOV      #1$,$ESCAPE
6208 031116 011637 001354          MOV      (SP),TRAPPC    ;STORE PC
6209 031122 104202          ERROR 202             ;UNEXP MEM PARITY ERROR
6210
6211 031124 005037 001176 1$:    CLR      $ESCAPE
6212 031130 032777 001000 150002 BIT      #SW9,@SWR        ;CHECK IF LOOP ON ERROR
6213 031136 001001          BNE      2$             ;YES, FORCE STACK AND TRY AGAIN
6214 031140 000002          RTI              ;ELSE RETURN
6215
6216 031142 012706 001100 2$:    MOV      #STACK,SP      ;INIT STACK
6217 031146 000177 147736          JMP      @SLPERR
6218
6219          .SBTTL RK06 INTERRUPT HANDLER
6220
6221 031152 000240          INTER: NOP
6222 031154 000240          NOP
6223 031156 000240          NOP
6224 031160 011600          MOV      (SP),R0        ;SAVE PC WHERE INT OCCURRED.
6225 031162 005740          TST      -(R0)         ;GET PC BEFORE UPDATE.
6226 031164 104401 037036          TYPE    ,MSG6          ;INT AT PC=
6227 031170 010046          MOV      R0,-(SP)      ;SAVE R0 FOR TYPEOUT
6228          ;TYPE PC
6229 031172 104403          TYPOS   ;GO TYPE--OCTAL ASCII
6230 031174          006          .BYTE 6                ;TYPE 6 DIGIT(S)
6231 031175          000          .BYTE 0                ;SUPPRESS LEADING ZEROS
6232 031176 000000          HALT
6233 031200 000240          NOP
6234 031202 000240          NOP
6235 031204 000002          RTI
6236
6237          .SBTTL POWER DOWN AND UP ROUTINES
6238
6239          ;POWER DOWN ROUTINE
6240
6241 031206 012737 031220 000024 $PWRDN: MOV      #SPWRUP,PWRVEC ;SET UP VECTOR
6242 031214 000000          HALT
6243 031216 000776          BR      .-2           ;HANG UP.
6244
6245          ;POWER UP ROUTINE
6246
6247 031220 005037 031272 $PWRUP: CLR      $PWRCT    ;WAIT LOOP FOR TTY
6248 031224 005237 031272 1$:    INC      $PWRCT        ;WAIT FOR THE INCR
6249 031230 001375          BNE      1$           ;OF WORD
6250 031232 012737 031206 000024 MOV      #SPWRDN,PWRVEC ;SET POWER DOWN VECTOR
6251 031240 012737 000340 000026 MOV      #PR7,PWRVEC+2 ;PRIORITY 7
6252 031246 012737 000340 000036 MOV      #PR7,TRAPVEC+2 ;LOCKOUT ALL INTERRUPTS FOR TRAPS
6253 031254 012706 001100          MOV      #STACK,SP    ;INITIALIZE STACK
6254 031260 104401 037232          TYPE    ,MSG11       ;REPORT POWER FAIL
6255 031264 000005          RESET
6256 031266 000137 012664          JMP      PFSRT
6257

```



```

6258 031272 000000      SPWRCT: 0                ;WAIT COUNT FOR TTY
6259
6260
6261      ;DIVISION UTILITY ROUTINE
6262
6263      ;R0-R1-R2-R3=DIVIDEND
6264      ;R4-R5=DIVISOR
6265      ;R0-R1=REMAINDER AFTER DIVISION
6266      ;R2-R3=QUOTIENT AFTER DIVISION
6267      ;ENTER WITH JSR PC,M.DPID
6268
6269 031274 012746 000040  M.DPID: MOV      #40,-(SP)      ;COUNTER FOR DIVISION CYCLES
6270 031300 010446          MOV      R4,-(SP)      ;HI ORDER
6271 031302 010546          MOV      R5,-(SP)      ;LO ORDER TO THE STACK
6272 031304 005466 000002  NEG      2(SP)         ;FORM NEGATIVE
6273 031310 005416          NEG      @SP           ;VERSION OF DIVISCR
6274 031312 005666 000002  SBC      2(SP)
6275 031316 061601          ADD      @SP,R1
6276 031320 005500          ADC      R0           ;PERFORM INIT SUBT.
6277 031322 066600 000002  ADD      2(SP),R0
6278 031326 103445          BCS     M.DP50        ;IF CARRY THEN OVERFLOW HAS OCCURRED
6279 031330 005046          CLR     -(SP)        ;THIS IS A LONGER LASTING CARRY BIT
6280 031332 006103  M.DP40: ROL     R3
6281 031334 006102          ROL     R2
6282 031336 006101          ROL     R1
6283 031340 006100          ROL     R0
6284 031342 005716          TST     @SP           ;TEST CARRY INDICATOR
6285 031344 001410          BEQ     M.DP41        ;IF TO CARRY THEN ADD, ELSE SUBT.
6286 031346 005016          CLR     @SP           ;CLEAR UP FOR NEXT TIME
6287 031350 066601 000002  ADD      2(SP),R1
6288 031354 005500          ADC     R0           ;ADD -(DIVISOR)
6289 031356 005516          ADC     @SP           ;SET CARRY
6290 031360 066600 000004  ADD      4(SP),R0
6291 031364 000404          BR      M.DP42
6292
6293 031366 060501  M.DP41: ADD     R5,R1
6294 031370 005500          ADC     R0           ;ADD +(DIVISOR)
6295 031372 005516          ADC     @SP           ;SET CARRY
6296 031374 060400          ADD     R4,R0
6297 031376 005516  M.DP42: ADC     @SP
6298 031400 005716          TST     @SP           ;SET CARRY
6299 031402 001401          BEQ     .+4           ;TEST THE UPDATE INDICATOR
6300 031404 005203          INC     R3           ;IF 0, FORGET IT
6301 031406 005366 000006  DEC     6(SP)        ;NO CARRY POSSIBLE HERE
6302 031412 003347          BGT     M.DP40        ;DECREMENT CTR
6303 031414 006003          ROR     R3           ;BR IF MORE TO DO
6304 031416 103404          BCS     M.DP44
6305 031420 060501          ADD     R5,R1
6306 031422 005500          ADC     R0
6307 031424 060400          ADD     R4,R0
6308 031426 000241          CLC
6309
6310 031430 006103  M.DP44: ROL     R3
6311 031432 062706 000010  ADD     #10,SP       ;ADJUST STACK BY 4 WORDS
6312 031436 000242          CLV
6313 031440 000207          RTS     PC
  
```

6314					
6315	031442	062706	000006	M.DP50:	ADD #6.SP
6316	031446	000262			SEV
6317	031450	000207			RTS PC
6318					

```

6319 .SBTTL SCOPE HANDLER ROUTINE
6320
6321 ;*****
6322 ;*THIS ROUTINE CONTROLS THE LOOPING OF SUBTESTS. IT WILL INCREMENT
6323 ;*AND LOAD THE TEST NUMBER($STNM) INTO THE DISPLAY REG.(DISPLAY<7:0>)
6324 ;*AND LOAD THE ERROR FLAG ($ERFLG) INTO DISPLAY<15:08>
6325 ;*THE SWITCH OPTIONS PROVIDED BY THIS ROUTINE ARE:
6326 ;*SW14=1 LOOP ON TEST
6327 ;*SW11=1 INHIBIT ITERATIONS
6328 ;*SW09=1 LOOP ON ERROR
6329 ;*SW08=1 LOOP ON TEST IN SWR<7:0>
6330 ;*CALL
6331 ;* SCOPE ;:SCOPE=IOT
6332
6333 $SCOPE:
6334 031452 104407 040000 147456 1$: CKSWR ;:TEST FOR CHANGE IN SOFT-SWR
6335 031454 032777 040000 147456 1$: BIT #BIT14,@SWR ;:LOOP ON PRESENT TEST?
6336 031462 001114 040000 147456 1$: BNE $OVER ;:YES IF SW14=1
6337 ;*****START OF CODE FOR THE XOR TESTER*****
6338 031464 000416 $XTSTR: BR 6$ ;:IF RUNNING ON THE 'XOR' TESTER CHANGE
6339 ;:THIS INSTRUCTION TO A 'NOP' (NOP=240)
6340 031466 013746 000004 000004 MOV @ERRVEC,-(SP) ;:SAVE THE CONTENTS OF THE ERROR VECTOR
6341 031472 012737 031512 000004 MOV #5,@ERRVEC ;:SET FOR TIMEOUT
6342 031500 005737 177060 000004 TST @177060 ;:TIME OUT ON XOR?
6343 031504 012637 000004 MOV (SP)+,@ERRVEC ;:RESTORE THE ERROR VECTOR
6344 031510 000463 000004 BR $SVLAD ;:GO TO THE NEXT TEST
6345 031512 022626 000004 5$: CMP (SP)+,(SP)+ ;:CLEAR THE STACK AFTER A TIME OUT
6346 031514 012637 000004 MOV (SP)+,@ERRVEC ;:RESTORE THE ERROR VECTOR
6347 031520 000423 000004 BR 7$ ;:LOOP ON THE PRESENT TEST
6348 031522 000400 147410 6$: ;*****END OF CODE FOR THE XOR TESTER*****
6349 031522 032777 000400 147410 BIT #BIT08,@SWR ;:LOOP ON SPEC. TEST?
6350 031530 001404 147402 001102 BEQ 2$ ;:BR IF NO
6351 031532 127737 147402 001102 CMPB @SWR,$STNM ;:ON THE RIGHT TEST? SWR<7:0>
6352 031540 001465 147402 001102 BEQ $OVER ;:BR IF YES
6353 031542 105737 001103 2$: TSTB $ERFLG ;:HAS AN ERROR OCCURRED?
6354 031546 001421 001103 BEQ 3$ ;:BR IF NO
6355 031550 123737 001115 001103 CMPB $ERMAX,$ERFLG ;:MAX. ERRORS FOR THIS TEST OCCURRED?
6356 031556 101015 001115 001103 BHI 3$ ;:BR IF NO
6357 031560 032777 001000 147352 BIT #BIT09,@SWR ;:LOOP ON ERROR?
6358 031566 001404 001000 147352 BEQ 4$ ;:BR IF NO
6359 031570 013737 001110 001106 7$: MOV $LPERR,$LPADR ;:SET LOOP ADDRESS TO LAST SCOPE
6360 031576 000446 001110 001106 BR $OVER
6361 031600 105037 001103 4$: CLRB $ERFLG ;:ZERO THE ERROR FLAG
6362 031604 005037 001174 001103 CLR $TIMES ;:CLEAR THE NUMBER OF ITERATIONS TO MAKE
6363 031610 000415 001174 001103 BR 1$ ;:ESCAPE TO THE NEXT TEST
6364 031612 032777 004000 147320 3$: BIT #BIT11,@SWR ;:INHIBIT ITERATIONS?
6365 031620 001011 004000 147320 BNE 1$ ;:BR IF YES
6366 031622 005737 001216 004000 147320 TST $PASS ;:IF FIRST PASS OF PROGRAM
6367 031626 001406 001216 004000 147320 BEQ 1$ ;: INHIBIT ITERATIONS
6368 031630 005237 001104 001216 INC $ICNT ;:INCREMENT ITERATION COUNT
6369 031634 023737 001174 001104 CMP $TIMES,$ICNT ;:CHECK THE NUMBER OF ITERATIONS MADE
6370 031642 002024 001174 001104 BGE $OVER ;:BR IF MORE ITERATION REQUIRED
6371 031644 012737 000001 001104 1$: MOV #1,$ICNT ;:REINITIALIZE THE ITERATION COUNTER
6372 031652 013737 031730 001174 MOV $MXCNT,$TIMES ;:SET NUMBER OF ITERATIONS TO DO
6373 031660 105237 001102 001174 $SVLAD: INCB $STNM ;:COUNT TEST NUMBERS
6374 031664 113737 001102 001214 MOVB $STNM,$TESTN ;:SET TEST NUMBER IN APT MAILBOX

```



```
6431 032116
6432 032116 022737 024022 000042 5$:      CMP      #SENDAD,@#42      ;;ACT-11 AUTO-ACCEPT?
6433 032124 001001                BNE      6$              ;;BRANCH IF NO
6434 032126 000000                HALT                    ;;YES
6435 032130
6436 032130 000002 6$:      RTI              ;;RETURN
6437
6438 .SBTTL  TYPE ROUTINE
6439
6440 ::*****
6441 ::*ROUTINE TO TYPE AS'IZ MESSAGE. MESSAGE MUST TERMINATE WITH A 0 BYTE.
6442 ::*THE ROUTINE WILL INSERT A NUMBER OF NULL CHARACTERS AFTER A LINE FEED.
6443 ::*NOTE1:      $NULL CONTAINS THE CHARACTER TO BE USED AS THE FILLER CHARACTER.
6444 ::*NOTE2:      $FILLS CONTAINS THE NUMBER OF FILLER CHARACTERS REQUIRED.
6445 ::*NOTE3:      $FILLC CONTAINS THE CHARACTER TO FILL AFTER.
6446 ::*
6447 ::*CALL:
6448 ::*1) USING A TRAP INSTRUCTION
6449 ::*      TYPE      ,MESADR      ;;MESADR IS FIRST ADDRESS OF AN ASCIZ STRING
6450 ::*OR
6451 ::*      TYPE
6452 ::*      MESADR
6453 ::*
6454 032132 105737 001157 $TYPE:  TSTB      $TPFLG      ;;IS THERE A TERMINAL?
6455 032136 100002        BPL      1$              ;;BR IF YES
6456 032140 000000        HALT                    ;;HALT HERE IF NO TERMINAL
6457 032142 000430        BR      3$              ;;LEAVE
6458 032144 010046 1$:      MOV      RO,-(SP)      ;;SAVE RO
6459 032146 017600 000002  MOV      @2(SP),RO      ;;GET ADDRESS OF ASCIZ STRING
6460 032152 122737 000001 001230  CMPB     #APTENV,$ENV      ;;RUNNING IN APT MODE
6461 032160 001011        BNE      62$             ;;NO,GO CHECK FOR APT CONSOLE
6462 032162 132737 000100 001231  BITB     #APTSPOOL,$ENVM  ;;SPOOL MESSAGE TO APT
6463 032170 001405        BEQ      62$             ;;NO,GO CHECK FOR CONSOLE
6464 032172 010037 032202  MOV      RO,61$          ;;SETUP MESSAGE ADDRESS FOR APT
6465 032176 004737 032720  JSR      PC,$ATY3       ;;SPOOL MESSAGE TO APT
6466 032202 000000 61$:      .WORD     0              ;;MESSAGE ADDRESS
6467 032204 132737 000040 001231 62$:      BITB     #APTCSUP,$ENVM  ;;APT CONSOLE SUPPRESSED
6468 032212 001003        BNE      60$             ;;YES,SKIP TYPE OUT
6469 032214 112046 2$:      MOVB     (RO)+,-(SP)      ;;PUSH CHARACTER TO BE TYPED ONTO STACK
6470 032216 001005        BNE      4$              ;;BR IF IT ISN'T THE TERMINATOR
6471 032220 005726        TST      (SP)+          ;;IF TERMINATOR POP IT OFF THE STACK
6472 032222 012600 60$:      MOV      (SP)+,RO      ;;RESTORE RO
6473 032224 062716 000002 3$:      ADD      #2,(SP)      ;;ADJUST RETURN PC
6474 032230 000002        RTI              ;;RETURN
6475 032232 122716 000011 4$:      CMPB     #HT,(SP)      ;;BRANCH IF <HT>
6476 032236 001430        BEQ      8$              ;;BRANCH IF NOT <CRLF>
6477 032240 122716 000200  CMPB     #CRLF,(SP)
6478 032244 001006        BNE      5$              ;;POP <CR><LF> EQUIV
6479 032246 005726        TST      (SP)+          ;;TYPE A CR AND LF
6480 032250 104401        TYPE
6481 032252 001205        $CRLF
6482 032254 105037 032462  CLRB     $CHARCNT      ;;CLEAR CHARACTER COUNT
6483 032260 000755        BR      2$              ;;GET NEXT CHARACTER
6484 032262 004737 032344 5$:      JSR      PC,$TYPEPC     ;;GO TYPE THIS CHARACTER
6485 032266 123726 001156 6$:      CMPB     $FILLC,(SP)+  ;;IS IT TIME FOR FILLER CHARS.?
6486 032272 001350        BNE      2$              ;;IF NO GO GET NEXT CHAR.
```

```
6487 032274 013746 001154      MOV      $NULL,-(SP)      ;;GET # OF FILLER CHARS. NEEDED
6488                               ;;AND THE NULL CHAR.
6489 032300 105366 000001      7$:  DECB      1(SP)      ;;DOES A NULL NEED TO BE TYPED?
6490 032304 002770              BLT      6$              ;;BR IF NO--GO POP THE NULL OFF OF STACK
6491 032306 004737 032344      JSR      PC,$TYPEC      ;;GO TYPE A NULL
6492 032312 105337 032462      DECB      $CHARCNT      ;;DO NOT COUNT AS A COUNT
6493 032316 000770              BR       7$              ;;LOOP

;HORIZONTAL TAB PROCESSOR
6494
6495
6496
6497 032320 112716 000040      8$:  MOVB      #' (SP)      ;;REPLACE TAB WITH SPACE
6498 032324 004737 032344      9$:  JSR      PC,$TYPEC      ;;TYPE A SPACE
6499 032330 132737 000007 032462  BITB      #7,$CHARCNT      ;;BRANCH IF NOT AT
6500 032336 001372              BNE      9$              ;;TAB STOP
6501 032340 005726              TST      (SP)+           ;;POP SPACE OFF STACK
6502 032342 000724              BR       2$              ;;GET NEXT CHARACTER
6503 032344
6504 032344 105777 146574      $TYPEC: TSTB     @STKS          ;  ;;CHAR IN KYBD BUFFER?      :MJD001
6505 032350 100022              BPL      10$             ;;BR IF NOT                :MJD001
6506 032352 017746 146570      MOV      @STKB,-(SP)      ;;GET CHAR                  :MJD001
6507 032356 042716 177600      BIC      #177600,(SP)     ;;STRIP EXTRANEOUS BITS    :MJD001
6508 032362 122716 000023      CMPB     #$XOFF,(SP)     ;;WAS CHAR XOFF           :MJD001
6509 032366 001012              BNE      102$           ;;BR IF NOT                :MJD001
6510 032370
6511 032370 105777 146550      101$: TSTB     @STKS          ;;WAIT FOR CHAR            :MJD001
6512 032374 100375              BPL      101$           ;;BR IF NOT                :MJD001
6513 032376 117716 146544      MOVB     @STKB,(SP)       ;;GET CHAR                  :MJD001
6514 032402 042716 177600      BIC      #177600,(SP)     ;;STRIP IT                  :MJD001
6515 032406 122716 000021      CMPB     #$XON,(SP)      ;;WAS IT XON?             :MJD001
6516 032412 001366              BNE      101$           ;;BR IF NOT                :MJD001
6517 032414
6518 032414 005726      102$: TST      (SP)+           ;;FIX STACK                :MJD001
6519 032416
6520 032416 105777 146526      10$:  TSTB     @STPS          ;;WAIT UNTIL PRINTER IS READY :MJD001
6521 032422 100375              BPL      10$              ;;BR IF NOT                :MJD001
6522 032424 116677 000002 146520  MOVB     2(SP),@STPB      ;;LOAD CHAR TO BE TYPED INTO DATA REG.
6523 032432 122766 000015 000002  CMPB     #CR,2(SP)       ;;IS CHARACTER A CARRIAGE RETURN?
6524 032440 001003              BNE      1$              ;;BRANCH IF NO
6525 032442 105037 032462      CLRB     $CHARCNT       ;;YES--CLEAR CHARACTER COUNT
6526 032446 000406              BR       $TYPEX          ;;EXIT
6527 032450 122766 000012 000002  1$:  CMPB     #LF,2(SP)     ;;IS CHARACTER A LINE FEED?
6528 032456 001402              BEQ      $TYPEX          ;;BRANCH IF YES
6529 032460 105227              INCB     (PC)+           ;;COUNT THE CHARACTER
6530 032462 000000      $CHARCNT: .WORD      0      ;;CHARACTER COUNT STORAGE
6531 032464 000207      $TYPEX:  RTS      PC
```

.SBTTL CONVERT BINARY TO DECIMAL AND TYPE ROUTINE

```
6532
6533
6534
6535
6536
6537
6538
6539
6540
6541
6542
;*****
;*THIS ROUTINE IS USED TO CHANGE A 16-BIT BINARY NUMBER TO A 5-DIGIT
;*SIGNED DECIMAL (ASCII) NUMBER AND TYPE IT. DEPENDING ON WHETHER THE
;*NUMBER IS POSITIVE OR NEGATIVE A SPACE OR A MINUS SIGN WILL BE TYPED
;*BEFORE THE FIRST DIGIT OF THE NUMBER. LEADING ZEROS WILL ALWAYS BE
;*REPLACED WITH SPACES.
;*CALL:
;*      MOV      NUM,-(SP)      ;;PUT THE BINARY NUMBER ON THE STACK
```

```

6543          :*      TYPDS          ;;GO TO THE ROUTINE
6544
6545          $TYPDS:
6546 032466    010046    MOV      R0,-(SP)          ;;PUSH R0 ON STACK
6547 032470    010146    MOV      R1,-(SP)          ;;PUSH R1 ON STACK
6548 032472    010246    MOV      R2,-(SP)          ;;PUSH R2 ON STACK
6549 032474    010346    MOV      R3,-(SP)          ;;PUSH R3 ON STACK
6550 032476    010546    MOV      R5,-(SP)          ;;PUSH R5 ON STACK
6551 032500    012746    020200  MOV      #20200,-(SP)      ;;SET BLANK SWITCH AND SIGN
6552 032504    016605    000020  MOV      20(SP),R5        ;;GET THE INPUT NUMBER
6553 032510    100004    BPL      1$                ;;BR IF INPUT IS POS.
6554 032512    005405    NEG      R5                ;;MAKE THE BINARY NUMBER POS.
6555 032514    112766    000055  000001  MOVVB   #'-,1(SP)         ;;MAKE THE ASCII NUMBER NEG.
6556 032522    005000    1$:     CLR      R0                ;;ZERO THE CONSTANTS INDEX
6557 032524    012703    032702  MOV      #SDBLK,R3        ;;SETUP THE OUTPUT POINTER
6558 032530    112723    000040  MOVVB   #' ,(R3)+         ;;SET THE FIRST CHARACTER TO A BLANK
6559 032534    005002    2$:     CLR      R2                ;;CLEAR THE BCD NUMBER
6560 032536    016001    032672  MOV      $DTBL(R0),R1     ;;GET THE CONSTANT
6561 032542    160105    3$:     SUB      R1,R5            ;;FORM THIS BCD DIGIT
6562 032544    002402    BLT      4$                ;;BR IF DONE
6563 032546    005202    INC      R2                ;;INCREASE THE BCD DIGIT BY 1
6564 032550    000774    BR       3$
6565 032552    060105    4$:     ADD      R1,R5            ;;ADD BACK THE CONSTANT
6566 032554    005702    TST      R2                ;;CHECK IF BCD DIGIT=0
6567 032556    001002    BNE      5$                ;;FALL THROUGH IF 0
6568 032560    105716    TSTB    (SP)              ;;STILL DOING LEADING 0'S?
6569 032562    100407    BMI      7$                ;;BR IF YES
6570 032564    106316    5$:     ASLB    (SP)              ;;MSD?
6571 032566    103003    BCC      6$                ;;BR IF NO
6572 032570    116663    000001  177777  MOVVB   1(SP),-1(R3)      ;;YES--SET THE SIGN
6573 032576    052702    000060  6$:     BIS      #'0,R2          ;;MAKE THE BCD DIGIT ASCII
6574 032602    052702    000040  7$:     BIS      #' ,R2          ;;MAKE IT A SPACE IF NOT ALREADY A DIGIT
6575 032606    110223    MOVVB   R2,(R3)+         ;;PUT THIS CHARACTER IN THE OUTPUT BUFFER
6576 032610    005720    TST      (R0)+           ;;JUST INCREMENTING
6577 032612    020027    000010  CMP      R0,#10          ;;CHECK THE TABLE INDEX
6578 032616    002746    BLT      2$                ;;GO DO THE NEXT DIGIT
6579 032620    003002    BGT      8$                ;;GO TO EXIT
6580 032622    010502    MOV      R5,R2            ;;GET THE LSD
6581 032624    000764    BR       6$                ;;GO CHANGE TO ASCII
6582 032626    105726    8$:     TSTB    (SP)+           ;;WAS THE LSD THE FIRST NON-ZERO?
6583 032630    100003    BPL      9$                ;;BR IF NO
6584 032632    116663    177777  177776  MOVVB   -1(SP),-2(R3)    ;;YES--SET THE SIGN FOR TYPING
6585 032640    105013    9$:     CLRB   (R3)              ;;SET THE TERMINATOR
6586 032642    012605    MOV      (SP)+,R5        ;;POP STACK INTO R5
6587 032644    012603    MOV      (SP)+,R3        ;;POP STACK INTO R3
6588 032646    012602    MOV      (SP)+,R2        ;;POP STACK INTO R2
6589 032650    012601    MOV      (SP)+,R1        ;;POP STACK INTO R1
6590 032652    012600    MOV      (SP)+,R0        ;;POP STACK INTO R0
6591 032654    104401    032702  TYPE    $SDBLK           ;;NOW TYPE THE NUMBER
6592 032660    016666    000002  000004  MOV      2(SP),4(SP)     ;;ADJUST THE STACK
6593 032666    012616    MOV      (SP)+,(SP)
6594 032670    000002    RTI                          ;;RETURN TO USER
6595 032672    023420    $DTBL: 1000.
6596 032674    001750    1000.
6597 032676    000144    100.
6598 032700    000012    10.
  
```

```
6599 032702 000004          $DBLK: .BLKW 4
6600                               .SBTTL APT COMMUNICATIONS ROUTINE
6601
6602                               ::*****
6603 032712 112737 000001 033156 $ATY1: MOVB #1,$FFLG      ;;TO REPORT FATAL ERROR
6604 032720 112737 000001 033154 $ATY3: MOVB #1,$MFLG      ;;TO TYPE A MESSAGE
6605 032726 000403          BR          $ATYC
6606 032730 112737 000001 033156 $ATY4: MOVB #1,$FFLG      ;;TO ONLY REPORT FATAL ERROR
6607 032736          $ATYC:
6608 032736 010046          MOV      R0,-(SP)      ;;PUSH R0 ON STACK
6609 032740 010146          MOV      R1,-(SP)      ;;PUSH R1 ON STACK
6610 032742 105737 033154          TSTB   $MFLG          ;;SHOULD TYPE A MESSAGE?
6611 032746 001450          BEQ    5$             ;;IF NOT: BR
6612 032750 122737 000001 001230  CMPB   #APTENV,$ENV   ;;OPERATING UNDER APT?
6613 032756 001031          BNE   3$             ;;IF NOT: BR
6614 032760 132737 000100 001231  BITB   #APTPOOL,$ENVM ;;SHOULD SPOOL MESSAGES?
6615 032766 001425          BEQ    3$             ;;IF NOT: BR
6616 032770 017600 000004          MOV    @4(SP),R0      ;;GET MESSAGE ADDR.
6617 032774 062766 000002 000004  ADD    #2,4(SP)       ;;BUMP RETURN ADDR.
6618 033002 005737 001210          1$:   TST    $MSGTYPE  ;;SEE IF DONE W/ LAST XMISSION?
6619 033006 001375          BNE   1$             ;;IF NOT: WAIT
6620 033010 010037 001224          MOV    R0,$MSGAD     ;;PUT ADDR IN MAILBOX
6621 033014 105720          2$:   TSTB   (R0)+     ;;FIND END OF MESSAGE
6622 033016 001376          BNE   2$
6623 033020 163700 001224          SUB    $MSGAD,R0      ;;SUB START OF MESSAGE
6624 033024 006200          ASR    R0             ;;GET MESSAGE LNTH IN WORDS
6625 033026 010037 001226          MOV    R0,$MSGGLT    ;;PUT LENGTH IN MAILBOX
6626 033032 012737 000004 001210  MOV    #4,$MSGTYPE    ;;TELL APT TO TAKE MSG.
6627 033040 000413          BR     5$
6628 033042 017637 000004 033066  3$:   MOV    @4(SP),4$    ;;PUT MSG ADDR IN JSR LINKAGE
6629 033050 062766 000002 000004  ADD    #2,4(SP)       ;;BUMP RETURN ADDRESS
6630 033056 013746 177776          MOV    177776,-(SP)  ;;PUSH 177776 ON STACK
6631 033062 004737 032132          JSR   PC,$TYPE      ;;CALL TYPE MACRO
6632 033066 000000          4$:   .WORD  0
6633 033070          5$:
6634 033070 105737 033156          10$:  TSTB   $FFLG          ;;SHOULD REPORT FATAL ERROR?
6635 033074 001416          BEQ   12$           ;;IF NOT: BR
6636 033076 005737 001230          TST   $ENV           ;;RUNNING UNDER APT?
6637 033102 001413          BEQ   12$           ;;IF NOT: BR
6638 033104 005737 001210          11$:  TST   $MSGTYPE       ;;FINISHED LAST MESSAGE?
6639 033110 001375          BNE   11$           ;;IF NOT: WAIT
6640 033112 017637 000004 001212  MOV    @4(SP),$FATAL  ;;GET ERROR #
6641 033120 062766 000002 000004  ADD    #2,4(SP)       ;;BUMP RETURN ADDR.
6642 033126 005237 001210          INC   $MSGTYPE       ;;TELL APT TO TAKE ERROR
6643 033132 105037 033156          12$:  CLRB   $FFLG          ;;CLEAR FATAL FLAG
6644 033136 105037 033155          CLRB   $LFLG         ;;CLEAR LOG FLAG
6645 033142 105037 033154          CLRB   $MFLG         ;;CLEAR MESSAGE FLAG
6646 033146 012601          MOV    (SP)+,R1      ;;POP STACK INTO R1
6647 033150 012600          MOV    (SP)+,R0      ;;POP STACK INTO R0
6648 033152 000207          RTS    PC            ;;RETURN
6649 033154 000          $MFLG: .BYTE 0      ;;MESSG. FLAG
6650 033155 000          $LFLG: .BYTE 0      ;;LOG FLAG
6651 033156 000          $FFLG: .BYTE 0      ;;FATAL FLAG
6652          033160          .EVEN
6653          000200
6654          000001
APTSIZE=200
APTENV=001
```



```

6655      000100      APTSPool=100
6656      000040      APTCSUP=040
6657                                     .SBTTL  BINARY TO OCTAL (ASCII) AND TYPE
6658
6659      ::*****
6660      ::*THIS ROUTINE IS USED TO CHANGE A 16-BIT BINARY NUMBER TO A 6-DIGIT
6661      ::*OCTAL (ASCII) NUMBER AND TYPE IT.
6662      ::*STYPOS---ENTER HERE TO SETUP SUPPRESS ZEROS AND NUMBER OF DIGITS TO TYPE
6663      ::*CALL:
6664      ::*      MOV      NUM,-(SP)      ::NUMBER TO BE TYPED
6665      ::*      TYPOS      ::CALL FOR TYPEOUT
6666      ::*      .BYTE  N      ::N=1 TO 6 FOR NUMBER OF DIGITS TO TYPE
6667      ::*      .BYTE  M      ::M=1 OR 0
6668      ::*                                     ::1=TYPE LEADING ZEROS
6669      ::*                                     ::0=SUPPRESS LEADING ZEROS
6670
6671      ::*STYPON----ENTER HERE TO TYPE OUT WITH THE SAME PARAMETERS AS THE LAST
6672      ::*STYPOS OR STYPOC
6673      ::*CALL:
6674      ::*      MOV      NUM,-(SP)      ::NUMBER TO BE TYPED
6675      ::*      TYPON      ::CALL FOR TYPEOUT
6676
6677      ::*STYPOC---ENTER HERE FOR TYPEOUT OF A 16 BIT NUMBER
6678      ::*CALL:
6679      ::*      MOV      NUM,-(SP)      ::NUMBER TO BE TYPED
6680      ::*      TYPOC      ::CALL FOR TYPEOUT
6681
6682  033160  017646  000000      STYPOS: MOV      @ (SP),-(SP)      ::PICKUP THE MODE
6683  033164  116637  000001  033403  MOV      1(SP), $OFILL      ::LOAD ZERO FILL SWITCH
6684  033172  112637  033405      MOV      (SP)+, $OMODE+1    ::NUMBER OF DIGITS TO TYPE
6685  033176  062716  000002      ADD      #2, (SP)          ::ADJUST RETURN ADDRESS
6686  033202  000406      BR      $TYPON
6687  033204  112737  000001  033403  STYPOC: MOV      #1, $OFILL      ::SET THE ZERO FILL SWITCH
6688  033212  112737  000006  033405  MOV      #6, $OMODE+1      ::SET FOR SIX(6) DIGITS
6689  033220  112737  000005  033402  STYPON: MOV      #5, $OCNT      ::SET THE ITERATION COUNT
6690  033226  010346      MOV      R3, -(SP)         ::SAVE R3
6691  033230  010446      MOV      R4, -(SP)         ::SAVE R4
6692  033232  010546      MOV      R5, -(SP)         ::SAVE R5
6693  033234  113704  033405      MOV      $OMODE+1, R4      ::GET THE NUMBER OF DIGITS TO TYPE
6694  033240  005404      NEG      R4
6695  033242  062704  000006      ADD      #6, R4            ::SUBTRACT IT FOR MAX. ALLOWED
6696  033246  110437  033404      MOV      R4, $OMODE        ::SAVE IT FOR USE
6697  033252  113704  033403      MOV      $OFILL, R4        ::GET THE ZERO FILL SWITCH
6698  033256  016605  000012      MOV      12(SP), R5        ::PICKUP THE INPUT NUMBER
6699  033262  005003      CLR      R3                ::CLEAR THE OUTPUT WORD
6700  033264  006105      1$:  ROL      R5            ::ROTATE MSB INTO 'C'
6701  033266  000404      BR      3$                ::GO DO MSB
6702  033270  006105      2$:  ROL      R5            ::FORM THIS DIGIT
6703  033272  006105      ROL      R5
6704  033274  006105      ROL      R5
6705  033276  010503      MOV      R5, R3
6706  033300  006103      3$:  ROL      R3            ::GET LSB OF THIS DIGIT
6707  033302  105337  033404      DECB    $OMODE            ::TYPE THIS DIGIT?
6708  033306  100016      BPL     7$                ::BR IF NO
6709  033310  042703  177770      BIC     #177770, R3       ::GET RID OF JUNK
6710  033314  001002      BNE     4$                ::TEST FOR 0
  
```

```

6711 033316 005704          TST      R4          ::SUPPRESS THIS 0?
6712 033320 001403          BEQ      5$          ::BR IF YES
6713 033322 005204          4$: INC      R4          ::DON'T SUPPRESS ANYMORE 0'S
6714 033324 052703 000060    BIS      #'0,R3      ::MAKE THIS DIGIT ASCII
6715 033330 052703 000040    5$: BIS      #' ,R3      ::MAKE ASCII IF NOT ALREADY
6716 033334 110337 033400    MOVB     R3,8$       ::SAVE FOR TYPING
6717 033340 104401 033400    TYPE     8$         ::GO TYPE THIS DIGIT
6718 033344 105337 033402    7$: DECB    $OCNT     ::COUNT BY 1
6719 033350 003347          BGT      2$         ::BR IF MORE TO DO
6720 033352 002402          BLT      6$         ::BR IF DONE
6721 033354 005204          INC      R4          ::INSURE LAST DIGIT ISN'T A BLANK
6722 033356 000744          BR       2$         ::GO DO THE LAST DIGIT
6723 033360 012605          6$: MOV      (SP)+,R5  ::RESTORE R5
6724 033362 012604          MOV      (SP)+,R4  ::RESTORE R4
6725 033364 012603          MOV      (SP)+,R3  ::RESTORE R3
6726 033366 016666 000002 000004  MOV      2(SP),4(SP) ::SET THE STACK FOR RETURNING
6727 033374 012616          MOV      (SP)+,(SP)
6728 033376 000002          RTI                    ::RETURN
6729 033400          8$: .BYTE    0          ::STORAGE FOR ASCII DIGIT
6730 033401          .BYTE    0          ::TERMINATOR FOR TYPE ROUTINE
6731 033402          .BYTE    0          ::OCTAL DIGIT COUNTER
6732 033403          .BYTE    0          ::ZERO FILL SWITCH
6733 033404 000000          $OCNT: .WORD    0     ::NUMBER OF DIGITS TO TYPE
6734          .SMTTL  TTY INPUT ROUTINE
6735
6736          ::*****
6737          .ENABL  LSB
6738 033406 000000          $TKCNT: .WORD    0     ::NUMBER OF ITEMS IN QUEUE
6739 033410 000000          $TKQIN: .WORD    0     ::INPUT POINTER
6740 033412 000000          $TKQOUT: .WORD   0     ::OUTPUT POINTER
6741 033414 000001          $TKQSRT: .BLKB   1     ::TTY KEYBOARD QUEUE
6742          $TKQEND=.
6743          .EVEN
6744
6745          ;*TK INITIALIZE ROUTINE
6746          ;*THIS ROUTINE WILL INITIALIZE THE TTY KEYBOARD INPUT QUEUE
6747          ;*SETUP THE INTERRUPT VECTOR AND TURN ON THE KEYBOARD INTERRUPT
6748
6749          ;*CALL:
6750          ;*
6751          ;* JSR      PC,$TKINT
6752          ;* RETURN
6753 033416 005037 033406          $TKINT: CLR      $TKCNT  ::CLEAR COUNT OF ITEMS IN QUEUE
6754 033422 012737 033414 033410  MOV      #$TKQSRT,$TKQIN ::MOVE THE STARTING ADDRESS OF THE
6755 033430 013737 033410 033412  MOV      $TKQIN,$TKQOUT  ::QUEUE INTO THE INPUT & OUTPUT POINTERS.
6756 033436 012737 033466 000060  MOV      #$TKSRV,@TKVEC  ::INITIALIZE THE KEYBOARD VECTOR
6757 033444 012737 000200 000062  MOV      #200,@TKVEC+2   ::'BR' LEVEL 4
6758 033452 005777 145470          TST      @TKB          ::CLEAR DONE FLAG
6759 033456 012777 000100 145460  MOV      #100,@TKS      ::ENABLE TTY KEYBOARD INTERRUPT
6760 033464 000207          RTS      PC          ::RETURN TO CALLER
6761
6762          ;*TK SERVICE ROUTINE
6763          ;*THIS ROUTINE WILL SERVICE THE TTY KEYBOARD INTERRUPT
6764          ;*BY READING THE CHARACTER FROM THE INPUT BUFFER AND PUTTING
6765          ;*IT IN THE QUEUE.
6766          ;*IF THE CHARACTER IS A "CONTROL-C" (^C) $TKINT IS CALLED AND

```

```

6767 ;*UPON RETURN EXIT IS MADE TO THE "CONTROL-C" RESTART ADDRESS (STOP)
6768
6769 033466 117746 145454 $TKSRV: MOVB @STKB, -(SP) ;; PICKUP THE CHARACTER
6770 033472 042716 177600 BIC #^C177, (SP) ;; STRIP THE JUNK
6771 033476 021627 000021 CMP (SP), #XON ;; IS IT A RANDOM XON? ;RAN001
6772 033502 001002 BNE 30$ ;; BRANCH IF NO ;RAN001
6773 033504 005726 TST (SP)+ ;; CLEAN RANDOM XON OFF STACK ;RAN001
6774 033506 000002 RTI ;; RETURN ;RAN001
6775 033510
6776 033510 021627 000003 30$: CMP (SP), #3 ;; IS IT A CONTROL C?
6777 033514 001007 BNE 1$ ;; BRANCH IF NO
6778 033516 104401 034626 TYPE ,SCNTLC ;; TYPE A CONTROL-C (^C)
6779 033522 (04737 033414 JSR PC, $TKINT ;; INIT THE KEYBOARD
6780 033526 (05726 TST (SP)+ ;; CLEAN UP STACK
6781 033530 000137 030500 JMP STOP ;; CONTROL C RESTART
6782 033534 021627 000007 1$: CMP (SP), #7 ;; IS IT A CONTROL G?
6783 033540 061004 BNE 2$ ;; BRANCH IF NO
6784 033542 022737 000176 001140 CMP #SWREG, SWR ;; IS SOFT-SWR SELECTED?
6785 033550 001500 BEQ 6$ ;; GO TO SWR CHANGE
6786
6787 033552
6788 033552 022737 000001 033406 2$: CMP #1, $TKCNT ;; IS THE QUEUE FULL?
6789 033560 001004 BNE 3$ ;; BRANCH IF NO
6790 033562 104401 001200 TYPE ,SBELL ;; RING THE TTY BELL
6791 033566 005726 TST (SP)+ ;; CLEAN CHARACTER OFF OF STACK
6792 033570 000451 BR 5$ ;; EXIT
6793 033572 021627 000023 3$: CMP (SP), #23 ;; IS IT A CONTROL-S?
6794 033576 001021 BNE 32$ ;; BRANCH IF NO
6795 033600 005077 145340 CLR @STKS ;; DISABLE TTY KEYBOARD INTERRUPTS
6796 033604 005726 TST (SP)+ ;; CLEAN CHAR OFF STACK
6797 033606 105777 145332 31$: TSTB @STKS ;; WAIT FOR A CHAR
6798 033612 100375 BPL 31$ ;; LOOP UNTIL ITS THERE
6799 033614 117746 145326 MOVB @STKB, -(SP) ;; GET THE CHARACTER
6800 033620 042716 177600 BIC #^C177, (SP) ;; MAKE IT 7-BIT ASCII
6801 033624 022627 000021 CMP (SP)+, #21 ;; IS IT A CONTROL-Q?
6802 033630 001366 BNE 31$ ;; BRANCH IF NO
6803 033632 012777 000100 145304 MOV #100, @STKS ;; REENABLE TTY KEYBOARD INTERRUPTS
6804 033640 000002 RTI ;; RETURN
6805 033642 005237 033406 32$: INC $TKCNT ;; COUNT THIS CHARACTER
6806 033646 021627 000140 CMP (SP), #140 ;; IS IT UPPER CASE?
6807 033652 002405 BLT 4$ ;; BRANCH IF YES
6808 033654 021627 000175 CMP (SP), #175 ;; IS IT A SPECIAL CHAR?
6809 033660 003002 BGT 4$ ;; BRANCH IF YES
6810 033662 042716 000040 BIC #40, (SP) ;; MAKE IT UPPER CASE
6811 033666 112677 177516 4$: MOVB (SP)+, @STKQIN ;; AND PUT IT IN QUEUE
6812 033672 005237 033410 INC $TKQIN ;; UPDATE THE POINTER
6813 033676 023727 033410 033415 CMP $TKQIN, #STKQEND ;; GO OFF THE END?
6814 033704 001003 BNE 5$ ;; BRANCH IF NO
6815 033706 012737 033414 033410 MOV #STKQSR, $TKQIN ;; RESET THE POINTER
6816 033714 000002 5$: RTI ;; RETURN
6817
6818
6819
6820
6821
6822

```

```

;*****
;*SOFTWARE SWITCH REGISTER CHANGE ROUTINE.
;*ROUTINE IS ENTERED FROM THE TRAP HANDLER, AND WILL
;*SERVICE THE TEST FOR CHANGE IN SOFTWARE SWITCH REGISTER TRAP
;*CALL WHEN OPERATING IN TTY INTERRUPT MODE.

```

```

6823 033716 022737 000176 001140 $CKSWR: CMP #SWREG,SWR ;;IS THE SOFT-SWR SELECTED
6824 033724 001124 BNE 15$ ;;EXIT IF NOT
6825 033726 105777 145212 TSTB @STKS ;;IS A CHAR WAITING?
6826 033732 100121 BPL 15$ ;;IF NOT, EXIT
6827 033734 117746 145206 MOVB @STKB,-(SP) ;;YES
6828 033740 042716 177600 BIC #^C177,(SP) ;;MAKE IT 7-BIT ASCII
6829 033744 021627 000007 CMP (SP),#7 ;;IS IT A CONTROL-G?
6830 033750 001300 BNE 2$ ;;IF NOT, PUT IT IN THE TTY QUEUE
6831 ;;AND EXIT
6832
6833 *****
6834 ;*CONTROL IS PASSED TO THIS POINT FROM EITHER THE TTY INTERRUPT SERVICE
6835 ;*ROUTINE OR FROM THE SOFTWARE SWITCH REGISTER TRAP CALL, AS A RESULT OF A
6836 ;*CONTROL-G BEING TYPED, AND THE SOFTWARE SWITCH REGISTER BEING SELECTED.
6837 033752 123727 001134 000001 6$: CMPB $AUTOB,#1 ;;ARE WE RUNNING IN AUTO-MODE?
6838 033760 001674 BEQ 2$ ;;BRANCH IF YES
6839 033762 005726 TST (SP)+ ;;CLEAR CONTROL-G OFF STACK
6840 033764 004737 033416 JSR PC,$TKINT ;;FLUSH THE TTY INPUT QUEUE
6841 033770 005077 145150 CLR @STKS ;;DISABLE TTY KEYBOARD INTERRUPTS
6842 033774 112737 000001 001135 MOVB #1,$INTAG ;;SET INTERRUPT MODE INDICATOR
6843
6844 034002 104401 034640 $GTSWR: TYPE ,SCNTLG ;;ECHO THE CONTROL-G (^G)
6845 034006 104401 034645 TYPE ,SMSWR ;;TYPE CURRENT CONTENTS
6846 034012 013746 000176 MOV SWREG,-(SP) ;;SAVE SWREG FOR TYPEOUT
6847 034016 104402 TYPOC ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
6848 034020 104401 034656 TYPE ,SMNEW ;;PROMPT FOR NEW SWR
6849 034024 005046 19$: CLR -(SP) ;;CLEAR COUNTER
6850 034026 005046 CLR -(SP) ;;THE NEW SWR
6851 034030 105777 145110 7$: TSTB @STKS ;;CHAR THERE?
6852 034034 100375 BPL 7$ ;;IF NOT TRY AGAIN
6853
6854 034036 117746 145104 MOVB @STKB,-(SP) ;;PICK UP CHAR
6855 034042 042716 177600 BIC #^C177,(SP) ;;MAKE IT 7-BIT ASCII
6856
6857 034046 021627 000003 CMP (SP),#3 ;;IS IT A CONTROL-C?
6858 034052 001015 BNE 9$ ;;BRANCH IF NOT
6859 034054 104401 034626 TYPE ,SCNTLC ;;YES, ECHO CONTROL-C (^C)
6860 034060 062706 000006 ADD #6,SP ;;CLEAN UP STACK
6861 034064 123727 001135 000001 CMPB $INTAG,#1 ;;REENABLE TTY KEYBOARD INTERRUPTS?
6862 034072 001003 BNE 8$ ;;BRANCH IF NO
6863 034074 012777 000100 145042 8$: MOV #100,@STKS ;;ALLOW TTY KEYBOARD INTERRUPTS
6864 034102 000137 030500 JMP STOP ;;CONTROL-C RESTART
6865
6866
6867 034106 021627 000025 9$: CMP (SP),#25 ;;IS IT A CONTROL-U?
6868 034112 001005 BNE 10$ ;;BRANCH IF NOT
6869 034114 104401 034633 TYPE ,SCNTLU ;;YES, ECHO CONTROL-U (^U)
6870 034120 062706 000006 20$: ADD #6,SP ;;IGNORE PREVIOUS INPUT
6871 034124 000737 BR 19$ ;;LET'S TRY IT AGAIN
6872
6873
6874 034126 021627 000015 10$: CMP (SP),#15 ;;IS IT A <CR>?
6875 034132 001022 BNE 16$ ;;BRANCH IF NO
6876 034134 005766 000004 TST 4(SP) ;;YES, IS IT THE FIRST CHAR?
6877 034140 001403 BEQ 11$ ;;BRANCH IF YES
6878 034142 016677 000002 144770 MOV 2(SP),@SWR ;;SAVE NEW SWR

```

```

6879 034150 062706 000006      11$:  ADD      #6,SP          ;;CLEAR UP STACK
6880 034154 104401 001205      14$:  TYPE     $SCRLF        ;;ECHO <CR> AND <LF>
6881 034160 123727 001135      000001  CMPB     $INTAG,#1        ;;RE-ENABLE TTY KBD INTERRUPTS?
6882 034166 001003          BNE      15$             ;;BRANCH IF NOT
6883 034170 012777 000100      144746  MOV      #100,@$TKS      ;;RE-ENABLE TTY KBD INTERRUPTS
6884 034176 000002          15$:  RTI                    ;;RETURN
6885 034200 004737 032344      16$:  JSR      PC,$TYPEC     ;;ECHO CHAR
6886 034204 021627 000060          CMP      (SP),#60        ;;CHAR < 0?
6887 034210 002420          BLT      18$             ;;BRANCH IF YES
6888 034212 021627 000067          CMP      (SP),#67        ;;CHAR > 7?
6889 034216 003015          BGT      18$             ;;BRANCH IF YES
6890 034220 042726 000060          BIC      #60,(SP)+       ;;STRIP-OFF ASCII
6891 034224 005766 000002          TST      2(SP)           ;;IS THIS THE FIRST CHAR
6892 034230 001403          BEQ      17$             ;;BRANCH IF YES
6893 034232 006316          ASL      (SP)            ;;NO, SHIFT PRESENT
6894 034234 006316          ASL      (SP)            ;;CHAR OVER TO MAKE
6895 034236 006316          ASL      (SP)            ;;ROOM FOR NEW ONE.
6896 034240 005266 000002      17$:  INC      2(SP)           ;;KEEP COUNT OF CHAR
6897 034244 056616 177776          BIS      -2(SP),(SP)     ;;SET IN NEW CHAR
6898 034250 000667          BR       7$              ;;GET THE NEXT ONE
6899 034252 104401 001204      18$:  TYPE     $QUES          ;;TYPE ?<CR><LF>
6900 034256 000720          BR       20$             ;;SIMULATE CONTROL-U
6901
6902
6903
6904
6905
6906
6907
6908
6909
6910
6911
6912 034260 011646          $RDCHR: MOV      (SP),-(SP) ;;PUSH DOWN THE PC AND
6913 034262 016666 000004      000002  MOV      4(SP),2(SP)     ;;THE PS
6914 034270 005066 000004          CLR      4(SP)           ;;GET READY FOR A CHARACTER
6915 034274 005046          CLR      -(SP)           ;;PUT NEW PS ON STACK
6916 034276 012746 034304          MOV      #64$,-(SP)     ;;PUT NEW PC ON STACK
6917 034302 000002          RTI                    ;;POP NEW PC AND PS
6918 034304
6919 034304 005737 033406      64$:  TST      $TKCNT         ;;WAIT ON A CHARACTER
6920 034310 001775          BEQ      1$              1$:
6921 034312 005337 033406          DEC      $TKCNT         ;;DECREMENT THE COUNTER
6922 034316 117766 177070      000004  MOVB    @$TKQOUT,4(SP)   ;;GET ONE CHARACTER
6923 034324 005237 033412          INC      $TKQOUT        ;;UPDATE THE POINTER
6924 034330 023727 033412      033415  CMP      $TKQOUT,#$TKQEND ;;DID IT GO OFF OF THE END?
6925 034336 001003          BNE      2$              2$:
6926 034340 012737 033414      033412  MOV      #$TKQSRRT,$TKQOUT ;;RESET THE POINTER
6927 034346 000002          RTI                    ;;RETURN
6928
6929
6930
6931
6932
6933
6934

```

6935	034350	010346				SRDLIN: MOV	R3,-(SP)	::SAVE R3
6936	034352	005046				CLR	-(SP)	::CLEAR THE RUBOUT KEY
6937	034354	012703	034604			1\$: MOV	#\$TTYIN,R3	::GET ADDRESS
6938	034360	022703	034626			2\$: CMP	#\$TTYIN+22,R3	::BUFFER FULL?
6939	034364	101456				BLOS	4\$	::BR IF YES
6940	034366	104410				RDCHR		::GO READ ONE CHARACTER FROM THE TTY
6941	034370	112613				MOVB	(SP)+,(R3)	::GET CHARACTER
6942	034372	122713	000177			10\$: CMPB	#177,(R3)	::IS IT A RUBOUT
6943	034376	001022				BNE	5\$	::BR IF NO
6944	034400	005716				TST	(SP)	::IS THIS THE FIRST RUBOUT?
6945	034402	001007				BNE	6\$	::BR IF NO
6946	034404	112737	000134	034602		MOVB	#'\,9\$	::TYPE A BACK SLASH
6947	034412	104401	034602			TYPE	,9\$	
6948	034416	012716	177777			MOV	#-1,(SP)	::SET THE RUBOUT KEY
6949	034422	005303				6\$: DEC	R3	::BACKUP BY ONE
6950	034424	020327	034604			CMP	R3,\$TTYIN	::STACK EMPTY?
6951	034430	103434				BLO	4\$	::BR IF YES
6952	034432	111337	034602			MOVB	(R3),9\$	::SETUP TO TYPEOUT THE DELETED CHAR.
6953	034436	104401	034602			TYPE	,9\$	::GO TYPE
6954	034442	000746				BR	2\$	::GO READ ANOTHER CHAR.
6955	034444	005716				5\$: TST	(SP)	::RUBOUT KEY SET?
6956	034446	001406				BEQ	7\$	::BR IF NO
6957	034450	112737	000134	034602		MOVB	#'\,9\$	::TYPE A BACK SLASH
6958	034456	104401	034602			TYPE	,9\$	
6959	034462	005016				CLR	(SP)	::CLEAR THE RUBOUT KEY
6960	034464	122713	000025			7\$: CMPB	#25,(R3)	::IS CHARACTER A CTRL U?
6961	034470	001003				BNE	8\$	::BR IF NO
6962	034472	104401	034633			TYPE	,SCNTLU	::TYPE A CONTROL 'U'
6963	034476	000726				BR	1\$	::GO START OVER
6964	034500	122713	000022			8\$: CMPB	#22,(R3)	::IS CHARACTER A 'R'?
6965	034504	001011				BNE	3\$	::BRANCH IF NO
6966	034506	105013				CLRB	(R3)	::CLEAR THE CHARACTER
6967	034510	104401	001205			TYPE	,\$CRLF	::TYPE A 'CR' & 'LF'
6968	034514	104401	034604			TYPE	,\$TTYIN	::TYPE THE INPUT STRING
6969	034520	000717				BR	2\$	::GO PICKUP ANOTHER CHARACTER
6970	034522	104401	001204			4\$: TYPE	,\$QUES	::TYPE A '?'
6971	034526	000712				BR	1\$	::CLEAR THE BUFFER AND LOOP
6972	034530	111337	034602			3\$: MOVB	(R3),9\$	::ECHO THE CHARACTER
6973	034534	104401	034602			TYPE	,9\$	
6974	034540	122723	000015			CMPB	#15,(R3)+	::CHECK FOR RETURN
6975	034544	001305				BNE	2\$	::LOOP IF NOT RETURN
6976	034546	105063	177777			CLRB	-1(R3)	::CLEAR RETURN (THE 15)
6977	034552	104401	001206			TYPE	,\$LF	::TYPE A LINE FEED
6978	034556	005726				TST	(SP)+	::CLEAN RUBOUT KEY FROM THE STACK
6979	034560	012603				MOV	(SP)+,R3	::RESTORE R3
6980	034562	011646				MOV	(SP)-,(SP)	::ADJUST THE STACK AND PUT ADDRESS OF THE
6981	034564	016666	000004	000002		MOV	4(SP),2(SP)	::FIRST ASCII CHARACTER ON IT
6982	034572	012766	034604	000004		MOV	#\$TTYIN,4(SP)	
6983	034600	000002				RTI		::RETURN
6984	034602	000				9\$: .BYTE	0	::STORAGE FOR ASCII CHAR. TO TYPE
6985	034603	000				.BYTE	0	::TERMINATOR
6986	034604	000022				\$TTYIN: .BLKB	22	::RESERVE 22 BYTES FOR TTY INPUT
6987	034626	041536	005015	000		\$CNTLC: .ASCIZ	/'^C/<15><12>	::CONTROL 'C'
6988	034633	136	006525	000012		\$CNTLU: .ASCIZ	/'^U/<15><12>	::CONTROL 'U'
6989	034640	043536	005015	000		\$CNTLG: .ASCIZ	/'^G/<15><12>	::CONTROL 'G'
6990	034645	015	051412	051127		\$MSWR: .ASCIZ	<15><12>/SWR = /	

```

6991 034652 036440 000040
6992 034656 020040 042516 020127 SMNEW: .ASCIZ / NEW = /
6993 034664 020075 000
6994 034670
6995
6996
6997
6998
6999
7000
7001
7002
7003
7004
7005
7006
7007
7008
7009 034670 011646 000004 000002 SRDOCT: MOV (SP),-(SP) ;;PROVIDE SPACE FOR THE
7010 034672 016666 MOV 4(SP),2(SP) ;;INPUT NUMBER
7011 034700 010046 MOV R0,-(SP) ;;PUSH R0 ON STACK
7012 034702 010146 MOV R1,-(SP) ;;PUSH R1 ON STACK
7013 034704 010246 MOV R2,-(SP) ;;PUSH R2 ON STACK
7014 034706 104411 1$: RDLIN ;;READ AN ASCIZ LINE
7015 034710 012600 MOV (SP)+,R0 ;;GET ADDRESS OF 1ST CHARACTER
7016 034712 010037 035016 MOV R0,5$ ;;AND SAVE IT
7017 034716 005001 CLR R1 ;;CLEAR DATA WORD
7018 034720 005002 CLR R2
7019 034722 112046 2$: MOV (R0)+,-(SP) ;;PICKUP THIS CHARACTER
7020 034724 001420 BEQ 3$ ;;IF ZERO GET OUT
7021 034726 122716 000060 CMPB #'0,(SP) ;;MAKE SURE THIS CHARACTER
7022 034732 003026 BGT 4$ ;;IS AN OCTAL DIGIT
7023 034734 122716 000067 CMPB #'7,(SP)
7024 034740 002423 BLT 4$
7025 034742 006301 ASL R1 ;;*2
7026 034744 006102 ROL R2
7027 034746 006301 ASL R1 ;;*4
7028 034750 006102 ROL R2
7029 034752 006301 ASL R1 ;;*8
7030 034754 006102 ROL R2
7031 034756 042716 177770 BIC #'C7,(SP) ;;STRIP THE ASCII JUNK
7032 034762 062601 ADD (SP)+,R1 ;;ADD IN THIS DIGIT
7033 034764 000756 BR 2$ ;;LOOP
7034 034766 005726 3$: TST (SP)+ ;;CLEAN TERMINATOR FROM STACK
7035 034770 010166 000012 MOV R1,12(SP) ;;SAVE THE RESULT
7036 034774 010237 035026 MOV R2,$HIOCT
7037 035000 012602 MOV (SP)+,R2 ;;POP STACK INTO R2
7038 035002 012601 MOV (SP)+,R1 ;;POP STACK INTO R1
7039 035004 012600 MOV (SP)+,R0 ;;POP STACK INTO R0
7040 035006 000002 RTI ;;RETURN
7041 035010 005726 4$: TST (SP)+ ;;CLEAN PARTIAL FROM STACK
7042 035012 105010 CLRB (R0) ;;SET A TERMINATOR
7043 035014 104401 TYPE ;;TYPE UP THRU THE BAD CHAR.
7044 035016 000000 5$: .WORD 0
7045 035020 104401 001204 TYPE $QUES ;; '?' 'CR' & 'LF'
7046 035024 000730 BR 1$ ;;TRY AGAIN
    
```

```
7047 035026 000000 $HI OCT: .WORD 0 ;;HIGH ORDER BITS GO HERE
7048 .SBTTL DOUBLE LENGTH BINARY TO OCTAL ASCII CONVERT ROUTINE
7049
7050
7051 ::*****
7052 :*THIS ROUTINE WILL CONVERT A 32-BIT UNSIGNED BINARY NUMBER TO AN
7053 :*UNSIGNED OCTAL ASCII NUMBER.
7054 :*CALL
7055 :* MOV #PNTR,-(SP) ;;POINTER TO LOW WORD OF BINARY NUMBER
7056 :* JSR PC,@#$DB20 ;;CALL THE ROUTINE
7057 :* RETURN ;;THE ADDRESS OF THE FIRST ASCII CHAR. IS ON THE STACK
7058
7059 035030 104413 $DB20: SAVREG ;;SAVE ALL REGISTERS
7060 035032 016601 000002 MOV 2(SP),R1 ;;PICKUP THE POINTER TO LOW WORD
7061 035036 012705 035147 MOV #SOCTVL+13.,R5 ;;POINTER TO DATA TABLE
7062 035042 012704 000014 MOV #12.,R4 ;;DO ELEVEN CHARACTERS
7063 035046 012703 177770 MOV #^C7,R3 ;;MASK
7064 035052 012100 MOV (R1)+,R0 ;;LOWER WORD
7065 035054 012101 MOV (R1)+,R1 ;;HIGH WORD
7066 035056 005002 CLR R2 ;;TERMINATOR
7067 035060 110245 1$: MOV B R2,-(R5) ;;PUT CHARACTER IN DATA TABLE
7068 035062 010002 MOV R0,R2 ;;GET THIS DIGIT
7069 035064 005304 DEC R4 ;;COUNT THIS CHARACTER
7070 035066 003007 BGT 3$ ;;BR IF NOT THE LAST DIGIT
7071 035070 001405 BEQ 2$ ;;BR IF IT IS THE LAST DIGIT
7072 035072 005205 INC R5 ;;ALL DIGITS DONE-ADJUST POINTER FOR FIRST
7073 035074 010566 000002 MOV R5,2(SP) ;;ASCII CHAR. & PUT IT ON THE STACK
7074 035100 104414 RESREG ;;RESTORE ALL REGISTERS
7075 035102 000207 RTS PC ;;RETURN TO USER
7076 035104 006203 2$: ASR R3 ;;POSITION THE MASK FOR THE LAST DIGIT
7077 035106 006001 3$: ROR R1 ;;POSITION THE BINARY NUMBER FOR
7078 035110 006000 ROR R0 ;; THE NEXT OCTAL DIGIT
7079 035112 006001 ROR R1
7080 035114 006000 ROR R0
7081 035116 006001 ROR R1
7082 035120 006000 ROR R0
7083 035122 040302 BIC R3,R2 ;;MASK OUT ALL JUNK
7084 035124 062702 000060 ADD #'0,R2 ;;MAKE THIS CHAR. ASCII
7085 035130 000753 BR 1$ ;;GO PUT IT IN THE DATA TABLE
7086 035132 000016 $SOCTVL: .BLKB 14. ;;RESERVE DATA TABLE
7087 .SBTTL DOUBLE LENGTH BINARY TO DECIMAL ASCII CONVERT ROUTINE
7088
7089 ::*****
7090 :*THIS ROUTINE WILL CONVERT A 32-BIT BINARY NUMBER TO AN UNSIGNED
7091 :*DECIMAL (ASCII) NUMBER. THE SIGN OF THE BINARY NUMBER MUST BE
7092 :*POSITIVE.
7093 :*CALL
7094 :* MOV #PNTR,-(SP) ;;POINTER TO LOW WORD OF BINARY NUMBER
7095 :* JSR PC,@#$DB20 ;;CALL THE ROUTINE
7096 :* RETURN ;;THE FIRST ADDRESS OF ASCII
7097 ;; IS ON THE STACK
7098
7099
7100 035150 104413 $DB2D: SAVREG ;;SAVE REGISTERS
7101 035152 016602 000002 MOV 2(SP),R2 ;;PICKUP THE DATA POINTER
7102 035156 012700 035330 MOV #SDECVL,R0 ;;GET ADDRESS OF 'SDECVL' STRING
```



```

7103 035162 010066 000002      MOV      R0,2(SP)      ;;PUT ADDRESS OF ASCIZ STRING ON STACK
7104 035166 012201      MOV      (R2)+,R1     ;;PICKUP THE BINARY NUMBER
7105 035170 012202      MOV      (R2)+,R2
7106 035172 012737 000012 035246  MOV      #10,,4$      ;;SET UP TO DO 10 CONVERSIONS
7107 035200 012704 035260  MOV      #STNPWR,R4    ;;ADDRESS OF TEN POWER
7108 035204 012705 035262  MOV      #STNPWR+2,R5
7109 035210 005003      1$: CLR      R3        ;;CLEAR PARTIAL
7110 035212 161401      2$: SUB      (R4),R1    ;;SUBTRACT TEN POWER
7111 035214 005602      SBC      R2
7112 035216 161502      SUB      (R5),R2
7113 035220 002402      BLT      3$          ;;BR IF TEN POWER TO LARGE
7114 035222 005203      INC      R3          ;;ADD 1 TO PARTIAL
7115 035224 000772      BR       2$          ;;LOOP
7116 035226 062401      3$: ADD      (R4)+,R1  ;;RESTORE SUBTRACTED VALUE
7117 035230 005502      ADC      R2
7118 035232 062402      ADD      (R4)+,R2
7119 035234 022525      CMP      (R5)+,(R5)+ ;;MOVE TO NEXT TEN POWER
7120 035236 052703 000060  BIS      #'0,R3      ;;CHANGE PARTIAL TO ASCII
7121 035242 110320      MOVB     R3,(R0)+    ;;SAVE IT
7122 035244 005327      DEC      (PC)+      ;;DONE?
7123 035246 000000      4$: .WORD    0
7124 035250 001357      BNE     1$          ;;BR IF NO
7125 035252 105020      CLRB    (R0)+      ;;TERMINATOR
7126 035254 104414      RESREG  ;;RESTORE REGISTERS
7127 035256 000207      RTS     PC         ;;RETURN
7128 035260 145000      $TNPWR: 145000     ;;1.0E09
7129 035262 035632      35632
7130 035264 160400      160400           ;;1.0E08
7131 035266 002765      2765
7132 035270 113200      113200           ;;1.0E07
7133 035272 000230      230
7134 035274 041100      041100           ;;1.0E06
7135 035276 000017      17
7136 035300 103240      103240           ;;1.0E05
7137 035302 000001      1
7138 035304 023420      23420           ;;1.0E04
7139 035306 000000      0
7140 035310 001750      1750           ;;1.0E03
7141 035312 000000      0
7142 035314 000144      144           ;;1.0E02
7143 035316 000000      0
7144 035320 000012      12           ;;1.0E01
7145 035322 000000      0
7146 035324 000001      1           ;;1.0E00
7147 035326 000000      0
7148 035330 000014      $DECVL: .BLKB 12.  ;;RESERVE STORAGE FOR ASCIZ STRING
7149      .SBTTL SINGLE LENGTH BINARY TO DECIMAL ASCII ROUTINE
7150
7151      ;*****
7152      ;*THIS ROUTINE WILL CONVERT A 16-BIT UNSIGNED BINARY NUMBER TO AN
7153      ;*UNSIGNED DECIMAL ASCII NUMBER.
7154      ;*CALL
7155      ;*      MOV      NUMBER,-(SP)  ;;PUT BINARY NUMBER ON THE STACK
7156      ;*      JSR     PC,@#$SB2D    ;;CALL
7157      ;*      RETURN  ;;ADDRESS OF THE 1ST ASCII CHAR.IS ON THE STACK
7158

```

CZ  
CZ

```
7159
7160 035344 016637 000002 035374 $SB2D: MOV 2(SP),1$      ;;SAVE BINARY NUMBER
7161 035352 012746 035374      MOV #1$,-(SP)      ;;SET POINTER
7162 035356 004737 035150      JSR PC,@$SDB2D    ;;CALL DOUBLE LENGTH CONVERT
7163 035362 062716 000005      ADD #5,(SP)       ;;ONLY ALLOW FIVE CHARACTERS
7164 035366 012666 000002      MOV (SP)+,2(SP)  ;;PICKUP POINTER
7165 035372 000207                RTS PC           ;;RETURN
7166 035374 000000 000000      1$: .WORD 0,0
7167                .SBTTL TYPE NUMERICAL ASCIZ STRING SUPPRESS LEADING ZEROS
7168
7169                ;;*****
7170                ;;THIS ROUTINE IS USED TO TYPE AN ASCIZ NUMBER SUPPRESSING THE
7171                ;;LEADING NUMBERS.
7172                ;;CALL
7173                ;;* MOV #NUMADR,-(SP) ;;FIRST ADDRESS OF ASCIZ STRING
7174                ;;* JSR PC,@$SUPRS
7175
7176                $SUPRS: MOV R0,-(SP)      ;;SAVE R0
7177 035400 010046                MOV 4(SP),R0     ;;PICKUP THE POINTER
7178 035402 016600 000004      1$: TSTB (R0)     ;;TERMINATEOR?
7179 035406 105710                BEQ 2$          ;;BR IF YES
7180 035410 001403                CMPB #'0,(R0)+  ;;IS THIS AN ASCII '0' ?
7181 035412 122720 000060      BEQ 1$          ;;BR IF YES
7182 035416 001773                DEC R0          ;;BACKUP BY '1'
7183 035420 005300                MOV R0,3$      ;;SAVE FOR TYPING
7184 035422 010037 035430      TYPE          ;;GO TYPE
7185 035426 104401                .WORD 0        ;;ASCIZ POINTER GOES HERE
7186 035430 000000                MOV (SP)+,R0   ;;RESTORE R0
7187 035432 012600                MOV (SP)+,(SP) ;;RESTORE THE STACK
7188 035434 012616                RTS PC        ;;RETURN
7189 035436 000207                .SBTTL INTEGER MULTIPLY ROUTINE
7190
7191                ;;*****
7192                ;;CALL
7193                ;;* MOV MULTIPLER,-(SP)
7194                ;;* MOV MULTIPLICAND,-(SP)
7195                ;;* JSR PC,@$SMULT
7196                ;;* RETURN ;;PRODUCT IS ON THE STACK
7197
7198                ;;*
7199                ;;* STACK PRODUCT
7200                ;;* -----
7201                ;;* TOP LSB'S
7202                ;;* +2 MSB'S
7203
7204                $SMULT:
7205 035440 010046                MOV R0,-(SP)   ;;PUSH R0 ON STACK
7206 035442 010146                MOV R1,-(SP)   ;;PUSH R1 ON STACK
7207 035444 010246                MOV R2,-(SP)   ;;PUSH R2 ON STACK
7208 035446 005046                CLR -(SP)     ;;CLEAR THE SIGN KEY
7209 035450 016601 000012      MOV 12(SP),R1  ;;GET THE MULTIPLICAND
7210 035454 100002                BPL 1$        ;;BR IF PLUS
7211 035456 005216                INC (SP)      ;;SET THE SIGN KEY
7212 035460 005401                NEG R1        ;;MAKE THE MULTIPLICAND POSTIVE
7213 035462 016602 000014      1$: MOV 14(SP),R2 ;;GET THE MULTIPLIER
7214 035466 100002                BPL 2$        ;;BR IF PLUS
```

```

7215 035470 005316          DEC      (SP)          ;;UPDATE THE SIGN KEY
7216 035472 005402          NEG      R2           ;;MAKE THE MULTIPLIER POSTIVE
7217 035474 012746 000021  2$:      MOV      #17,-(SP)      ;;SET THE LOOP COUNT
7218 035500 005000          CLR      R0           ;;SETUP FOR THE MULTIPLY LOOP
7219 035502 103001          3$:      BCC      4$           ;;DON'T ADD IF MULTIPLICAND = 0
7220 035504 060200          ADD      R2,R0
7221 035506 006000          4$:      ROR      R0           ;;POSITION THE PARITIAL PRODUCT AND
7222 035510 006001          ROR      R1           ;;THE MULTIPLICAND
7223 035512 005316          DEC      (SP)          ;;HAS ALL BITS OF THE MULTIPLICAND BEEN DONE?
7224 035514 001372          BNE      3$           ;;BR IF NO
7225 035516 022616          CMP      (SP)+,(SP)    ;;SHOULD PRODUCT BE NEGATIVE?
7226 035520 001403          BEQ      5$           ;;GO TO EXIT IF NO
7227 035522 005400          NEG      R0           ;;YES--SO MAKE IT SO
7228 035524 005401          NEG      R1
7229 035526 005600          SBC      R0
7230 035530 005726          5$:      TST      (SP)+          ;;CLEAR SIGN INFO. OFF OF STACK
7231 035532 010066 000012  MOV      R0,12(SP)      ;;PUT THE PRODUCT ON THE STACK (MSB'S)
7232 035536 010166 000010  MOV      R1,10(SP)      ;;LSB'S
7233 035542 012602          MOV      (SP)+,R2      ;;POP STACK INTO R2
7234 035544 012601          MOV      (SP)+,R1      ;;POP STACK INTO R1
7235 035546 012600          MOV      (SP)+,R0      ;;POP STACK INTO R0
7236 035550 000207          RTS      PC
    
```

.SBTTL SAVE AND RESTORE R0-R5 ROUTINES

\*\*\*\*\*

```

; *SAVE R0-R5
; *CALL:
; * SAVREG
; *UPON RETURN FROM $SAVREG THE STACK WILL LOOK LIKE:
; *
; *TOP---(+16)
; * +2---(+18)
; * +4---R5
; * +6---R4
; * +8---R3
; *+10---R2
; *+12---R1
; *+14---R0
    
```

\$SAVREG:

```

7254 035552          MOV      R0,-(SP)      ;;PUSH R0 ON STACK
7255 035552 010046          MOV      R1,-(SP)      ;;PUSH R1 ON STACK
7256 035554 010146          MOV      R2,-(SP)      ;;PUSH R2 ON STACK
7257 035556 010246          MOV      R3,-(SP)      ;;PUSH R3 ON STACK
7258 035560 010346          MOV      R4,-(SP)      ;;PUSH R4 ON STACK
7259 035562 010446          MOV      R5,-(SP)      ;;PUSH R5 ON STACK
7260 035564 010546          MOV      22(SP),-(SP)  ;;SAVE PS OF MAIN FLOW
7261 035566 016646 000022  MOV      22(SP),-(SP)  ;;SAVE PC OF MAIN FLOW
7262 035572 016646 000022  MOV      22(SP),-(SP)  ;;SAVE PS OF CALL
7263 035576 016646 000022  MOV      22(SP),-(SP)  ;;SAVE PC OF CALL
7264 035602 016646 000022  MOV      22(SP),-(SP)
7265 035606 000002          RTI
    
```

; \*RESTORE R0-R5

```

; *CALL:
; * RESREG
$RESREG:
    
```

7270 035610

```

7271 035610 012666 000022      MOV      (SP)+,22(SP)      ;;RESTORE PC OF CALL
7272 035614 012666 000022      MOV      (SP)+,22(SP)      ;;RESTORE PS OF CALL
7273 035620 012666 000022      MOV      (SP)+,22(SP)      ;;RESTORE PC OF MAIN FLOW
7274 035624 012666 000022      MOV      (SP)+,22(SP)      ;;RESTORE PS OF MAIN FLOW
7275 035630 012605              MOV      (SP)+,R5          ;;POP STACK INTO R5
7276 035632 012604              MOV      (SP)+,R4          ;;POP STACK INTO R4
7277 035634 012603              MOV      (SP)+,R3          ;;POP STACK INTO R3
7278 035636 012602              MOV      (SP)+,R2          ;;POP STACK INTO R2
7279 035640 012601              MOV      (SP)+,R1          ;;POP STACK INTO R1
7280 035642 012600              MOV      (SP)+,R0          ;;POP STACK INTO R0
7281 035644 000002              RTI
    
```

.SBTTL TRAP DECODER

```

*****
*THIS ROUTINE WILL PICKUP THE LOWER BYTE OF THE 'TRAP' INSTRUCTION
*AND USE IT TO INDEX THROUGH THE TRAP TABLE FOR THE STARTING ADDRESS
*OF THE DESIRED ROUTINE. THEN USING THE ADDRESS OBTAINED IT WILL
*GO TO THAT ROUTINE.
    
```

```

7290 035646 010046 000002      $TRAP:  MOV      R0,-(SP)      ;;SAVE R0
7291 035650 016600 000002      MOV      2(SP),R0          ;;GET TRAP ADDRESS
7292 035654 005740              TST      -(R0)             ;;BACKUP BY 2
7293 035656 111000              MOV      (R0),R0          ;;GET RIGHT BYTE OF TRAP
7294 035660 006300              ASL      R0                ;;POSITION FOR INDEXING
7295 035662 016000 035702      MOV      $TRPAD(R0),R0     ;;INDEX TO TABLE
7296 035666 000200              RTS      R0                ;;GO TO ROUTINE
    
```

;;THIS IS USE TO HANDLE THE 'GETPRI' MACRO

```

7301 035670 011646 000002      $TRAP2: MOV      (SP),-(SP)    ;;MOVE THE PC DOWN
7302 035672 016666 000004 000002  MOV      4(SP),2(SP)      ;;MOVE THE PSW DOWN
7303 035700 000002              RTI                        ;;RESTORE THE PSW
    
```

.SBTTL TRAP TABLE

```

*THIS TABLE CONTAINS THE STARTING ADDRESSES OF THE ROUTINES CALLED
*BY THE 'TRAP' INSTRUCTION.
    
```

```

:      ROUTINE
:      -----
7312 035702 035670      $TRPAD: .WORD   $TRAP2
7313 035704 032132      $TYPE   ;;CALL=TYPE      TRAP+1(104401)  TTY TYPEOUT ROUTINE
7314 035706 033204      $TYPOC  ;;CALL=TYPOC      TRAP+2(104402)  TYPE OCTAL NUMBER (WITH LEADING ZEROS)
7315 035710 033160      $TYPOS  ;;CALL=TYPOS      TRAP+3(104403)  TYPE OCTAL NUMBER (NO LEADING ZEROS)
7316 035712 033220      $TYPON  ;;CALL=TYPON      TRAP+4(104404)  TYPE OCTAL NUMBER (AS PER LAST CALL)
7317 035714 032466      $TYPDS  ;;CALL=TYPDS      TRAP+5(104405)  TYPE DECIMAL NUMBER (WITH SIGN)
7318
7319 035716 034006      $GTSWR  ;;CALL=GTSWR      TRAP+6(104406)  GET SOFT-SWR SETTING
7320
7321 035720 033716      $CKSWR  ;;CALL=CKSWR      TRAP+7(104407)  TEST FOR CHANGE IN SOFT-SWR
7322 035722 034260      $RDCHR  ;;CALL=RDCHR      TRAP+10(104410) TTY TYPEIN CHARACTER ROUTINE
7323 035724 034350      $RDLIN  ;;CALL=RDLIN       TRAP+11(104411) TTY TYPEIN STRING ROUTINE
7324 035726 034670      $RDOCT  ;;CALL=RDOCT      TRAP+12(104412) READ AN OCTAL NUMBER FROM TTY
7325 035730 035552      $SAVREG ;;CALL=SAVREG      TRAP+13(104413) SAVE R0-R5 ROUTINE
7326 035732 035610      $RESREG ;;CALL=RESREG      TRAP+14(104414) RESTORE R0-R5 ROUTINE
    
```

CZR6JFO UNIBUS RK6 DR-3 MACY11 30(1046) 04-JAN-82 13:06 PAGE 141<sup>K 11</sup>  
CZR6JF.P11 04-JAN-82 12:48 TRAP TABLE

SEQ 0140

7327 035734 030360  
7328

SCOP1\$ ;:CALL=SCOP1 TRAP+15(104415) INTERNAL LOOP ON ERROR

CZ  
CZ

```
7329
7330
7331
7332
7333 035736 005015 047125 041111 MSG1: .ASCII <CR><LF>/UNIBUS RK6 DR-3/
7334 035744 051525 051040 033113
7335 035752 042040 026522 063
7336 035757 015 041412 051132 .ASCII <CR><LF>/CZR6JFO/<CR><LF>
7337 035764 045066 030106 005015
7338 035772 005015 025011 025052 .ASCII <CR><LF>/ ***** CAUTION *****/<CR><LF>
7339 036000 025052 041440 052501
7340 036006 044524 047117 025040
7341 036014 025052 025052 005015
7342 036022 005015 044124 051511 .ASCII <CR><LF>/THIS PROGRAM SHOULD BE HALTED ONLY BY TYPING CONTROL-C/
7343 036030 050040 047522 051107
7344 036036 046501 051440 047510
7345 036044 046125 020104 041040
7346 036052 020105 040510 052114
7347 036060 042105 047440 046116
7348 036066 020131 054502 052040
7349 036074 050131 047111 020107
7350 036102 047503 052116 047522
7351 036110 026514 103
7352 036113 015 047412 044124 .ASCII <CR><LF>/OTHERWISE CARTRIDGE FORMATTING AND, OR THE DEVICE/
7353 036120 051105 044527 042523
7354 036126 041440 051101 051124
7355 036134 042111 042507 043040
7356 036142 051117 040515 052124
7357 036150 047111 020107 047101
7358 036156 026104 047440 020122
7359 036164 044124 020105 042504
7360 036172 044526 042503
7361 036176 005015 040515 020131 .ASCII <CR><LF>/MAY BE LEFT IN AN UNDETERMINED STATE/<CR><LF>
7362 036204 042502 046040 043105
7363 036212 020124 047111 040440
7364 036220 020116 047125 042504
7365 036226 042524 046522 047111
7366 036234 042105 051440 040524
7367 036242 042524 005015
7368 036246 005015 047111 052111 .ASCII <CR><LF>/INITIALLY, DRIVES TO BE TESTED SHOULD HAVE:/<CR><LF>
7369 036254 040511 046114 026131
7370 036262 042040 044522 042526
7371 036270 020123 047524 041040
7372 036276 020105 042524 052123
7373 036304 042105 051440 047510
7374 036312 046125 020104 040510
7375 036320 042526 006472 012
7376 036325 015 040412 020056 .ASCII <CR><LF>/A. HEADS MANUALLY LOADED/
7377 036332 042510 042101 020123
7378 036340 040515 052516 046101
7379 036346 054514 046040 040517
7380 036354 042504 104
7381 036357 015 041012 020056 .ASCII <CR><LF>/B. CORRECT PORT SELECTED/
7382 036364 047503 051122 041505
7383 036372 020124 047520 052122
7384 036400 051440 046105 041505
```

7385	036406	042524	104		
7386	036411	015	041412	020056	.ASCII <CR><LF>/C. WRITE LOCK DISABLED/
7387	036416	051127	052111	020105	
7388	036424	047514	045503	042040	
7389	036432	051511	041101	042514	
7390	036440	104			
7391	036441	015	042012	020056	.ASCII <CR><LF>/D. DRIVE READY INDICATOR ON/<CR><LF>
7392	036446	051104	053111	020105	
7393	036454	042522	042101	020131	
7394	036462	047111	044504	040503	
7395	036470	047524	020122	047117	
7396	036476	005015			
7397	036500	005015	051104	053111	.ASCII <CR><LF>/DRIVES NOT TO BE TESTED MUST HAVE/
7398	036506	051505	047040	052117	
7399	036514	052040	020117	042502	
7400	036522	052040	051505	042524	
7401	036530	020104	052515	052123	
7402	036536	044040	053101	105	
7403	036543	015	041012	052117	.ASCIZ <CR><LF>/BOTH PORTS DESELECTED/<CR><LF>
7404	036550	020110	047520	052122	
7405	036556	020123	042504	042523	
7406	036564	042514	052103	042105	
7407	036572	005015	000		
7408					
7409	036575	015	041412	040510	MSG2: .ASCII <CR><LF>/CHANGE XXDP PACK/
7410	036602	043516	020105	054130	
7411	036610	050104	050040	041501	
7412	036616	113			
7413	036617	015	041412	042514	.ASCIZ <CR><LF>/CLEAR LOC 40,RESTART PROGRAM/
7414	036624	051101	046040	041517	
7415	036632	032040	026060	042522	
7416	036640	052123	051101	020124	
7417	036646	051120	043517	040522	
7418	036654	000115			
7419	036656	005015	051104	053111	MSG3: .ASCIZ <CR><LF>/DRIVE(S) TO BE TESTED: /
7420	036664	024105	024523	052040	
7421	036672	020117	042502	052040	
7422	036700	051505	042524	035104	
7423	036706	000040			
7424	036710	005015	054524	042520	MSG4: .ASCIZ <CR><LF>/TYPE BUSS ADDRESS IF NOT 177440: /
7425	036716	041040	051525	020123	
7426	036724	042101	051104	051505	
7427	036732	020123	043111	047040	
7428	036740	052117	030440	033467	
7429	036746	032064	035060	020040	
7430	036754	000			
7431	036755	015	052012	050131	MSG5: .ASCIZ <CR><LF>/TYPE CONTROLLER INTERRUPT VECTOR IF NOT 210: /
7432	036762	020105	047503	052116	
7433	036770	047522	046114	051105	
7434	036776	044440	052116	051105	
7435	037004	052522	052120	053040	
7436	037012	041505	047524	020122	
7437	037020	043111	047040	052117	
7438	037026	031040	030061	020072	
7439	037034	000040			
7440	037036	005015	047111	042524	MSG6: .ASCIZ <CR><LF>/INTERRUPT OCCURRED AT PC=/

7441	037044	051122	050125	020124		
7442	037052	041517	052503	051122		
7443	037060	042105	040440	020124		
7444	037066	041520	000075			
7445	037072	005015	051104	053111	MSG7:	.ASCIZ <CR><LF>/DRIVE 0 WILL NOT BE TESTED/
7446	037100	020105	020060	044527		
7447	037106	046114	047040	052117		
7448	037114	041040	020105	042524		
7449	037122	052123	042105	000		
7450	037127	015	044412	052116	MSG8:	.ASCIZ <CR><LF>/INTERLOCKS TEST/<CR><LF>
7451	037134	051105	047514	045503		
7452	037142	020123	042524	052123		
7453	037150	005015	000			
7454	037153	015	051012	046505	MSG9:	.ASCIZ <CR><LF>/REMOVE UNIT SELECT PLUG/
7455	037160	053117	020105	047125		
7456	037166	052111	051440	046105		
7457	037174	041505	020124	046120		
7458	037202	043525	000			
7459	037205	015	005012	044527	MSG10:	.ASCIZ <CR><LF><LF>/WILL TEST DRIVES:/
7460	037212	046114	052040	051505		
7461	037220	020124	051104	053111		
7462	037226	051505	000072			
7463	037232	005015	050012	053517	MSG11:	.ASCIZ <CR><LF><LF>/POWER UP RESTART TO TEST 1/<CR><LF>
7464	037240	051105	052440	020120		
7465	037246	042522	052123	051101		
7466	037254	020124	047524	052040		
7467	037262	051505	020124	006461		
7468	037270	000012				
7469	037272	047516	042516	005015	MSG12:	.ASCIZ /NONE/<CR><LF>
7470	037300	000				
7471	037301	015	047012	020117	MSG13:	.ASCII <CR><LF>/NO L OR P CLOCKS PRESENT/
7472	037306	020114	051117	050040		
7473	037314	041440	047514	045503		
7474	037322	020123	051120	051505		
7475	037330	047105	124			
7476	037333	015	040412	046114		.ASCIZ <CR><LF>/ALL TIMING TESTS BYPASSED/
7477	037340	052040	046511	047111		
7478	037346	020107	042524	052123		
7479	037354	020123	054502	040520		
7480	037362	051523	042105	000		
7481	037367	015	041012	050131	MSG14:	.ASCIZ <CR><LF>/BYPASSING DRIVE /
7482	037374	051501	044523	043516		
7483	037402	042040	044522	042526		
7484	037410	000040				
7485	037412	005015	042012	044522	MSG15:	.ASCIZ <CR><LF><LF>/DRIVE /
7486	037420	042526	000040			
7487	037424	005015	051104	053111	MSG16:	.ASCIZ <CR><LF>/DRIVE SERIAL NO. /
7488	037432	020105	042523	044522		
7489	037440	046101	047040	027117		
7490	037446	000040				
7491	037450	005015	040503	052122	MSG17:	.ASCIZ <CR><LF>/CARTRIDGE SERIAL NO. /
7492	037456	044522	043504	020105		
7493	037464	042523	044522	046101		
7494	037472	047040	027117	000040		
7495	037500	005015	047125	052111	MSG18:	.ASCIZ <CR><LF>/UNIT SELECT PLUG TEST/<CR><LF>
7496	037506	051440	046105	041505		



7497	037514	020124	046120	043525	
7498	037522	052040	051505	006524	
7499	037530	000012			
7500	037532	005015	047520	052122	MSG19: .ASCIZ <CR><LF>/PORT SELECTION TESTS/<CR><LF>
7501	037540	051440	046105	041505	
7502	037546	044524	047117	052040	
7503	037554	051505	051524	005015	
7504	037562	000			
7505	037563	015	040412	020103	MSG21: .ASCIZ <CR><LF>/AC LOW DETECTION TEST-PART 1/<CR><LF>
7506	037570	047514	020127	042504	
7507	037576	042524	052103	047511	
7508	037604	020116	042524	052123	
7509	037612	050055	051101	020124	
7510	037620	006461	000012		
7511	037624	005015	047516	026516	MSG22: .ASCIZ <CR><LF>/NON-EXECUTABLE FUNCTION (NXF) DETECTION TEST/<CR><LF>
7512	037632	054105	041505	052125	
7513	037640	041101	042514	043040	
7514	037646	047125	052103	047511	
7515	037654	020116	047050	043130	
7516	037662	020051	042504	042524	
7517	037670	052103	047511	020116	
7518	037676	042524	052123	005015	
7519	037704	000			
7520	037705	015	053412	044522	MSG24: .ASCIZ <CR><LF>/WRITE LOCK TEST/<CR><LF>
7521	037712	042524	046040	041517	
7522	037720	020113	042524	052123	
7523	037726	005015	000		
7524	037731	015	040412	020103	MSG26: .ASCIZ <CR><LF>/AC LOW DETECTION TEST-PART 2/<CR><LF>
7525	037736	047514	020127	042504	
7526	037744	042524	052103	047511	
7527	037752	020116	042524	052123	
7528	037760	050055	051101	020124	
7529	037766	006462	000012		
7530	037772	005015	040412	046114	MSG27: .ASCIZ <CR><LF><LF>/ALL DRIVES TESTED/<CR><LF><LF>
7531	040000	042040	044522	042526	
7532	040006	020123	042524	052123	
7533	040014	042105	005015	000012	
7534	040022	005015	052515	052114	MSG28: .ASCIZ <CR><LF>/MULTIPLE DRIVE DETECTION TEST/<CR><LF>
7535	040030	050111	042514	042040	
7536	040036	044522	042526	042040	
7537	040044	052105	041505	044524	
7538	040052	047117	052040	051505	
7539	040060	006524	000012		
7540	040064	005015	046120	040505	MSG29: .ASCIZ <CR><LF>/PLEASE WAIT, HEADS BEING LOADED/<CR><LF>
7541	040072	042523	053440	044501	
7542	040100	026124	044040	040505	
7543	040106	051504	041040	044505	
7544	040114	043516	046040	040517	
7545	040122	042504	006504	000012	
7546	040130	005015	042526	044522	MSG30: .ASCIZ <CR><LF>/VERIFY DOOR CAN NOW BE OPENED & LEAVE IT OPEN/<CR><LF>
7547	040136	054506	042040	047517	
7548	040144	020122	040503	020116	
7549	040152	047516	020127	042502	
7550	040160	047440	042520	042516	
7551	040166	020104	020046	042514	
7552	040174	053101	020105	052111	

7553	040202	047440	042520	006516	
7554	040210	000012			
7555	040212	005015	006477	000012	MSG31: .ASCIZ <CR><LF>/?/<CR><LF>
7556	040220	005015	042522	047515	MSG32: .ASCIZ <CR><LF>/REMOVE DISK PACK & CLOSE DOOR/
7557	040226	042526	042040	051511	
7558	040234	020113	040520	045503	
7559	040242	023040	041440	047514	
7560	040250	042523	042040	047517	
7561	040256	000122			
7562	040260	005015	042504	051120	MSG33: .ASCIZ <CR><LF>/DEPRESS 'RUN-STOP' SWITCH TO 'RUN' WHILE DOOR OPEN/
7563	040266	051505	020123	051047	
7564	040274	047125	051455	047524	
7565	040302	023520	051440	044527	
7566	040310	041524	020110	047524	
7567	040316	023440	052522	023516	
7568	040324	053440	044510	042514	
7569	040332	042040	047517	020122	
7570	040340	050117	047105	000	
7571	040345	015	053012	051105	MSG34: .ASCIZ <CR><LF>/VERIFY SPINDLE DOES NOT START & HEADS DO NOT LOAD/<CR><LF>
7572	040352	043111	020131	050123	
7573	040360	047111	046104	020105	
7574	040366	047504	051505	047040	
7575	040374	052117	051440	040524	
7576	040402	052122	023040	044040	
7577	040410	040505	051504	042040	
7578	040416	020117	047516	020124	
7579	040424	047514	042101	005015	
7580	040432	000			
7581	040433	015	042012	050105	MSG35: .ASCIZ <CR><LF>/DEPRESS 'RUN-STOP' SWITCH TO 'RUN' WHILE CARTRIDGE REMOVED/
7582	040440	042522	051523	023440	
7583	040446	052522	026516	052123	
7584	040454	050117	020047	053523	
7585	040462	052111	044103	052040	
7586	040470	020117	051047	047125	
7587	040476	020047	044127	046111	
7588	040504	020105	040503	052122	
7589	040512	044522	043504	020105	
7590	040520	042522	047515	042526	
7591	040526	000104			
7592	040530	005015	047111	042523	MSG36: .ASCIZ <CR><LF>/INSERT DISK PACK, DEPRESS 'RUN-STOP' TO 'RUN' & CLOSE DOOR/
7593	040536	052122	042040	051511	
7594	040544	020113	040520	045503	
7595	040552	020054	042504	051120	
7596	040560	051505	020123	051047	
7597	040566	047125	051455	047524	
7598	040574	023520	052040	020117	
7599	040602	051047	047125	020047	
7600	040610	020046	046103	051517	
7601	040616	020105	047504	051117	
7602	040624	000			
7603	040625	015	042012	050105	MSG37: .ASCIZ <CR><LF>/DEPRESS SPACE BAR WHEN FINISHED/<CR><LF>
7604	040632	042522	051523	051440	
7605	040640	040520	042503	041040	
7606	040646	051101	053440	042510	
7607	040654	020116	044506	044516	
7608	040662	044123	042105	005015	

7609	040670	000			
7610	040671	015	044412	051516	MSG38: .ASCII <CR><LF>/INSERT UNIT SELECT PLUGS. IN ANY ORDER/
7611	040676	051105	020124	047125	
7612	040704	052111	051440	046105	
7613	040712	041505	020124	046120	
7614	040720	043525	026123	044440	
7615	040726	020116	047101	020131	
7616	040734	051117	042504	122	
7617	040741	015	052012	042510	.ASCIZ <CR><LF>/THE PROGRAM WILL ECHO THE UNIT SELECT PLUG NUMBER/<CR><LF>
7618	040746	050040	047522	051107	
7619	040754	046501	053440	046111	
7620	040762	020114	041505	047510	
7621	040770	052040	042510	052440	
7622	040776	044516	020124	042523	
7623	041004	042514	052103	050040	
7624	041012	052514	020107	052516	
7625	041020	041115	051105	005015	
7626	041026	000			
7627	041027	015	042012	050105	MSG39: .ASCIZ <CR><LF>/DEPRESS CONTROL-E TO EXIT TEST/<CR><LF>
7628	041034	042522	051523	041440	
7629	041042	047117	051124	046117	
7630	041050	042455	052040	020117	
7631	041056	054105	052111	052040	
7632	041064	051505	006524	000012	
7633	041072	005015	047526	052514	MSG40: .ASCII <CR><LF>/VOLUME VALID NOT SET/
7634	041100	042515	053040	046101	
7635	041106	042111	047040	052117	
7636	041114	051440	052105		
7637	041120	005015	040515	042513	.ASCIZ <CR><LF>/MAKE SURE ORIGINAL UNIT SELECT PLUG IS INSERTED/<CR><LF>
7638	041126	051440	051125	020105	
7639	041134	051117	043511	047111	
7640	041142	046101	052440	044516	
7641	041150	020124	042523	042514	
7642	041156	052103	050040	052514	
7643	041164	020107	051511	044440	
7644	041172	051516	051105	042524	
7645	041200	006504	000012		
7646	041204	005015	042504	042523	MSG41: .ASCIZ <CR><LF>/DESELECT PORT IN USE & SELECT OPPOSITE PORT/<CR><LF>
7647	041212	042514	052103	050040	
7648	041220	051117	020124	047111	
7649	041226	052440	042523	023040	
7650	041234	051440	046105	041505	
7651	041242	020124	050117	047520	
7652	041250	044523	042524	050040	
7653	041256	051117	006524	000012	
7654	041264	005015	042504	042523	MSG42: .ASCIZ <CR><LF>/DESELECT WRONG PORT & SELECT CORRECT PORT/<CR><LF>
7655	041272	042514	052103	053440	
7656	041300	047522	043516	050040	
7657	041306	051117	020124	020046	
7658	041314	042523	042514	052103	
7659	041322	041440	051117	042522	
7660	041330	052103	050040	051117	
7661	041336	006524	000012		
7662	041342	005015	042504	042523	MSG43: .ASCIZ <CR><LF>/DESELECT BOTH PORTS/<CR><LF>
7663	041350	042514	052103	041040	
7664	041356	052117	020110	047520	

7665	041364	052122	006523	000012	
7666	041372	005015	042523	042514	MSG44: .ASCIZ <CR><LF>/SELECT CORRECT PORT/<CR><LF>
7667	041400	052103	041440	051117	
7668	041406	042522	052103	050040	
7669	041414	051117	006524	000012	
7670	041422	005015	047503	051122	MSG45: .ASCIZ <CR><LF>/CORRECT PORT NOT SELECTED, TRY AGAIN/<CR><LF>
7671	041430	041505	020124	047520	
7672	041436	052122	047040	052117	
7673	041444	051440	046105	041505	
7674	041452	042524	026104	052040	
7675	041460	054522	040440	040507	
7676	041466	047111	005015	000	
7677	041473	015	053012	051105	MSG47: .ASCIZ <CR><LF>/VERIFY DOOR CANNOT BE OPENED (DO NOT FORCE)/<CR><LF>
7678	041500	043111	020131	047504	
7679	041506	051117	041440	047101	
7680	041514	047516	020124	042502	
7681	041522	047440	042520	042516	
7682	041530	020104	042050	020117	
7683	041536	047516	020124	047506	
7684	041544	041522	024505	005015	
7685	041552	000			
7686	041553	015	052012	051125	MSG49: .ASCIZ <CR><LF>/TURN OFF AC POWER FROM BEHIND THE RK06/<CR><LF>
7687	041560	020116	043117	020106	
7688	041566	041501	050040	053517	
7689	041574	051105	043040	047522	
7690	041602	020115	042502	044510	
7691	041610	042116	052040	042510	
7692	041616	051040	030113	006466	
7693	041624	000012			
7694	041626	005015	040527	052111	MSG50: .ASCIZ <CR><LF>/WAIT 30 SEC & SWITCH AC POWER BACK ON/
7695	041634	031440	020060	042523	
7696	041642	020103	020046	053523	
7697	041650	052111	044103	040440	
7698	041656	020103	047520	042527	
7699	041664	020122	040502	045503	
7700	041672	047440	000116		
7701	041676	005015	047515	042515	MSG51: .ASCII <CR><LF>/MOMENTARILY REMOVE & INSERT THE SAME UNIT SELECT PLUG/
7702	041704	052116	051101	046111	
7703	041712	020131	042522	047515	
7704	041720	042526	023040	044440	
7705	041726	051516	051105	020124	
7706	041734	044124	020105	040523	
7707	041742	042515	052440	044516	
7708	041750	020124	042523	042514	
7709	041756	052103	050040	052514	
7710	041764	107			
7711	041765	015	052012	020117	.ASCIZ <CR><LF>/TO RESET VOLUME VALID/<CR><LF>
7712	041772	042522	042523	020124	
7713	042000	047526	052514	042515	
7714	042006	053040	046101	042111	
7715	042014	005015	000		
7716	042017	015	042012	050105	MSG52: .ASCII <CR><LF>/DEPRESS SPACE TO DO TEST/
7717	042024	042522	051523	051440	
7718	042032	040520	042503	052040	
7719	042040	020117	047504	052040	
7720	042046	051505	124		

7721	042051	015	047412	020122		.ASCIZ <CR><LF>/OR CONTROL-E TO BYPASS ENTIRE TEST/<CR><LF>
7722	042056	047503	052116	047522		
7723	042064	026514	020105	047524		
7724	042072	041040	050131	051501		
7725	042100	020123	047105	044524		
7726	042106	042522	052040	051505		
7727	042114	006524	000012			
7728	042120	005015	044504	040523	MSG53:	.ASCIZ <CR><LF>/DISABLE THE WRITE LOCK SWITCH, VERIFY LIGHT GOES OFF/<CR><LF>
7729	042126	046102	020105	044124		
7730	042134	020105	051127	052111		
7731	042142	020105	047514	045503		
7732	042150	051440	044527	041524		
7733	042156	026110	053040	051105		
7734	042164	043111	020131	044514		
7735	042172	044107	020124	047507		
7736	042200	051505	047440	043106		
7737	042206	005015	000			
7738	042211	015	042412	040516	MSG54:	.ASCIZ <CR><LF>/ENABLE THE WRITE LOCK SWITCH, VERIFY LIGHT GOES ON/<CR><LF>
7739	042216	046102	020105	044124		
7740	042224	020105	051127	052111		
7741	042232	020105	047514	045503		
7742	042240	051440	044527	041524		
7743	042246	026110	053040	051105		
7744	042254	043111	020131	044514		
7745	042262	044107	020124	047507		
7746	042270	051505	047440	006516		
7747	042276	000012				
7748	042300	005015	054105	052111	MSG55:	.ASCIZ <CR><LF>/EXIT TEST WITH ORIGINAL UNIT SELECT PLUG NO. /
7749	042306	052040	051505	020124		
7750	042314	044527	044124	047440		
7751	042322	044522	044507	040516		
7752	042330	020114	047125	052111		
7753	042336	051440	046105	041505		
7754	042344	020124	046120	043525		
7755	042352	047040	027117	000040		
7756	042360	005015	046101	043511	MSG56:	.ASCII <CR><LF>/ALIGNMENT CARTRIDGE USED/
7757	042366	046516	047105	020124		
7758	042374	040503	052122	044522		
7759	042402	043504	020105	051525		
7760	042410	042105				
7761	042412	005015	051120	043517		.ASCII <CR><LF>/PROGRAM WILL BYPASS THE WRITE LOCK TEST/
7762	042420	040522	020115	044527		
7763	042426	046114	041040	050131		
7764	042434	051501	020123	044124		
7765	042442	020105	051127	052111		
7766	042450	020105	047514	045503		
7767	042456	052040	051505	124		
7768	042463	015	040412	042116		.ASCIZ <CR><LF>/AND READ-WRITE DATA PORTION OF AC LOW DETECTION TEST-PART 2/<CR>
7769	042470	051040	040505	026504		
7770	042476	051127	052111	020105		
7771	042504	040504	040524	050040		
7772	042512	051117	044524	047117		
7773	042520	047440	020106	041501		
7774	042526	046040	053517	042040		
7775	042534	052105	041505	044524		
7776	042542	047117	052040	051505		

7777 042550 026524 040520 052122  
7778 042556 031040 005015 000012  
7779 042564 005015 042526 044522  
7780 042572 054506 041040 052101  
7781 042600 042524 054522 051040  
7782 042606 052105 040522 052103  
7783 042614 043040 047125 052103  
7784 042622 047511 040516 006514  
7785 042630 000012  
7786 042632 005015 047514 042101  
7787 042640 044040 040505 051504  
7788 042646 047440 020116 046101  
7789 042654 020114 051104 053111  
7790 042662 051505 052040 020117  
7791 042670 042502 052040 051505  
7792 042676 042524 020104 047506  
7793 042704 020122 042115 006523  
7794 042712 000012  
7795 042714 005015 047111 042523  
7796 042722 052122 051440 046501  
7797 042730 020105 047125 052111  
7798 042736 051440 046105 041505  
7799 042744 020124 046120 043525  
7800 042752 047040 046525 042502  
7801 042760 020122 047111 040440  
7802 042766 054516 031040 042040  
7803 042774 044522 042526 020123  
7804 043002 047524 041040 020105  
7805 043010 042524 052123 042105  
7806 043016 000  
7807 043017 015 044412 051516  
7808 043024 051105 020124 047503  
7809 043032 051122 041505 020124  
7810 043040 047125 052111 051440  
7811 043046 046105 041505 020124  
7812 043054 046120 043525 020123  
7813 043062 020046 047514 042101  
7814 043070 044040 040505 051504  
7815 043076 005015 047117 050040  
7816 043104 042522 044526 052517  
7817 043112 020123 020062 051104  
7818 043120 053111 051505 005015  
7819 043126 000  
7820 043127 015 046412 046125  
7821 043134 044524 046120 020105  
7822 043142 051104 053111 051505  
7823 043150 043040 052517 042116  
7824 043156 047440 020116 051104  
7825 043164 053111 020105 047516  
7826 043172 020056 000  
7827 043175 015 041012 050131  
7828 043202 051501 044523 043516  
7829 043210 046440 046125 027124  
7830 043216 042040 044522 042526  
7831 043224 051440 046105 041505  
7832 043232 020124 042524 052123

MSG57: .ASCIZ <CR><LF>/VERIFY BATTERY RETRACT FUNCTIONAL/<CR><LF>

MSG58: .ASCIZ <CR><LF>/LOAD HEADS ON ALL DRIVES TO BE TESTED FOR MDS/<CR><LF>

MSG59: .ASCIZ <CR><LF>/INSERT SAME UNIT SELECT PLUG NUMBER IN ANY 2 DRIVES TO BE TESTE

MSG60: .ASCII <CR><LF>/INSERT CORRECT UNIT SELECT PLUGS & LOAD HEADS/

.ASCIZ <CR><LF>/ON PREVIOUS 2 DRIVES/<CR><LF>

MSG61: .ASCIZ <CR><LF>/MULTIPLE DRIVES FOUND ON DRIVE NO. /

MSG62: .ASCII <CR><LF>/BYPASSING MULT. DRIVE SELECT TEST/

7833	043240	005015	047117	054514		.ASCIZ <CR><LF>/ONLY 1 DRIVE PRESENT/<CR><LF>
7834	043246	030440	042040	044522		
7835	043254	042526	050040	042522		
7836	043262	042523	052116	005015		
7837	043270	000				
7838	043271	015	042012	050105	MSG63:	.ASCII <CR><LF>/DEPRESS SPACE BAR TO DO ANOTHER 2 DRIVES/
7839	043276	042522	051523	051440		
7840	043304	040520	042503	041040		
7841	043312	051101	052040	020117		
7842	043320	047504	040440	047516		
7843	043326	044124	051105	031040		
7844	043334	042040	044522	042526		
7845	043342	123				
7846	043343	015	047412	020122		.ASCII <CR><LF>/OR CONTROL-E TO EXIT TEST/
7847	043350	047503	052116	047522		
7848	043356	026514	020105	047524		
7849	043364	042440	044530	020124		
7850	043372	042524	052123			
7851	043376	005015	044450	051516		.ASCIZ <CR><LF>/((INSERT CORRECT UNIT SELECT PLUGS BEFORE EXITING))<CR><LF>
7852	043404	051105	020124	047503		
7853	043412	051122	041505	020124		
7854	043420	047125	052111	051440		
7855	043426	046105	041505	020124		
7856	043434	046120	043525	020123		
7857	043442	042502	047506	042522		
7858	043450	042440	044530	044524		
7859	043456	043516	006451	000012		
7860	043464	005015	042504	051120	MSG65:	.ASCIZ <CR><LF>/DEPRESS 'RUN-STOP' SWITCH TO 'STOP'/'
7861	043472	051505	020123	051047		
7862	043500	047125	051455	047524		
7863	043506	023520	051440	044527		
7864	043514	041524	020110	047524		
7865	043522	023440	052123	050117		
7866	043530	000047				
7867	043532	005015	042504	042523	MSG67:	.ASCIZ <CR><LF>/DESELECT PORT SWITCH ON ALL OTHER DRIVES/<CR><LF>
7868	043540	042514	052103	050040		
7869	043546	051117	020124	053523		
7870	043554	052111	044103	047440		
7871	043562	020116	046101	020114		
7872	043570	052117	042510	020122		
7873	043576	051104	053111	051505		
7874	043604	005015	000			
7875	043607	015	053012	051105	MSG68:	.ASCIZ <CR><LF>/VERIFY BOTH DRIVES UNLOADED/<CR><LF>
7876	043614	043111	020131	047502		
7877	043622	044124	042040	044522		
7878	043630	042526	020123	047125		
7879	043636	047514	042101	042105		
7880	043644	005015	000			
7881	043647	015	042012	050105	MSG69:	.ASCIZ <CR><LF>/DEPRESS SPACE WHEN 'READY' LIGHT GOES ON/<CR><LF>
7882	043654	042522	051523	051440		
7883	043662	040520	042503	053440		
7884	043670	042510	020116	051047		
7885	043676	040505	054504	020047		
7886	043704	044514	044107	020124		
7887	043712	047507	051505	047440		
7888	043720	006516	000012			

7889	043724	005015	042526	044522	MSG70: .ASCIZ <CR><LF>/VERIFY HEADS LOAD/<CR><LF>
7890	043732	054506	044040	040505	
7891	043740	051504	046040	040517	
7892	043746	006504	000012		
7893	043752	005015	042523	042514	MSG71: .ASCIZ <CR><LF>/SELECT CORRECT PORT SWITCH ON ALL OTHER DRIVES/<CR><LF>
7894	043760	052103	041440	051117	
7895	043766	042522	052103	050040	
7896	043774	051117	020124	053523	
7897	044002	052111	044103	047440	
7898	044010	020116	046101	020114	
7899	044016	052117	042510	020122	
7900	044024	051104	053111	051505	
7901	044032	005015	000		
7902	044035	015	040412	047502	MSG72: .ASCIZ <CR><LF>/ABORTING BALANCE OF TESTS/
7903	044042	052122	047111	020107	
7904	044050	040502	040514	041516	
7905	044056	020105	043117	052040	
7906	044064	051505	051524	000	
7907					
7908	044071	015	050012	047522	MSG74: .ASCIZ <CR><LF>/PROGRAM ABORT PENDING...PLEASE WAIT/
7909	044076	051107	046501	040440	
7910	044104	047502	052122	050040	
7911	044112	047105	044504	043516	
7912	044120	027056	050056	042514	
7913	044126	051501	020105	040527	
7914	044134	052111	000		
7915	044137	015	044012	046101	MSG75: .ASCIZ <CR><LF>/HALT PENDING...PLEASE WAIT/
7916	044144	020124	042520	042116	
7917	044152	047111	027107	027056	
7918	044160	046120	040505	042523	
7919	044166	053440	044501	000124	
7920	044174	005015	051120	043517	MSG76: .ASCIZ <CR><LF>/PROGRAM ABORTED/
7921	044202	040522	020115	041101	
7922	044210	051117	042524	000104	
7923	044216	005015	050103	020125	MSG77: .ASCIZ <CR><LF>/CPU HALTED/
7924	044224	040510	052114	042105	
7925	044232	000			
7926	044233	015	051412	051117	MSG100: .ASCII <CR><LF>/SORRY, WRITE LOCK SHOULD NOT BE DEPRESSED/
7927	044240	054522	020054	051127	
7928	044246	052111	020105	047514	
7929	044254	045503	051440	047510	
7930	044262	046125	020104	047516	
7931	044270	020124	042502	042040	
7932	044276	050105	042522	051523	
7933	044304	042105			
7934	044306	005015	044127	046111	.ASCII <CR><LF>/WHILE ON TRACK 2 SECTORS 19, 20 OR 21/
7935	044314	020105	047117	052040	
7936	044322	040522	045503	031040	
7937	044330	051440	041505	047524	
7938	044336	051522	030440	026071	
7939	044344	031040	020060	051117	
7940	044352	031040	061		
7941	044355	015	050012	042514	.ASCIZ <CR><LF>/PLEASE TRY AGAIN/<CR><LF>
7942	044362	051501	020105	051124	
7943	044370	020131	043501	044501	
7944	044376	006516	000012		



```
7945
7946 .SBTTL ERROR MESSAGES
7947
7948 044402 005015 051105 047522 EM1: .ASCIZ <CR><LF>/ERROR, ONLY 0 THRU 7 ALLOWED, TRY AGAIN/<CR><LF>
7949 044410 026122 047440 046116
7950 044416 020131 020060 044124
7951 044424 052522 033440 040440
7952 044432 046114 053517 042105
7953 044440 020054 051124 020131
7954 044446 043501 044501 006516
7955 044454 000012
7956 044456 051104 053111 020105 EM2: .ASCIZ /DRIVE # IN RKCS2 CANNOT BE READ BACK CORRECTLY IN RKMR2/
7957 044464 020043 047111 051040
7958 044472 041513 031123 041440
7959 044500 047101 047516 020124
7960 044506 042502 051040 040505
7961 044514 020104 040502 045503
7962 044522 041440 051117 042522
7963 044530 052103 054514 044440
7964 044536 020116 045522 051115
7965 044544 000062
7966 044546 005015 041101 051117 EM3: .ASCIZ <CR><LF>/ABORT TESTS...UNEXPECTED TIME OUT AT PC=/
7967 044554 020124 042524 052123
7968 044562 027123 027056 047125
7969 044570 054105 042520 052103
7970 044576 042105 052040 046511
7971 044604 020105 052517 020124
7972 044612 052101 050040 036503
7973 044620 000
7974 044621 015 040412 047502 EM4: .ASCIZ <CR><LF>/ABORT TESTS...UNEXPECTED INTERRUPT AT PC=/
7975 044626 052122 052040 051505
7976 044634 051524 027056 052456
7977 044642 042516 050130 041505
7978 044650 042524 020104 047111
7979 044656 042524 051122 050125
7980 044664 020124 052101 050040
7981 044672 036503 000
7982 044675 115 051504 051440 EM5: .ASCIZ /MDS SET IN RKCS2/
7983 044702 052105 044440 020116
7984 044710 045522 051503 000062
7985 044716 043125 020105 042523 EM6: .ASCIZ /UFE SET IN RKCS2/
7986 044724 020124 047111 051040
7987 044732 041513 031123 000
7988 044737 104 040522 044440 EM7: .ASCIZ /DRA IN RKDS & NED IN RKCS2 RESET; WRONG PORT SELECTED?/
7989 044744 020116 045522 051504
7990 044752 023040 047040 042105
7991 044760 044440 020116 045522
7992 044766 051503 020062 042522
7993 044774 042523 035524 053440
7994 045002 047522 043516 050040
7995 045010 051117 020124 042523
7996 045016 042514 052103 042105
7997 045024 000077
7998 045026 051104 053111 020105 EM8: .ASCIZ /DRIVE PRESENT BUT NOT SPECIFIED BY OPERATOR/
7999 045034 051120 051505 047105
8000 045042 020124 052502 020124
```

8001	045050	047516	020124	050123	
8002	045056	041505	043111	042511	
8003	045064	020104	054502	047440	
8004	045072	042520	040522	047524	
8005	045100	000122			
8006	045102	051104	053111	020105	EM9: .ASCIZ /DRIVE NOT PRESENT BUT SPECIFIED BY OPERATOR/
8007	045110	047516	020124	051120	
8008	045116	051505	047105	020124	
8009	045124	052502	020124	050123	
8010	045132	041505	043111	042511	
8011	045140	020104	054502	047440	
8012	045146	042520	040522	047524	
8013	045154	000122			
8014	045156	041101	051117	020124	EM10: .ASCIZ /ABORT TESTS...CANNOT REFERENCE CONTROLLER REGISTER/
8015	045164	042524	052123	027123	
8016	045172	027056	040503	047116	
8017	045200	052117	051040	043105	
8018	045206	051105	047105	042503	
8019	045214	041440	047117	051124	
8020	045222	046117	042514	020122	
8021	045230	042522	044507	052123	
8022	045236	051105	000		
8023	045241	104	040522	044440	EM11: .ASCIZ /DRA IN RKDS & NED IN RKCS2 BOTH SET/
8024	045246	020116	045522	051504	
8025	045254	023040	047040	042105	
8026	045262	044440	020116	045522	
8027	045270	051503	020062	047502	
8028	045276	044124	051440	052105	
8029	045304	000			
8030	045305	103	047117	051124	EM12: .ASCIZ /CONTROLLER NOT READY IN RKCS1/
8031	045312	046117	042514	020122	
8032	045320	047516	020124	042522	
8033	045326	042101	020131	047111	
8034	045334	051040	041513	030523	
8035	045342	000			
8036	045343	116	020117	052101	EM13: .ASCIZ /NO ATTN IN RKASOF/
8037	045350	047124	044440	020116	
8038	045356	045522	051501	043117	
8039	045364	000			
8040	045365	127	047522	043516	EM14: .ASCIZ /WRONG ATTN IN RKASOF/
8041	045372	040440	052124	020116	
8042	045400	047111	051040	040513	
8043	045406	047523	000106		
8044	045412	051104	054504	047040	EM15: .ASCIZ /DRDY NOT CLEARED IN RKMR2/
8045	045420	052117	041440	042514	
8046	045426	051101	042105	044440	
8047	045434	020116	045522	051115	
8048	045442	000062			
8049	045444	051504	020103	047516	EM16: .ASCIZ /DSC NOT SET IN RKMR2/
8050	045452	020124	042523	020124	
8051	045460	047111	051040	046513	
8052	045466	031122	000		
8053	045471	115	051505	040523	EM17: .ASCIZ /MESSAGE A0 ERROR/
8054	045476	042507	040440	020060	
8055	045504	051105	047522	000122	
8056	045512	042515	051523	043501	EM18: .ASCIZ /MESSAGE B0 ERROR/

8057	045520	020105	030102	042440	
8058	045526	051122	051117	000	
8059	045533	115	051505	040523	EM19: .ASCIZ /MESSAGE A1 ERROR/
8060	045540	042507	040440	020061	
8061	045546	051105	047522	000122	
8062	045554	042515	051523	043501	EM20: .ASCIZ /MESSAGE B1 ERROR/
8063	045562	020105	030502	042440	
8064	045570	051122	051117	000	
8065	045575	103	051105	020122	EM21: .ASCIZ /CERR SET IN RKCS1/
8066	045602	042523	020124	047111	
8067	045610	051040	041513	030523	
8068	045616	000			
8069	045617	104	047517	020122	EM22: .ASCIZ /DOOR STATUS IN RKMR2 NOT CLEARED/
8070	045624	052123	052101	051525	
8071	045632	044440	020116	045522	
8072	045640	051115	020062	047516	
8073	045646	020124	046103	040505	
8074	045654	042522	000104		
8075	045660	050123	047111	046104	EM23: .ASCIZ /SPINDLE ON SET IN RKMR2/
8076	045666	020105	047117	051440	
8077	045674	052105	044440	020116	
8078	045702	045522	051115	000062	
8079	045710	047526	052514	042515	EM24: .ASCIZ /VOLUME VALID NOT SET IN RKMR2/
8080	045716	053040	046101	042111	
8081	045724	047040	052117	051440	
8082	045732	052105	044440	020116	
8083	045740	045522	051115	000062	
8084	045746	040503	052122	044522	EM25: .ASCIZ /CARTRIDGE STATUS IN RKMR2 NOT CLEARED/
8085	045754	043504	020105	052123	
8086	045762	052101	051525	044440	
8087	045770	020116	045522	051115	
8088	045776	020062	047516	020124	
8089	046004	046103	040505	042522	
8090	046012	000104			
8091	046014	047526	052514	042515	EM26: .ASCIZ /VOLUME VALID NOT CLEARED IN RKMR2/
8092	046022	053040	046101	042111	
8093	046030	047040	052117	041440	
8094	046036	042514	051101	042105	
8095	046044	044440	020116	045522	
8096	046052	051115	000062		
8097	046056	042516	020104	047516	EM27: .ASCIZ /NED NOT SET IN RKCS2/
8098	046064	020124	042523	020124	
8099	046072	047111	051040	041513	
8100	046100	031123	000		
8101	046103	126	046117	046525	EM28: .ASCIZ /VOLUME VALID SET IN RKMR2/
8102	046110	020105	040526	044514	
8103	046116	020104	042523	020124	
8104	046124	047111	051040	046513	
8105	046132	031122	000		
8106	046135	104	041523	047040	EM29: .ASCIZ /DSC NOT SET IN RKMR2/
8107	046142	052117	051440	052105	
8108	046150	044440	020116	045522	
8109	046156	051115	000062		
8110	046162	052101	047124	047040	EM30: .ASCIZ /ATTN NOT RESET IN RKASOF/
8111	046170	052117	051040	051505	
8112	046176	052105	044440	020116	

8113	046204	045522	051501	043117	
8114	046212	000			
8115	046213	116	042105	047040	EM31: .ASCIZ /NED NOT CLEARED IN RKCS2/
8116	046220	052117	041440	042514	
8117	046226	051101	042105	044440	
8118	046234	020116	045522	051503	
8119	046242	000062			
8120	046244	050123	047111	046104	EM32: .ASCIZ /SPINDLE ON NOT SET IN RKMR2/
8121	046252	020105	047117	047040	
8122	046260	052117	051440	052105	
8123	046266	044440	020116	045522	
8124	046274	051115	000062		
8125	046300	051104	053111	020105	EM33: .ASCIZ /DRIVE NOT READY IN RKMR2/
8126	046306	047516	020124	042522	
8127	046314	042101	020131	047111	
8128	046322	051040	046513	031122	
8129	046330	000			
8130	046331	104	047517	020122	EM34: .ASCIZ /DOOR STATUS BIT NOT SET IN RKMR2/
8131	046336	052123	052101	051525	
8132	046344	041040	052111	047040	
8133	046352	052117	051440	052105	
8134	046360	044440	020116	045522	
8135	046366	051115	000062		
8136	046372	042510	042101	020123	EM35: .ASCIZ /HEADS HOME NOT SET IN RKMR2/
8137	046400	047510	042515	047040	
8138	046406	052117	051440	052105	
8139	046414	044440	020116	045522	
8140	046422	051115	000062		
8141	046426	041501	046040	053517	EM37: .ASCIZ /AC LOW NOT SET IN RKMR3/
8142	046434	047040	052117	051440	
8143	046442	052105	044440	020116	
8144	046450	045522	051115	000063	
8145	046456	042516	020104	047516	EM42: .ASCIZ /NED NOT SET IN RKCS2/
8146	046464	020124	042523	020124	
8147	046472	047111	051040	041513	
8148	046500	031123	000		
8149	046503	101	046103	020117	EM43: .ASCIZ /ACLO NOT CLEARED IN RKMR3/
8150	046510	047516	020124	046103	
8151	046516	040505	042522	020104	
8152	046524	047111	051040	046513	
8153	046532	031522	000		
8154	046535	126	046117	046525	EM44: .ASCIZ /VOLUME VALID NOT CLEARED IN RKMR2/
8155	046542	020105	040526	044514	
8156	046550	020104	047516	020124	
8157	046556	046103	040505	042522	
8158	046564	020104	047111	051040	
8159	046572	046513	031122	000	
8160	046577	126	046117	046525	EM45: .ASCIZ /VOLUME VALID SET IN RKMR2 AFTER HEADS LOADED/
8161	046604	020105	040526	044514	
8162	046612	020104	042523	020124	
8163	046620	047111	051040	046513	
8164	046626	031122	040440	052106	
8165	046634	051105	044040	040505	
8166	046642	051504	046040	040517	
8167	046650	042504	000104		
8168	046654	047516	026516	054105	EM46: .ASCIZ /NON-EXECUTABLE FUNCTION (NXF) NOT SET IN RKMR3/

8169	046662	041505	052125	041101	
8170	046670	042514	043040	047125	
8171	046676	052103	047511	020116	
8172	046704	047050	043130	020051	
8173	046712	047516	020124	042523	
8174	046720	020124	047111	051040	
8175	046726	046513	031522	000	
8176	046733	103	046131	047111	EM47: .ASCIZ /CYLINDER ADDRESS CHANGED FROM 0/
8177	046740	042504	020122	042101	
8178	046746	051104	051505	020123	
8179	046754	044103	047101	042507	
8180	046762	020104	051106	046517	
8181	046770	030040	000		
8182	046773	127	044522	042524	EM48: .ASCIZ /WRITE LOCK IN RKMR2 NOT CLEARED/
8183	047000	046040	041517	020113	
8184	047006	047111	051040	046513	
8185	047014	031122	047040	052117	
8186	047022	041440	042514	051101	
8187	047030	042105	000		
8188	047033	127	044522	042524	EM49: .ASCIZ /WRITE LOCK IN RKMR2 NOT SET/
8189	047040	046040	041517	020113	
8190	047046	047111	051040	046513	
8191	047054	031122	047040	052117	
8192	047062	051440	052105	000	
8193	047067	127	044522	042524	EM50: .ASCIZ /WRITE LOCK ERROR IN RKMR3 NOT SET/
8194	047074	046040	041517	020113	
8195	047102	051105	047522	020122	
8196	047110	047111	051040	046513	
8197	047116	031522	047040	052117	
8198	047124	051440	052105	000	
8199	047131	127	044522	042524	EM51: .ASCIZ /WRITE LOCK DID NOT OCCUR AT SECTOR BOUNDRY/
8200	047136	046040	041517	020113	
8201	047144	044504	020104	047516	
8202	047152	020124	041517	052503	
8203	047160	020122	052101	051440	
8204	047166	041505	047524	020122	
8205	047174	047502	047125	051104	
8206	047202	000131			
8207	047204	047125	020123	047516	EM52: .ASCIZ /UNS NOT SET IN RKMR3/
8208	047212	020124	042523	020124	
8209	047220	047111	051040	046513	
8210	047226	031522	000		
8211					
8212	047231	125	046116	040517	EM53: .ASCIZ /UNLOAD NOT SET IN RKMR2/
8213	047236	020104	047516	020124	
8214	047244	042523	020124	047111	
8215	047252	051040	046513	031122	
8216	047260	000			
8217	047261	103	047101	047516	EM54: .ASCIZ /CANNOT FIND MULT. DRIVE SELECT IN RKCS2/
8218	047266	020124	044506	042116	
8219	047274	046440	046125	027124	
8220	047302	042040	044522	042526	
8221	047310	051440	046105	041505	
8222	047316	020124	047111	051040	
8223	047324	041513	031123	000	
8224	047331	101	052124	020116	EM55: .ASCIZ /ATTN NOT CLEARED IN RKASOF/

8225	047336	047516	020124	046103	
8226	047344	040505	042522	020104	
8227	047352	047111	051040	040513	
8228	047360	047523	000106		
8229	047364	047125	054105	042520	EM56: .ASCIZ /UNEXPECTED MEMORY PARITY ERROR TRAP/
8230	047372	052103	042105	046440	
8231	047400	046505	051117	020131	
8232	047406	040520	044522	054524	
8233	047414	042440	051122	051117	
8234	047422	052040	040522	000120	
8235	047430	042503	051122	044440	EM57: .ASCIZ /CERR IN RKCS1 NOT SET/
8236	047436	020116	045522	051503	
8237	047444	020061	047516	020124	
8238	047452	042523	000124		
8239	047456	040520	044522	054524	EM58: .ASCIZ /PARITY NOT SET IN RKMR3/
8240	047464	047040	052117	051440	
8241	047472	052105	044440	020116	
8242	047500	045522	051115	000063	
8243	047506	052103	020117	042523	EM59: .ASCIZ /CTO SET IN RKCS1/
8244	047514	020124	047111	051040	
8245	047522	041513	030523	000	
8246	047527	116	043130	042040	EM61: .ASCIZ /NXF DID NOT SET FAULT/
8247	047534	042111	047040	052117	
8248	047542	051440	052105	043040	
8249	047550	052501	052114	000	
8250	047555	104	052114	051440	EM63: .ASCIZ /DLT SET IN RKCS2/
8251	047562	052105	044440	020116	
8252	047570	045522	051503	000062	
8253	047576	045522	041504	023040	EM64: .ASCII /RKDC & RKDA INDICATE THAT WRITE CHECK ERROR/
8254	047604	051040	042113	020101	
8255	047612	047111	044504	040503	
8256	047620	042524	052040	040510	
8257	047626	020124	051127	052111	
8258	047634	020105	044103	041505	
8259	047642	020113	051105	047522	
8260	047650	122			
8261	047651	015	047412	041503	.ASCIZ <CR><LF>/OCCURRED AT CYL 411, TRACK 2, SECTOR 21/
8262	047656	051125	042522	020104	
8263	047664	052101	041440	046131	
8264	047672	032040	030461	020054	
8265	047700	051124	041501	020113	
8266	047706	026062	051440	041505	
8267	047714	047524	020122	030462	
8268	047722	000			
8269	047723	116	042105	051440	EM67: .ASCIZ /NED SET IN RKCS2/
8270	047730	052105	044440	020116	
8271	047736	045522	051503	000062	
8272	047744	040503	047116	052117	EM68: .ASCIZ /CANNOT READ BAD SECTOR INFORMATION/
8273	047752	051040	040505	020104	
8274	047760	040502	020104	042523	
8275	047766	052103	051117	044440	
8276	047774	043116	051117	040515	
8277	050002	044524	047117	000	
8278	050007	116	020117	051104	EM69: .ASCII /NO DRIVES FOUND ON BUSS/
8279	050014	053111	051505	013040	
8280	050022	052517	042116	047440	

8281	050030	020116	052502	051523	
8282	050036	005015	042523	052524	.ASCIZ <CR><LF>/SETUP CORRECTLY & PRESS 'CONTINUE'/<CR><LF>
8283	050044	020120	047503	051122	
8284	050052	041505	046124	020131	
8285	050060	020046	051120	051505	
8286	050066	020123	041447	047117	
8287	050074	044524	052516	023505	
8288	050102	005015	000		
8289	050105	127	044510	042514	EM70: .ASCIZ /WHILE WAITING FOR CONTR READY OR AFTER CONTR READY REC'D/
8290	050112	053440	044501	044524	
8291	050120	043516	043040	051117	
8292	050126	041440	047117	051124	
8293	050134	051040	040505	054504	
8294	050142	047440	020122	043101	
8295	050150	042524	020122	047503	
8296	050156	052116	020122	042522	
8297	050164	042101	020131	042522	
8298	050172	023503	000104		
8299	050176	042504	042524	052103	EM71: .ASCIZ /DETECTED 10 BAD SECTORS...ABORTING TEST/
8300	050204	042105	030440	020060	
8301	050212	040502	020104	042523	
8302	050220	052103	051117	027123	
8303	050226	027056	041101	051117	
8304	050234	044524	043516	052040	
8305	050242	051505	000124		
8306	050246	042504	042524	052103	EM72: .ASCIZ /DETECTED BSE BUT NOT LISTED IN BAD SECTOR FILE/
8307	050254	042105	041040	042523	
8308	050262	041040	052125	047040	
8309	050270	052117	046040	051511	
8310	050276	042524	020104	047111	
8311	050304	041040	042101	051440	
8312	050312	041505	047524	020122	
8313	050320	044506	042514	000	
8314	050325	104	052105	041505	EM73: .ASCII /DETECTED BSE IN READ COMMAND/
8315	050332	042524	020104	051502	
8316	050340	020105	047111	051040	
8317	050346	040505	020104	047503	
8318	050354	046515	047101	104	
8319	050361	015	041012	052125	.ASCIZ <CR><LF>/BUT NOT IN PREVIOUS WRITE COMMAND TO SAME SECTOR/
8320	050366	047040	052117	044440	
8321	050374	020116	051120	053105	
8322	050402	047511	051525	053440	
8323	050410	044522	042524	041440	
8324	050416	046517	040515	042116	
8325	050424	052040	020117	040523	
8326	050432	042515	051440	041505	
8327	050440	047524	000122		
8328	050444	052122	020132	047516	EM74: .ASCIZ /RTZ NOT SET IN RKMR2/
8329	050452	020124	042523	020124	
8330	050460	047111	051040	046513	
8331	050466	031122	000		
8332	050471	015	042012	052105	EM75: .ASCIZ <CR><LF>/DETECTED 10 BAD CYLINDERS...ABORTING TEST/
8333	050476	041505	042524	020104	
8334	050504	030061	041040	042101	
8335	050512	041440	046131	047111	
8336	050520	042504	051522	027056	

8337	050526	040456	047502	052122	
8338	050534	047111	020107	042524	
8339	050542	052123	000		
8340	050545	116	020117	051104	EM76: .ASCII /NO DRIVES FOUND IN DEVICE MAP (\$DEVN)/
8341	050552	053111	051505	043040	
8342	050560	052517	042116	044440	
8343	050566	020116	042504	044526	
8344	050574	042503	046440	050101	
8345	050602	024040	042044	053105	
8346	050610	024515			
8347	050612	005015	042523	052524	.ASCIIZ <CR><LF>/SETUP CORRECTLY & RESTART/<CR><LF>
8348	050620	020120	047503	051122	
8349	050626	041505	046124	020131	
8350	050634	020046	042522	052123	
8351	050642	051101	006524	000012	
8352	050650	051127	052111	020105	EM80: .ASCIIZ /WRITE CHECK ERROR SET IN RKCS2/
8353	050656	044103	041505	020113	
8354	050664	051105	047522	020122	
8355	050672	042523	020124	047111	
8356	050700	051040	041513	031123	
8357	050706	000			
8358	050707	104	052101	020101	EM83: .ASCIIZ /DATA CHECK ERROR SET IN RKER/
8359	050714	044103	041505	020113	
8360	050722	051105	047522	020122	
8361	050730	042523	020124	047111	
8362	050736	051040	042513	000122	
8363	050744	042522	042101	047111	EM93: .ASCIIZ /READING WRONG CYLINDER # IN HEADER/
8364	050752	020107	051127	047117	
8365	050760	020107	054503	044514	
8366	050766	042116	051105	021440	
8367	050774	044440	020116	042510	
8368	051002	042101	051105	000	
8369					
8370					.SBTTL DATA HEADERS
8371					
8372	051007	124	051505	020124	DH1: .ASCIIZ /TEST NO. PC/
8373	051014	047516	020056	050040	
8374	051022	000103			
8375	051024	045522	051115	004461	DH2: .ASCIIZ /RKMR1 RKMR2 RKMR3 RKER RKDS RKCS1 RKCS2/
8376	051032	045522	051115	004462	
8377	051040	045522	051115	004463	
8378	051046	045522	051105	051011	
8379	051054	042113	004523	045522	
8380	051062	051503	004461	045522	
8381	051070	051503	000062		
8382	051074	045522	041527	051011	DH3: .ASCIIZ /RKWC RKBA RKDA RKASOF RKDC RKECPS RKECPT/
8383	051102	041113	004501	045522	
8384	051110	040504	051011	040513	
8385	051116	047523	004506	045522	
8386	051124	041504	051011	042513	
8387	051132	050103	004523	045522	
8388	051140	041505	052120	000	
8389	051145	101	052106	051105	DH4: .ASCIIZ /AFTER PACK RE-INSERTED & HEADS LOADED/
8390	051152	050040	041501	020113	
8391	051160	042522	044455	051516	
8392	051166	051105	042524	020104	



8393	051174	020046	042510	042101		
8394	051202	020123	047514	042101		
8395	051210	042105	000			
8396	051213	101	052106	051105	DH5:	.ASCIZ /AFTER AC SWITCHED OFF/
8397	051220	040440	020103	053523		
8398	051226	052111	044103	042105		
8399	051234	047440	043106	000		
8400	051241	101	052106	051105	DH8:	.ASCIZ /AFTER DRIVE UNLOADED & DOOR OPENED/
8401	051246	042040	044522	042526		
8402	051254	052440	046116	040517		
8403	051262	042504	020104	020046		
8404	051270	047504	051117	047440		
8405	051276	042520	042516	000104		
8406	051304	043101	042524	020122	DH11:	.ASCIZ /AFTER MANUALLY LOADING HEADS WITH DOOR OPEN/
8407	051312	040515	052516	046101		
8408	051320	054514	046040	040517		
8409	051326	044504	043516	044040		
8410	051334	040505	051504	053440		
8411	051342	052111	020110	047504		
8412	051350	051117	047440	042520		
8413	051356	000116				
8414	051360	043101	042524	020122	DH12:	.ASCIZ /AFTER DISK PACK REMOVED/
8415	051366	044504	045523	050040		
8416	051374	041501	020113	042522		
8417	051402	047515	042526	000104		
8418	051410	043101	042524	020122	DH13:	.ASCIZ /AFTER MANUALLY LOADING HEADS WITH DISK PACK REMOVED/
8419	051416	040515	052516	046101		
8420	051424	054514	046040	040517		
8421	051432	044504	043516	044040		
8422	051440	040505	051504	053440		
8423	051446	052111	020110	044504		
8424	051454	045523	050040	041501		
8425	051462	020113	042522	047515		
8426	051470	042526	000104			
8427	051474	044527	044124	052517	DH15:	.ASCIZ /WITHOUT PACK COMMAND/
8428	051502	020124	040520	045503		
8429	051510	041440	046517	040515		
8430	051516	042116	000			
8431	051521	101	052106	051105	DH16:	.ASCIZ /AFTER UNIT SELECT PLUG REMOVED/
8432	051526	052440	044516	020124		
8433	051534	042523	042514	052103		
8434	051542	050040	052514	020107		
8435	051550	042522	047515	042526		
8436	051556	000104				
8437	051560	043101	042524	020122	DH17:	.ASCIZ /AFTER RECAL COMMAND/
8438	051566	042522	040503	020114		
8439	051574	047503	046515	047101		
8440	051602	000104				
8441	051604	043101	042524	020122	DH18:	.ASCIZ /AFTER UNLOAD COMMAND/
8442	051612	047125	047514	042101		
8443	051620	041440	046517	040515		
8444	051626	042116	000			
8445	051631	101	052106	051105	DH19:	.ASCIZ /AFTER PACK COMMAND/
8446	051636	050040	041501	020113		
8447	051644	047503	046515	047101		
8448	051652	000104				

8449	051654	043101	042524	020122	DH20:	.ASCIZ	/AFTER SELECT DRIVE COMMAND/
8450	051662	042523	042514	052103			
8451	051670	042040	044522	042526			
8452	051676	041440	046517	040515			
8453	051704	042116	000				
8454	051707	101	052106	051105	DH21:	.ASCIZ	/AFTER SUBSYSTEM CLEAR/
8455	051714	051440	041125	054523			
8456	051722	052123	046505	041440			
8457	051730	042514	051101	000			
8458	051735	101	052106	051105	DH22:	.ASCIZ	/AFTER DRIVE CLEAR COMMAND/
8459	051742	042040	044522	042526			
8460	051750	041440	042514	051101			
8461	051756	041440	046517	040515			
8462	051764	042116	000				
8463	051767	101	052106	051105	DH23:	.ASCIZ	/AFTER WRONG PORT SELECTED/
8464	051774	053440	047522	043516			
8465	052002	050040	051117	020124			
8466	052010	042523	042514	052103			
8467	052016	042105	000				
8468	052021	101	052106	051105	DH25:	.ASCIZ	/AFTER SEEK COMMAND/
8469	052026	051440	042505	020113			
8470	052034	047503	046515	047101			
8471	052042	000104					
8472	052044	043101	042524	020122	DH26:	.ASCIZ	/AFTER READ DATA COMMAND/
8473	052052	042522	042101	042040			
8474	052060	052101	020101	047503			
8475	052066	046515	047101	000104			
8476	052074	043101	042524	020122	DH27:	.ASCIZ	/AFTER WRITE DATA COMMAND/
8477	052102	051127	052111	020105			
8478	052110	040504	040524	041440			
8479	052116	046517	040515	042116			
8480	052124	000					
8481	052125	101	052106	051105	DH28:	.ASCIZ	/AFTER BOTH PORTS DESELECTED/
8482	052132	041040	052117	020110			
8483	052140	047520	052122	020123			
8484	052146	042504	042523	042514			
8485	052154	052103	042105	000			
8486	052161	101	052106	051105	DH29:	.ASCIZ	/AFTER CORRECT PORT SELECTED/
8487	052166	041440	051117	042522			
8488	052174	052103	050040	051117			
8489	052202	020124	042523	042514			
8490	052210	052103	042105	000			
8491	052215	101	052106	051105	DH30:	.ASCIZ	/AFTER READ HEADER COMMAND/
8492	052222	051040	040505	020104			
8493	052230	042510	042101	051105			
8494	052236	041440	046517	040515			
8495	052244	042116	000				
8496	052247	101	052106	051105	DH31:	.ASCIZ	/AFTER DRIVE MANUALLY LOADED/
8497	052254	042040	044522	047526			
8498	052262	046440	047101	040525			
8499	052270	046114	020131	047514			
8500	052276	042101	042105	000			
8501	052303	101	052106	051105	DH32:	.ASCIZ	/AFTER WRITE CHECK COMMAND/
8502	052310	053440	044522	042524			
8503	052316	041440	042510	045503			
8504	052324	041440	046517	040515			

8505	052332	042116	000		
8506	052335	101	052106	051105	DH34: .ASCIZ /AFTER START SPINDLE COMMAND/
8507	052342	051440	040524	052122	
8508	052350	051440	044520	042116	
8509	052356	042514	041440	046517	
8510	052364	040515	042116	000	
8511	052371	101	052106	051105	DH35: .ASCIZ /AFTER MANUALLY UNLOADING/
8512	052376	046440	047101	040525	
8513	052404	046114	020131	047125	
8514	052412	047514	042101	047111	
8515	052420	000107			
8516	052422	043101	042524	020122	DH37: .ASCIZ /AFTER TIMEOUT TO POWER DOWN/
8517	052430	044524	042515	052517	
8518	052436	020124	047524	050040	
8519	052444	053517	051105	042040	
8520	052452	053517	000116		
8521	052456	043101	042524	020122	DH38: .ASCIZ /AFTER AC POWERED UP/
8522	052464	041501	050040	053517	
8523	052472	051105	042105	052440	
8524	052500	000120			
8525	052502	043101	042524	020122	DH39: .ASCIZ /AFTER WRITE HEADER COMMAND/
8526	052510	051127	052111	020105	
8527	052516	042510	042101	051105	
8528	052524	041440	046517	040515	
8529	052532	042116	000		
8530	052535	104	051125	047111	DH41: .ASCIZ /DURING RECAL COMMAND/
8531	052542	020107	042522	040503	
8532	052550	020114	047503	046515	
8533	052556	047101	000104		
8534	052562	047117	051440	041505	DH42: .ASCIZ /ON SECTORS 0,2,4,6 OR 8 CYL 410 TRACK 2/
8535	052570	047524	051522	030040	
8536	052576	031054	032054	033054	
8537	052604	047440	020122	020070	
8538	052612	041440	046131	032040	
8539	052620	030061	052040	040522	
8540	052626	045503	031040	000	
8541	052633	106	051117	040515	DH44: .ASCIZ /FORMAT & ALL READ-WRITE TESTS WILL BE BYPASSED/
8542	052640	020124	020046	046101	
8543	052646	020114	042522	042101	
8544	052654	053455	044522	042524	
8545	052662	052040	051505	051524	
8546	052670	053440	046111	020114	
8547	052676	042502	041040	050131	
8548	052704	051501	042523	000104	
8549	052712	043101	042524	020122	DH45: .ASCIZ /AFTER SEEK WITH VOLUME VALID=0/
8550	052720	042523	045505	053440	
8551	052726	052111	020110	047526	
8552	052734	052514	042515	053040	
8553	052742	046101	042111	030075	
8554	052750	000			
8555	052751	101	052106	051105	DH46: .ASCIZ /AFTER WRITE DATA WITH VOLUME VALID=0/
8556	052756	053440	044522	042524	
8557	052764	042040	052101	020101	
8558	052772	044527	044124	053040	
8559	053000	046117	046525	020105	
8560	053006	040526	044514	036504	

8561	053014	000060							
8562	053016	043101	042524	020122	DH48:	.ASCIZ	/AFTER WRITE LOCK SWITCH DISABLED/		
8563	053024	051127	052111	020105					
8564	053032	047514	045503	051440					
8565	053040	044527	041524	020110					
8566	053046	044504	040523	046102					
8567	053054	042105	000						
8568	053057	101	052106	051105	DH49:	.ASCIZ	/AFTER WRITE LOCK SWITCH ENABLED/		
8569	053064	053440	044522	042524					
8570	053072	046040	041517	020113					
8571	053100	053523	052111	044103					
8572	053106	042440	040516	046102					
8573	053114	042105	000						
8574	053117	101	052106	051105	DH50:	.ASCIZ	/AFTER WRITING WITH WRITE LOCK ENABLED/		
8575	053124	053440	044522	044524					
8576	053132	043516	053440	052111					
8577	053140	020110	051127	052111					
8578	053146	020105	047514	045503					
8579	053154	042440	040516	046102					
8580	053162	042105	000						
8581	053165	103	046131	021440	DH56:	.ASCIZ	/CYL # HEADER WORD 0/		
8582	053172	044011	040505	042504					
8583	053200	020122	047527	042122					
8584	053206	030040	000						
8585	053211	101	052106	051105	DH59:	.ASCIZ	/AFTER DRIVE SELECT COMMAND WITH EVEN PARITY/		
8586	053216	042040	044522	042526					
8587	053224	051440	046105	041505					
8588	053232	020124	047503	046515					
8589	053240	047101	020104	044527					
8590	053246	044124	042440	042526					
8591	053254	020116	040520	044522					
8592	053262	054524	000						
8593	053265	101	052106	051105	DH60:	.ASCIZ	/AFTER WRITE LOCK ENABLED DURING CONTINUOUS WRITING/		
8594	053272	053440	044522	042524					
8595	053300	046040	041517	020113					
8596	053306	047105	041101	042514					
8597	053314	020104	052504	044522					
8598	053322	043516	041440	047117					
8599	053330	044524	052516	052517					
8600	053336	020123	051127	052111					
8601	053344	047111	000107						
8602	053350	045522	040504	053411	DH61:	.ASCII	/RKDA WORD EXPECTED/		
8603	053356	051117	004504	054105					
8604	053364	042520	052103	042105					
8605	053372	005015	020100	051127		.ASCIZ	<CR><LF>/@ WRL WAS WORD/		
8606	053400	004514	040527	004523					
8607	053406	047527	042122	000					
8608	053413	122	042113	004501	DH62:	.ASCII	/RKDA WORD EXPECTED/		
8609	053420	047527	042122	042411					
8610	053426	050130	041505	042524					
8611	053434	104							
8612	053435	015	041012	043105		.ASCII	<CR><LF>/BEFORE WAS WORD/		
8613	053442	051117	004505	040527					
8614	053450	004523	047527	042122					
8615	053456	005015	051127	000114		.ASCIZ	<CR><LF>/WRL/		
8616	053464	043101	042524	020122	DH63:	.ASCIZ	/AFTER WRITE LOCK ENABLED FROM AC FAILURE/		

```
8617 053472 051127 052111 020105
8618 053500 047514 045503 042440
8619 053506 040516 046102 042105
8620 053514 043040 047522 020115
8621 053522 041501 043040 044501
8622 053530 052514 042522 000
8623 053535 101 052106 051105 DH64: .ASCIZ /AFTER MDS DETECTED IN RKCS2/
8624 053542 046440 051504 042040
8625 053550 052105 041505 042524
8626 053556 020104 047111 051040
8627 053564 041513 031123 000
8628 053571 101 052106 051105 DH65: .ASCIZ /AFTER SEARCHING ALL DRIVES PRESENT/
8629 053576 051440 040505 041522
8630 053604 044510 043516 040440
8631 053612 046114 042040 044522
8632 053620 042526 020123 051120
8633 053626 051505 047105 000124
8634 053634 042524 052123 047040 DH66: .ASCIZ /TEST NO. TRAP PC/
8635 053642 027117 052011 040522
8636 053650 020120 041520 000
8637 053655 101 052106 051105 DH67: .ASCIZ /AFTER TIMEOUT TO ENABLE WRITE LOCK/
8638 053662 052040 046511 047505
8639 053670 052125 052040 020117
8640 053676 047105 041101 042514
8641 053704 053440 044522 042524
8642 053712 046040 041517 000113
8643 053720 042411 050130 041505 DH69: .ASCIZ / EXPECT/
8644 053726 000124
8645 053730 040411 052103 040525 DH70: .ASCIZ / ACTUAL/
8646 053736 000114
8647 053740 030101 041011 004460 DH71: .ASCIZ /A0 B0 A1 B1 A2 B2 B3/
8648 053746 030501 041011 004461
8649 053754 031101 041011 004462
8650 053762 031502 000
8651 053765 115 043523 040440 DH73: .ASCIZ /MSG A & B IN RKMR2 & RKMR3 RESP ARE INVALID/
8652 053772 023040 041040 044440
8653 054000 020116 045522 051115
8654 054006 020062 020046 045522
8655 054014 051115 020063 042522
8656 054022 050123 040440 042522
8657 054030 044440 053116 046101
8658 054036 042111 000
8659 054041 117 020116 042523 DH74: .ASCIZ /ON SECTORS 10, 12, 14, 16, 18 OR 20 CYL 410 TRACK 2/
8660 054046 052103 051117 020123
8661 054054 030061 020054 031061
8662 054062 020054 032061 020054
8663 054070 033061 020054 034061
8664 054076 047440 020122 030062
8665 054104 041440 046131 032040
8666 054112 030061 052040 040522
8667 054120 045503 031040 000
8668
8669 .SBTTL ERROR OUTPUT DATA
8670
8671 .EVEN
8672 054126 001214 001116 DT1: $TESTN,$ERRPC
```

8673	054132	005364	005366	005370		HMR1,HMR2,HMR3,HER,HDS,HCS1,HCS2
8674	054140	005354	005352	005340		
8675	054146	005342				
8676	054150	005344	005346	005350		HWC,HBA,HDA,HASOF,HDC,HPOS,HPAT
8677	054156	005356	005360	005372		
8678	054164	005374				
8679	054166	001214	001116	001166	DT3:	\$TESTN,\$ERRPC,\$TMP3,WD1,WD2
8680	054174	001434	001436			
8681	054200	005364	005366	005370		HMR1,HMR2,HMR3,HER,HDS,HCS1,HCS2
8682	054206	005354	005352	005340		
8683	054214	005342				
8684	054216	005344	005346	005350		HWC,HBA,HDA,HASOF,HDC,HPOS,HPAT
8685	054224	005356	005360	005372		
8686	054232	005374				
8687	054234	001214	001354		DT6:	\$TESTN,TRAPPC
8688	054240	001214	001116	001366	DT9:	\$TESTN,\$ERRPC,TOCYL,RHTAB
8689	054246	001712				
8690	054250	005364	005366	005370		HMR1,HMR2,HMR3,HER,HDS,HCS1,HCS2
8691	054256	005354	005352	005340		
8692	054264	005342				
8693	054266	005344	005346	005350		HWC,HBA,HDA,HASOF,HDC,HPOS,HPAT
8694	054274	005356	005360	005372		
8695	054302	005374				
8696	054304	001214	001116	005430	DT13:	\$TESTN,\$ERRPC,E.A0,E.B0,E.A1,E.B1,H.A0,H.B0,H.A1,H.B1
8697	054312	005432	005434	005436		
8698	054320	005410	005412	005414		
8699	054326	005416				
8700	054330	005364	005366	005370		HMR1,HMR2,HMR3,HER,HDS,HCS1,HCS2
8701	054336	005354	005352	005340		
8702	054344	005342				
8703	054346	005344	005346	005350		HWC,HBA,HDA,HASOF,HDC,HPOS,HPAT
8704	054354	005356	005360	005372		
8705	054362	005374				
8706						
8707	054364	001214	001116	005430	DT14:	\$TESTN,\$ERRPC,E.A0,E.B0,E.A1,E.B1,E.A2,E.B2
8708	054372	005432	005434	005436		
8709	054400	005440	005442			
8710	054404	005410	005412	005414		H.A0,H.B0,H.A1,H.B1,H.A2,H.B2
8711	054412	005416	005420	005422		
8712	054420	005364	005366	005370		HMR1,HMR2,HMR3,HER,HDS,HCS1,HCS2
8713	054426	005354	005352	005340		
8714	054434	005342				
8715	054436	005344	005346	005350		HWC,HBA,HDA,HASOF,HDC,HPOS,HPAT
8716	054444	005356	005360	005372		
8717	054452	005374				
8718						
8719	054454	001214	001116	005430	DT15:	\$TESTN,\$ERRPC,E.A0,E.B0,E.A1,E.B1,E.A2,E.B2,E.B3
8720	054462	005432	005434	005436		
8721	054470	005440	005442	005446		
8722	054476	005410	005412	005414		H.A0,H.B0,H.A1,H.B1,H.A2,H.B2,H.B3
8723	054504	005416	005420	005422		
8724	054512	005426				
8725	054514	005364	005366	005370		HMR1,HMR2,HMR3,HER,HDS,HCS1,HCS2
8726	054522	005354	005352	005340		
8727	054530	005342				
8728	054532	005344	005346	005350		HWC,HBA,HDA,HASOF,HDC,HPOS,HPAT

Line	Code	Subcode	Value	Value	Format
8729	054540	005356	005360	005372	
8730	054546	005374			
8731					
8732					.SBTTL ERROR DATA FORMATS
8733					
8734	054550	000003			DF1: 3
8735	054552	002	000		.BYTE 2.0
8736	054554	051024			DH2
8737	054556	007	000		.BYTE 7.0
8738	054560	051074			DH3
8739	054562	007	000		.BYTE 7.0
8740					
8741					
8742	054564	000005			DF3: 5
8743	054566	000	000		.BYTE 0.0
8744	054570	051007			DH1
8745	054572	002	000		.BYTE 2.0
8746	054574	053350			DH61
8747	054576	003	000		.BYTE 3.0
8748	054600	051024			DH2
8749	054602	007	000		.BYTE 7.0
8750	054604	051074			DH3
8751	054606	007	000		.BYTE 7.0
8752					
8753	054610	000005			DF4: 5
8754	054612	000	000		.BYTE 0.0
8755	054614	051007			DH1
8756	054616	002	000		.BYTE 2.0
8757	054620	053413			DH62
8758	054622	003	000		.BYTE 3.0
8759	054624	051024			DH2
8760	054626	007	000		.BYTE 7.0
8761	054630	051074			DH3
8762	054632	007	000		.BYTE 7.0
8763					
8764	054634	000001			DF5: 1
8765	054636	002	000		.BYTE 2.0
8766					
8767	054640	000003			DF7: 3
8768	054642	002	000		.BYTE 2.0
8769	054644	051024			DH2
8770	054646	007	000		.BYTE 7.0
8771	054650	051074			DH3
8772	054652	007	000		.BYTE 7.0
8773	054654	000004			DF10: 4
8774	054656	000	000		.BYTE 0.0
8775	054660	051007			DH1
8776	054662	002	000		.BYTE 2.0
8777	054664	051024			DH2
8778	054666	007	000		.BYTE 7.0
8779	054670	051074			DH3
8780	054672	007	000		.BYTE 7.0
8781					
8782	054674	000005			DF12: 5
8783	054676	000	000		.BYTE 0.0
8784	054700	053765			DH73

8785	054702	000	000		.BYTE	0.0
8786	054704	051007			DH1	
8787	054706	002	000		.BYTE	2.0
8788	054710	051024			DH2	
8789	054712	007	000		.BYTE	7.0
8790	054714	051074			DH3	
8791	054716	007	000		.BYTE	7.0
8792						
8793						
8794	054720	000004		DF15:	4	
8795	054722	000	000		.BYTE	0.0
8796	054724	051007			DH1	
8797	054726	002	000		.BYTE	2.0
8798	054730	051024			DH2	
8799	054732	007	000		.BYTE	7.0
8800	054734	051074			DH3	
8801	054736	007	000		.BYTE	7.0
8802						
8803						
8804	054740	000005		DF17:	5	
8805	054742	000	000		.BYTE	0.0
8806	054744	052633			DH44	
8807	054746	000	000		.BYTE	0.0
8808	054750	051007			DH1	
8809	054752	002	000		.BYTE	2.0
8810	054754	051024			DH2	
8811	054756	007	000		.BYTE	7.0
8812	054760	051074			DH3	
8813	054762	007	000		.BYTE	7.0
8814	054764	000005		DF20:	5	
8815	054766	000	000		.BYTE	0.0
8816	054770	051007			DH1	
8817	054772	002	000		.BYTE	2.0
8818	054774	053165			DH56	
8819	054776	002	000		.BYTE	2.0
8820	055000	051024			DH2	
8821	055002	007	000		.BYTE	7.0
8822	055004	051074			DH3	
8823	055006	007	000		.BYTE	7.0
8824						
8825	055010	000007		DF21:	7	
8826	055012	000	000		.BYTE	0.0
8827	055014	051007			DH1	
8828	055016	002	000		.BYTE	2.0
8829	055020	053720			DH69	
8830	055022	000	000		.BYTE	0.0
8831	055024	053740			DH71	
8832	055026	004	000		.BYTE	4.0
8833	055030	053730			DH70	
8834	055032	004	000		.BYTE	4.0
8835	055034	051024			DH2	
8836	055036	007	000		.BYTE	7.0
8837	055040	051074			DH3	
8838	055042	007	000		.BYTE	7.0
8839						
8840	055044	000007		DF22:	7	



8841	055046	000	000	.BYTE	0,0
8842	055050	051007		DH1	
8843	055052	002	000	.BYTE	2,0
8844	055054	053720		DH69	
8845	055056	000	000	.BYTE	0,0
8846	055060	053740		DH71	
8847	055062	006	000	.BYTE	6,0
8848	055064	053730		DH70	
8849	055066	006	000	.BYTE	6,0
8850	055070	051024		DH2	
8851	055072	007	000	.BYTE	7,0
8852	055074	051074		DH3	
8853	055076	007	000	.BYTE	7,0
8854					
8855	055100	000007		DF23:	7
8856	055102	000	000	.BYTE	0,0
8857	055104	051007		DH1	
8858	055106	002	000	.BYTE	2,0
8859	055110	053720		DH69	
8860	055112	000	000	.BYTE	0,0
8861	055114	053740		DH71	
8862	055116	007	000	.BYTE	7,0
8863	055120	053730		DH70	
8864	055122	007	000	.BYTE	7,0
8865	055124	051024		DH2	
8866	055126	007	000	.BYTE	7,0
8867	055130	051074		DH3	
8868	055132	007	000	.BYTE	7,0

```

*****
:SBTTL TYPE ERROR ROUTINE
:*ENTRY JSR PC,TYP ERR
:*RETURN RTS PC
:*
:*THIS ROUTINE USES THE "ITEM CONTROL BYTE" ($ITEMB) TO DETERMINE WHICH
:*ERROR IS TO BE REPORTED. IT THEN USES THE "ERROR TABLE" ($ERRTB)
:*ENTRY TO DEFINE WHAT INFORMATION IS TO BE REPORTED CONCERNING
:*THE ERROR.
*****
TYPERR: SAVREG
BIT #SW13,@SWR ;INHIBIT ERROR TYPEOUTS?
BNE 20$ ;YES-BRANCH
MOVB $ITEMB,RO ;ENTER ERROR NUMBER
BIC #177400,RO ;CLEAR SIGN EXTENSION
DEC RO ;FORM INDEX FOR ERROR TABLE
ASL RO
ASL RO
ASL RO
1$: ADD #$ERRTB,RO ;FORM ADDRESS OF ERROR ENTRY
MOV (RO)+,2$ ;GET EM POINTER
BEQ 3$ ;BRANCH IF THERE ISN'T ONE
TYPE .%CRLF ;TYPE CARRIAGE RETURN LINE FEED
TYPE ;TYPE ERROR MESSAGE (EM)
2$: .WORD 0 ;EM POINTER GOES HERE
3$: MOV (RO)+,4$ ;GET DH POINTER
BEQ 5$ ;BRANCH IF THERE ISN'T ONE
TYPE .%CRLF ;TYPE CR-LF

```

CZ  
 CZ  
 AB  
 AC  
 AC  
 AC  
 AC  
 AD  
 AD  
 AD  
 AD  
 AD  
 AD  
 AD  
 AD  
 AD  
 AD  
 AE  
 AE  
 AF  
 AH  
 AH  
 AH  
 AH  
 AH  
 AH  
 AH  
 AH  
 AH  
 AP  
 AP  
 AP  
 AP  
 AS  
 AT  
 AT  
 AU  
 AU  
 AV  
 AV  
 BA

8897	055222	104401		TYPE		:TYPE DATA HEADER
8898	055224	000000		4\$: .WORD	0	:DH POINTER GOES HERE
8899	055226	012001		5\$: MOV	(R0)+,R1	:GET DT POINTER
8900	055230	001455		BEQ	20\$	:BRANCH IF THERE ARE NONE
8901	055232	005004		CLR	R4	:SET INDENT SWITCH
8902	055234	012000		MOV	(R0)+,R0	:GET DF POINTER
8903	055236	012002		MOV	(R0)+,R2	:STORE NUMBER OF DH'S
8904	055240	001446		BEQ	17\$	:DH NUM IS 0-BRANCH
8905	055242	005104		COM	R4	:NO INDENT
8906	055244	104401	001205	TYPE	,\$CRLF	
8907	055250	112003		10\$: MOVB	(R0)+,R3	:GET & STORE NUMBER OF DATA WORDS
8908	055252	105720		TSTB	(R0)+	:BUMP PAST FORMAT WORD
8909	055254	005703		TST	R3	:TEST IF ANY DATA FOR THIS HEADER
8910	055256	001407		BEQ	14\$	:NO - SKIP DATA PRINT
8911	055260	013146		11\$: MOV	@(R1)+,-(SP)	:PUT FIRST DATA WORD ON STACK
8912	055262	104402		TYPOC		:TYPE IT
8913	055264	005303		DEC	R3	:MORE DATA WORDS
8914	055266	001403		BEQ	14\$	:NO-BRANCH
8915	055270	104401	055420	TYPE	,\$SPACE2	:TYPE SEPARATORS
8916	055274	000771		BR	11\$	:LOOP
8917	055276	005302		14\$: DEC	R2	:MORE DH'S?
8918	055300	003431		BLE	20\$	:NO-BRANCH
8919	055302	104401	001205	TYPE	,\$CRLF	
8920	055306	005760	000002	TST	2(R0)	:ONLY A DH IN THIS REQUEST?
8921	055312	001404		BEQ	15\$	:YES-BRANCH BYPASS INDENT
8922	055314	005104		COM	R4	:INDENT?
8923	055316	001002		BNE	15\$	:NO-BRANCH
8924	055320	104401	055420	TYPE	,\$SPACE2	:YES-TYPE SPACES
8925	055324	012037	055332	15\$: MOV	(R0)+,16\$	:GET NEXT DH POINTER
8926	055330	104401		TYPE		:TYPE DH
8927	055332	000000		16\$: .WORD	0	:DH POINTER GOES HERE
8928	055334	105710		TSTB	(R0)	:TYPE A DT?
8929	055336	001003		BNE	21\$	:YES-BRANCH
8930	055340	062700	000002	ADD	#2,R0	:INCREMENT DF POINTER
8931	055344	000754		BR	14\$	:SEE IF END OF DF BLOCK
8932	055346	104401	001205	21\$: TYPE	,\$CRLF	
8933	055352	005704		TST	R4	:INDENT?
8934	055354	001335		BNE	10\$	:NO-BRANCH
8935	055356	104401	055420	17\$: TYPE	,\$SPACE2	:YES-TYPE SPACES
8936	055362	000732		BR	10\$	:LOOP
8937	055364	104414		20\$: RESREG		
8938	055366	032777	010000 123544	BIT	#SW12,@SWR	:SEE IF EXIT AFTER 20 ERRORS
8939	055374	001410		BEQ	25\$	:BR IF NO
8940	055376	023727	001103 000024	CMPL	\$ERFLG,#20.	:ELSE SEE IF HAVE 20 ERRORS
8941	055404	001004		BNE	25\$	:BR IF NO
8942	055406	012706	001100	MOV	#STACK,SP	:ELSE RESTORE STACK
8943	055412	000137	023374	JMP	ENDRV	:AND BYPASS DRIVE
8944	055416	000207		25\$: RTS	PC	
8945	055420	020040	000	SPACE2: .ASCIZ/ /		:2 SPACES
8946				: ODT-11 -- V005A		
8947						
8948				: DEC-11-UODPA-A-LA		
8949						
8950				: COPYRIGHT 1969,1970,1972		
8951				: DIGITAL EQUIPMENT CORPORATION		
8952				: MAYNARD, MASSACHUSETTS 01754		

CZ  
CZ  
BA  
BA  
BA  
BA  
BI  
BI  
BI  
BI  
BI  
BI  
BI  
BI  
BI  
BI  
BI  
BI  
BI  
BI  
BI  
BI  
BI  
BI  
BI  
BI  
BI  
BI  
BI  
BP  
BS  
BS  
BS  
BS  
BY  
BY  
CA  
CA  
CC  
CC  
CC  
CD  
CE  
CF  
CH  
CH  
CK  
CK  
CL  
CL  
CL  
CL















```

9267
9268 056652 012637 055462      0.BRK:  MOV      (SP)+,0.UPC      ;PRIORITY IS 7 UPON ENTRY
9269 056656 012637 055464      MOV      (SP)+,0.UST      ;SAVE STATUS AND PC
9270 056662 004037 057070      0.BK1:  JSR      0,0.SVR      ;SAVE VARIOUS REGISTERS
9271 056666 105737 057510      TSTB    0.T              ;CHECK FOR T-BIT SET
9272 056672 001311              BNE     0.TBIT           ;JUMP IF SET
9273 056674 013777 055502 176574  MOV     0.UIN,00.ADR1    ;REMOVE BREAKPOINTS
9274 056702 105737 055466      TSTB    0.PRI           ;CHECK IF PRIORITY
9275 056706 100CJ3              BPL     0.BK2           ;IS AS SAME AS USER PGM
9276 056710 113705 055464      MOV     0.UST,R5        ;PICK UP USER UST IF SO
9277 056714 000407              BR      0.BK3           ;AND DON'T COMPUTE THE PRIORITY
9278 056716 113705 055466      0.BK2:  MOV     0.PRI,R5    ;OTHERWISE PICK UP ACTUAL PRIORITY
9279 056722 000257              CCC     ;CLEAR CARRY
9280 056724 106005              RORB   R5               ;SHIFT LOW ORDER BITS
9281 056726 106005              RORB   R5               ;INTO
9282 056730 106005              RORB   R5               ;HIGH ORDER
9283 056732 106005              RORB   R5               ;POSITION
9284 056734 110537 177776      0.BK3:  MOV     R5,ST        ;PUT THE STATUS AWAY WHERE IT BELONGS
9285 056740 013705 055462      MOV     0.UPC,R5        ;GET PC, IT POINTS TO THE TRT
9286 056744 005745              TST    -(R5)           ;SUBTRACT TWO
9287 056746 010537 055462      MOV     R5,0.UPC        ;FROM THE USER'S PC
9288 056752 020537 055476      CMP     R5,0.ADR1       ;COMPARE WITH LIST
9289 056756 001417              BEQ    0.B2            ;JUMP IF FOUND
9290 056760 004537 057126      JSR    5,0.SVTT         ;SAVE TELETYPE STATUS
9291 056764 004537 057462      JSR    5,0.CRLF        ;
9292 056770 012704 057514      MOV     #0.BD,R4        ;ERROR, NOTHING FOUND
9293 056774 012703 057515      MOV     #0.BD+1,R3     ;
9294 057000 004537 057354      JSR    5,0.TYPE        ;OUTPUT 'BE' FOR BAD ENTRY
9295 057004 010500              MOV     R5,R0          ;
9296 057006 042737 000020 055464  BIC     #0.TBT,0.UST    ;CLEAR OUT ANY POSSIBLE FAKE T-BIT
9297 057014 000420              BR      0.B3           ;AND CONTINUE
9298 057016 005337 055500      0.B2:  DEC     0.CT          ;
9299 057022 003302              BGT    0.C1           ;JUMP IF REPEAT
9300 057024 012737 000001 055500  MOV     #1,0.CT        ;RESET COUNT TO 1
9301 057032 105237 057511      INCB   0.P             ;ALLOW PROCEED
9302 057036 004537 057126      JSR    5,0.SVTT         ;SAVE TELETYPE STATUS, R4 IS SAFE
9303 057042 012700 000102      MOV     #1B,R0         ;
9304 057046 004537 057370      JSR    5,0.FTYP        ;TYPE 'B'
9305 057052 013700 055476      MOV     0.ADR1,R0      ;GET ADDRESS OF BREAK
9306 057056 004537 057226      0.B3:  JSR    5,0.CADV       ;TYPE ADDRESS
9307 057062 005005              CLR    R5              ;CLEAR CAD
9308 057064 000137 055774      JMP    0.DCD           ;GO TO DECODER
9309
9310      ; SAVE REGISTERS R0-R6 IN INTERNAL STACK
9311
9312 057070 012637 057506      0.SVR:  MOV     (SP)+,0.XXX    ;PICK REGISTER FROM STACK AND SAVE
9313 057074 010637 055460      MOV     SP,0.USP       ;SAVE USER STACK ADDRESS
9314 057100 012706 055460      MOV     #0.USP,SP      ;SET TO INTERNAL STACK
9315 057104 010546              MOV     R5,-(SP)       ;SAVE
9316 057106 010446              MOV     R4,-(SP)       ;REGISTERS
9317 057110 010346              MOV     R3,-(SP)       ;1
9318 057112 010246              MOV     R2,-(SP)       ;THRU
9319 057114 010146              MOV     R1,-(SP)       ;5
9320 057116 013746 057506      MOV     0.XXX,-(SP)    ;PUT SAVED REGISTER ON STACK
9321 057122 005746              TST    -(SP)          ;
9322 057124 000200              RTS     R0             ;

```

CZ  
 CZ  
 HC  
 HD  
 HD  
 HD  
 HD  
 HD  
 HD  
 HE  
 HE  
 HE  
 HM  
 HM  
 HO  
 HP  
 HP  
 HP  
 HT  
 HV  
 HW  
 HZ  
 H.  
 H.  
 H.  
 H.  
 H.  
 H.  
 ID  
 IE  
 IL  
 IN  
 IO  
 IR  
 LC  
 LC  
 LC  
 LF  
 LI  
 LK

























ESEC	001420	1921#															
E.A0	005430	2022#	3562*	3587*	3641*	4052*	4127*	4304*	4346*	4395*	4430*	4585*	4643*	4747*			
		4822*	4900*	4958*	5420*	5425*	5429	5431*	5460	8696	8707	8719					
E.A1	005434	2024#	3564*	3589*	3643*	4054*	4129*	4306*	4348*	4397*	4432*	4587*	4645*	4749*			
		4824*	4902*	4960*	5421*	5433	5435*	5480	8696	8707	8719						
E.A2	005440	2026#	4131*	4308*	4350*	4434*	4589*	4647*	4751*	4904*	4962*	5422*	5437	5439*			
		8707	8719														
E.A3	005444	2028#	5423*														
E.B0	005432	2023#	3563*	3588*	3642*	4053*	4128*	4305*	4347*	4396*	4431*	4586*	4644*	4748*			
		4823*	4901*	4959*	5441	5443*	5470	8696	8707	8719							
E.B1	005436	2025#	3565*	3590*	3644*	4055*	4130*	4307*	4349*	4398*	4433*	4588*	4646*	4750*			
		4825*	4903*	4961*	5445	5447*	5490	8696	8707	8719							
E.B2	005442	2027#	4132*	4309*	4351*	4435*	4590*	4648*	4752*	4905*	4963*	5449	5451*	8707			
		8719															
E.B3	005446	2029#	4133*	4310*	4352*	4436*	4591*	4649*	4753*	4906*	4964*	5453	5455*	8719			
E.DDT	012662	3410*	3416*	3424#	5425												
FATT1	025056	4000	5349#	6119	6140												
FATT2	025152	4039	4196	5377#													
FCYL	027206	5736#															
FHDHM	027276	5761#	5770														
FHDTAB	027420	4248	4709	5795#													
FLGTST	027650	5837	5841	5856#													
FLOAD	027352	5779#															
FMT#	= 000020	1161#															
FMT1	001452	1940#	5805*	5806*	5807*	5812											
FORM	031022	6154	6167#														
FORMAT	001450	1939#	4246*	4707*	5805	5834											
FRCYL	001364	1904#															
FRDY	024542	3147	3162	3171	3275	3284	5254	5265	5277#	5280	5615						
FRDY1	024610	5291#	5294	5602													
FSEC17	027116	5712#															
FTITLE	001360	1896#	5147	5149*													
GBA	024350	2972	5214#														
GDRVS	024210	2970	5169#														
GETSP	030452	3550	3555	3580	3605	3620	3631	3716	3733	3822	3848	3858	3869	3878			
		3893	3946	4017	4235	4370	4454	4489	4508	4696	4780	4811	4839	5041			
		5048	6078#	6084													
GIN#	024376	2974	5227#														
GNS	= ***** U	1279	7313	7314	7315	7316	7317	7319	7321	7322	7323	7324	7325	7326			
		7327															
GO	= 000001	1125#															
GSTAT	026106	3445	3557	3582	3608	3622	3636	3700	3710	3719	3725	3813	3850	3870			
		3888	3925	3936	4019	4122	4230	4237	4260	4284	4335	4373	4387	4419			
		4455	4574	4632	4690	4698	4720	4741	4803	4889	4947	5359	5364	5384			
		5388	5550#	5619	5630	5642	5658	5763	5781	6135							
GSTAT1	026142	5418	5561#														
GSTAT2	026372	5566	5574	5582	5589	5598#											
GTSWR	= 104406	5158	7319#														
HASOF	005356	1994#	5316*	5332	8676	8684	8693	8703	8715	8728							
HBA	005346	1990#	5312*	8676	8684	8693	8703	8715	8728								
HCS1	005340	1987#	3165*	3166*	3167	3173	3278*	3279*	3280	3286	3485*	3511*	3655*	3668*			
		3766*	3781*	3795*	3926	3967*	3995*	4034*	4072*	4119*	4123	4191*	4199	4257*			
		4261	4281*	4285	4332*	4336	4383*	4388	4416*	4420	4478*	4482	4571*	4575			
		4629*	4633	4717*	4721	4738*	4742	4799*	4804	4851*	4886*	4890	4944*	4948			
		5054*	5055*	5056	5251*	5252	5262*	5263	5309*	5524	5528	5552*	5598*	5599*			
		5600	5620	5931	6114*	6130*	6145*	8673	8681	8690	8700	8712	8725				









O.BKPT	055730	9048#	9106						
O.BK1	056662	8996	9270#						
O.BK2	056716	9275	9278#						
O.BK3	056734	9277	9284#						
O.BRK	056652	9013	9268#						
O.BUF	057543	9350	9359	9364	9469#				
O.B2	057016	9289	9298#						
O.B3	057056	9297	9306#						
O.CADV	057226	9119	9136	9159	9163	9197	9201	9306	9349#
O.CLGL	056044	9075	9077	9085#					
O.CLGT=	000014	9089	9459#						
O.CLS1	057452	9417	9419#						
O.CR	057516	9426	9428	9429	9443#				
O.CRET	056156	9097	9126#						
O.CRLF	057462	9134	9195	9291	9330	9335	9426#		
O.CRLS	057470	9070	9428#						
O.CRS	057474	9427	9429#						
O.CSR1	057512	9326*	9342	9437#					
O.CSR2	057513	9327*	9343	9438#					
O.CT	055500	9255*	9298*	9300*	9497#				
O.C1	056630	9258#	9299						
O.DCD	055774	9055	9060	9070#	9128	9183	9308		
O.DCD1	056000	9071#	9120	9164					
O.EFF	056304	9105	9174#						
O.EFF1	056422	9186	9205#						
O.ENTR	055510	8993#							
O.ERR	055760	9040	9051	9067#	9090	9116	9178	9421	
O.ERR1	056314	9178#	9251						
O.ERR2	056134	9116#	9150	9227					
O.FTYP	057370	9069	9138	9153	9199	9304	9375	9391	9396#
O.GET	057312	9019	9073	9372#	9377	9379	9381		
O.GO	056466	9099	9225#						
O.GO2	056544	9236#	9261						
O.LG =	000006	9023	9467#						
O.LGCH	057521	9086	9447#	9459					
O.LGDR	056074	9093	9096#	9108					
O.LGL =	000030	9108#							
O.LGL1	056046	9086#	9091						
O.LGL2	056066	9087	9092#						
O.MSK	055470	9179	9180	9182	9489#				
O.ODT	055504	1288	8991#	9478					
O.OFST	056232	9103	9149#						
O.OF1	056300	9162	9164#						
O.OP1	056170	9100	9132#						
O.OP2	056176	9044	9134#	9145					
O.ORPC	055702	9036#	9101						
O.P	057511	9011*	9225*	9250	9252*	9301*	9435#		
O.PRI	055466	9003	9274	9278	9488#				
O.PROC	056574	9107	9250#						
O.PR1	056616	9254	9256#						
O.RALL	055750	9014	9049	9059#					
O.RCSR=	177560	8978#	9326	9328*	9338	9340	9342*	9372	
O.RDB =	177562	8977#	9374						
O.REGT	055632	9019#	9098						
O.RSE1	057210	9339	9342#						
O.RSP	055642	9021#	9024						

O.RST	055546	8992	9001#						
U.RSTT	057160	9231	9257	9335#					
O.RST1	055576	9000	9008#						
O.RTIT	056572	9010*	9245#						
O.SCAN	056004	9030	9073#	9084					
O.SP	055674	9022	9031#						
O.SPC	057240	9352#	9360						
O.SP1	055662	9027#	9032						
O.STM =	000340	8967#	9012	9230	9256	9258			
O.STRT	055534	8991	8998#						
O.SVR	057070	9001	9270	9312#					
O.SVTT	057126	9290	9302	9326#					
O.T	057510	9232*	9259*	9271	9434#				
O.TBIT	056516	9232#	9272						
O.TBT =	000020	8968#	9233	9260	9296				
O.TC	057454	9414	9420#						
O.TCLS	057436	9036	9126	9132	9143	9413#			
O.TCSR=	177564	8980#	9327	9329*	9336	9343*	9396	9402	
O.TDB =	177566	8979#	9398*	9404*					
O.TL	057535	9020	9023	9031	9461#	9467			
O.TRTC	057552	9059	9235	9474#					
O.TVEC=	000014	8966#	8994	9012*	9013*				
O.TYPE	057354	9294	9365	9388#	9392	9430			
O.TYP1	057434	9389	9400	9409#					
O.TYP2	057414	9402#	9403	9406					
O.UIN	055502	9002	9234*	9273	9498#				
O.UPC	055462	8995*	9229*	9244	9268*	9285	9287*	9486#	
O.URO	055444	8998	9028	9479#					
O.USP	055460	8999*	9313*	9314	9485#				
O.UST	055464	8993*	9233*	9243	9260*	9269*	9276	9296*	9487#
O.WDS	056312	9175	9177#						
O.WDS2	056330	9182#	9204						
O.WDS3	056360	9193#	9206	9212	9221				
O.WDS4	056416	9193	9203#						
O.WRD	056124	9096	9112#						
O.WRDA	056150	9113	9121#						
O.WRD1	056132	9115#	9122	9139					
O.WSCH	056310	9104	9176#						
O.XXX	057506	9312*	9320	9433#					
O.45	055610	9009	9011#						
PACK =	000003	1112#	3485	3655	3781	3967	4072	4851	
PARAM	001356	1897#	2852*	2855*	2966				
PARSRT	010002	1285	2852#						
PAT =	000020	1193#	3699						
PCA =	004000	1200#							
PCD =	010000	1201#							
PCLKF	005504	2060#	3011*	3018*	5982	6003			
PCVEC	001352	1890#	3012	3019					
PCYL	001372	1907#							
PFSRT	012664	3426#	6256						
PGE =	002000	1148#							
PIP =	020000	1186#							
PIRQ =	177772	991#							
PIRQVE=	000240	1085#							
PKRB	001344	1886#							
PKS	001340	1884#	3010	3017	5987*	6007*			

PKSB	001342	1885#	5986*																	
PPTP	005456	2044#	2957*																	
PRGSRT	010016	2853	2856#	5201																
PRO =	000000	1008#	2858	3049																
PR1 =	000040	1009#																		
PR2 =	000100	1010#																		
PR3 =	000140	1011#																		
PR4 =	000200	1012#																		
PR5 =	000240	1013#	1883																	
PR6 =	000300	1014#	3022																	
PR7 =	000340	1015#	2923	2929	3073	6251	6252													
PS =	177776	988#	989																	
PSEC	001416	1920#																		
PSW =	177776	989#																		
PWRVEC=	000024	1080#	2892*	2893*	6241*	6250*	6251*													
RCYLA	026774	5505	5682#																	
RCYLD	026722	5502	5669#																	
RDCHR =	104410	6062	6079	6940	7322#															
RDCYLA	026652	5657#																		
RDCYLD	026566	5641#	5737	5741	5744															
RDDATA=	000021	1119#	4119																	
RDGATE=	100000	1204#																		
RDHEAD=	000025	1121#																		
RDLIN =	104411	5169	7014	7323#																
RDOCT =	104412	5214	5227	7324#																
RDSEC	026516	5629#	5713	5717	5720	5886														
RDTAB	004322	1964#																		
RDY =	000200	1127#	5277	5291																
RECAL =	000013	1116#	6130																	
RESREG=	104414	7074	7126	7326#	8937															
RESVEC=	000010	1075#																		
RHEAD	027070	5510	5702#																	
RHTAB	001712	1959#	5893	5895	5900	5903	8688													
RKASOF=	000016	1098#	3063	5316	5352	5379														
RKBA =	000004	1093#	3059	4113*	4153*	4171*	4250*	4277*	4326*	4379*	4410*	4470*	4473*	4559*						
		4564*	4568	4620*	4623*	4626	4711*	4734*	4791*	4794*	4874*	4879*	4883	4935*						
		4938*	4941	5312																
RKCS1 =	000000	1091#	3056	3167*	3280*	3483*	3634*	3653*	3709*	3718*	3724*	3764*	3779*	3793*						
		3887*	3935*	3965*	4070*	4299*	4372*	4849*	5052*	5056*	5252*	5263*	5277	5291						
		5309	5563*	5571*	5579*	5587*	5600*	6128*	6143*											
RKCS2 =	000010	1095#	3057	3140*	3160*	3164*	3273*	3277*	3484*	3654*	3726	3765*	3770	3780*						
		3794*	3799	3851	3863	3871	3883	3889	3937	3966*	4071*	4278*	4327*	4380*						
		4411*	4474*	4565*	4624*	4735*	4795*	4850*	4880*	4939*	5053*	5060	5310	5551*						
		5564*	5572*	5580*	5588*	5613*	5617*	6129*	6144*											
RKDA =	000006	1094#	3060	4115*	4155*	4173*	4253*	4275*	4330*	4382*	4414*	4476*	4535*	4540*						
		4544*	4546	4567*	4608*	4612*	4616*	4617	4714*	4797*	4882*	4923*	4927*	4931*						
		4932	5313																	
RKDB =	000024	1100#	3065	4340	4424	4579	4637	4894	4952											
RKDC =	000020	1099#	3064	4033*	4117*	4477*	4569*	4627*	4737*	4798*	4884*	4942*	5317							
RKDS =	000012	1096#	3061	5314																
RKECPS=	000030	1104#	3069	5321																
RKECPT=	000032	1105#	3070	5322																
RKER =	000014	1097#	3062	5315																
RKMR1 =	000026	1101#	3066	3444*	3556*	3607*	3635*	3699*	5318	5358*	5363*	5383*	5387*	5565*						
		5573*	5581*	5618*	5629*	5641*	5657*	5762*	5780*	6134*										
RKMR2 =	000034	1102#	3067	5295	5300	5319														

B  
C  
D  
E  
F  
G  
H  
I  
J  
K  
L  
M  
N  
O  
P  
Q  
R  
S  
T  
U  
V  
W  
X  
Y  
Z

RKMR3 = 000036	1103#	3068	5296	5301	5320									
RKPRI 001336	1883#	5243												
RKVEC 001334	1882#	2993*	5231*	5234*	5241									
RKWC = 000002	1092#	3058	4118*	4251*	4279*	4328*	4381*	4412*	4475*	4566*	4625*	4712*	4736*	
	4796*	4881*	4940*	5311										
RLS = 000010	1141#													
RSEC 027032	5509	5692#												
RTT = 000006	8970#													
SAVREG= 104413	7059	7100	7325#	8879										
SBPAR 030306	5430	5434	5438	5442	5446	5450	5454	6020#						
SCLR = 000040	1143#	3140	3160	3273	5613									
SCOP1 = 104415	3157	3270	7327#											
SCOP1\$ 030360	6044#	7327												
SEC 001410	1917#	5996*												
SECFLG 027600	5813	5833#												
SECNT 001414	1919#													
SECTOR 001422	1922#	4274*	4275	4293*	4294	4330	4382	4414	5631*	5632*	5633*	5634*	5635*	
	5636*	5692*	5693*	5694*	5695*	5696*	5697*	5714	5718	5721	5887*	5891	5892*	
	5893*	5905	5915*	5916	5917*	5918*								
	1118#	4034	4191											
SEEK = 000017	1111#	3165	3278	3766	3795	5054	5552	5598						
SELDRV= 000001	2994	5232	5241#											
SETINT 024430	2062#	2995*	3150	5187*	5207*									
SIZFLG 005510	1158#													
SKI = 000002	5884#													
SORT 027726	8915	8924	8935	8945#										
SPACE2 055420	1903#	6064	6081											
SPBAR = 000040	1115#	3995	6114											
SRTSPL= 000011	1960#	5896												
SRTTAB 002116	8964#	8993	8994*	9007*	9230*	9256*	9258*	9284*						
ST = 177776	979#	2857	2884	3047	3098	3158	3266	3271	3369	3436	3476	3504	3536	
STACK = 001100	3689	3836	3911	3986	4104	4217	4675	4993	5023	5515	6216	6253	8942	
	1283	2855#												
START 010012	2947#													
START1 010420	990#													
STKLMT= 177774	6093#	6781	6864											
STOP 030500	3758	6095#												
STOP1 030502	2955	2966#												
ST2 010470	2956	2984#												
ST3 010534	2968	2989	2992#											
ST4 010566	2975	2997#	3137	3206	5130	6165								
ST5 010612	5119	5124#												
ST5X1 024056	3440	3478	3508	3538	3574	3599	3614	3665	3691	3810	3838	3860	3880	
SUBCI R 026436	3913	3948	3988	4007	4030	4067	4106	4141	4187	4222	4271	4323	4365	
	4407	4449	4465	4550	4677	4731	4771	4786	4841	4865	5043	5084	5613#	
	6095	6107												
SVAL = 100000	1188#													
SWR 001140	1790#	2882	2904*	2906	2912*	2919*	5156	5466	5476	5486	5496	6044	6179	
	6187	6192	6212	6335	6349	6351	6357	6364	6402	6409	6421	6425	6784	
	6823	6878*	8880	8938										
	1281#	2912	5156	6784	6823	6846								
SWREG 000176	1043#													
SWO = 000001	1033#	1043												
SWO0 = 000001	1032#	1042												
SWO1 = 000002	1031#	1041												
SWO2 = 000004	1030#	1040												
SWO3 = 000010														

ABCDEFGHIJKLMNPOQRSTUVWXYZ















CALIB	1409#	6128													
CHECK	1366#	3566	3591	3645	4056	4134	4311	4353	4399	4437	4592	4650	4754	4826	4907
	4965														
COMMEN	1#	1086#													
CWD2	1381#														
DRCLR	1394#	6142													
ENDCOM	1#	1086#													
ERROR	980#	3077	3135	3148	3163	3172	3195	3204	3223	3227	3238	3242	3276	3285	3303
	3330	3334	3338	3342	3441	3479	3487	3491	3509	3513	3515	3539	3560	3569	3570
	3571	3572	3575	3585	3594	3595	3596	3597	3600	3611	3615	3625	3639	3648	3649
	3650	3651	3657	3661	3666	3670	3672	3692	3703	3707	3712	3722	3728	3768	3783
	3797	3811	3839	3853	3861	3865	3873	3881	3885	3914	3933	3941	3949	3953	3958
	3962	3969	3973	3989	3997	4001	4008	4022	4027	4031	4036	4040	4044	4049	4059
	4060	4061	4062	4065	4068	4074	4078	4107	4121	4125	4137	4138	4139	4140	4142
	4148	4150	4188	4193	4197	4201	4223	4240	4259	4263	4272	4283	4296	4302	4314
	4315	4316	4317	4320	4324	4334	4341	4344	4356	4357	4358	4359	4366	4376	4385
	4390	4393	4402	4403	4404	4405	4408	4418	4425	4428	4440	4441	4442	4443	4450
	4466	4480	4486	4499	4504	4551	4573	4580	4583	4595	4596	4597	4598	4631	4638
	4641	4653	4654	4655	4656	4678	4701	4719	4723	4732	4740	4757	4758	4759	4760
	4766	4772	4787	4801	4808	4819	4829	4830	4831	4832	4842	4846	4853	4857	4866
	4888	4895	4898	4910	4911	4912	4913	4946	4953	4956	4968	4969	4970	4971	5044
	5065	5085	5530	5535	5539	5554	5567	5575	5583	5590	5616	6096	6108	6116	6120
	6132	6138	6141	6147	6150	6209									
ESCAPE	1#	1086#													
F.EAB	1346#	4126	4303	4345	4429	4584	4642	4746	4899	4957					
GETPRI	1#	1086#													
GETSWR	1#	1086#	5151												
HDCHK3	1483#														
HDTBL	1496#														
LOOP	1330#	3156	3269												
MSG	3035#	3037	3081#	3083	3249#	3251	3358#	3360	3427#	3429	3468#	3470	3495#	3497	3520#
	3522	3675#	3677	3826#	3828	3897#	3899	3977#	3979	4082#	4084	4206#	4208	4664#	4666
	4981#	4983	5000#	5002											
MULT	1#	1086#													
NEWTST	1#	1086#	3035	3081	3249	3358	3427	3468	3495	3520	3675	3826	3897	3977	4082
	4206	4664	4981	5000											
OWNTAG	1324#	1881													
POP	1#	1086#	6586	6646	6647	7037	7233	7275							
PUSH	1#	1086#	6545	6607	6609	6630	7011	7204	7255						
QKPACK	1730#	3483	3653	3964	4070	4849									
QKSEEK	1508#	4190													
QKSRT	1706#	3987	6105												
QKUNLD	1746#	3507	3664												
RDATA	1577#														
RDHDR	1454#														
REPORT	1#	1086#													
SBOUND	1651#	4549	4863												
SCOPE	981#	3045	3096	3264	3367	3434	3474	3502	3534	3687	3834	3909	3984	4102	4215
	4673	4991	5021	5095											
SETPRI	1#	1086#	6915												
SETTRA	7305#	7314	7315	7316	7317	7319	7321	7322	7323	7324	7325	7326	7327		
SETUP	1#	1086#	2878												
SKIP	1#	1086#	3023	3074	3300	3413	3419	3439	3482	3884	3890	4159	4997		
SLASH	1#	1086#													
SPACE	1086#														
STARS	1#	1086#	1292	1303	1305	1312	1765	1812	1815	3035	3044	3081	3095	3249	3263



.STRAP	1#	941#	7282
.STYPB	1#		
.STYPD	1#	941#	6533
.STYPE	1#	941#	6437
.STYPO	1#	941#	6657
.S4OCA	1#		
.1170	1#		

. ABS. 057554 000

ERRORS DETECTED: 0

CZR6JF,CZR6JF.LST/SOL/CRF/NL:TOC=SYSMAC.SML,CZR6JF.P11  
RUN-TIME: 20 24 2 SECONDS  
RUN-TIME RATIO: 178/47=3.7  
CORE USED: 43K (86 PAGES)