

RK611

UNIBUS RK6 DRIVE PART 3
CZR6JE0

AH-9126E-MC
COPYRIGHT © 76-78
FICHE 1 OF 1

MAR 1978
digital
MADE IN USA

The main body of the document consists of a grid of 15 columns and 15 rows of small, illegible technical diagrams or data tables. Each cell in the grid contains a small, dark rectangular area with some faint, unreadable text or symbols. The overall appearance is that of a microfiche or a high-resolution scan of a technical manual page where the individual elements are too small to be legible.

88
89
90
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

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 SUCCESSFULLY FOLLOWED BY THE RK06 DRIVE DIAGNOSTICS - PARTS 1 & 2.

3.0 PROGRAM CONSIDERATIONS

144
145
146
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

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 AU'0 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

200
201
202
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

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

3. SWITCH 1 (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

256
257
258
259
260
261
262
263
264
265
266
267
268
269
270
271
272
273
274
275
276
277
278
279
280
281
282
283
284
285
286
287
288
289
290
291
292
293
294
295
296
297
298
299
300
301
302
303
304
305
306
307
308
309
310
311

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

312
313
314
315
316
317
318
319
320
321
322
323
324
325
326
327
328
329
330
331
332
333
334
335
336
337
338
339
340
341
342
343
344
345
346
347
348
349
350
351
352
353
354
355
356
357
358
359
360
361
362
363
364
365
366
367

TTY PRINTER STATUS REG 1150 177564
TTY PRINTER BUFFER 1152 177566

4.0 OPERATING PROCEDURE & CONTROL FUNCTIONS

4.1 PROGRAM LOADING

THE PROGRAM CAN BE LOADED FROM PAPER TAPE USING STANDARD PROCEDURE FOR ABSOLUTE LOADER TAPES; OR FROM ANY MEDIA SUPPORTED BY XXDP.

4.1.1 LOAD THE STARTING ADDRESS (SEE SEC 4.2).

4.1.2 SET SWITCH REGISTERS AS DESIRED (SEE SEC 4.3).

4.1.3 SET DRIVES TO BE TESTED IN THE 'LOAD' CONDITION & WITH THE APPROPRIATE PORT SELECTED & WRITE LOCK DISABLED. DRIVES NOT TO BE TESTED MUST HAVE BOTH PORTS DESELECTED.

NOTE: THE DRIVE WILL NOT RESPOND TO THE 'START SPINDLE' COMMAND IF THE RUN/STOP SWITCH IS IN THE 'STOP' POSITION.

4.1.4 PRESS 'START'

THE PROGRAM WILL IDENTIFY ITSELF AND WILL BEGIN A DIALOGUE WITH THE OPERATOR TO DETERMINE DRIVES TO BE TESTED (SEE SEC 4.5).

THE PROGRAM BEGINS TESTING ONLY THOSE DRIVES SPECIFIED BY THE INPUT DIALOGUE. IF A SPECIFIED DRIVE CANNOT BE FOUND BY THE PROGRAM IT WILL BE FLAGGED AS AN ERROR THAT THE DRIVE IS NOT AVAILABLE. THEN BEGINNING WITH THE LOWEST NUMERICAL DRIVE AND PROCEEDING IN SEQUENTIAL ORDER, ALL VALID DRIVES WILL BE TESTED. ONE PASS THROUGH THE TEST SEQUENCE WILL BE PERFORMED ON EACH DRIVE BEFORE MOVING TO THE NEXT DRIVE IN SEQUENCE. THE DRIVE TO BE TESTED WILL BE TYPED AT THE BEGINNING OF EACH PASS. "END OF PASS" WILL BE TYPED AFTER TESTING ALL DRIVES.

4.2 STARTING LOCATIONS

LOCATION 200 - STARTING ADDRESS TO DEFAULT THE BUSS ADDRESS & THE CONTROLLER INTERRUPT VECTOR & TEST ALL DRIVES IN THE 'DRIVE PRESENT' CONDITION.

NOTE: THE DRIVE PRESENT CONDITION IS:

368
369
370
371
372
373
374
375
376
377
378
379
380
381
382
383
384
385
386
387
388
389
390
391
392
393
394
395
396
397
398
399
400
401
402
403
404
405
406
407
408
409
410
411
412
413
414
415
416
417
418
419
420
421
422
423

- 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>

424
425
426
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

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 = NNNN IN 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

480
481
482
483
484
485
486
487
488
489
490
491
492
493
494
495
496
497
498
499
500
501
502
503
504
505
506
507
508
509
510
511
512
513
514
515
516
517
518
519
520
521
522
523
524
525
526
527
528
529
530
531
532
533
534
535

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.

UNIBUS RK06-RK07 DRIVE DIAGNOSTIC
PART 3
CZR6JED

DRIVES TO BE TESTED: 1,3<CR>

TYPE IN BUSS ADDRESS IF NOT 177440 <CR>

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.

UNIBUS RK06-RK07 DRIVE DIAGNOSTIC
PART 3
CZR6JEO

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 A ^/APT/XXDP

536
537
538
539
540
541
542
543
544
545
546
547
548
549
550
551
552
553
554
555
556
557
558
559
560
561
562
563
564
565
566
567
568
569
570
571
572
573
574
575
576
577
578
579
580
581
582
583
584
585
586
587
588
589
590
591

592
593
594
595
596
597
598
599
600
601
602
603
604
605
606
607
608
609
610
611
612
613
614
615
616
617
618
619
620
621
622
623
624
625
626
627
628
629
630
631
632
633
634
635
636
637
638
639
640
641
642
643
644
645
646
647

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

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 <IE>. ONLY AT THE
BEGINNING OF EACH TEST, AS INSTRUCTED BY THE TYPEOUT.

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 'MCNITOR 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

648
649
650
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
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

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.

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
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 EX'STENT 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

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

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

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.

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

816
817
818
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
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

ERROR.

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

F02

CZR6JED UNIBUS RK6 DR PRT3
CZR6JE.P11 25-JAN-78 12:08

MACY11 30A(1052) 25-JAN-78 12:16 PAGE 18

SEQ 0018

928
929
930
931

[END OF DOCUMENT]

%

932
933
934
935
936
937
938
939
940
941
942
943
944
945
946
947
948
949
950
951
952
953
954
955
956
957
958
959
960
961
962
963
964
965
966
967
968
969
970
971
972
973
974
975
976

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  CZR6JED UNIBUS RK6 DR PRT3
*COPYRIGHT (C) 1976,1977
*DIGITAL EQUIPMENT CORP.
*MAYNARD, MASS. 01754
*
*PROGRAM BY GARY PAPIAZIAN
*
*THIS PROGRAM WAS ASSEMBLED USING THE PDP-11 MAINDEC SYSMAC
*PACKAGE (MAINDEC-11-DZQAC-C3), JAN 19, 1977.

```

```

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

```

977      .SBTTL  BASIC DEFINITIONS
978
979      ;*INITIAL ADDRESS OF THE STACK POINTER *** 1100 ***
980      001100  STACK= 1100
981      .EQUIV  EMT,ERROR      ;;BASIC DEFINITION OF ERROR CALL
982      .EQUIV  IOT,SCOPE      ;;BASIC DEFINITION OF SCOPE CALL
983
984      ;*MISCELLANEOUS DEFINITIONS
985      000011  HT= 11          ;;CODE FOR HORIZONTAL TAB
986      000012  LF= 12          ;;CODE FOR LINE FEED
987      000015  CR= 15          ;;CODE FOR CARRIAGE RETURN
988      000200  CRLF= 200       ;;CODE FOR CARRIAGE RETURN-LINE FEED
989      177776  PS= 177776     ;;PROCESSOR STATUS WORD
990      .EQUIV  PS,PSW
991      177774  STKLMT= 177774  ;;STACK LIMIT REGISTER
992      177772  PIRQ= 177772   ;;PROGRAM INTERRUPT REQUEST REGISTER
993      177570  DSWR= 177570   ;;HARDWARE SWITCH REGISTER
994      177570  DDISP= 177570  ;;HARDWARE DISPLAY REGISTER
995
996      ;*GENERAL PURPOSE REGISTER DEFINITIONS
997      000000  R0= %0          ;;GENERAL REGISTER
998      000001  R1= %1          ;;GENERAL REGISTER
999      000002  R2= %2          ;;GENERAL REGISTER
1000     000003  R3= %3          ;;GENERAL REGISTER
1001     000004  R4= %4          ;;GENERAL REGISTER
1002     000005  R5= %5          ;;GENERAL REGISTER
1003     000006  R6= %6          ;;GENERAL REGISTER
1004     000007  R7= %7          ;;GENERAL REGISTER
1005     000006  SP= %6         ;;STACK POINTER
1006     000007  PC= %7         ;;PROGRAM COUNTER
1007
1008     ;*PRIORITY LEVEL DEFINITIONS
1009     000000  PR0= 0           ;;PRIORITY LEVEL 0
1010     000040  PR1= 40         ;;PRIORITY LEVEL 1
1011     000100  PR2= 100        ;;PRIORITY LEVEL 2
1012     000140  PR3= 140        ;;PRIORITY LEVEL 3
1013     000200  PR4= 200        ;;PRIORITY LEVEL 4
1014     000240  PR5= 240        ;;PRIORITY LEVEL 5
1015     000300  PR6= 300        ;;PRIORITY LEVEL 6
1016     000340  PR7= 340        ;;PRIORITY LEVEL 7
1017
1018     ;*"SWITCH REGISTER" SWITCH DEFINITIONS
1019     100000  SW15= 100000
1020     040000  SW14= 40000
1021     020000  SW13= 20000
1022     010000  SW12= 10000
1023     004000  SW11= 4000
1024     002000  SW10= 2000
1025     001000  SW09= 1000
1026     000400  SW08= 400
1027     000200  SW07= 200
1028     000100  SW06= 100
1029     000040  SW05= 40
1030     000020  SW04= 20
1031     000010  SW03= 10
1032     000004  SW02= 4

```

1033 000002
 1034 000001
 1035
 1036
 1037
 1038
 1039
 1040
 1041
 1042
 1043
 1044
 1045
 1046
 1047 100000
 1048 040000
 1049 020000
 1050 010000
 1051 004000
 1052 002000
 1053 001000
 1054 000400
 1055 000200
 1056 000100
 1057 000040
 1058 000020
 1059 000010
 1060 000004
 1061 000002
 1062 000001
 1063
 1064
 1065
 1066
 1067
 1068
 1069
 1070
 1071
 1072
 1073
 1074
 1075 000004
 1076 000010
 1077 000014
 1078 000014
 1079 000014
 1080 000020
 1081 000024
 1082 000030
 1083 000034
 1084 000060
 1085 000064
 1086 000240
 1087
 1088

SW01= 2
 SW00= 1
 .EQUIV SW09,SW9
 .EQUIV SW08,SW8
 .EQUIV SW07,SW7
 .EQUIV SW06,SW6
 .EQUIV SW05,SW5
 .EQUIV SW04,SW4
 .EQUIV SW03,SW3
 .EQUIV SW02,SW2
 .EQUIV SW01,SW1
 .EQUIV SW00,SW0

.*DATA BIT DEFINITIONS (BIT00 TO BIT15)

BIT15= 100000
 BIT14= 40000
 BIT13= 20000
 BIT12= 10000
 BIT11= 4000
 BIT10= 2000
 BIT09= 1000
 BIT08= 400
 BIT07= 200
 BIT06= 100
 BIT05= 40
 BIT04= 20
 BIT03= 10
 BIT02= 4
 BIT01= 2
 BIT00= 1
 .EQUIV BIT09,BIT9
 .EQUIV BIT08,BIT8
 .EQUIV BIT07,BIT7
 .EQUIV BIT06,BIT6
 .EQUIV BIT05,BIT5
 .EQUIV BIT04,BIT4
 .EQUIV BIT03,BIT3
 .EQUIV BIT02,BIT2
 .EQUIV BIT01,BIT1
 .EQUIV BIT00,BIT0

.*BASIC "CPU" TRAP VECTOR ADDRESSES

ERRVEC= 4 ; TIME OUT AND OTHER ERRORS
 RESVEC= 10 ; RESERVED AND ILLEGAL INSTRUCTIONS
 TBITVEC= 14 ; "T" BIT
 TRTVEC= 14 ; TRACE TRAP
 BPTVEC= 14 ; BREAKPOINT TRAP (BPT)
 IOTVEC= 20 ; INPUT/OUTPUT TRAP (IOT) **SCOPE**
 PWFVEC= 24 ; POWER FAIL
 EMTVEC= 30 ; EMULATOR TRAP (EMT) **ERROR**
 TRAPVEC= 34 ; "TRAP" TRAP
 TKVEC= 60 ; TTY KEYBOARD VECTOR
 TPVEC= 64 ; TTY PRINTER VECTOR
 PIRQVEC= 240 ; PROGRAM INTERRUPT REQUEST VECTOR

.SBTTL RK06 CONTROLLER REGISTER DEFINITION


```

1089
1090
1091
1092          000000          RKCS1= 0          ; CONTROL AND STATUS REGISTER 1
1093          000002          RKWC= 2          ; WORD COUNT REGISTER
1094          000004          RKBA= 4          ; BUS ADDRESS REGISTER
1095          000006          RKDA= 6          ; DESIRED TRACK SECTOR REGISTER
1096          000010          RKCS2= 10         ; CONTROL AND STATUS REGISTER 2
1097          000012          RKDS= 12         ; DRIVE STATUS REGISTER
1098          000014          RKER= 14         ; ERROR REGISTER
1099          000016          RKASOF= 16        ; ATTENTION SUMMARY AND OFFSET REGISTER
1100          000020          RKDC= 20         ; DESIRED CYLINDER REGISTER
1101          000024          RKDB= 24         ; DATA BUFFER
1102          000026          RKMRI= 26        ; MAINTENANCE REGISTER 1
1103          000034          RKMRI= 34        ; MAINTENANCE REGISTER 2 (MESSAGE LINE A)
1104          000036          RKMRI= 36        ; MAINTENANCE REGISTER 3 (MESSAGE LINE B)
1105          000030          RKECPS= 30       ; ECC POSITION INFORMATION
1106          000032          RKECPT= 32       ; ECC PATTERN INFORMATION
1107
1108          .SBTTL CONTROL AND STATUS REGISTER 1 BITS (RKCS1:0)
1109
1110          ; DRIVE COMMANDS
1111
1112          000001          SELDRV= 1         ; SELECT DRIVE (GET STATUS)
1113          000003          PACK= 3          ; PACK ACKNOWLEDGE
1114          000005          CLEAR= 5         ; DRIVE CLEAR
1115          000007          UNLOAD= 7        ; UNLOAD
1116          000011          SRTSPL= 11       ; START SPINDLE
1117          000013          RECAL= 13        ; RECALIBRATE
1118          000015          OFFSET= 15       ; OFFSET
1119          000017          SEEK= 17         ; SEEK
1120          000021          RDATA= 21        ; READ DATA
1121          000023          WRDATA= 23       ; WRITE DATA
1122          000025          RDHEAD= 25       ; READ HEADER
1123          000027          WRHEAD= 27       ; WRITE HEADER AND DATA
1124          000031          WRTCHK= 31       ; WRITE CHECK
1125
1126          000001          GO= BIT0         ; GO BIT
1127          000100          IE= BIT6         ; INTERRUPT ENABLE
1128          000200          RDY= BIT7         ; CONTROLLER READY
1129          000400          BA16= BIT8        ; BUS ADDRESS BIT 16
1130          001000          BA17= BIT9        ; BUS ADDRESS BIT 17
1131          002000          CDT= BIT10       ; CONTROLLER DRIVE TYPE (0=RK06,1=RK07)
1132          004000          CTO= BIT11       ; CONTROLLER TIMEOUT
1133          010000          CFMT= BIT12      ; CONTROLLER DRIVE FORMAT (0=22 SECTOR, 1=20 SECTOR)
1134          020000          DCPAR= BIT13     ; SERCON PARITY ERROR DETECTED BY CONTROLLER
1135          040000          DI= BIT14        ; DRIVE INTERRUPT
1136          100000          CERR= BIT15      ; CONTROLLER ERROR
1137          100000          CCLR= BIT15      ; CONTROLLER CLEAR
1138
1139          .SBTTL CONTROL AND STATUS REGISTER 2 BITS (RKCS2:10)
1140
1141          000007          DRVMSK= 7        ; MASK FOR DRIVE SELECTION CODE
1142          000010          RLS= BIT3        ; DESELECT OR RELEASE DRIVE IN BITS 0-2
1143          000020          BAI= BIT4        ; BUS ADDRESS INCREMENT INHIBIT
1144          000040          SCLR= BITS       ; SUBSYSTEM CLEAR CONTROLLER AND ALL DRIVES

```

1145 000100
1146 000200
1147 000400
1148 001000
1149 002000
1150 004000
1151 010000
1152 020000
1153 040000
1154 100000

IR= BIT5 ; INPUT READY
OR= BIT7 ; OUTPUT READY
UFE= BIT8 ; UNIT FIELD ERROR
MDS= BIT9 ; MULTIPLE DRIVE SELECT
PGE= BIT10 ; PROGRAMMING ERROR
NEM= BIT11 ; NON-EXISTENT MEMORY
NED= BIT12 ; NON-EXISTENT DRIVE
UPE= BIT13 ; UNIBUS PARITY ERROR
WCE= BIT14 ; WRITE CHECK ERROR
DLT= BIT15 ; DATA LATE ERROR

.SBTTL ERROR REGISTER BIT DEFINITION (RKER:14)

1158 000001
1159 000002
1160 000004
1161 000010
1162 000020
1163 000040
1164 000100
1165 000200
1166 000400
1167 001000
1168 002000
1169 004000
1170 010000
1171 020000
1172 040000

ILF= BIT0 ; ILLEGAL FUNCTION CODE
SKI= BIT1 ; SEEK INCOMPLETE
NXF= BIT2 ; NON-EXECUTABLE FUNCTION
DRPAR= BIT3 ; DRIVE DETECTED SERCON PARITY ERROR
FMTE= BIT4 ; FORMAT ERROR
DTYE= BIT5 ; DRIVE TYPE ERROR
ECH= BIT6 ; ECC HARD
BSE= BIT7 ; BAD SECTOR ERROR
HVRC= BIT8 ; HEADER VRC ERROR
COE= BIT9 ; CYLINDER ADDRESS OVERFLOW ERROR
IDAE= BIT10 ; INVALID DISK ADDRESS ERROR: HEAD/CYL
WLE= BIT11 ; WRITE LOCK ERROR
DTE= BIT12 ; DRIVE TIMING ERROR
OPI= BIT13 ; OPERATION (SEARCH) INCOMPLETE
UNS= BIT14 ; DRIVE UNSAFE

```

1173      100000      DCK=      BIT15      ;DATA CHECK
1174
1175      .SBTTL STATUS REGISTER BIT DEFINITION (RKDS:12)
1176
1177      000001      DRA=      BIT0      ;DRIVE AVAILABLE (CONTROLLER IS SET IF
1178      ; THIS BIT IS RESET)
1179      000004      OFST=     BIT2      ;DRIVE OFFSET
1180      000010      ACLO=     BIT3      ;AC LOW
1181      000020      DCLO=     BIT4      ;DC LOW
1182      000040      DROT=     BIT5      ;DRIVE OFF TRACK
1183      000100      VV=       BIT6      ;VOLUME VALID
1184      000200      DRDY=     BIT7      ;DRIVE READY
1185      000400      DDT=      BIT8      ;DRIVE TYPE (0=RK06,1=RK07)
1186      004000      WRL=      BIT11     ;WRITE LOCK
1187      020000      PIP=      BIT13     ;POSITIONING IN PROGRESS
1188      040000      DSC=      BIT14     ;DRIVE STATUS CHANGE
1189      100000      SVAL=     BIT15     ;STATUS VALID
1190
1191      .SBTTL MAINTENANCE REGISTER 1 BIT DEFINITION (RKMR1:22)
1192
1193      000017      MESMSK= 17      ;MESSAGE MASK
1194      000020      PAT=      BIT4      ;FORCE EVEN PARITY ON SERCON MESSAGE LINES
1195      000040      DMD=      BITS      ;DIAGNOSTIC MODE
1196      000100      MSP=      BIT6      ;MAINTENANCE SECTOR PULSE
1197      000200      MIND=     BIT7      ;MAINTENANCE INDEX
1198      000400      MCLK=     BIT8      ;MAINTENANCE CLOCK
1199      001000      MERD=     BIT9      ;MAINTENANCE ENCODED READ DATA
1200      002000      MEWD=     BIT10     ;MAINTENANCE ENCODED WRITE DATA
1201      004000      PCA=      BIT11     ;PRECOMPENSATION ADVANCE
1202      010000      PCD=      BIT12     ;PRECOMPENSATION DELAY
1203      020000      ECCW=     BIT13     ;ECC WORD IS BEING READ OR WRITTEN
1204      040000      WRTGAT=  BIT14     ;WRITE GATE
1205      100000      RDGATE=  BIT15     ;READ GATE
1206
1207      .SBTTL DEFINITION OF DRIVE STATUS BYTE 00 MESSAGE A (RKMR2:34)
1208
1209      000040      D.DRA=    BITS      ;DRIVE AVAILABLE
1210      000100      D.VV=     BIT6      ;VOLUME VALID
1211      000200      D.DRDY=   BIT7      ;DRIVE READY
1212      000400      D.DDT=    BIT8      ;DRIVE TYPE (0=RK06,1=RK07)
1213      001000      D.FORM=   BIT9      ;DRIVE FORMAT
1214      002000      D.OFF=    BIT10     ;OFFSET ON
1215      004000      D.WRL=    BIT11     ;WRITE LOCK
1216      010000      D.SPIN=   BIT12     ;SPINDLE ON
1217      020000      D.PIP=    BIT13     ;POSITIONING IN PROGRESS
1218      040000      D.DSC=    BIT14     ;DRIVE STATUS CHANGE
1219
1220      .SBTTL DEFINITION OF DRIVE STATUS BYTE 01 MESSAGE A (RKMR2:34)
1221
1222      000020      D.SSP=    BIT4      ;SERVO SIG PRESENT
1223      000040      D.HOHM=   BITS      ;HEADS HOME
1224      000100      D.BPHM=   BIT6      ;BRUSHES HOME
1225      000200      D.DOOR=   BIT7      ;DOOR INTERLOCKED
1226      000400      D.CART=   BIT9      ;CARTRIDGE INTERLOCK
1227      001000      D.SPOK=   BIT9      ;SPEED OK
1228      002000      D.FWD=    BIT10     ;FORWARD

```

M02

CZR6JED UNIBUS RK6 DR PRT3
CZR6JE.P11 25-JAN-78 12:08

MACY11 30A(1052) 25-JAN-78 12:16 PAGE 25
DEFINITION OF DRIVE STATUS BYTE 01 MESSAGE A (RKMR2:34)

SEQ 0025

1229	004000	D.REV= BIT11	; REVERSE
1230	010000	D.LOAD= BIT12	; HEADS LOADING
1231	020000	D.RTZ= BIT13	; RETURN TO ZERO
1232	040000	D.UNLD= BIT14	; HEADS UNLOADING
1233			
1234		.SBTTL DEFINITION OF DRIVE STATUS BYTE 00 MESSAGE B (RKMR3:36)	
1235			
1236	000040	D.IDAE= BIT5	; INVALID DISK ADDRESS ERROR:HEAD/CYL
1237	000100	D.ACLO= BIT6	; AC LOW
1238	000200	D.FLT= BIT7	; DRIVE FAULT
1239	000400	D.NXF= BIT8	; NON-EXECUTABLE FUNCTION CODE
1240	001000	D.PAR= BIT9	; DRIVE DETECTED SERCON PARITY ERROR
1241	002000	D.SKI= BIT10	; SEEK INCOMPLETE
1242	004000	D.WLE= BIT11	; WRITE LOCK ERROR
1243	010000	D.SPLS= BIT12	; SPEED LOSS
1244	020000	D.DROT= BIT13	; DRIVE OFF TRACK
1245	040000	D.UNS= BIT14	; R/W UNSAFE
1246			
1247		.SBTTL DEFINITION OF DRIVE STATUS BYTE 01 MESSAGE B (RKMR3:36)	
1248			
1249	000020	D.SECT= BIT4	; SECTOR ERROR
1250	000040	D.WCUR= BIT5	; WRITE CURRENT AND NO WRITE GATE
1251	000100	D.WGAT= BIT6	; WRITE GATE AND NO TRANSISTIONS
1252	000200	D.HOFL= BIT7	; HEAD FAULT
1253	000400	D.MHD= BIT8	; MULTIPLE HEAD SELECT
1254	001000	D.XERR= BIT9	; INDEX ERROR
1255	002000	D.TIB= BIT10	; TRIBIT ERROR
1256	004000	D.PLO= BIT11	; PLO ERROR
1257	010000	D.NMOV= BIT12	; SEEK AND NO MOTION
1258	020000	D.LIMD= BIT13	; LIMIT DETECT ON SEEK
1259	040000	D.SUNS= BIT14	; SERVO UNSAFE
1260			
1261		.SBTTL COMMON MASKS AND OTHER BITS: MESSAGE A (RKMR2:34)	
1262			
1263	000007	M.DRV= 7	; DRIVE CODE, ALL BYTES
1264	077770	M.SER= 77770	; DRIVE SERIAL #, BYTE 11
1265			
1266		.SBTTL COMMON MASKS AND OTHER BITS: MESSAGE B (RKMR3:36)	
1267			
1268	000003	M.ID= 3	; BYTE ID, ALL BYTES
1269	040000	M.ALGN= BIT14	; ALIGN SIGN, BYTE 10
1270	000760	M.SECT= 760	; SECTOR COUNT, BYTE 11
1271	007000	M.HEAD= 7000	; HEAD DECODE, BYTE 11
1272	100000	M.PAR= BIT15	; PARITY, MESS A/B, ALL BYTES

1273
1274
1275
1276
1277
1278
1279
1280
1281
1282
1283
1284
1285
1286
1287
1288
1289
1290
1291
1292
1293
1294
1295
1296
1297
1298
1299
1300
1301
1302
1303
1304
1305
1306
1307
1308
1309
1310
1311
1312
1313
1314
1315
1316
1317
1318
1319
1320
1321
1322
1323
1324
1325
1326
1327
1328

```

.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 0.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
;*****
      $X=.        ;;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.

$AFTHD: .WORD 0      ;;TWO HIGH BITS OF 18 BIT MAILBOX ADDR.
$HIBTS: .WORD 0
$MBADR: .WORD $MAIL  ;;ADDRESS OF APT MAILBOX (BITS 0-15)
$TSTM:  .WORD 300.   ;;RUN TIM 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 FO' EACH ADDITIONAL UNIT
      .WORD $ETEND-$MAIL/2 ;;LENGTH MAILBOX-ETABLE(WORDS)

.LIST MD

```


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
1384

```
;USE LOOP X TO OMIT SUBCLR
;MACRO LOOP A
    SCOP1
    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 AC, 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' & 'HEADA'
;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
```

```

1385      JSR      PC,GSTAT
1386      TST      CYLDIF      ;SEE IF MSG A2=0
1387      BEQ      D           ;BR IF YES
1388      ERROR   A           ;MSG A2 NOT CLEARED C
1389      D:      TST      CYLADD ;SEE IF MSG B2=0
1390      BEQ      E           ;BR IF YES
1391      ERROR   B           ;MSG B2 NOT CLEARED C
1392      E:
1393      .ENDM   CWL '
1394
1395      .MACRO  DRCLR  ?A
1396
1397      MOV      #CCLR,RKCS1(R5)
1398      MOV      $UNIT,RKCS2(R5) ;DRIVE#
1399      MOV      #CLEAR,HCS1
1400      JSR      PC,DOCMD      ;DO DRIVE CLEAR CMD & GET CONTR RDY
1401      ERROR   151           ;NO RDY AFTER DRIVE CLEAR CMD
1402      JSR      PC,TSTATN     ;TEST FOR ATTN
1403      BR      A
1404      ERROR   154           ;ATTN NOT CLEARED AFTER DRIVE CLEAR CMD
1405      A:
1406
1407      .ENDM   DRCLR
1408
1409      .MACRO  CALIB  ?A
1410
1411      MOV      #CCLR,RKCS1(R5)
1412      MOV      $UNIT,RKCS2(R5)
1413      MOV      #RECAL,HCS1
1414      JSR      PC,DOCMD      ;DO RECAL CMD & GET CONTR RDY
1415      ERROR   124           ;RDY NOT SET AFTER RECAL CMD
1416
1417      MOV      #1,RKMR1(R5)   ;SELECT WORD 1
1418      JSR      PC,GSTAT
1419      BIT      #D.RTZ,HMR2
1420      BNE     A
1421      ERROR   214           ;RTZ NOT SET DURING RECAL CMD
1422      A:      MOV      T10,TEMP2 ;SETUP TIMEOUT
1423      JSR      PC,FATT1      ;FIND ATTN
1424      ERROR   55           ;NO ATTN AFTER RECAL CMD
1425      DRCLR
1426
1427      .ENDM   CALIB
1428
1429
1430      ;
1431      ; A=WRHEAD/<CFMT!WRHEAD>
1432      ; USE WRHDR <A>,X TO OMIT CHECKING A0, B0, A1, B1
1433      ;
1434      .MACRO  WRHDR  A,C,'D
1435
1436      MOV      #<A>,HCS1
1437      JSR      PC,DATCMD     ;DO DATA X FOR CMD & GET CONTR RDY
1438      ERROR   200           ;NO RDY AFTER WRITE HEADER CMD
1439      JSR      PC,GSTAT      ;GET FRESH STATUS
1440      BIT      #CERR,HCS1

```

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
1496

```

        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,RO
            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),(RO)+          ;1'ST WORD FROM SILO TO RHTAB
  MOV      RKDB(R5),(RO)+          ;2'ND WORD
  MOV      RKDB(R5),(RO)+          ;3'RD WORD

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

.MACRO      DCHK3      ?A
            RDHDR    RDHEAD,X
            CMP      RHTAB,↑OCYL          ;CHECK WORD 0 ONLY, CYL#
            BEQ      A          ;BR IF SAME
            ERROR    S1          ;WRONG CYL# ON HEADER
A:
  .ENDM      HDCHK3

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

```

```

1497 .MACRO HDTBL A,B,C
1498
1499     MOV A,CALADD ;SETUP
1500     MOV #B,HEAD ;TO FILL
1501     MOV #C,FORMAT ;HEADER
1502     JSR PC,FHDTAB ;TABLE
1503
1504 .ENDM HDTBL
1505
1506 ;
1507 ;QUICK SEEK. ENTER WITH CYL# IN RKDC
1508 ;
1509 .MACRO QKSEEK ?A
1510
1511     MOV #SEEK,HCSI
1512     JSR PC,DOCMD ;DO SEEK CMD & GET CONTR READY
1513     ERROR 131 ;NO RDY AFTER SEEK CMD
1514
1515     MOV T5000,TEMP1 ;SETUP TIMEOUT
1516     JSR PC,FATT2 ;FIND ATTN
1517     ERROR 132 ;NO ATTN AFTER SEEK CMD
1518
1519     BIT #CERR,HCSI
1520     BEQ A
1521     ERROR 210 ;CERR AFTER SEEK CMD
1522
1523 A:
1524
1525 .ENDM QKSEEK
1526
1527 ;
1528 ;A=WRDATA/<CFMT!WRDATA>
1529 ;C=ADDR TO JMP TO ATTEMPT TO WRITE ON ANOTHER SECTOR
1530 ;D=ADDR TO JMP TO BYPASS TEST
1531 ;E: IF BLANK WILL CHECK AD, BO, A1 & B1 AT THE END OF WRITING
1532 ;E: IF NON BLANK WILL OMIT CHECKING AD THRU B1
1533 ;
1534 ;
1535 .MACRO WDATA A,C,D,E,?F,?G,?H,?I
1536
1537     MOV #<A>,HCSI
1538     JSR PC,DATCMD ;DO DATA X FOR CMD & GET CONTR RDY
1539     ERROR 11 ;NO RDY AFTER WRITE DATA CMD
1540     JSR PC,GSTAT ;GET FRESH STATUS
1541     BIT #CERR,HCSI
1542     BEQ I ;BR IF NO ERRORS
1543
1544     BIT #BSE,HER ;SEE IF BAD SECTOR FLAG
1545     BEQ G ;BR IF NO
1546     JSR PC,TRUEERR ;ELSE SEE IF SECTOR LISTED IN BSE TABLE
1547     BR H ;RETURN HERE IF NO
1548
1549     INC SECTOR ;RETURN HERE IF YES
1550     CMP SECTOR,#10. ;ARE 10 CONSEC. SECTORS BAD
1551     BNE F ;BR IF NO
1552

```

```

1553          ERROR 40          ;ABORTING TEST DETECTED 10 BAD SECTORS
1554          JMP      0          ;BYPASS TEST
1555
1556 F:         MOV      #CCLR,RKCS1(R5) ;TRY ANOTHER SECTOR
1557          JMP      C
1558
1559 G:         ERROR 12          ;CERR WITH WRITE DATA CMD
1560          F.EAB  0
1561          CHECK  52,23,53,25,<AFTER WRITE DATA CMD>,T.A2,T.B2,0
1562          TYPE  MSG72          ;ABORTING BALANCE OF TESTS
1563          JMP      $EOP
1564 H:         ERROR 47          ;BAD SECTOR NOT LISTED IN TABLE
1565 I:
1566 .IF      B      E
1567          F.EAB  0
1568          CHECK  52,23,53,25,<AFTER WRITE DATA CMD>,T.A2,T.B2,0
1569
1570 .ENDC
1571 .ENDM      WDATA
1572
1573
1574 ;A=RDDATA/<CFMT!RDDATA>
1575 ;USE RDDATA <A>,X TO OMIT CKWD12
1576 ;
1577
1578 .MACRO    RDATA  A,C,?D,?E,?F,?G,?H
1579
1580          MOV      #<A>,HCS1
1581          JSR      PC,DATCMD          ;DO DATA X FOR CMD & GET CONTR RDY
1582          ERROR 13          ;NO RDY AFTER READ DATA CMD
1583          JSR      PC,GSTAT          ;GET FRESH STATUS
1584          BIT      #CERR,HCS1
1585          BEQ      G
1586          BIT      #BSE,HER          ;SEE IF BAD SECTOR
1587          BEQ      E
1588          ERROR 50          ;DETECTED BSE IN READ BUT NOT IN WRITE CMD.
1589          BR      H
1590 D:         TYPE  MSG72          ;ABORTING BALANCE OF TESTS
1591          JMP      $EOP
1592
1593 E:         BIT      #DCK,HER          ;SEE IF DATA CHECK ERROR
1594          BEQ      F
1595          ERROR 21          ;DATA CHECK ERROR AFTER READ CMD (ECC)
1596          BR      H
1597
1598 F:         ERROR 14          ;CERR AFTER READ DATA CMD.
1599
1600 H:         F.EAB  0
1601          CHECK  54,26,56,30,<AFTER READ DATA CMD>,T.A2,T.B2,0
1602          BR      0
1603
1604 G:
1605 .IF      B      C
1606          F.EAB  0
1607          CHECK  54,26,56,30,<AFTER READ DATA CMD>,T.A2,T.B2,0
1608 .ENDC

```

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
1664

```
.ENDM RDATA

:
:A=WRCHK/<CFMT!WRCHK>
: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:
      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
```



```

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
1720

K:      JMP      C
D:      MOV      #DATA0,RKBA(R5) ; WRITING 1'S:WLE SECTOR SHOULD HAVE ALL 0'S
E:      BIS      #BA1,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      #25.,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      #BIT0,$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      #BA1,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
    
```

```

1721          JSR      PC,FATT1          ;FIND ATTN
1722          ERROR    144                ;NO ATTN AFTER CMD
1723
1724          CLR      UNLD
1725
1726          A:
1727          .ENDM    QKSRT
1728
1729          ;ROUTINE TO ISSUE PACK COMMAND
1730
1731          .MACRO   QKPACK  ?A
1732                  MOV      #CLR,RKCS1(R5)
1733                  MOV      $UNIT,RKCS2(R5) ;DRIVE #
1734                  MOV      #PACK,HCS1
1735                  JSR      PC,DOCMD      ;DO PACK CMD & GET CONTR RDY
1736                  ERROR    116          ;CONTR NOT RDY
1737
1738                  BIT      #D.VV,HMR2
1739                  BNE      A
1740                  ERROR    27            ;VOLUME VALID NOT SET AFTER PACK CMD
1741          A:
1742          .ENDM    QKPACK
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

```

1764
1765
1766
1767
1768
1769
1770 001100 001100
1771 001100 00J000
1772 001102 000
1773 001103 000
1774 001104 000000
1775 001106 000000
1776 001110 000000
1777 001112 000000
1778 001114 000
1779 001115 001
1780 001116 000000
1781 001120 000000
1782 001122 000000
1783 001124 000000
1784 001126 000000
1785 001130 000000
1786 001132 000000
1787 001134 000
1788 001135 000
1789 001136 000000
1790 001140 177570
1791 001142 177570
1792 001144 177560
1793 001146 177562
1794 001150 177564
1795 001152 177566
1796 001154 000
1797 001155 002
1798 001156 012
1799 001157 000
1800 001160 000000
1801 001162 000000
1802 001164 000000
1803 001166 000000
1804 001170 000000
1805 001172 000000
1806 001174 000000
1807 001176 000000
1808 001200 177607 000377
1809 001204 077
1810 001205 015
1811 001206 000012
1812
1813
1814
1815
1816
1817
1818 001210
1819 001210 000000

.SBTTL COMMON TAGS

```

*****
; THIS TABLE CONTAINS VARIOUS COMMON STORAGE LOCATIONS
; USED IN THE PROGRAM.

```

SCMTAG: =1100

```

; START OF COMMON TAGS
SCMTAG: .WORD 0
$STNM: .BYTE 0
$ERFLG: .BYTE 0
$ICNT: .WORD 0
$LPADR: .WORD 0
$LPERR: .WORD 0
$ERTTL: .WORD 0
$ITEMB: .BYTE 0
$ERMAX: .BYTE 1
$ERRPC: .WORD 0
$GDADR: .WORD 0
$BDADR: .WORD 0
$GDDAT: .WORD 0
$BDDAT: .WORD 0
$AUTOB: .BYTE 0
$INTAG: .BYTE 0
SWR: .WORD DSWR
DISPLAY: .WORD DDISP
$TKS: 177560
$TKB: 177562
$TPS: 177564
$TPB: 177566
$NULL: .BYTE 0
$FILLS: .BYTE 2
$FILLC: .BYTE 12
$TPFLG: .BYTE 0
$TMP0: .WORD 0
$TMP1: .WORD 0
$TMP2: .WORD 0
$TMP3: .WORD 0
$TMP4: .WORD 0
$TMP5: .WORD 0
$TIMES: 0
$ESCAPE: 0
$BELL: .ASCIZ <207><377><377>
$QUES: .ASCII /?/
$CRLF: .ASCII <15>
$LF: .ASCIZ <12>

```

```

; CONTAINS THE TEST NUMBER
; CONTAINS ERROR FLAG
; CONTAINS SUBTEST ITERATION COUNT
; CONTAINS SCOPE LOOP ADDRESS
; CONTAINS SCOPE RETURN FOR ERRORS
; CONTAINS TOTAL ERRORS DETECTED
; CONTAINS ITEM CONTROL BYTE
; CONTAINS MAX. ERRORS PER TEST
; CONTAINS PC OF LAST ERROR INSTRUCTION
; CONTAINS ADDRESS OF 'GOOD' DATA
; CONTAINS ADDRESS OF 'BAD' DATA
; CONTAINS 'GOOD' DATA
; CONTAINS 'BAD' DATA
; RESERVED--NOT TO BE USED

; AUTOMATIC MODE INDICATOR
; INTERRUPT MODE INDICATOR

; ADDRESS OF SWITCH REGISTER
; ADDRESS OF DISPLAY REGISTER
; TTY KBD STATUS
; TTY KBD BUFFER
; TTY PRINTER STATUS REG. ADDRESS
; TTY PRINTER BUFFER REG. ADDRESS
; CONTAINS NULL CHARACTER FOR FILLS
; CONTAINS # OF FILLER CHARACTERS REQUIRED
; INSERT FILL CHARS. AFTER A "LINE FEED"
; "TERMINAL AVAILABLE" FLAG (BIT<07>=0=YES)
; USER DEFINED
; USER DEFINED
; USER DEFINED
; USER DEFINED
; USER DEFINED
; USER DEFINED
; MAX. NUMBER OF ITERATIONS
; ESCAPE ON ERROR ADDRESS
; CODE FOR BELL
; QUESTION MARK
; CARRIAGE RETURN
; LINE FEED

```

.SBTTL APT MAILBOX-ETABLE

```

*****
.EVEN
$MAIL: .WORD 0 ; APT MAILBOX
$MSGTY: .WORD 0 ; MESSAGE TYPE CODE

```

K03

CZR6JEO UNIBUS RK6 DR PRT3
CZR6JE.P11 25-JAN-78 12:09

MACY11 30A(1052) 25-JAN-78 12:16 PAGE 36
APT MAILBOX-ETABLE

SEQ 0036

1820	001212	000000	\$FATAL:	.WORD	AFATAL	:: FATAL ERROR NUMBER
1821	001214	000000	\$TESTN:	.WORD	ATESTN	:: TEST NUMBER
1822	001216	000000	\$PASS:	.WORD	APASS	:: PASS COUNT
1823	001220	000000	\$DEVCT:	.WORD	ADEVCT	:: DEVICE COUNT
1824	001222	000000	\$UNIT:	.WORD	AUNIT	:: I/O UNIT NUMBER
1825	001224	000000	\$MSGAD:	.WORD	AMSGAD	:: MESSAGE ADDRESS
1826	001226	000000	\$MSGLG:	.WORD	AMSGLG	:: MESSAGE LENGTH
1827	001230		\$ETABLE:			:: APT ENVIRONMENT TABLE
1828	001230	000	\$ENV:	.BYTE	AENV	:: ENVIRONMENT BYTE
1829	001231	000	\$ENVM:	.BYTE	AENVM	:: ENVIRONMENT MODE BITS
1830	001232	000000	\$SWREG:	.WORD	ASWREG	:: APT SWITCH REGISTER
1831	001234	000000	\$USWR:	.WORD	AUSWR	:: USER SWITCHES
1832	001236	000000	\$CPUOP:	.WORD	ACPUOP	:: CPU TYPE, OPTIONS
1833			*			BITS 15-11=CPU TYPE
1834			*			11/04=01, 11/05=02, 11/20=03, 11/40=04, 11/45=05
1835			*			11/70=06, PDQ=07, Q=10
1836			*			BIT 10=REAL TIME CLOCK
1837			*			BIT 9=FLOATING POINT PROCESSOR
1838			*			BIT 8=MEMORY MANAGEMENT
1839	001240	000	\$MAMS1:	.BYTE	AMAMS1	:: HIGH ADDRESS, M.S. BYTE
1840	001241	000	\$MTYP1:	.BYTE	AMTYP1	:: MEM. TYPE, BLK#1
1841			*			MEM. TYPE BYTE -- (HIGH BYTE)
1842			*			900 NSEC CORE=001
1843			*			300 NSEC BIPOLAR=002
1844			*			500 NSEC MOS=003
1845	001242	000000	\$MADR1:	.WORD	AMADR1	:: HIGH ADDRESS, BLK#1
1846			*			MEM. LAST ADDR.=3 BYTES, THIS WORD AND LOW OF "TYPE" ABOVE
1847	001244	000	\$MAMS2:	.BYTE	AMAMS2	:: HIGH ADDRESS, M.S. BYTE
1848	001245	000	\$MTYP2:	.BYTE	AMTYP2	:: MEM. TYPE, BLK#2
1849	001246	000000	\$MADR2:	.WORD	AMADR2	:: MEM. LAST ADDRESS, BLK#2
1850	001250	000	\$MAMS3:	.BYTE	AMAMS3	:: HIGH ADDRESS, M.S. BYTE
1851	001251	000	\$MTYP3:	.BYTE	AMTYP3	:: MEM. TYPE, BLK#3
1852	001252	000000	\$MADR3:	.WORD	AMADR3	:: MEM. LAST ADDRESS, BLK#3
1853	001254	000	\$MAMS4:	.BYTE	AMAMS4	:: HIGH ADDRESS, M.S. BYTE
1854	001255	000	\$MTYP4:	.BYTE	AMTYP4	:: MEM. TYPE, BLK#4
1855	001256	000000	\$MADR4:	.WORD	AMADR4	:: MEM. LAST ADDRESS, BLK#4
1856	001260	000000	\$VECT1:	.WORD	AVECT1	:: INTERRUPT VECTOR#1, BUS PRIORITY#1
1857	001262	000000	\$VECT2:	.WORD	AVECT2	:: INTERRUPT VECTOR#2, BUS PRIORITY#2
1858	001264	177440	\$BASE:	.WORD	ABASE	:: BASE ADDRESS OF EQUIPMENT UNDER TEST
1859	001266	000000	\$DEVN:	.WORD	ADEVN	:: DEVICE MAP
1860	001270	000000	\$CDW1:	.WORD	ACDW1	:: CONTROLLER DESCRIPTION WORD#1
1861	001272	000000	\$CDW2:	.WORD	ACDW2	:: CONTROLLER DESCRIPTION WORD#2
1862	001274	000000	\$DDW0:	.WORD	ADDW0	:: DEVICE DESCRIPTOR WORD#0
1863	001276	000000	\$DDW1:	.WORD	ADDW1	:: DEVICE DESCRIPTOR WORD#1
1864	001300	000000	\$DDW2:	.WORD	ADDW2	:: DEVICE DESCRIPTOR WORD#2
1865	001302	000000	\$DDW3:	.WORD	ADDW3	:: DEVICE DESCRIPTOR WORD#3
1866	001304	000000	\$DDW4:	.WORD	ADDW4	:: DEVICE DESCRIPTOR WORD#4
1867	001306	000000	\$DDW5:	.WORD	ADDW5	:: DEVICE DESCRIPTOR WORD#5
1868	001310	000000	\$DDW6:	.WORD	ADDW6	:: DEVICE DESCRIPTOR WORD#6
1869	001312	000000	\$DDW7:	.WORD	ADDW7	:: DEVICE DESCRIPTOR WORD#7
1870	001314	000000	\$DDW8:	.WORD	ADDW8	:: DEVICE DESCRIPTOR WORD#8
1871	001316	000000	\$DDW9:	.WORD	ADDW9	:: DEVICE DESCRIPTOR WORD#9
1872	001320	000000	\$DDW10:	.WORD	ADDW10	:: DEVICE DESCRIPTOR WORD#10
1873	001322	000000	\$DDW11:	.WORD	ADDW11	:: DEVICE DESCRIPTOR WORD#11
1874	001324	000000	\$DDW12:	.WORD	ADDW12	:: DEVICE DESCRIPTOR WORD#12
1875	001326	000000	\$DDW13:	.WORD	ADDW13	:: DEVICE DESCRIPTOR WORD#13

1876	001330	000000	\$DDW14: .WORD	ADDW14	::;DEVICE	DESCRIPTOR	WORD#14
1877	001332	000000	\$DDW15: .WORD	ADDW15	::;DEVICE	DESCRIPTOR	WORD#15
1878							
1879							
1880	001334		SETEND:				
1881							
1882		177440	ABASE=	177440			; DEFAULT BUSS ADDRESS
1883	001334	000210	RKVEC:	210			; DEFAULT CONTROLLER INTERRUPT VECTOR
1884	001336	000240	RKPRI:	PR5			; PRIORITY
1885	001340	172540	PKS:	172540			; P-CLOCK STATUS REG
1886	001342	172542	PKSB:	172542			; P-CLOCK SET BUFFER
1887	001344	172544	PKRB:	172544			; P-CLOCK READ BUFFER
1888	001346	177546	LKS:	177546			; L-CLOCK STATUS REG.
1889							
1890	001350	000100	LCVEC:	100			; L-CLOCK INTERRUPT VECTOR
1891	001352	000104	PCVEC:	104			; P-CLOCK INTERRUPT VECTOR.
1892							
1893		000114	MEMVEC=	114			; MEMORY PARITY VECTOR
1894		172100	MEMBAS=	172100			; MEMORY PARITY OPTION CSR START ADDR
1895							
1896	001354	000000	TRAPPC:	0			; PC FOR MEMORY PARITY ERROR TRAP
1897							
1898	001356	000000	PARAM:	0			; 1 FOR 220 START, NO DEFAULT
1899	001360	000000	FTITLE:	0			; FLAG FOR PRINTING OUT 1ST PROGRAM TITLE
1900							
1901	001362	000000	DRVPTR:	0			; CONTAINS THE POINTER TO THE DRIVE FLAG
1902							; (DRIVO-DRIV7) OF THE DRIVE TO BE CHECKED NEXT.
1903							
1904		000040	SPBAR=	40			; SPACE BAR
1905	001364	000000	FRCYL:	0			; FROM CYLINDER
1906	001366	000000	TOCYL:	0			; TO CYLINDER
1907	001370	000000	CCYL:	0			; CURRENT CYL, USED IN N SQUARE TEST
1908	001372	000000	PCYL:	0			; PREV CYL, USED IN N SQUARE TEST
1909	001374	000000	CALDIF:	0			; CALC CYL DIFF USED IN N SQUARE TEST
1910	001376	000000	CYLDIF:	0			; CYL DIFF, RIGHT JUSTIFIED FROM RKMR3
1911	001400	000000	CYLADD:	0			; CYL ADDR, RIGHT JUSTIFIED FROM RKMR3
1912	001402	000000	CALADD:	0			; CYL ADDR USED IN FHDTAB ROUTINE
1913							
1914	001404	000074	HZ:	60.			; 60 FOR 60 CPS
1915							; 50 FOR 50 CPS
1916	001406	000000	COUNT:	0			; LOADED TO 50 OR 60 TO COUNT TO 1 SEC
1917							; OR ANY OTHER NUMBER TO COUNT OFF FRACTIONAL SECOND
1918	001410	000000	SEC:	0			; SECOND COUNTER
1919	001412	000000	TIMUP:	0			; FLAG TO INDICATE TIME IS UP
1920	001414	000000	SECNT:	0			; SECTOR COUNT
1921	001416	000000	PSEC:	0			; PREVIOUS SECTOR
1922	001420	000000	ESEC:	0			; EXPECTED SECTOR
1923	001422	000000	SECTOR:	0			; SECTOR COUNT, RIGHT JUSTIFIED FROM RKMR3
1924							
1925	001424	000012	T10:	10.			; TIMEOUT CONSTANTS
1926	001426	000144	T100:	100.			
1927	001430	011610	T5000:	5000.			
1928	001432	141520	T50000:	50000.			
1929							
1930							
1931	001434	000000	WD1:	0			; ACTUAL HEADER/DATA WORD

M03

CZR6JED UNIBUS RK6 DR PRT3
CZR6JE.P11 25-JAN-78 12:08

MACY11 30A(1052) 25-JAN-78 12:16 PAGE 38
APT MAILBOX-ETABLE

SEQ 0038

```

1932 001436 000000      WD2: 0 ;EXPECTED DATA WORD
1933 001440 000000      OFFERR: 0 ;SET WHEN WRITE CHECK ERROR ON OFFSET
1934 001442 000000      HEAD: 0 ;HEAD NUMBER
1935 001444 000000      HEAD# FROM H.B3 RT. JUSTIFIED
1936 001446 000000      HD1: 0 ;SHIFTED HEAD# FOR FORMATTER ROUTINE
1937 001450 000000      FORMAT: 0 ;FORMAT TYPE
1938 001452 000000      FMT1: 0 ;SHIFTED FORMAT FOR FORMATTER ROUTINE
1939 001454 000000      WDCNT: 0 ;WORD COUNT
1940 001456 000000      DATA0: 0 ;ALL 0'S
1941 001460 052525      DATA01: 52525 ;0101 PATT
1942 001462 177777      DATA1: 177777 ;ALL 1'S
1943 001464 133467      DPAT1: 133467
1944 001466 070627      DPAT2: 70627
1945 001470 000000      WORD: 0 ;HEADER/DATA WORD
1946 001472 000000      HDWD: 0 ;HEADER WORD FROM RKDB
1947 001474 000000      BSERR: 0 ;CANNOT READ BSE INFO WHEN SET
1948 001476 000000      LIMERR: 0 ;LIMIT DETECT ERROR FLAG
1949 001500 000000      MDSERR: 0 ;MULT DRIVE SEL ERROR FLAG
1950 001502 000000      BYPCERR: 0 ;SET TO 1 TO BYPASS CKCERR IN 'GSTAT1'
1951 001504 000000      CHKFLG: 0 ;WORDS TO BE TESTED
1952 001506 000102      HDTAB: .BLKW 66. ;CALCULATED HEADER WORD TABLE
1953 001712 000102      RHTAB: .BLKW 66. ;FILLED AFTER READ HEADER CMD
1954 002116 000102      SRTTAB: .BLKW 66. ;ABOVE RHTAB SORTED STARTING FORM
1955 002322 000400      BSE22H: .BLKW 256. ;SECTOR 0 BY SORT ROUTINE
1956 003322 000400      BSE22S: .BLKW 256. ;22 SECTOR HARDWARE BSE INFO.
1957 004322 000400      RDTAB: .BLKW 256. ;22 SECTOR SOFTWARE BSE INFO.
1958 005322 000000      UNLD: 0 ;FILLED AFTER READ DATA CMD
1959 005324 000000      BADHOR: 0 ;SET TO 0 IF HEADS ARE LOADED
1960 005326 000000      HPEND: 0 ;SET TO 1 IF HEADS UNLOADED
1961 005328 000000      ;SET TO 0 IF FORMATTING OK
1962 005330 000000      ;SET TO 1 IF FORMATTING ALTERED
1963 005332 000000      ;SET TO 0 IF HALT NOT PENDING
1964 005334 000000      ;SET TO 1 IF HALT PENDING
1965 005336 000000      ;THE ABOVE 3 FLAGS ARE USED
1966 005338 000000      ;BY 'STOP' ROUTINE TO BRING
1967 005340 000000      ;THE CPU TO A VALID HALT.
1968 005330 001 002 004 ATTN: .BYTE 1,2,4,10,20,40,100,200 ;ATN 0-7 RESP.
1969 005333 010 020 040
1970 005336 100 200
1971 .EVEN
1972 ;THE FOLLOWING ARE HOLDING REGISTERS FOR THE RK611 REGISTERS
1973 ;THEY ARE LOADED AFTER RDY IS REC'D FROM WRDY ROUTINE.
1974 ;
1975 ;
1976 ;
1977 ;
1978 ;
1979 ;
1980 ;
1981 ;
1982 ;
1983 ;
1984 ;
1985 ;
1986 ;
1987 ;

```

1989 005340 000000
 1989 005342 000000
 1990 005344 000000
 1991 005346 000000
 1992 005350 000000
 1993 005352 000000
 1994 005354 000000
 1995 005356 000000
 1996 005360 000000
 1997 005362 000000
 1998 005364 000000
 1999 005366 000000
 2000 005370 000000
 2001 005372 000000
 2002 005374 000000
 2003
 2004 005376 000000
 2005 005400 000000
 2006 005402 000000
 2007 005404 000000
 2008 005406 000000
 2009
 2010
 2011
 2012 005410 000000
 2013 005412 000000
 2014 005414 000000
 2015 005416 000000
 2016 005420 000000
 2017 005422 000000
 2018 005424 000000
 2019 005426 000000
 2020
 2021
 2022
 2023 005430 000000
 2024 005432 000000
 2025 005434 000000
 2026 005436 000000
 2027 005440 000000
 2028 005442 000000
 2029 005444 000000
 2030 005446 000000
 2031
 2032
 2033
 2034 000001
 2035 000002
 2036 000004
 2037
 2038
 2039
 2040
 2041
 2042 005450 000000
 2043 005452 000000

HCS1: 0 ;HOLD RKCS1
 HCS2: 00 ;HOLD RKCS2
 HWC: 00 ;HOLD RKWC
 HBA: 00 ;ETC.
 HDA: 00
 HDS: 00
 HER: 00
 HASOF: 00
 HDC: 00
 HDB: 00
 HMR1: 00
 HMR2: 00
 HMR3: 00
 HPOS: 00
 HPAT: 0

;TEMPORARY STORAGE.

TEMP1: 0
 TEMP2: 00
 TEMP3: 00
 TEMP4: 00
 TEMP5: 0

THE FOLLOWING ARE HOLDING REGISTERS FOR MSGA(0-3) & MSGB(0-3).

H.A0: 0
 H.B0: 00
 H.A1: 00
 H.B1: 00
 H.A2: 00
 H.B2: 00
 H.A3: 00
 H.B3: 0

THE FOLLOWING ARE 'EXPECTED' REGISTER FOR THE ABOVE.

E.A0: 0
 E.B0: 00
 E.A1: 00
 E.B1: 00
 E.A2: 00
 E.B2: 00
 E.A3: 00
 E.B3: 0

THE FOLLOWING ARE IDENTIFIERS FOR DRIVE MSG WORDS TO BE TESTED.

T.A2=BIT0 ;TEST MSG A2 IF SET
 T.B2=BIT1
 T.B3=BIT2

ALL THE FLAGS BELOW ARE CLEARED INITIALLY BY THE CLRFLG ROUTINE.

DDUMP: 0 ;FLAG - SET WHEN IN DDP DUMP MODE
 DDPCH: 0 ;FLAG - SET WHEN IN DDP CHAIN MODE

2044 005454 000000
 2045 005456 000000
 2046 005460 000000
 2047
 2048
 2049
 2050
 2051 005462 000000
 2052 005464 000000
 2053 005466 000000
 2054 005470 000000
 2055 005472 000000
 2056 005474 000000
 2057 005476 000000
 2058 005500 000000
 2059
 2060 005502 000000
 2061 005504 000000
 2062 005506 000000
 2063 005510 000000

ACT11: 0
 PPTP: 0
 DRIVS: 0

; FLAG - SET WHEN IN ACT11 MODE OF OPERATION
 ; FLAG - SET WHEN PROGRAM LOADED BY PAPER TAPE
 ; CONTAINS THE NUMBER OF DRIVES PRESENT

; THE FLAGS BELOW ARE SET TO 1 TO INDICATE THAT A PARTICULAR DRIVE
 ; IS PRESENT AND IS TO BE TESTED.

DRIV0: 0
 DRIV1: 0
 DRIV2: 0
 DRIV3: 0
 DRIV4: 0
 DRIV5: 0
 DRIV6: 0
 DRIV7: 0

; FLAG SET TO 1 WHEN DRIVE 0 PRESENT
 ; FOR DRIVE 1
 ; FOR DRIVE 2
 ; FOR DRIVE 3
 ; FOR DRIVE 4
 ; FOR DRIVE 5
 ; FOR DRIVE 6
 ; FOR DRIVE 7

LCLKF: 0
 PCLKF: 0
 DOTIM: 0
 SIZFLG: 0

; L-CLOCK FLAG PRESENT FLAG
 ; P-CLOCK FLAG PRESENT FLAG
 ; SET IF EITHER CLOCK PRESENT FOR TIMING TESTS.
 ; SET IF DEFAULT DO SIZING IN TEST 1

ERROR POINTER TABLE

.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:

2078	005512				
2081	005512	044424	EM2		;DR # IN RKCS2 CANNOT BE READ BACK CORRECTLY IN RKMR2
2082	005514	050755	DH1		
2083	005516	054074	DT1		
2084	005520	054516	DF1		
2087	005522	044643	EM5		;DETECTED MDS
2088	005524	050755	DH1		
2089	005526	054074	DT1		
2090	005530	054516	DF1		
2093	005532	044664	EM6		;DETECTED UFE
2094	005534	050755	DH1		
2095	005536	054074	DT1		
2096	005540	054516	DF1		
2099	005542	044705	EM7		;DETECTED DRA & NED RESET (WRONG PORT SELECTED?)
2100	005544	050755	DH1		
2101	005546	054074	DT1		
2102	005550	054516	DF1		
2104	005552	000000	0		
2105	005554	000000	0		
2106	005556	000000	0		
2107	005560	000000	0		
2110	005562	045050	EM9		;DR NOT PRESENT BUT SPECIFIED BY OPERATOR
2111	005564	050755	DH1		
2112	005566	054074	DT1		
2113	005570	054516	DF1		
2116	005572	045124	EM10		;ABORT TEST, COULD NOT REFERENCE CONTROLLER REGISTER
2117	005574	050755	DH1		
2118	005576	054074	DT1		
2119	005600	054516	DF1		

2120				
2121			; ERROR 10	
2122	005602	045207	EM11	; DRA & NED BOTH SET
2123	005604	050755	DH1	
2124	005606	054074	DT1	
2125	005610	054516	DF1	
2126			; ERR 11	
2127	005612	045253	EM12	; NO RDY
2128	005614	052042	DH27	; AFTER WRITE DATA CMD
2129	005616	054074	DT1	
2130	005620	054622	DF10	
2131			; ERR 12	
2132	005622	045543	EM21	; CERR SET
2133	005624	052042	DH27	
2134	005626	054074	DT1	
2135	005630	054622	DF10	
2136			; ERR 13	
2137	005632	045253	EM12	; NO RDY
2138	005634	052012	DH26	; AFTER READ DATA CMD
2139	005636	054074	DT1	
2140	005640	054622	DF10	
2141			; ERR 14	
2142	005642	045543	EM21	; CERR SET
2143	005644	052012	DH26	
2144	005646	054074	DT1	
2145	005650	054622	DF10	
2146			; ERR 15	
2147	005652	045253	EM12	; NO RDY
2148	005654	052251	DH32	; AFTER WRITE CHECK CMD
2149	005656	054074	DT1	
2150	005660	054622	DF10	
2151			; ERR 16	
2152	005662	050616	EM80	; WRITE CHECK ERROR SET
2153	005664	052251	DH32	; AFTER WRITE CHECK CMD
2154	005666	054074	DT1	
2155	005670	054622	DF10	
2156			; ERR 17	
2157	005672	045253	EM12	; CONTR NOT RDY
2158	005674	051552	DH18	; AFTER UNLD CMD
2159	005676	054074	DT1	
2160	005700	054622	DF10	
2161			; ERR 20	
2162	005702	045311	EM13	; NO ATTN
2163	005704	051552	DH18	
2164	005706	054074	DT1	
2165	005710	054622	DF10	
2166			; ERR 21	
2167	005712	050655	EM83	; DATA CHECK ERROR
2168	005714	052012	DH26	; AFTER READ DATA CMD
2169	005716	054074	DT1	
2170	005720	054622	DF10	
2171			; ERR 22	
2172	005722	045543	EM21	; CERR SET
2173	005724	052251	DH32	; AFTER WRITE CHECK CMD
2174	005726	054074	DT1	
2175	005730	054622	DF10	

2176					
2177	005732	045460	;ERR 23	EM18	;MSG B0 ERROR
2178	005734	052042		DH27	;AFTER WRITE DATA CMD
2179	005736	054252		DT13	
2180	005740	054756		DF21	
2181			;ERROR 24		
2182	005742	045543		EM21	;CERR SET
2183	005744	051655		DH21	;AFTER SCLR
2184	005746	054074		DT1	
2185	005750	054622		DF10	
2186			;ERR 25		
2187	005752	045522		EM20	;MSG B1 ERROR
2188					
2189					
2190					
2191					
2192					
2193	005754	052042		DH27	
2194	005756	054252		DT13	
2195	005760	054756		DF21	
2196			;ERR 26		
2197	005762	045460		EM18	
2198	005764	052012		DH26	;AFTER READ DATA CMD
2199	005766	054252		DT13	
2200	005770	054756		DF21	
2201					
2202			;ERROR 27		
2203	005772	045656		EM24	;VOL VALID NOT SET
2204	005774	051577		DH19	;AFTER PACK CMD
2205	005776	054074		DT1	
2206	006000	054622		DF10	
2207			;ERR 30		
2208	006002	045522		EM20	;MSG B1 ERROR
2209	006004	052012		DH26	;AFTER READ DATA CMD.
2210	006006	054252		DT13	
2211	006010	054756		DF21	
2212			;ERR 31		
2213	006012	045460		EM18	;MSG B0 ERROR
2214	006014	052251		DH32	;AFTER WRITE CHECK CMD
2215	006016	054252		DT13	
2216	006020	054756		DF21	
2217			;ERR 32		
2218	006022	045522		EM20	;MSG B1 ERROR
2219	006024	052251		DH32	
2220	006026	054252		DT13	
2221	006030	054756		DF21	
2222			;ERR 33		
2223	006032	046503		EM44	;VV NOT CLEARED
2224	006034	051113		DH4	;AFTER PACK RE-INSERTED
2225	006036	054074		DT1	
2226	006040	054622		DF10	
2227			;ERR 34		
2228	006042	050513		EM76	;NO DRIVES FOUND IN DEVICE MAP
2229	006044	050755		DH1	
2230	006046	054074		DT1	
2231	006050	054516		DF1	

2232			;ERR 35		
2233	006052	045437		EM17	;MSG AD ERROR
2234	006054	051161		DH5	;AFTER AC SW OFF
2235	006056	054252		DT13	
2236	006060	054756		DF21	
2237			;ERR 36		
2238	006062	045501		EM19	;MSG A1 ERROR
2239	006064	051161		DH5	
2240	006066	054252		DT13	
2241	006070	054756		DF21	
2242			;ERR 37		
2243	006072	045522		EM20	;MSG A1 ERROR
2244	006074	051161		DH5	;AFTER OFFSET CMD
2245	006076	054252		DT13	
2246	CJ6100	054756		DF21	
2247			;ERR 40		
2248	006102	050144		EM71	;DETECTED 10 BAD SECTORS
2249	006104	052042		DH27	;AFTER WRITE DATA CMD
2250	006106	054074		DT1	
2251	006110	054622		DF10	
2252			;ERR 41		
2253	006112	045333		EM14	;WRONG ATTN
2254	006114	051552		DH18	;AFTER UNLOAD CMD
2255					
2256	006116	054074		DT1	
2257	006120	054622		DF10	
2258			;ERR 42		
2259	006122	045360		EM15	;DRDY NOT CLEARED
2260	006124	051552		DH18	
2261					
2262	006126	054074		DT1	
2263	006130	054622		DF10	
2264			;ERR 43		
2265	006132	045412		EM16	;DSC NOT SET
2266	006134	051552		DH18	
2267	006136	054074		DT1	
2268	006140	054622		DF10	
2269			;ERR 44		
2270	006142	045565		EM22	;DOOR NOT CLEARED
2271	006144	051207		DH8	;AFTER DRIVE UNLOADED & DOOR OPENED
2272	006146	054074		DT1	
2273	006150	054622		DF10	
2274			;ERR 45		
2275	006152	045437		EM17	;MSG AD ERROR
2276	006154	051207		DH8	
2277	006156	054252		DT13	
2278	006160	054756		DF21	
2279			;ERR 46		
2280	006162	045460		EM18	;MSG B0 ERROR
2281	006164	051207		DH8	
2282	006166	054252		DT13	
2283	006170	054756		DF21	
2284			;ERR 47		
2285	006172	050214		EM72	;BSE ERROR IN WRITE CMD NOT ON BSE TABLE
2286	006174	052042		DH27	;AFTER WRITE DATA CMD
2287	006176	054074		DT1	

2288	006200	054622	:	DF10	
2289					
2290					
2291	006202	050273		EM73	: DETECTED BSE IN READ BUT NOT IN WRITE CMD.
2292	006204	050755		DH1	
2293	006206	054074		DT1	
2294	006210	054516		DF1	
2295					
2296	006212	050712		EM93	: WRONG CYL# IN HEADER WORD
2297	006214	051767		DH25	: AFTER SEEK CMD
2298	006216	054206		DT9	
2299	006220	054732		DF20	
2300					
2301	006222	045437		EM17	: MSG A0 ERROR
2302	006224	052042		DH27	: AFTER WRITE DATA CMD
2303	006226	054252		DT13	
2304	006230	054756		DF21	
2305					
2306	006232	045501		EM19	: MSG A1 ERROR

2307	006234	052042	DH27	
2308	006236	054252	DT13	
2309	006240	054756	DF21	
2310				:ERR 54
2311	006242	045437	EM17	:MSG A0 ERROR
2312	006244	052012	DH26	:AFTER READ DATA CMD
2313	006246	054252	DT13	
2314	006250	054756	DF21	
2315				:ERROR 55
2316	006252	045311	EM13	:NO ATTN
2317	006254	051526	DH17	:AFTER RECAL CMD
2318	006256	054074	DT1	
2319	006260	054622	DF10	
2320				:ERR 56
2321	006262	045501	EM19	:MSG A1 ERROR
2322	006264	052012	DH26	
2323	006266	054252	DT13	
2324	006270	054756	DF21	
2325				:ERR 57
2326	006272	045437	EM17	:MSG A0 ERROR
2327	006274	052251	DH32	:AFTER WRITE CHECK CMD
2328	006276	054252	DT13	
2329	006300	054756	DF21	
2330				:ERR 60
2331	006302	045501	EM19	:MSG A1 ERROR
2332	006304	052251	DH32	
2333	006306	054252	DT13	
2334	006310	054756	DF21	
2335				:ERR 61
2336	006312	047755	EM69	:NO DRIVES PRESENT
2337	006314	050755	DH1	
2338	006316	054074	DT1	
2339	006320	054516	DF1	
2340				:ERR 62
2341	006322	050437	EM75	:FOUND 10 BAD CYL
2342	006324	052042	DH27	:AFTER WRITE DATA CMD
2343	006326	054074	DT1	
2344	006330	054622	DF10	
2345				:ERR 63
2346	006332	045501	EM19	:MSG A1 ERROR
2347	006334	051207	DH8	:AFTER DRIVE UNLOADED & DOOR OPENED
2348	006336	054252	DT13	
2349	006340	054756	DF21	
2350				:ERR 64
2351	006342	045522	EM20	:MSG B1 ERROR
2352	006344	051207	DH8	
2353	006346	054252	DT13	
2354	006350	054756	DF21	
2355				:ERR 65
2356	006352	045626	EM23	:SPIN SET
2357	006354	051252	DH11	:AFTER LOADING HEADS WITH DOOR OPEN
2358	006356	054074	DT1	
2359	006360	054622	DF10	
2360				:ERR 66
2361	006362	045437	EM17	:MSG A0 ERROR
2362	006364	051252	DH11	

2363	006366	054252	DT13	
2364	006370	054756	DF21	
2365			:ERR 67	
2366	006372	045460	EM18	;MSG B0 ERROR
2367	006374	051252	DH11	
2368	006376	054252	DT13	
2369	006400	054756	DF21	
2370			:ERR 70	
2371	006402	045501	EM19	;MSG A1 ERROR
2372	006404	051252	DH11	
2373	006406	054252	DT13	
2374	006410	054756	DF21	
2375			:ERR 71	
2376	006412	045522	EM20	;MSG B1 ERROR
2377	006414	051252	DH11	
2378	006416	054252	DT13	
2379	006420	054756	DF21	
2380			:ERR 72	
2381	006422	045714	EM25	;CARTRIDGE NOT CLEARED
2382	006424	051326	DH12	;AFTER DISK PACK REMOVED
2383	006426	054074	DT1	
2384	006430	054622	DF10	
2385			:ERR 73	
2386	006432	000000	0	
2387	006434	000000	0	
2388	006436	000000	0	
2389	006440	000000	0	
2390			:ERR 74	
2391	006442	045437	EM17	;MSG A0 ERROR
2392	006444	051113	DH4	;AFTER PACK RE-INSERTED & HDS LOADED
2393	006446	054252	DT13	
2394	006450	054756	DF21	
2395			:ERR 75	
2396	006452	045460	EM18	;MSG B0 ERROR
2397	006454	051113	DH4	
2398	006456	054252	DT13	
2399	006460	054756	DF21	
2400			:ERR 76	
2401	006462	045501	EM19	;MSG A1 ERROR
2402	006464	051113	DH4	
2403	006466	054252	DT13	
2404	006470	054756	DF21	
2405			:ERR 77	
2406	006472	045522	EM20	;MSG B1 ERROR
2407	006474	051113	DH4	
2408	006476	054252	DT13	
2409	006500	054756	DF21	
2410			:ERR 100	
2411	006502	045626	EM23	;SPIN SET
2412	006504	051356	DH13	;AFTER LOADING HEADS WITH CART. OUT
2413	006506	054074	DT1	
2414	006510	054622	DF10	
2415			:ERR 101	
2416	006512	000000	0	
2417	006514	000000	0	
2418	006516	000000	0	

2419	006520	000000		
2420			:ERR 102	0
2421	006522	000000		0
2422	006524	000000		0
2423	006526	000000		0
2424	006530	000000		0
2425			:ERR 103	0
2426	006532	000000		0
2427	006534	000000		0
2428	006536	000000		0
2429	006540	000000		0
2430			:ERR 104	0
2431	006542	000000		0
2432	006544	000000		0
2433	006546	000000		0
2434	006550	000000		0
2435			:ERR 105	
2436	006552	047152		EM52
2437	006554	053503		DH64
2438	006556	054074		DT1
2439	006560	054622		DF10
2440			:ERR 106	
2441	006562	046051		EM28
2442	006564	051442		DH15
2443	006566	054074		DT1
2444	006570	054622		DF10
2445			:ERR 107	
2446	006572	046103		EM29
2447	006574	053157		DH59
2448	006576	054074		DT1
2449	006600	054622		DF10
2450			:ERR 110	
2451	006602	046741		EM48
2452	006604	052764		DH48
2453	006606	054074		DT1
2454	006610	054622		DF10
2455			:ERR 111	
2456	006612	046130		EM30
2457	006614	051467		DH16
2458	006616	054074		DT1
2459	006620	054622		DF10
2460			:ERR 112	
2461	006622	046024		EM27
2462	006624	051467		DH16
2463	006626	054074		DT1
2464	006630	054622		DF10
2465			:ERR 113	
2466	006632	045311		EM13
2467	006634	053157		DH59
2468	006636	054074		DT1
2469	006640	054622		DF10
2470			:ERROR 114	
2471	006642	047424		EM58
2472	006644	053157		DH59
2473	006646	054074		DT1
2474	006650	054622		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

```

2475			:ERR 115		
2476	006652	047001		EM49	:WRL NOT SET
2477	006654	053025		DH49	:AFTER WRITE LOCK SW ENABLED
2478	006656	054074		DT1	
2479	006660	054622		DF10	
2480			:ERROR 116		
2481	006662	045253		EM12	:CONT NOT RDY
2482	006664	051577		DH19	:AFTER PACK CMD
2483	006666	054074		DT1	
2484	006670	054622		DF10	
2485			:ERROR 117		
2486	006672	045253		EM12	:CONT NOT RDY
2487	006674	051622		DH20	:AFTER SEL DR CMD
2488	006676	054074		DT1	
2489	006700	054622		DF10	
2490			:ERROR 120		
2491	006702	045253		EM12	
2492	006704	051655		DH21	:AFTER SUBSYS CLEAR
2493	006706	054074		DT1	
2494	006710	054622		DF10	
2495			:ERR 121		
2496	006712	047035		EM50	:WLE NOT SET
2497	006714	053065		DH50	:AFTER WRITING WITH WRITE LOCK SET
2498	006716	054074		DT1	
2499	006720	054622		DF10	
2500			:ERR 122		
2501	006722	046024		EM27	:MED NOT SET
2502	006724	051735		DH23	:AFTER WRONG PORT SELECTED
2503	006726	054074		DT1	
2504	006730	054622		DF10	
2505			:ERR 123		
2506	006732	045437		EM17	:MSG AD ERROR
2507	006734	053065		DH50	:AFTER WRITING WITH WRITE LOCK ENABLED
2508	006736	054252		DT13	
2509	006740	054756		DF21	
2510			:ERROR 124		
2511	006742	045253		EM12	
2512	006744	051526		DH17	:AFTER RECAL CMD
2513	006746	054074		DT1	
2514	006750	054622		DF10	
2515			:ERR 125		
2516	006752	045460		EM18	:MSG BD ERROR
2517	006754	053065		DH50	:AFTER WITING WITH WRL FNABLED
2518	006756	054252		DT13	
2519	006760	054756		DF21	
2520			:ERR 126		
2521	006762	045501		EM19	:MSG A1 ERROR
2522	006764	053065		DH50	
2523	006766	054252		DT13	
2524	006770	054756		DF21	
2525			:ERR 127		
2526	006772	045522		EM20	:MSG B1 ERROR
2527	006774	053065		DH50	
2528	006776	054252		DT13	
2529	007000	054756		DF21	
2530			:ERR 130		

2531	007002	046161	EM31	:NED NOT CLEARED
2532	007004	052127	DH29	:AFTER CORRECT PORT SELECTED
2533	007006	054074	DT1	
2534	007010	054622	DF10	
2535			;ERR 131	
2536	007012	045253	EM12	:NO RDY
2537	007014	051767	DH25	:AFTER SEEK CMD
2538	007016	054074	DT1	
2539	007020	054622	DF10	
2540			;ERR 132	
2541	007022	045311	EM13	:NO ATTN
2542	007024	051767	DH25	
2543	007026	054074	DT1	
2544	007030	054622	DF10	
2545			;ERR 133	
2546	007032	046024	EM27	:NED NOT SET
2547	007034	052073	DH28	:AFTER L TH PORTS DESELECTED
2548	007036	054074	DT1	
2549	007040	054622	DF10	
2550			;ERR 134	
2551	007042	047077	EM51	:WRITE LOCK NOT SET SECTOR BOUNDRY
2552	007044	053065	DH50	:AFTER WRITING WITH WRL ENABLED
2553	007046	054134	DT3	
2554	007050	054532	DF3	
2555			;ERR 135	
2556	007052	047077	EM51	
2557	007054	053233	DH60	:AFTER WRITE LOCK ENABLED WHILE WRITING
2558	007056	054134	DT3	
2559	007060	054556	DF4	
2560			;ERR 136	
2561	007062	047077	EM51	
2562	007064	053432	DH63	:AFTER WRITE LOCK ENABLED FROM AC OFF
2563	007066	054134	DT3	
2564	007070	054532	DF3	
2565			;ERR 137	
2566	007072	047077	EM51	
2567	007074	053432	DH63	
2568	007076	054134	DT3	
2569	007100	054556	DF4	
2570			;ERR 140	
2571	007102	046212	EM32	:SPINDLE ON NOT SET
2572	007104	052215	DH31	:AFTER DRIVE MANUALLY LOADED
2573	007106	054074	DT1	
2574	007110	054622	DF10	
2575			;ERR 141	
2576	007112	046246	EM33	:DRIVE NOT READY
2577	007114	052424	DH38	:AFTER AC POWERED UP
2578	007116	054074	DT1	
2579	007120	054622	DF10	
2580			;ERR 142	
2581	007122	046277	EM34	
2582	007124	052215	DH31	
2583	007126	054074	DT1	
2584	007130	054622	DF10	
2585			;ERR 143	
2586	007132	045253	EM12	:CONT NOT READY

2587	007134	052303	DH34	;AFTER ST SPIN. CMD
2588	007136	054074	DT1	
2589	007140	054622	DF10	
2590			;ERR 144	
2591	007142	045311	EM13	;NO ATTN
2592	007144	052303	DH34	
2593	007146	054074	DT1	
2594	007150	054622	DF10	
2595			;ERR 145	
2596	007152	046340	EM35	;HEADS NOT HOME
2597	007154	052337	DH35	;AFTER MANUAL UNLOAD
2598	007156	054074	DT1	
2599	007160	054622	DF10	
2600			;ERR 146	
2601	007162	047376	EM57	;CERR NOT SET
2602	007164	052370	DH37	;AFTER TIMEOUT TO POWER DOWN
2603	007166	054074	DT1	
2604	007170	054622	DF10	
2605			;ERR 147	
2606	007172	046374	EM37	;AC LOW NOT SET
2607	007174	052370	DH37	
2608	007176	054074	DT1	
2609	007200	054622	DF10	
2610			;ERR 150	
2611	007202	046424	EM42	;NED NOT SET
2612	007204	052370	DH37	
2613	007206	054074	DT1	
2614	007210	054622	DF10	
2615			;ERROR 151	
2616	007212	045253	EM12	;NO RDY
2617	007214	051703	DH22	;AFTER CLEAR CMD
2618	007216	054074	DT1	
2619	007220	054622	DF10	
2620			;ERR 152	
2621	007222	046451	EM43	;AC LO NOT CLEARED
2622	007224	052424	DH38	;AFTER AC POWERED UP
2623	007226	054074	DT1	
2624	007230	054622	DF10	
2625			;ERR 153	
2626	007232	046503	EM44	;VV NOT CLEARED
2627	007234	052424	DH38	
2628	007236	054074	DT1	
2629	007240	054622	DF10	
2630			;ERROR 154	
2631	007242	047277	EM55	;ATTN NOT CLEARED
2632	007244	051703	DH22	
2633	007246	054074	DT1	
2634	007250	054622	DF10	
2635			;ERR 155	
2636	007252	046545	EM45	;VV SET AFTER HDS LOADED
2637	007254	051442	DH15	;WITHOUT 'PACK' CMD
2638	007256	054074	DT1	
2639	007260	054622	DF10	
2640			;ERR 156	
2641	007262	046622	EM46	;NXF=0
2642	007264	052660	DH45	;AFTER SEEK WITH VV=0

2643	007266	054074	DT1	
2644	007270	054622	DF10	
2645			;ERR 157	
2646	007272	046701	EM47	;CYL ADDR CHANGED FROM 0
2647	007274	052660	DH45	
2648	007276	054332	DT14	
2649	007300	055012	DF22	
2650			;ERR 160	
2651	007301	J45437	EM17	;MSG A0 ERROR
2652	007304	052660	DH45	
2653	007306	054252	DT13	
2654	007310	054756	DF21	
2655			;ERR 161	
2656	007312	045460	EM18	;MSG B0 ERROR
2657	007314	052660	DH45	
2658	007316	054252	DT13	
2659	007320	054756	DF21	
2660			;ERR 162	
2661	007322	045501	EM19	;MSG A1 ERROR
2662	007324	052660	DH45	
2663	007326	054252	DT13	
2664	007330	054756	DF21	
2665			;ERR 163	
2666	007332	045522	EM20	;MSG B1 ERROR
2667	007334	052660	DH45	
2668	007336	054252	DT13	
2669	007340	054756	DF21	
2670			;ERR 164	
2671	007342	046622	EM46	;NXF NOT SET
2672	007344	052717	DH46	;AFTER WRITE DATA WITH VV=0
2673	007346	054074	DT1	
2674	007350	054622	DF10	
2675			;ERR 165	
2676	007352	047712	EM68	;CANNOT READ BSE INFO
2677	007354	052530	DH42	;ON SECTORS 0, 2, 4, 6, 8
2678	007356	054074	DT1	
2679	007360	054706	DF17	
2680			;ERR 166	
2681	007362	000000	0	
2682	007364	000000	0	
2683	007366	000000	0	
2684	007370	000000	0	
2685			;ERR 167	
2686	007372	047712	EM68	
2687	007374	054007	DH74	;ON SEC 10, 12....20
2688	007376	054074	DT1	
2689	007400	054706	DF17	
2690			;ERR 170	
2691	007402	000000	0	
2692	007404	000000	0	
2693	007406	000000	0	
2694	007410	000000	0	
2695			;ERROR 171	
2696	007412	045253	EM12	;NO RDY
2697	007414	052163	DH30	;AFTER READ HEADER CMD
2698	007416	054074	DT1	

2699	007420	054622		DF10	
2700			;ERROR 172	EM61	;NXF DID NOT SET FAULT
2701	007422	047475		DH45	;AFTER SEEK WITH VV=0
2702	007424	052660		DT1	
2703	007426	054074		DF10	
2704	007430	054622		DF10	
2705			;ERROR 173	EM63	:DLT SET
2706	007432	047523		DH30	
2707	007434	052163		DT1	
2708	007436	054074		DF15	
2709	007440	054666		DF15	
2710			;ERROR 174	EM21	;CERR SET
2711	007442	045543		DH30	
2712	007444	052163		DT1	
2713	007446	054074		DF15	
2714	007450	054666		DF15	
2715			;ERR 175	EM53	;UNLD NOT SET
2716	007452	047177		DH64	;AFTER MDS FOUND
2717	007454	053503		DT1	
2718	007456	054074		DF10	
2719	007460	054622		DF10	
2720			;ERR 176	EM54	;CANNOT FIND MDS
2721	007462	047227		DH65	;AFTER SEARCHING ALL DRIVES
2722	007464	053537		DT1	
2723	007466	054074		DF10	
2724	007470	054622		DF10	
2725			;ERROR 177	EM26	;VV NOT CLEARED
2726	007472	045762		DH16	;AFTER UNIT SEL PLUG REMOVED
2727	007474	051467		DT1	
2728	007476	054074		DF10	
2729	007500	054622		DF10	
2730			;ERROR 200	EM12	;NO RDY
2731	007502	045253		DH39	;AFTER WRITE HEADER CMD
2732	007504	052450		DT1	
2733	007506	054074		DF15	
2734	007510	054666		DF15	
2735			;ERROR 201	EM21	;CERR SET
2736	007512	045543		DH39	
2737	007514	052450		DT1	
2738	007516	054074		DF15	
2739	007520	054666		DF15	
2740			;ERROR 202	EM56	;UNEXP MEMORY PARITY ERROR
2741	007522	047332		DH66	;TEST #,PRAP PC
2742	007524	053602		DT6	
2743	007526	054202		DF5	
2744	007530	054602		DF5	
2745			;ERROR 203	EM18	;MSG BO ERROR
2746	007532	045460		DH5	;AFTER AC SWITCHED OFF
2747	007534	051161		DT13	
2748	007536	054252		DF21	
2749	007540	054756		DF21	
2750			;ERR 204	EM57	;CERR NOT SET
2751	007542	047376		DH67	;AFTER TIMEOUT TO ENABLE WRL
2752	007544	053623		DT1	
2753	007546	054074		DF10	
2754	007550	054622		DF10	

2755			;ERR 205	
2756	007552	047035	EM50	;WRL NOT SET
2757	007554	053623	DH67	
2758	007556	054074	DT1	
2759	007560	054622	DF10	
2760			;ERROR 206	
2761	007562	047544	EM64	;WCE AT CYL 411,TRK 2, SEC 21
2762	007564	050755	DH1	
2763	007566	054074	DT1	
2764	007570	054606	DF7	
2765			;ERROR 207	
2766	007572	047376	EM57	;CERR NOT SET
2767	007574	053065	DH50	;AFTER WRITING WITH WRL ENABLED.
2768	007576	054074	DT1	
2769	007600	054622	DF10	
2770			;ERROR 210	
2771	007602	045543	EM21	;CERR SET
2772	007604	051767	DH25	
2773	007606	054074	DT1	
2774	007610	054622	DF10	
2775			;ERR 211	
2776	007612	047454	EM59	;CTO SET
2777	007614	050053	EM70	;WHILE WAITING FOR OR REC'D CONTR RDY MSG A & B BAD
2778	007616	054074	DT1	
2779	007620	054642	DF12	
2780			;ERR 212	
2781	007622	047671	EM67	;NED SET
2782	007624	050053	EM70	
2783	007626	054074	DT1	
2784	007630	054642	DF12	
2785			;ERR 213	
2786	007632	044643	EMS	;MDS SET
2787	007634	050053	EM70	
2788	007636	054074	DT1	
2789	007640	054642	DF12	
2790			;ERR 214	
2791	007642	050412	EM74	;RTZ NOT SET
2792	007644	052503	DH41	;DURING RECD CMD
2793	007646	054074	DT1	
2794	007650	054622	DF10	
2795			;ERR 215	
2796	007652	045437	EM17	;MSG AD ERROR
2797	007654	052450	DH39	;AFTER WRITE HEADER CMD.
2798	007656	054252	DT13	
2799	007660	054756	DF21	
2800			;ERR 216	
2801	007662	045460	EM18	;BO ERROR
2802	007664	052450	DH39	
2803	007666	054252	DT13	
2804	007670	054756	DF21	
2805			;ERR 217	
2806	007672	045501	EM19	;A1 ERROR
2807	007674	052450	DH39	
2808	007676	054252	DT13	
2809	007700	054756	DF21	
2810			;ERR 220	

2811	007702	045522	EM20	
2812	007704	052450	DH39	;B1 ERROR
2813	007706	054252	DT13	
2814	007710	054756	DF21	
2815				
2816	007712	000000		
2817	007714	000000		
2818	007716	000000		
2819	007720	000000		
2820				
2821	007722	000000		
2822	007724	000000		
2823	007726	000000		
2824	007730	000000		
2825				
2826	007732	000000		
2827	007734	000000		
2828	007736	000000		
2829	007740	000000		
2830				
2831	007742	000000		
2832	007744	000000		
2833	007746	000000		
2834	007750	000000		
2835				
2836	007752	000000		
2837	007754	000000		
2838	007756	000000		
2839	007760	000000		
2840				
2841	007762	045253	EM12	;NO RDY
2842	007764	052012	DH26	;AFTER READ DATA CMD
2843	007766	054074	DT1	
2844	007770	054622	DF10	
2845				
2846	007772	045543	EM21	;CERR SET
2847	007774	052012	DH26	
2848	007776	054074	DT1	
2849	010000	054666	DF15	


```

2850
2851 .SBTTL PROGRAM SETUP
2852
2853 010002 012737 000001 001356 PARSRT: MOV #1,PARAM ;SET FLAG FOR 220 START
2854 010010 000402 BR PRGSRT ;START PROGRAM
2855
2856 010012 005037 001356 START: CLR PARAM ;CLEAR FOR 200 START
2857 010016 000005 PRGSRT: RESET ;CLEAR ALL INT ENABLE & INIT
2858 010020 012706 001100 MOV #STACK,SP ;SETUP STACK POINTER
2859 010024 012746 000000 MOV #PRO,-(SP) ;PSW LOADED TO BE
2860 010030 012746 010036 MOV #1$,-(SP) ;LSI-11 COMPATABLE
2861 010034 000002 RTI ;ENABLE ALL INTERRUPTS
2862
2863 010036 004737 033344 1$: JSR PC,$TKINT ;SETUP KB VECTOR ADDR, PRIORITY 4
2864 ;& TURN ON KB INTERRUPT
2865
2866
2867 ;*** CPU PRIORITY LEVEL NOW AT 0 ***
2868 ;*** ANY DEVICE WHICH SETS ITS ***
2869 ;*** INTERRUPT ENABLE BIT WILL ***
2870 ;*** SERVICED. ***
2871
2872 ;CLOCK INTERRUPTS WILL CHANGE CPU PRIORITY TO LEVEL 6 (IN 'STS')
2873 ;RK06 CONTROLLER INTERRUPTS WILL CHANGE CPU PRIORITY TO LEVEL 5 IN 'SETINT')
2874 ;KEYBOARD INTERRUPTS WILL CHANGE CPU PRIORITY TO LEVEL 4 (SEE ABOVE)
2875
2876 ;ALL 'SYSMAC' TRAPS WILL CHANGE CPU PRIORITY TO LEVEL 7 (SEE BELOW)
2877
2878 ;SYSMAC 'SETUP'
2879 .SBTTL INITIALIZE THE COMMON TAGS
2880 ;;CLEAR THE COMMON TAGS ($CMTAG) AREA
2881 010042 012706 001100 MOV #CMTAG,R6 ;:FIRST LOCATION TO BE CLEARED
2882 010046 005026 CLR (R6)+ ;:CLEAR MEMORY LOCATION
2883 010050 022706 001140 CMP #SWR,R6 ;:DONE?
2884 010054 001374 BNE -6 ;:LOOP BACK IF NO
2885 010056 012706 001100 MOV #STACK,SP ;:SETUP THE STACK POINTER
2886 ;;INITIALIZE A FEW VECTORS
2887 010062 012737 031452 000020 MOV #SCOPE,@IOTVEC ;:IOT VECTOR FOR SCOPE ROUTINE
2888 010070 012737 000340 000022 MOV #340,@IOTVEC+2 ;:LEVEL 7
2889 010076 012737 031732 000030 MOV #ERROR,@EMTVEC ;:EMT VECTOR FOR ERROR ROUTINE
2890 010104 012737 000340 000032 MOV #340,@EMTVEC+2 ;:LEVEL 7
2891 010112 012737 035562 000034 MOV #TRAP,@TRAPVEC ;:TRAP VECTOR FOR TRAP CALLS
2892 010120 012737 000340 000036 MOV #340,@TRAPVEC+2 ;:LEVEL 7
2893 010126 012737 031206 000024 MOV #SPWRON,@PWRVEC ;:POWER FAILURE VECTOR
2894 010134 012737 000340 000026 MOV #340,@PWRVEC+2 ;:LEVEL 7
2895 010142 013737 023770 023762 MOV SENDCT,$EOPCT ;:SETUP END-OF-PROGRAM COUNTER
2896 010150 005037 001174 CLR $TIMES ;:INITIALIZE NUMBER OF ITERATIONS
2897 010154 005037 001176 CLR $ESCAPE ;:CLEAR THE ESCAPE ON ERROR ADDRESS
2898 010160 112737 000001 001115 MOVB #1,$ERMAX ;:ALLOW ONE ERROR PER TEST
2899 010166 012737 010166 001106 MOV #,$SLPADR ;:INITIALIZE THE LOOP ADDRESS FOR SCOPE
2900 010174 012737 010174 001110 MOV #,$SLPERR ;:SETUP THE ERROR LOOP ADDRESS
2901 ;;SIZE FOR A HARDWARE SWITCH REGISTER. IF NOT FOUND OR IT IS
2902 ;;EQUAL TO A "-1" SETUP FOR A SOFTWARE SWITCH REGISTER.
2903 010202 013746 000004 MOV @ERRVEC,-(SP) ;:SAVE ERROR VECTOR
2904 010206 012737 010242 000004 MOV #64,$ERRVEC ;:SET UP ERROR VECTOR
2905 010214 012737 177570 001140 MOV #DSWR,$SWR ;:SETUP FOR A HARDWARE SWICH REGISTER

```

```

2906 010222 012737 17 570 001142      MOV    #DDISP,DISPLAY ; AND A HARDWARE DISPLAY REGISTER
2907 010230 022777 177777 170702     CMP    #-1,JSWR      ; TRY TO REFERENCE HARDWARE SWR
2908 010236 001012                BNE    66$           ; BRANCH IF NO TIMEOUT TRAP OCCURRED
2909                                ; AND THE HARDWARE SWR IS NOT = -1
2910 010240 000403                BR     65$           ; BRANCH IF NO TIMEOUT
2911 010242 012716 010250 64$:      MOV    #65$, (SP)    ; SET UP FOR TRAP RETURN
2912 010246 000002                RTI
2913 010250 012737 000176 001140 65$:    MOV    #SWREG,SWR    ; POINT TO SOFTWARE SWR
2914 010256 012737 000174 001142     MOV    #DISPREG,DISPLAY
2915 010264 012637 000004 66$:      MOV    (SP)+,2#ERRVEC ; RESTORE ERROR VECTOR
2916
2917 010270 005037 001216                CLR    $PASS         ; CLEAR PASS COUNT
2918 010274 132737 000200 001231     BITB  #APTSIZE,$ENVM ; TEST USER SIZE UNDER APT
2919 010302 001403                BEQ    67$           ; YES, USE NON-APT SWITCH
2920 010304 012737 001232 001140     MOV    #SSWREG,SWR  ; NO, USE APT SWITCH REGISTER
2921 010312
2922
2923 010312 012737 010356 000004 MEMPAR: MOV #1$,ERRVEC ; TIMEOUT VECTOR
2924 010320 012737 000340 000006     MOV    #PR7,ERRVEC+2
2925
2926 010326 012701 172100                MOV    #MEMBAS,R1   ; ADDR OF MEM CSR
2927 010332 005011 3$:          CLR    (R1)         ; SEE IF CAN REFERENCE
2928 010334 012711 000001                MOV    #1,(R1)      ; SET ENABLE BIT IF YES
2929 010340 012737 031110 000114     MOV    #MEMERR,MEMVEC ; LD MEMORY CHK VECTOR IF DONT TIMEOUT
2930 010346 012737 000340 000116     MOV    #PR7,MEMVEC+2
2931 010354 000401                BR     2$
2932
2933 010356 022626 1$:          CMP    (SP)+,(SP)+  ; ADJ STACK
2934 010360 062701 000002 2$:      ADD    #2,R1        ; TRY NEXT CSR
2935 010364 020127 172140                CMP    R1,#MEMBAS+40 ; ALL TRIED?
2936 010370 001360                BNE    3$           ; BR IN NO
2937 010372 012737 000006 000004     MOV    #ERRVEC+2,ERRVEC ; RESTORE TRAP CATCHER
2938 010400 005037 000006                CLR    ERRVEC+2
2939
2940 010404 004737 024110                JSR    PC,CLRFLG    ; CLEAR DDUMP THRU SIZFLG
2941 010410 005037 001220                CLR    $DEVCT
2942 010414 005037 001222                CLR    $UNIT
2943
2944
2945 ; FIND OUT IF XXDP, ACT, APT; CHAIN OR DUMP MODE
2946 ;
2947
2948 010420 005737 000042 START1: TST 42
2949 010424 001015                BNE    1$           ; BR IF AUTO
2950 010426 004737 024130                JSR    PC,TITLE    ; MANUAL, TYPE PROG ID
2951 010432 123727 000041 000013     CMPB  41,#13      ; 13=LOADED BY XXDP
2952 010440 001011                BNE    2$           ;
2953 010442 005237 005450                INC    DDUMP        ; SET RK06 DUMP MODE FLAG
2954 010446 104401 036543                TYPE  ,MSG2        ; REPLACE DRD PACK W/SCRATCH & DO<CR>
2955 010452 000000                HALT                ; HALT
2956 010454 000137 010470                JMP    ST2
2957 010460 000137 010534 1$:      JMP    ST3
2958 010464 005237 005456 2$:      INC    PPTP        ; SET ACT/APT/PTP DUMP MODE FLAG
2959
2960
2961 ; CHECK IF ALL PARAMETERS DEFAULTED. IF NOT, BEGIN INPUT DIALOGUE

```

G05

CZR6JED UNIBUS RK6 DR PRT3
CZR6JE.P11 25-JAN-78 12:08

MACY11 30A(1052) 25-JAN-78 12:16 PAGE 58
INITIALIZE THE COMMON TAGS

SEQ 0058

```

2962 ;WITH OPERATOR. THE REPLY TO 'DRIVES TO BE TESTED' SHOULD BE
2963 ;DRIVE NOS. SEPERATED BY COMMAS & TERMINATED BY <CR>
2964 ; EX: DRIVES TO BE TESTED: 1,2,4<CR>
2965 ;
2966 ;
2967 010470 005737 001356 ST2: TST PARAM ;BR IF 220 START
2968 010474 001002 BNE 1$ ;200 START, DEFAULT & SIZE THE BUSS
2969 010476 000137 010566 JMP ST4 ;DRIVES TO BE TESTED
2970 010502 104401 036624 1$: TYPE MSG3 ;GET DR NOS.
2971 010506 004737 024210 JSR PC,GDRVS ;BUSS ADDR
2972 010512 104401 036656 TYPE MSG4 ;GET BA
2973 010516 004737 024350 JSP PC,GBA ;CONT INT VECTOR
2974 010522 104401 036723 TYPE MSG5 ;GET INT VECTOR
2975 010526 004737 024376 JSF PC,GINT
2976 010532 000427 BR ST$
2977 ;
2978 ;
2979 ;AUTO MODE
2980 ;CHECK IF LOADED BY YXDP OR OTHER. SET FLAGS & NO INPUT DIALOGUE.
2981 ;DEFAULT ALL PARAMETERS. TEST ONLY THOSE DRIVES THAT ARE READY
2982 ;ON THE BUSS
2983 ;
2984 ;
2985 010534 123727 000041 000013 ST3: CMPB 41,#13 ;13=LOADED BY XXDP
2986 010542 001007 BNE 1$
2987 010544 005237 005452 INC DDPCH ;SET RK06 CHAIN MODE FLAG
2988 010550 004737 024130 JSR PC,TITLE
2989 010554 104401 037040 TYPE MSG7 ;DRO NOT TSTD
2990 010560 000402 BR ST4
2991 010562 005237 005454 1$: INC ACT11 ;SET ACT AUTO FLAG.
2992 ;
2993 010566 012737 177440 001264 ST4: MOV #177440,$BASE ;DEFAULT VALUE
2994 010574 012737 000210 001334 MOV #210,RKVEC ;DEFAULT VALUE
2995 010602 004737 024430 JSR PC,SETINT
2996 010606 005237 005510 INC SI2FLG ;DO "SIZE THE BUSS" TEST
2997 ;
2998 010612 005037 005322 ST5: CLR UNLD ;INITIALIZE FLAGS
2999 010616 005037 005324 CLR BADHDR ;USED IN 'STOP' ROUTINE
3000 010622 005037 005326 CLR HPEND ;FOR VALID PROGRAM HALTS
3001 010626 005037 001176 CLR $ESCAPE
3002 010632 005037 001172 CLR $TMPS ;CLEAR RK07 FLAG
3003 010636 012737 005462 001362 MOV #DRIVO,DRVPTR ;SETUP
3004 010644 005037 001220 CLR $DEVCT ;NO. OF DRVS DONE
3005 010650 005037 001222 CLR $UNIT ;CURRENT DRV UNDER TEST
3006 010654 012737 010722 000004 MOV #1$,ERRVEC ;SETUP TIMEOUT ERROR VECTOR
3007 010662 005777 170460 TST JLK$ ;SEE IF L-CLOCK THERE
3008 010666 005237 005502 INC LCLKF ;PRESENT, SET FLAG.
3009 010672 013700 001350 MOV LCVEC,RO ;VECTOR ADDR
3010 010676 012737 010764 000004 MOV #2$,ERRVEC
3011 010704 005777 170430 TST JPK$ ;SEE IF P-CLOCK THERE
3012 010710 005237 005504 INC PCLKF ;PRESENT, SET FLAG
3013 010714 013700 001352 MOV PCVEC,RO ;VECTOR ADDR
3014 010720 000412 BR 3$
3015 ;
3016 010722 022626 1$: CMP (SP)+,(SP)+ ;L-CLOCK NOT THERE, CLEAR STACK
3017 010724 012737 010770 000004 MOV #4$,ERRVEC

```

H05

CZR6JED UNIBUS RK6 DR PRT3
CZR6JE.P11 - 25-JAN-78 12:08

MACY11 30A(1052) 25-JAN-78 12:16 PAGE 59
INITIALIZE THE COMMON TAGS

SEQ 0059

3018	010732	005777	170402	TST	2PKS	;SEE IF P-CLOCK THERE
3019	010736	005237	005504	INC	PCLKF	;PRESENT, SET FLAG
3020	010742	013700	001352	MOV	PCVEC,RC	;VECTOR ADDR
3021	010746	005237	005506	3\$: INC	DOTIM	;INDICATES TIMING TESTS CAN BE DONE
3022	010752	012720	030230	MOV	#CLOCK,(R0)+	;SERVICE ROUTINE FOR CLOCKS
3023	010756	012710	000300	MOV	#PR6,(R0)	
3024	010762	000407		BR	TST	::GO TO NEXT TEST
3025						
3026	010764	022626		2\$: CMP	(SP)+,(SP)+	;P-CLOCK NOT THERE, CLEAR STACK
3027	010766	000767		BR	3\$	
3028						
3029	010770	022626		4\$: CMP	(SP)+,(SP)+	;NEITHER CLOCK THERE, CLEAR STACK
3030	010772	005037	005506	CLR	DOTIM	;TIMING TESTS CANNOT BE DONE.
3031	010776	104401	037247	TYPE	,MSG13	;ALL TIMING TESTS BYPASSED
3032						
3033						

.SBTTL BASIC CONTROLLER TESTS, SIZING & SETUP

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
3089

011062 000004
011004 012737 000001 001174
011012 012706 001100
011016 012746 000000
011022 012746 011030
011026 000002
011030
011030 012737 011154 000004
011036 013705 001264
011042 005765 000000
011046 005765 000010
011052 005765 000002
011056 005765 000004
011062 005765 000006
011066 005765 000012
011072 005765 000014
011076 005765 000016
011102 005765 000020
011106 005765 000024
011112 005765 000026
011116 005765 000034
011122 005765 000036
011126 005765 000030
011132 005765 000032
011136 012737 031022 000004
011144 012737 000340 000006
011152 000404
011154 022626
011156 104007
011160 000137 023734

```
*****  
*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'  
*****  
*ST1: SCOPE ;DO 1 ITERATION  
MOV #1,STIMES ;RESTORE STK PTR  
MOV #STACK,SP ;RESET PSW TO PRIORITY 0  
MOV #PRO,-(SP) ;& MAKE IT LSI COMPATABLE  
MOV #SS,-(SP)  
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
```

J05

CZR6JED UNIBUS RK6 DR PRT3
CZR6JE.P11 25-JAN-78 12:08

MACY11 30A(1052) 25-JAN-78 12:16 PAGE 61
T2 SIZE THE BUSS

SEQ 0061

```

3090      ;*      DRIVE WILL BE TESTED AS AN RK06.  IF SET, THE PROGRAM WILL BYPASS
3091      ;*      TESTING THAT DRIVE ONLY IF THE ERROR WAS A RESULT OF
3092      ;*      MDS, UFE OR NED BEING SET; OR BOTH NED & DRA RESET IN-
3093      ;*      DICATING THE OTHER PORT IS ACCESSED.
3094      ;*      IF CERR DUE TO DTYE, DRIVE WILL BE TESTED AS AN RK07.
3095      ;*
3096      ;*****
3097      011164 000004          ;ST2:  SCOPE
3098      011166 012737 000001 001174  MOV      #1,$TIMES      ;;DO 1 ITERATION
3099      011174 012706 001100          MOV      #STACK,SP      ;RESTORE STK PTR
3100
3101      011200 005237 001502          INC      BYPCERR        ;DO NOT TEST CERR IN FROY
3102
3103      011204 132737 000200 001231  BITB     #BIT7,$ENVM    ;SEE IF USE APT SELECTED DRIVES
3104      011212 001002          BNE     14$            ;BR IF YES
3105      011214 000137 011334          JMP      12$            ;ELSE DO NORM SIZING OR VERIFY
3106
3107      011220 104401 037153          14$:   TYPE     MSG10    ;WILL TEST DRIVES
3108      011224 005037 005460          CLR     DRIVS        ;# OF DRIVES PRESENT
3109      011230 005000          CLR     RD           ;DRV ADDR
3110      011232 012701 005462          MOV     #DRIVO,R1    ;DRV FLAG
3111      011236 013702 001266          MOV     $DEVN,R2     ;APT DEVICE MAP
3112
3113      011242 032702 000001          15$:   BIT      #BIT0,R2 ;SEE IF DRV IN DEVICE MAP
3114      011246 001410          BEQ     16$            ;BR IF NO
3115      011250 005237 005460          INC     DRIVS        ;ELSE INCR DRIVE COUNT
3116      011254 005211          INC     (R1)         ;& SET DRIVE PRESENT FLAG
3117      011256 104401 001205          TYPE   $SCLF
3118      011262 010046          MOV     RD,-(SP)     ;;SAVE RD FOR TYPEOUT
3119
3120      011264 104403          TYPOS  .BYTE 1      ;;TYPE DRIVE #
3121      011266          .BYTE 0              ;;GO TYPE--OCTAL ASCII
3122      011267          .BYTE 0              ;;TYPE 1 DIGIT(S)
3123
3124      011270 005721          16$:   TST     (R1)+     ;ADV POINTER TO NEXT FLAG
3125      011272 005200          INC     RD           ;INC DRIVE #
3126      011274 022700 000010          CMP     #8.,RD      ;ALL 8 TESTED?
3127      011300 001402          BEQ     17$            ;BR IF YES
3128
3129      011302 006002          ROR     R2           ;ELSE GET NEXT BIT OFF DEVICE MAP
3130      011304 000756          BR      15$          ;& TRY AGAIN
3131
3132      011306 005737 005460          17$:   TST     DRIVS    ;SEE IF MORE DRIVES PRESENT
3133      011312 001402          BEQ     18$            ;BR IF NO
3134      011314 000137 012010          JMP     VERIFY        ;ELSE EXIT TEST & SETUP FOR RK07'S.
3135
3136      011320 104034          18$:   ERROR   34      ;NO DRIVES FOUND IN $DEVN
3137      011322 000000          HALT
3138      011324 000137 010612          JMP     ST5           ;SETUP CORRECTLY & PRESS 'CONTINUE'
3139      011330 000137 012010          20$:   JMP     VERIFY        ;TO TRY AGAIN
3140
3141      011334 012765 000040 000010  12$:   MOV     #SCLR,RKCS2(R5);SUBSYSTEM CLEAR
3142
3143
3144
3145

```

K05

CZR6JEO UNIBUS RK6 DR PRT3
CZR6JE.P11 25-JAN-78 12:08

MACY11 30A(1052) 25-JAN-78 12:16 PAGE 62
T2 SIZE THE BUSS

SEQ 0062

```

3146
3147 011342 013737 001424 005376      MOV    T10,TEMP1      ;SET TIMEOUT
3148 011350 004737 024542                JSR    PC,FRDY        ;FIND RDY
3149 011354 104120                        ERROR  120            ;RDY NOT SET BY END OF SCLR
3150
3151 011356 005737 005510                TST    SIZFLG         ;SIZE BUS?
3152 011362 001762                        BEQ    20$            ;BR IF NO
3153 011364 104401 037153                TYPE  MSG10          ;WILL TEST DRIVES
3154 011370 005037 005460                CLR    DRVS          ;# OF DRIVES PRESENT
3155 011374 005000                        CLR    RD            ;DRV ADDR
3156 011376 012701 005462                MOV    #DRIV0,R1     ;DRV FLAG
3157 011402
3158 011402 104415                        1$:  SCOP1
3159 011404 012706 001100                MOV    #STACK,SP    ;RESTORE STK PTR
3160
3161 011410 012765 000040 000010      MOV    #SCLR,RKCS2(R5) ;SUBSYSTEM CLEAR
3162 011416 013737 001424 005376      MOV    T10,TEMP1     ;SET TIMEOUT
3163 011424 004737 024542                JSR    PC,FRDY        ;FIND RDY
3164 011430 104120                        ERROR  120            ;RDY NOT SET BY END OF SCLR
3165 011432 010065 000010                MOV    RD,RKCS2(R5)  ;SELECT THE DRIVE ADDR
3166 011436 012737 000001 005340      MOV    #SELDRV,HCS1
3167 011444 053737 001170 005340      BIS    $TMP4,HCS1    ;ADD CDT IF RK07
3168 011452 013765 005340 000000      MOV    HCS1,RKCS1(R5) ;GET STATUS
3169 011460 013737 001432 005376      MOV    T50000,TEMP1
3170 011466 004737 025232                JSR    PC,DLY        ;DO DELAY TO CATCH MDS
3171 011472 013737 001424 005376      MOV    T10,TEMP1
3172 011500 004737 024542                JSR    PC,FRDY        ;FIND RDY
3173 011504 104117                        ERROR  117            ;NO RDY AFTER SELECT DRIVE CMD.
3174 011506 032737 100000 005340      BIT    #CERR,HCS1
3175 011514 001052                        BNE    2$
3176 011516 013737 005366 005376      MOV    HMR2,TEMP1
3177 011524 042737 177770 005376      BIC    #1C<DRVMSK>,TEMP1
3178 011532 020037 005376                CMP    RD,TEMP1     ;S/B SAME
3179 011536 001020                        BNE    3$
3180 011540 005700                        TST    RD
3181 011542 001003                        BNE    4$
3182 011544 005737 005452                TST    DDPCH        ;SEE IF XXDP CHAIN MODE
3183 011550 001016                        BNE    5$
3184 011552 005237 005460                4$: INC    DRVS        ;INC DRIVE COUNT.
3185 011556 005211                        INC    (R1)          ;SET DRIVE PRESENT FLAG
3186 011560 053711 001170                BIS    $TMP4,(R1)   ;ADD CDT IF RK07
3187 011564 104401 001205                TYPE  $SCLRF
3188 011570 010046                        MOV    RD,-(SP)     ;SAVE RD FOR TYPEOUT
3189
3190 011572 104403                        TYPOS
3191 011574 001                                .BYTE 1
3192 011575 000                                .BYTE 0
3193 011576 000403                        BR    5$
3194
3195 011600 004737 025250                3$: JSR    PC,BYP     ;TYPE BYPASS DR #
3196 011604 104001                        ERROR  1             ;WRITTEN DR # DOES NOT MATCH RkMR2 DR #
3197
3198 011606 005721                        5$: TST    (R1)+      ;SHIFT PTR TO NEXT DR. FLAG
3199 011610 005200                        INC    RD            ;INC DR #
3200 011612 005037 001170                CLR    $TMP4        ;CLEAR RK07 FLAG FOR NEXT DRIVE
3201 011616 022700 000010                CMP    #B.,RD

```

L05

CZR6JEO UNIBUS RK6 DR PRT3
CZR6JE.P11 25-JAN-78 12:08

MAC111 30A(1052) 25-JAN-78 12:16 PAGE 63
T2 SIZE THE BUSS

SEQ 0063

```

3202 011622 001257      BNE      1$      ;MORE LEFT.
3203 011624 005737 005460 TST      DRVS
3204 011630 001065      BNE      10$
3205 011632 104061      ERROR    61      ;NO DRIVES SEEN
3206 011634 000000      HALT
3207 011636 000137 010612 JMP      ST5      ;SETUP & PRESS 'CONTINUE'
3208
3209 011642 032737 000040 005354 2$: BIT      #DTYE,HER
3210 011650 001405      BEQ      13$
3211 011652 012737 002000 001170 MOV      #CDT,$TMP4 ;ADD CDT
3212 011660 000137 011402 JMP      1$      ;TRY AGAIN
3213 011664 032737 001000 005342 13$: BIT      #MDS,HCS2
3214 011672 001015      BNE      6$
3215 01 674 032737 000400 005342 BIT      #UFE,HCS2
3216 01 702 001015      BNE      7$
3217 011704 032737 000001 005352 BIT      #DRA,HDS
3218 011712 001015      BNE      8$
3219 011714 032737 010000 005342 BIT      #NED,HCS2
3220 011722 001424      BEQ      9$
3221 011724 000739      BR       5$
3222
3223 011726 004737 025250      6$: JSR      PC,BYP ;TYPE BYP DR #
3224 011732 104002      ERROR    2$      ;MDS DETECTED
3225 011734 000724      BR       5$
3226
3227 011736 004737 025250      7$: JSR      PC,BYP
3228 011742 104003      ERROR    3$      ;UFE DETECTED
3229 011744 000720      BR       5$
3230
3231 011746 032737 010000 005342 8$: BIT      #NED,HCS2
3232 011754 001676      BEQ      4$
3233 011756 104401 037360 TYPE     MSG15 ;DRV#
3234 011762 010046 MOV      RO,-(SP) ;SAVE RO FOR TYPEOUT
3235 ;TYPE DR#
3236 011764 104403 TYPOS ;GO TYPE--OCTAL ASCII
3237 011766 001 .BYTE 1 ;TYPE 1 DIGIT(S)
3238 011767 000 .BYTE 0 ;SUPPRESS LEADING ZEROS
3239 011770 104010 ERROR 10 ;DRA & NED BOTH SET
3240 011772 000705 BR       5$
3241
3242 011774 004737 025250      9$: JSR      PC,BYP
3243 012000 104004 ERROR 4$ ;NO DRA & NO NED = OTHER PORT SELECTED
3244 012002 000701 BR       5$
3245 012004 000137 012402 10$: JMP      NUDRV
3246
3247 012010 VERIFY:

```

```

3248
3249
3250 *****
3251 *TEST 3 VERIFY OPERATOR DRIVE SELECTIONS
3252 *
3253 * THIS TEST IS ENTERED ONLY IF DRIVE SELECTION IS NOT
3254 * DEFAULTED. EVERY DRIVE FROM 0 TO 7 IS ADDRESSED &
3255 * CONTROLLER ERROR (CERR) IS EXAMINED. IF NOT SET, THE
3256 * PROGRAM WILL ASSUME THE DRIVE IS PRESENT AS AN RK06.
3257 * IF CERR WAS SET, THAT DRIVE WILL BE BYPASSED

```


MOS

CZR6JED UNIBUS RK6 DR PRT3
CZR6JE.P11 25-JAN-78 12:08

MACY11 30A(1052) 25-JAN-78 12:16 PAGE 64
T3 VERIFY OPERATOR DRIVE SELECTIONS

SEQ 0064

C
C

```

3258
3259
3260
3261
3262
3263
3264
3265 012010 000004
3266 012012 012737 000001 001174
3267 012020 012706 001100
3268 012024 005000
3269 012026 012701 005462
3270 012032
3271 012032 104415
3272 012034 012706 001100
3273
3274 012040 012765 000040 000010
3275 012046 013737 001424 005376
3276 012054 004737 024542
3277 012060 104120
3278 012062 010065 000010
3279 012066 012737 000001 005340
3280 012074 053737 001170 005340
3281 012102 013765 005340 000000
3282 012110 013737 001432 005376
3283 012116 004737 025232
3284 012122 013737 001424 005376
3285 012130 004737 024542
3286 012134 104117
3287 012136 032737 100000 005340
3288 012144 001036
3289 012146 013737 005366 005376
3290 012154 042737 177770 005376
3291 012162 020037 005376
3292 012166 001014
3293 012170 005711
3294 012172 001402
3295 012174 053711 001170
3296 012200 005721
3297 012202 005200
3298 012204 005037 001170
3299 012210 022700 000010
3300 012214 001306
3301 012216 000475
3302
3303 012220 004737 025250
3304 012224 104001
3305 012226 005711
3306 012230 001763
3307 012232 005337 005460
3308 012236 005011
3309 012240 000757
3310 012242 032737 000040 005354
3311 012250 001405
3312 012252 012737 002000 001170
3313 012260 000137 012032

```

```

;* ONLY IF THE ERROR WAS A RESULT OF MDS OR LIFE 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.
*****
↑ST3: SCOPE
MOV #1,STIMES ;DO 1 ITERATION
MOV #STACK,SP ;RESTORE STK PTR
CLR RO ;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 RO,RKCS2(R5) ;DRV ADDR
MOV #SELDRV,HCS1
BIS $TMP4,HCS1 ;ADD CDT IF RK07
MOV HCS1,RKCS1(R5) ;GET STATUS
MOV T5000,TEMP1
JSR PC,DLY ;DO DELAY TO CATCH MDS
MOV T10,TEMP1
JSR PC,FRDY ;FIND RDY
ERROR 117 ;NO RDY AFTER SELECT DRIVE CMD.
BIT #CERR,HCS1
BNE 2$
MOV HMR2,TEMP1
BIC #C<DRVMSK>,TEMP1
CMP RO,TEMP1 ;S/B SAME
BNE 3$

11$: TST (R1)
BEQ 4$
BIS $TMP4,(R1) ;ADD CDT IF RK07
TST (R1)+ ;SHIFT PTR TO NEXT DR FLAG
INC RO ;INC DR#
CLR $TMP4 ;CLEAR RK07 FLAG FOR NEXT DRIVE
CMP #8,RO
BNE 1$ ;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
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

```

```

3311
3315
3316 012264 032737 001000 005342 13$: BIT #MDS,HCS2
3317 012272 001027 BNE 6$
3318 012274 032737 000400 005342 BIT #UFE,HCS2
3319 012302 001027 BNE 7$
3320 012304 032737 000001 005352 BIT #DRA,HDS
3321 012312 001005 BNE 8$
3322 012314 032737 010000 005342 BIT #NED,HCS2
3323 012322 001423 BEQ 9$
3324 012324 000404 BR 10$
3325 012326 032737 010000 005342 8$: BIT #NED,HCS2
3326 012334 001715 BEQ 11$
3327 012336 005711 10$: TST (R1)
3328 012340 001717 BEQ 4$
3329
3330 012342 004737 025250 JSR PC,BYP ;TYPE BYPASS DRIVE#
3331 012346 104006 ERROR 6
3332 012350 000730 BR 12$
3333
3334 012352 004737 025250 6$: JSR PC,BYP ;TYPE BYPASS DRIVE#
3335 012356 104002 ERROR 2 ;MDS DETECTED
3336 012360 000724 BR 12$
3337
3338 012362 004737 025250 7$: JSR PC,BYP
3339 012366 104003 ERROR 3 ;UFE DETECTED
3340 012370 000720 BR 12$
3341
3342 012372 004737 025250 9$: JSR PC,BYP
3343 012376 104004 ERROR 4 ;DRA & NED RESET - OTHER PORT SELECTED
3344 012400 000714 BR 12$
3345
3346
3347
3348
3349
3350 ; THIS PART OF THE PROGRAM WILL BE REPEATED FOR EACH
3351 ; DRIVE PRESENT
3352 ; 'SUNIT' CONTAINS THE ADDRESS OF THE DRIVE CURRENTLY
3353 ; UNDER TEST
3354
3355 012402 005037 001502 NUDRV: CLR BYPCERR ;ENTER HERE FROM LAST TEST
3356 012406 005037 001170 CLR $TMP4 ;TEST CERR IN 'FRDY'
3357 ;RK07 FLAG
3358

```

```

3359
3360
3361
3362
3363
3364
3365
3366
3367
3368 012412 000004
3369 012414 012737 000001 001174
3370 012422 012706 001100
3371 012426 012737 000004 001214
3372 012434 012737 000004 001102
3373
3374 012442 005737 005460
3375 012446 001004
3376 012450 104401 037740
3377 012454 000137 023734
3378
3379 012460 013701 001362
3380 012464 005737 001220
3381 012470 001402
3382 012472 005237 001222
3383 012476 005711
3384 012500 001002
3385 012502 005721
3386 012504 000772
3387 012506 005737 005452
3388 012512 001405
3389 012514 005737 001222
3390 012520 001002
3391 012522 005721
3392 012524 000762
3393
3394 012526 032721 002000
3395 012532 001403
3396 012534 012737 002000 001170
3397 012542 010137 001362
3398 012546 104401 037360
3399 012552 013700 001222
3400 012556 010046
3401
3402 012560 104403
3403 012562 001
3404 012563 000
3405
3406
3407
3408 012564 005737 001170
3409 012570 001014
3410 012572 012737 000632 012654
3411 012600 005037 012662
3412 012604 012737 000777 012656
3413 012612 012737 160017 012660
3414 012620 000421

```

```

*****
TEST 4 FIND NEXT DRIVE TO BE TESTED
*****
THIS TEST FINDS THE NEXT DRIVE PRESENT & PUTS THAT
ADDRESS IN 'SUNIT' & $TMP4 IS SET TO CDT IF DRIVE IS RK07.
THROUGHOUT THE FOLLOWING TESTS, THE DRIVE TESTED IS
THE DRIVE WHOSE ADDRESS IS IN 'SUNIT'.
*****
TST4: SCOPE
MOV #1,$TIMES ;:DO 1 ITERATION
MOV #STACK,$SP ;:RESTORE STK PTR
MOV #STN-1,$TESTN
MOV #STN-1,$STNM
TST DRVS ;:ANY DRIVES PRESENT?
BNE 4$ ;:YES BRANCH
TYPE $MSG27 ;:ALL DRIVES TESTED
JMP $EOP ;:NO, GO TO END
4$: MOV DRVPTR,R1 ;:ADDN OF NEXT DRIVE FLAG
TST $DEVCT ;:IS FIRST DRIVE BEING CHECKED
BEQ 2$ ;:YES BRANCH
1$: INC SUNIT ;:INCR DRIVE ADDR TO NEXT DRIVE
2$: TST (R1) ;:IS DRIVE PRESENT?
BNE 5$ ;:BR IF YES
TST (R1)+ ;:ELSE FIND NEXT DRIVE
BR 1$
5$: TST DDPCH ;:DDP CHAIN MODE?
BEQ 3$ ;:BR IF NO
TST SUNIT ;:ELSE SEE IF DRV 0
BNE 3$ ;:BR IF NO
TST (R1)+ ;:ELSE FIND NEXT DRIVE PRESENT
BR 1$
3$: BIT #CDT,(R1)+ ;:SEE IF DRIVE UNDER TEST IS RK07
BEQ 6$ ;:BR IF NO
6$: MOV #CDT,$TMP4 ;:ELSE SET RK07 FLAG
MOV R1,DRVPTR ;:STORE POINTER TO NEXT DR FLAG
TYPE $MSG15 ;:"DRIVE"
MOV $SUNIT,R0 ;:SAVE R0 FOR TYPEOUT
MOV R0,-($SP) ;:DRIVE #
;:GO TYPE--OCTAL ASCII
;:TYPE 1 DIGIT(S)
;:SUPPRESS LEADING ZEROS
; TYPE , $CRLF
TST $TMP4 ;:SEE IF RK07 UNDER TEST
BNE 7$ ;:BR IF YES
MOV #632,$LC ;:ELSE LOAD RK06 PARAMETERS
CLR E.DD↑
MOV #777,$MASK
MOV #160017,$MASK1
BR TST5 ;:GOTO NEXT TEST

```

C06

CZR6JED UNIBUS RK6 DR PRT3
CZR6JE.P11 25-JAN-78 12:08

MACY11 3CA(1052) 25-JAN-78 12:16 PAGE 67
T4 FIND NEXT DRIVE TO BE TESTED

SEQ 0067

```
3415  
3416 012622 012737 001456 012654 7S: MOV #1456,LC ;LOAD RK07 PARAMETERS  
3417 012630 012737 000400 012662 MOV #0,DDT,E,DDT  
3418 012636 012737 001777 012656 MOV #1777,MASK  
3419 012644 012737 140017 012660 MOV #140017,MASK1  
3420 012652 000404 BR TSTS ;;GOTO NEXT TEST  
3421  
3422 012654 000000 LC: 0 ;LAST CYL  
3423 012656 000000 MASK: 0  
3424 012660 000000 MASK1: 0  
3425 012662 000000 E.DDT: 0 ;EXPECTED DRIVE TYPE TO E.A0
```

```

3426
3427 012664
3428
3429
3430
3431
3432
3433
3434
3435 012664 000004
3436 012666 012737 000001 001174
3437 012674 012706 001100
3438
3439 012700 005737 001216
3440 012704 001042
3441 012706 004737 026436
3442 012712 104024
3443
3444 012714 104401 037372
3445 012720 012765 000003 000026
3446 012726 004737 026106
3447 012732 013701 005366
3448 012736 012704 035046
3449 012742 010446
3450 012744 012703 000003
3451 012750 006101
3452 012752 006101
3453 012754 006101
3454 012756 006101
3455 012760 006101
3456 012762 006101
3457 012764 010100
3458 012766 042700 177760
3459 012772 052700 000060
3460 012776 110024
3461 013000 005303
3462 013002 001364
3463 013004 105014
3464
3465 013006 004737 035314
3466
3467
3468
3469
3470
3471
3472
3473
3474
3475 013012 000004
3476 013014 012737 000001 001174
3477 013022 012706 001100
3478
3479 013026 004737 026436
3480 013032 104024
3481

```

```

PFSRT: ;ENTER HERE FOR POWER FAIL RESTART
;*****
;TEST 5 PRINT DRIVE SERIAL NUMBER
;
; THIS TEST READS & PRINTS THE DRIVE SERIAL # FROM MSG A, WORD 11
; IN BCD & IS PERFORMED ON THE 1ST PASS ONLY
;*****
↑ST5: SCOPE ;DO 1 ITERATION
MOV #1,$TIMES ;RESTORE STK PTR
MOV #STACK,$SP
TST $PASS
BNE TST6 ;GO TO NEXT IF NOT FIRST PASS
JSR PC,$SUBCLR ;DO SUBSYS CLEAR
ERROR 24 ;CERR AFTER SCLR

TYPE MSG16 ;DRIVE SERIAL NO.
MOV #3,$RKMRI($R5) ;SELECT BYTE 3
JSR PC,$GSTAT ;GET STATUS
MOV #R1,$R1 ;GET SERIAL #
MOV #SOCTVL,$R4 ;GET ADDR CHAR BUFF
MOV $R4,-($SP) ;STORE ON STACK FOR $SUPRS
MOV #3,$R3 ;SETUP CHAR COUNT
ROL $R1 ;INITIALIZE BIT POSITIONS
1$: ROL $R1 ;GET NEXT 4 BITS
ROL $R1
ROL $R1
ROL $R1
MOV $R1,$R0 ;GET WORKING COPY
BIC #177760,$R0 ;CLEAR ALL BUT LOW 4 BITS
BIS #60,$R0 ;CONVERT TO ASCII DIGIT
MOV $R0,$R4+ ;PUT ASCII DIGIT INTO CHAR BUFF
DEC $R3
BNE 1$ ;BR IF ALL 3 CHARS NOT DONE
CLRB ($R4) ;ELSE INSERT NULL TERMINATOR

JSR PC,$SUPRS ;TYPE
;
;
;*****
;TEST 6 SET VV WITH PACK COMMAND
;
; IF VV IS RESET, THE PACK COMMAND IS USED TO SET IT.
;*****
↑ST6: SCOPE ;DO 1 ITERATION
MOV #1,$TIMES ;RESTORE STK PTR
MOV #STACK,$SP
JSR PC,$SUBCLR
ERROR 24 ;CERR AFTER SCLR

```

E06

CZR6JED UNIBUS RK6 DR PRT3
CZR6JE.P11 25-JAN-78 12:08

MACY11 30A(1052) 25-JAN-78 12:16 PAGE 69
T6 SET VV WITH PACK COMMAND

SEQ 0069

```

3482 013034 032737 000100 005366 BIT #D.VV,HMR2
3483 013042 001021 BNE TST7 ;;GO TO NEXT TEST IF VV SET
3484 013044 012765 100000 000000 MOV #CCLR,RKCS1(R5)
3485 013052 013765 001222 000010 MOV $UNIT,RKCS2(R5) ;DRIVE #
3486 013060 012737 000003 005340 MOV #PACK,HCS1
3487 013066 004737 024446 JSR PC,DOCMD ;DO PACK CMD & GET CONTR RDY
3488 013072 104116 ERROR 116 ;CONTR NOT RDY
3489
3490 013074 032737 000100 005366 BIT #D.VV,HMR2
3491 013102 001001 BNE 64$
3492 013104 104027 ERROR 27 ;VOLUME VALID NOT SET AFTER PACK CMD
3493 64$:
3494
3495
3496
3497 *****
3498 *TEST 7 UNLOAD DRIVE TO BE TESTED
3499 *
3500 * THIS TEST UNLOADS THE DRIVE TO BE TESTED NEXT
3501 * WAITS FOR ATTN & VERIFIES IT CAME FROM THE CORRECT DRIVE.
3502 *
3503 *****
3504 *TST7: SCOPE
3505 013106 000004 MOV #1,$TIMES ;;DO 1 ITERATION
3506 013110 012737 000001 001174 MOV #STACK,SP
3507 013116 012706 001100
3508 INC UNLD ;USED TO CHECK FOR VALID HALT
3509
3510 013122 005237 005322 JSR PC,SUBCLR
3511 013126 004737 026436 ERROR 24 ;CERR AFTER SCLR
3512 013132 104024
3513 013134 012737 000007 005340 MOV #UNLOAD,HCS1
3514 013142 004737 024446 JSR PC,DOCMD ;DO UNLOAD CMD & GET CONTR READY
3515 013146 104017 ERROR 17 ;NO RDY AFTER UNLD CMD
3516 013150 004737 025024 JSR PC,TSTATN
3517 013154 104020 ERROR 20 ;NO ATTN AFTER UNLOAD CMD
3518
3519
3520 .SBTTL OPERATOR INTERVENTION TESTS
3521 *****
3522 *TEST 10 INTERLOCKS TESTS
3523 *
3524 * THIS TEST VERIFIES THAT THE DOOR & CARTRIDGE STATUS BITS
3525 * ARE OPERATING PROPERLY IN MESSAGE AD & THAT THE REMOVAL
3526 * OF THE CARTRIDGE CLEARS VOLUME VALID.
3527 * IT FURTHER VERIFIES ALL REMAINING STATUS BITS OF
3528 * MESSAGE A & B WORDS 0 & 1.
3529 * THIS TEST ALSO CHECKS THAT THE SPINDLE CANNOT BE STARTED
3530 * WHEN THE DOOR IS OPEN OR THE CARTRIDGE IS REMOVED.
3531 * IT ALSO VERIFIES THAT LOSS OF VOLUME VALID RESETS SACK WHICH
3532 * ASSERTS NON EXISTENT DRIVE IN RKCS2
3533 *
3534 *****
3535 *TST10: SCOPE
3536 013156 000004 MOV #1,$TIMES ;;DO 1 ITERATION
3537 013160 012737 000001 001174 MOV #STACK,SP ;RESET STACK PTR
3538 013166 012706 001100

```

```

3538
3539 013172 004737 026436 JSR PC,SUBCLR
3540 013176 104024 ERROR 24 ;CERR AFTER SCLR
3541
3542 013200 005237 001502 INC BYPCERR ;DONT DO CKCERR ROUTINE
3543
3544 013204 104401 037075 TYPE ,MSG8 ;INTERLOCKS TEST
3545 013210 104401 041765 TYPE ,MSG52 ;CONT-E TO EXIT OR SPACE TO CONTINUE
3546 013214 004737 030412 JSR PC,CCSP ;INPUT CONT-E OR SPACE
3547 013220 000137 014020 JMP B$ ;RET HERE FOR CONT-E
3548
3549 013224 104401 041441 TYPE ,MSG47 ;VERIFY DOOR CANNOT BE OPENED
3550 013230 104401 040573 TYPE ,MSG37 ;DEPRESS SPACE WHEN DONE
3551 013234 004737 030452 JSR PC,GETSP ;GET SPACE
3552
3553 013240 104401 043432 TYPE ,MSG65 ;DEPRESS RUN/STOP SW TO 'STOP'
3554 013244 104401 040076 TYPE ,MSG30 ;OPEN DOOR & LEAVE IT OPEN
3555 013250 104401 040573 TYPE ,MSG37 ;DEPRESS SPACE BAR WHEN DONE
3556 013254 004737 030452 JSR PC,GETSP ;INPUT SPACE
3557 013260 012765 000001 000026 MOV #1,RKMR1(R5) ;SELECT WORD 1
3558 013266 004737 026106 JSR PC,GSTAT
3559 013272 032737 000200 005366 BIT #D.DOOR,HMR2
3560 013300 001401 BEQ 1$
3561 013302 104044 ERROR 44 ;DOOR STATUS BIT NOT CLEARED
3562
3563 013304 012737 000140 005430 1$: MOV #<D.DRA!D.VV>,E.A0 ;EXPECTED A0
3564 013312 005037 005432 CLR E.B0
3565 013316 012737 000540 005434 MOV #<D.HDHM!D.BRHM!D.CART>,E.A1
3566 013324 012737 000001 005436 MOV #1,E.B1
3567
3568 013332 004737 025264 JSR PC,CHKMSG ;CHECK MSGS A0, B0, A1, B1
3569 013336 000000 .WORD 0!0!0 ;& MSGS SPECIFIED HERE
3570 013340 104045 ERROR 45 ;MSG A0 ERROR AFTER DRIVE UNLOADED & DOOR OPENED
3571 013342 104046 ERROR 46 ;MSG B0 ERROR
3572 013344 104063 ERROR 63 ;MSG A1 ERROR
3573 013346 104064 ERROR 64 ;MSG B1 ERROR
3574
3575 013350 004737 026436 JSR PC,SUBCLR
3576 013354 104024 ERROR 24 ;CERR AFTER SCLR
3577
3578 013356 104401 040226 TYPE ,MSG33 ;PRESS 'RUN-STOP' TO 'RUN'
3579 013362 104401 040313 TYPE ,MSG34 ;VERIFY DOES NOT START
3580 013366 104401 040573 TYPE ,MSG37
3581 013372 004737 030452 JSR PC,GETSP ;GET SPACE FROM TTY
3582
3583 013376 004737 026106 JSR PC,GSTAT
3584 013402 032737 010000 005366 BIT #D.SPIN,HMR2
3585 013410 001401 BEQ 2$
3586 013412 104065 ERROR 65 ;SPIN SET IN MSGA0
3587
3588 013414 012737 000140 005430 2$: MOV #<D.DRA!D.VV>,E.A0 ;EXPECTED MSG A0
3589 013422 005037 005432 CLR E.B0
3590 013426 012737 000540 005434 MOV #<D.HDHM!D.BRHM!D.CART>,E.A1
3591 013434 012737 000001 005436 MOV #1,E.B1
3592
3593 013442 004737 025264 JSR PC,CHKMSG ;CHECK MSGS A0, B0, A1, B1

```

3594	013446	000000				.WORD	0!0!0		; & MSGS SPECIFIED HERE
3595	013450	104066				ERROR	66		;MSG A0 ERROR AFTER ATTEMPT TO START SPIN & DOOR OPEN
3596	013452	104067				ERROR	67		;MSH B0 ERROR
3597	013454	104070				ERROR	70		;MSG A1 ERROR
3598	013456	104071				ERROR	71		;MSG B1 ERROR
3599									
3600	013460	004737	026436			JSR	PC, SUBCLR		
3601	013464	104024				ERROR	24		;CERR AFTER SCLR
3602									
3603	013466	104401	043432			TYPE	,MSG65		;DEPRESS 'RUN-STOP' SW TO STOP
3604	013472	104401	040166			TYPE	,MSG32		;REMOVE PACK & CLOSE DOOR
3605	013476	104401	040573			TYPE	,MSG37		
3606	013502	004737	030452			JSR	PC, GETSP		;GET SPACE FROM TTY
3607									
3608	013506	012765	000001	000026		MOV	#1, RKMR1(R5)		;SELECT WORD 1
3609	013514	004737	026106			JSR	PC, GSTAT		
3610	013520	032737	000400	005366		BIT	#D. CART, HMR2		
3611	013526	001401				BEQ	3\$		
3612	013530	104072				ERROR	72		;CARTRIDGE STATUS BIT NOT RESET
3613									
3614									
3615	013532	004737	026436		3\$:	JSR	PC, SUBCLR		
3616	013536	104024				ERROR	24		;CERR AFTER SCLR
3617									
3618	013540	104401	040401			TYPE	,MSG35		;PRESS 'RUN-STOP' TO 'RUN'
3619	013544	104401	040313			TYPE	,MSG34		;VERIFY DOES NOT START
3620	013550	104401	040573			TYPE	,MSG37		
3621	013554	004737	030452			JSR	PC, GETSP		;GET SPACE FROM TTY
3622									
3623	013560	004737	026106			JSR	PC, GSTAT		
3624	013564	032737	010000	005366		BIT	#D. SPIN, HMR2		
3625	013572	001401				BEQ	5\$		
3626	013574	104100				ERROR	100		;SPIN SET IN MSG A0
3627									
3628	013576	104401	043432		5\$:	TYPE	,MSG65		;PRESS 'RUN-STOP' TO 'STOP'
3629	013602	104401	040476			TYPE	,MSG36		;INSERT CARTRIDGE, SET 'RUN' SW. & CLOSE DOOR
3630	013606	104401	043672			TYPE	,MSG70		;VERIFY HEADS LOAD
3631	013612	104401	043615			TYPE	,MSG69		;DEPRESS SPACE WHEN READY GOES ON
3632	013616	004737	030452			JSR	PC, GETSP		;GET SPACE FROM TTY
3633	013622	005037	005322			CLR	UNLD		;FOR VALID HALT
3634									
3635	013626	012765	100000	000000		MOV	#CCLR, RKCS1(R5)		
3636	013634	005065	000026			CLR	RKMR1(R5)		;SELECT WORD 0
3637	013640	004737	026106			JSR	PC, GSTAT		
3638	013644	032737	000100	005366		BIT	#D. VV, HMR2		
3639	013652	001401				BEQ	7\$		
3640	013654	104033				ERROR	33		;VV NOT CLEARED AFTER PACK RE-INSERTED
3641									
3642	013656	012737	050240	005430	7\$:	MOV	#<D. DSC!D. SPIN!D. DRDY!D. DRA>, E. A0		;EXPECTED A0
3643	013664	005037	005432			CLR	E. B0		
3644	013670	012737	001720	005434		MOV	#<D. SPOK!D. CART!D. SSP!D. BRHM!D. DOOR>, E. A1		
3645	013676	012737	000001	005436		MOV	#1, E. B1		
3646									
3647	013704	004737	025264			JSR	PC, CHKMSG		;CHECK MSGS A0, B0, A1, B1
3648	013710	000000				.WORD	0!0!0		; & MSGS SPECIFIED HERE
3649	013712	104074				ERROR	74		;MSG A0 ERROR AFTER PACK REINSERTED & HEADS LOADED


```

3650 013714 104075          ERROR 75          ;MSH B0 ERROR
3651 013716 104076          ERROR 76          ;MSG A1 ERROR
3652 013720 104077          ERROR 77          ;MSG B1 ERROR
3653
3654 013722 012765 100000 000000      MOV      #CCLR,RKCS1(R5)
3655 013730 013765 001222 000010      MOV      $UNIT,RKCS2(R5) ;DRIVE #
3656 013736 012737 000003 005340      MOV      #PACK,HCS1
3657 013744 004737 024446          JSR      PC,DOCMD      ;DO PACK CMD & GET CONTR RDY
3658 013750 104116          ERROR 116         ;CONTR NOT RDY
3659
3660 C13752 032737 000100 005366      BIT      #D.VV,HMR2
3661 013760 001001          BNE     64$
3662 013762 104027          ERROR 27          ;VOLUME VALID NOT SET AFTER PACK CMD
3663 013764
3664 013764 005237 005322      64$: INC      UNLD          ;FOR VALID HALT
3665
3666 013770 004737 026436      JSR      PC,SUBCLR
3667 013774 104024          ERROR 24          ;CERR AFTER SCLR
3668
3669 013776 012737 000007 005340      MOV      #UNLOAD,HCS1
3670 014004 004737 024446          JSR      PC,DOCMD      ;DO UNLOAD CMD & GET CONTR READY
3671 014010 104017          ERROR 17          ;NO RDY AFTER UNLD CMD
3672 014012 004737 025024      JSR      PC,TSTATN
3673 014016 104020          ERROR 20          ;NO ATTN AFTER UNLOAD CMD
3674 014020
3675
3676
3677
3678
3679
3680
3681
3682
3683
3684
3685
3686
3687
3688 014020 000004          TST11: SCOPE
3689 014022 012737 000001 001174      MOV      #1,$TIMES      ;:DO 1 ITERATION
3690 014030 012706 001100          MOV      #STACK,SP      ;:RESTORE STACK PTR
3691
3692 014034 004737 026436      JSR      PC,SUBCLR
3693 014040 104024          ERROR 24          ;CERR AFTER SCLR
3694
3695 014042 104401 037446      TYPE     ,MSG18         ;UNIT SELECT PLUG TEST
3696 014046 104401 041765      TYPE     ,MSG52         ;CONT-E TO EXIT OR SPACE TO CONTINUE
3697 014052 004737 030412      JSR      PC,CCSP        ;INPUT CONT-E OR SPACE
3698 014056 000137 014652      JMP      12$           ;RETURN HERE FOR CONT-E
3699
3700 014062 012765 000020 000026      MOV      #PAT,RKMR1(R5) ;EVEN PARITY TO GET DSC & ATTN
3701 014070 004737 026106      JSR      PC,GSTAT
3702 014074 032737 001000 005370      BIT      #D.PAR,HMR3   ;SEE IF PARITY SET
3703 014102 001001          BNE     2$            ;BR IF YES
3704 014104 104114          ERROR 114         ;PARITY BIT NOT SET AFTER SEL DRIVE CMD
3705

```

```

*****
: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 CONT-E
:
*****

```

CZR6JED UNIBUS RK6 DR PRT3
CZR6JE.P11 25-JAN-78 12:08MACY11 30A(1052) 25-JAN-78 12:16 PAGE 73
T11 UNIT SELECT PLUG TEST

SEQ 0073

```

3706 014106 032737 040000 005366 2$: BIT #D.DSC,HMR2 ;SEE IF DSC SET
3707 014114 001001 BNE 1$
3708 014116 104107 ERROR 107 ;DSC NOT SET AFTER SEL DRV WITH EVEN PARITY
3709
3710 014120 012765 100000 000000 1$: MOV #CCLR,RKCS1(R5)
3711 014126 004737 026106 JSR PC,GSTAT
3712 014132 004737 025024 JSR PC,TSTATN
3713 014136 104113 ERROR 113 ;NO ATTN AFTER SEL DRV WITH EVEN PARITY
3714
3715 014140 104401 037121 TYPE ,MSG9 ;REMOVE UNIT SELECT PLUG
3716 014144 104401 040573 TYPE ,MSG37 ;DEPRESS SPACE WHEN DONE
3717 014150 004737 030452 JSR PC,GETSP ;GET SPACE
3718
3719 014154 012765 100000 000000 MOV #CCLR,RKCS1(R5)
3720 014162 004737 026106 JSR PC,GSTAT
3721 014166 004737 025024 JSR PC,TSTATN ;RETURN HERE FOR SPACE
3722 014172 000401 BR 3$ ;NO ATTN
3723 014174 104111 ERROR 111 ;REMOVING UNIT SEL PLUG. DID NOT
3724 ;DISABLE ATTN.
3725 014176 012765 100000 000000 3$: MOV #CCLR,RKCS1(R5)
3726 014204 004737 026106 JSR PC,GSTAT
3727 014210 032765 010000 000010 BIT #NED,RKCS2(R5)
3728 014216 001001 BNE 4$
3729 014220 104112 ERROR 112 ;REMOVING UNIT SEL PLUG DID NOT
3730 ;ASSERT NON-EXISTENT DRIVE
3731
3732 014222 104401 043500 4$: TYPE ,MSG67 ;DESELECT ALL OTHER PORTS
3733 014226 104401 040573 TYPE ,MSG37 ;TYPE SPACE WHEN DONE
3734 014232 004737 030452 JSR PC,GETSP ;GET SPACE
3735
3736 014236 104401 040637 TYPE ,MSG38 ;INSERT UNIT SELECT PLUGS
3737 014242 104401 040775 TYPE ,MSG39 ;DEPRESS CONTROL-E WHEN FINISHED
3738 014246 104401 042246 TYPE ,MSG55 ;EXIT WITH CORRECT UNIT SELECT PLUG #
3739 014252 013746 001222 MOV $UNIT,-(SP) ;SAVE $UNIT FOR TYPEOUT
3740 ;TYPE CORRECT#
3741 014256 104403 TYPOS ;GO TYPE--OCTAL ASCII
3742 014260 001 .BYTE 1 ;TYPE 1 DIGIT(S)
3743 014261 000 .BYTE 0 ;SUPPRESS LEADING ZEROS
3744 014262 104401 001205 TYPE ,$CRLF
3745
3746 014266 105037 033342 CLRB $TKQSRT ;CLEAR PREVIOUS INFO
3747 014272 005037 001160 CLR $TMPO
3748
3749 014276 004737 033344 5$: JSR PC,$TKINT
3750 014302 113737 033342 001160 MOV $TKQSRT,$TMPO ;GET CHAR IF THERE
3751 014310 042737 177600 001160 BIC #1C<177>,$TMPO ;GET RID OF JUNK, IF ANY
3752 014316 005737 001160 TST $TMPO
3753 014322 001422 BEQ 6$ ;BR IF NOTHING TYPED YET
3754 014324 023727 001160 000005 CMP $TMPO,#5 ;SEE IF CONT-E
3755 014332 001470 BEQ 9$ ;BR IF YES
3756 014334 023727 001160 000003 CMP $TMPO,#3 ;SEE IF CONT-C
3757 014342 001004 BNE 13$ ;BR IF NO
3758 014344 004737 033344 JSR PC,$TKINT ;ENABLE KB INT
3759 014350 000137 030502 JMP STOP1 ;ELSE DO VALID HALT
3760 014354 104401 040160 13$: TYPE ,MSG31 ;?
3761 014360 105037 033342 CLRB $TKQSRT

```

J06

CZR6JED UNIBUS RK6 DR PRT3
CZR6JE.P11 25-JAN-78 12:08

MACY11 30A(1052) 25-JAN-78 12:16 PAGE 74
T11 UNIT SELECT PLUG TEST

SEQ 0074

3762	014364	004737	033344			JSR	PC, \$TKINT	
3763								
3764	014370	005000			6\$:	CLR	RD	
3765	014372	012765	100000	000000	7\$:	MOV	#CCLR, RKCS1(R5)	
3766	014400	010065	000010			MOV	RD, RKCS2(R5)	; DRIVE NO.
3767	014404	012737	000001	005340		MOV	#SELDRV, HCS1	
3768	014412	004737	024446			JSR	PC, DOCMD	; DO SELDRV (STATUS) CMD & GET CONTR RDY
3769	014416	104117				ERROR	117	; CONTR RDY NOT SET
3770								
3771	014420	032765	010000	000010		BIT	#NED, RKCS2(R5)	; SEE IF UNIT SELECT PLUG INSERTED
3772	014426	001405				BEQ	8\$; & IF MATCH DRIVE NO IN CS2. BR IF YES
3773	014430	005200				INC	RD	
3774	014432	020027	000010			CMP	RD, #8.	; ALL 8 DRIVE NOS TRIED?
3775	014436	001355				BNE	7\$; BR IF NO
3776	014440	000716				BR	5\$; ELSE RETURN
3777								
3778	014442	032737	000100	005366	8\$:	BIT	#D.VV, HMR2	
3779	014450	001312				BNE	5\$; TYPE SAME UNIT SELECT PLUG RDY ONCE
3780	014452	012765	100000	000000		MOV	#CCLR, RKCS1(R5)	
3781	014460	010065	000010			MOV	RD, RKCS2(R5)	; DRIVE ADDR
3782	014464	012737	000003	005340		MOV	#PACK, HCS1	
3783	014472	004737	024446			JSR	PC, DOCMD	; DO PACK CMD & GET CONTR RDY
3784	014476	104116				ERROR	116	; CONTR NOT RDY
3785	014500	010046				MOV	RD, -(SP)	; SAVE RD FOR TYPEOUT
3786								; TYPE UNIT SEL PLUG NO.
3787	014502	104403				TYPOS		; GO TYPE--OCTAL ASCII
3788	014504	001				.BYTE	1	; TYPE 1 DIGIT(S)
3789	014505	000				.BYTE	0	; SUPPRESS LEADING ZEROS
3790	014506	104401	001205			TYPE	\$CRLF	
3791	014512	000671				BR	5\$	
3792								
3793	014514	004737	033344		9\$:	JSR	PC, \$TKINT	; ENABLE KB INT
3794	014520	012765	100000	000000		MOV	#CCLR, RKCS1(R5)	
3795	014526	013765	001222	000010		MOV	\$UNIT, RKCS2(R5)	
3796	014534	012737	000001	005340		MOV	#SELDRV, HCS1	
3797	014542	004737	024446			JSR	PC, DOCMD	; DO SELDRV (STATUS) CMD & GET CONTR RDY
3798	014546	104117				ERROR	117	; CONT NOT RDY AFTER SEL DRV CMD
3799								
3800	014550	032765	010000	000010		BIT	#NED, RKCS2(R5)	; SEE IF CORRECT PLUG FOR EXIT
3801	014556	001411				BEQ	10\$; BR IF YES
3802	014560	104401	042246			TYPE	MSG55	; EXIT WITH CORRECT UNIT SEL PLUG NO.
3803	014564	013746	001222			MOV	\$UNIT, -(SP)	; SAVE \$UNIT FOR TYPEOUT
3804								; TYPE CORRECT UNIT SEL PLUG #
3805	014570	104403				TYPOS		; GO TYPE--OCTAL ASCII
3806	014572	001				.BYTE	1	; TYPE 1 DIGIT(S)
3807	014573	000				.BYTE	0	; SUPPRESS LEADING ZEROS
3808	014574	104401	040775			TYPE	MSG39	; DEPRESS CONT-E WHEN DONE
3809	014600	000636				BR	5\$	
3810								
3811	014602	004737	026436		10\$:	JSR	PC, SUBCLR	
3812	014606	104024				ERROR	24	; CERR AFTER SCLR
3813								
3814	014610	004737	026106			JSP	PC, GSTAT	
3815	014614	032737	000100	005366		BIT	#D.VV, HMR2	
3816	014622	001005				BNE	11\$	
3817	014624	104401	041040			TYPE	,MSG40	; VV NOT SET. INSERT UNIT SELECT PLUG

3818 014630 104401 040775
 3819 014634 000620
 3820
 3821 014636 104401 043720
 3822 014642 1044 040573
 3823 014646 004737 030452
 3824 014652
 3825
 3826
 3827
 3828
 3829
 3830
 3831
 3832
 3833
 3834
 3835 014652 000004
 3836 014654 012737 000001 001174
 3837 014662 012706 001100
 3838
 3839 014666 004737 026436
 3840 014672 104024
 3841
 3842 014674 104401 037500
 3843 014700 104401 041765
 3844 014704 004737 030412
 3845 014710 000137 015070
 3846
 3847 014714 104401 041152
 3848 014720 104401 040573
 3849 014724 004737 030452
 3850
 3851 014730 004737 026106
 3852 014734 032765 010000 000010
 3853 014742 001001
 3854 014744 104122
 3855
 3856
 3857 014746 104401 041232
 3858 014752 104401 040573
 3859 014756 004737 030452
 3860
 3861 014762 004737 026436
 3862 014766 104024
 3863
 3864 014770 032765 010000 000010
 3865 014776 001401
 3866 015000 104130
 3867
 3868 015002 104401 041310
 3869 015006 104401 040573
 3870 015012 004737 030452
 3871 015016 004737 026106
 3872 015022 032765 010000 000010
 3873 015030 001001

```

TYPE MSG39 ;DEPRESS CONT-E TO EXIT TEST
BR 5$

11$: TYPE ,MSG71 ;SELECT CORRECT PORT ON OTHER DRIVES
TYPE MSG37 ;TYPE SPACE WHEN DONE
JSR PC,GETSP ;GET SPACE

12$:

*****
*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
*****
TST12: SCOPE
MOV #1,STIMES ;DO 1 ITERATION
MOV #STACK,SP ;RESTORE STACK PTR

JSR PC,SUBCLR
ERROR 24 ;CERR AFTER SCLR

TYPE ,MSG19 ;PORT SELECTION TESTS
TYPE ,MSG52 ;DEPRESS CONT-E TO EXIT OR SPACE BAR TO CONTINUE
JSR PC,CCSP ;INPUT CONT-E OR SPACE
JMP 4$ ;RETURN HERE FOR CONT-E

TYPE ,MSG41 ;SWITCH TO OTHER PORT
TYPE ,MSG37 ;DEPRESS SPACE WHEN DONE
JSR PC,GETSP ;GET SPACE

JSR PC,GSTAT
BIT #NED,RKCS2(R5)
BNE 1$
ERROR 122 ;NED NOT SET AFTER WRONG PORT SELECTED

1$: TYPE ,MSG42 ;SELECT CORRECT PORT
TYPE ,MSG37 ;DEPRESS SPACE BAR WHEN DONE
JSR PC,GETSP ;GET SPACE

JSR PC,SUBCLR
ERROR 24 ;CERR AFTER SCLR

BIT #NED,RKCS2(R5)
BEQ 2$
ERROR 130 ;NED NOT CLEARED AFTER CORRECT PORT SELECTED

2$: TYPE ,MSG43 ;DESELECT BOTH PORTS
TYPE ,MSG37 ;DEPRESS SPACE BAR WHEN DONE
JSR PC,GETSP
JSR PC,GSTAT
BIT #NED,RKCS2(R5)
BNE 3$

```

```

3874 015032 104133          ERROR 133          ;NED NOT SET AFTER BOTH PORTS DESELECTED
3875
3876
3877 015034 104401 041340    3$:  TYPE      ,MSG44          ;SELECT CORRECT PORT
3878 015040 104401 040573    TYPE      ,MSG37          ;DEPRESS SPACE BAR WHEN DONE
3879 015044 004737 030452    JSR      PC,GETSP
3880
3881 015050 004737 026436    JSR      PC,SUBCLR
3882 015054 104024          ERROR 24          ;CERR AFTER SCLR
3883
3884 015056 032765 010000 000010  BIT      #NED,RKCS2(R5)
3885 015064 001421          BEQ      TST13          ;;GOTO NEXT TEST
3886 015066 104130          ERROR 130          ;NED NOT CLEARED AFTER CORRECT PORT SELECTED
3887
3888 015070 012765 100000 000000 4$:  MOV      #CCLR,RKCS1(R5)
3889 015076 004737 026106    JSR      PC,GSTAT
3890 015102 032765 010000 000010  BIT      #NED,RKCS2(R5)
3891 015110 001407          BEQ      TST13          ;;GOTO NEXT TEST
3892 015112 104401 041370    TYPE      ,MSG45          ;CORRECT PORT NOT SELECTED, TRY AGAIN
3893 015116 104401 040573    TYPE      ,MSG37          ;DEPRESS SPACE BAR WHEN DONE
3894 015122 004737 030452    JSR      PC,GETSP
3895 015126 000760          BR       4$
3896
3897
3898
3899
3900
3901
3902
3903
3904
3905
3906
3907
3908
3909

```

```

*****
*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.
*****

```

```

3910 015130 000004          ↑TST13: SCOPE
3911 015132 012737 000001 001174  MOV      #1,STIMES          ;;DO 1 ITERATION
3912 015140 012706 001100    MOV      #S+ACK,SP          ;RESTORE STK PTR
3913
3914 015144 004737 026436    JSR      PC,SUBCLR
3915 015150 104024          ERROR 24          ;CERR AFTER SCLR
3916
3917 015152 104401 037531    TYPE      ,MSG21          ;AC LOW TEST
3918 015156 104401 041765    TYPE      ,MSG52          ;CONT-E TO BYPASS TEST
3919                                ;OR SPACE TO CONTINUE
3920 015162 004737 030412    JSR      PC,CCSP
3921 015166 000137 015442    JMP      10$
3922 015172 104401 041521    TYPE      ,MSG49          ;INPUT CONT-E OR SPACE
3923                                ;RETURN HERE IF CONT-E
3924                                ;TURN OFF AC(RET HERE IF SPACE)
3924 015176 005037 005400
3925 015202 012737 177777 005376 12$:  MOV      #-1,TEMP1          ;SETUP TIMEOUT
3926 015210 004737 026106    1$:  JSR      PC,GSTAT
3927 015214 032737 100000 005340  BIT      #CERR,HCS1
3928 015222 001012          BNE     3$
3929 015224 005337 005376          DEC     TEMP1

```

```

3930 015230 001367 BNE 15
3931 015232 005237 005400 INC TEMP2
3932 015236 023727 005400 000002 CMP TEMP2,#2 ;SEE IF GONE THRU 2 TIMES
3933 015244 001356 BNE 125 ;BR IF NO
3934 015246 104146 ERROR 146 ;CERR NOT DETECTED BEFORE TIMEOUT
3935 015250 013737 001432 005376 3$: MOV #5000,TEMP1
3936 015256 012765 100000 000000 4$: MOV #CCLR,RKCS1(R5)
3937 015264 004737 026106 JSR PC,GS1AT
3938 015270 032765 010000 000010 BIT #NED,RKCS2(R5)
3939 015276 001004 BNE 55
3940 015300 005337 005376 DEC TEMP1
3941 015304 001364 BNE 45
3942 015306 104150 ERROR 150 ;NED NOT SET BEFORE TIMEOUT
3943
3944 015310 104401 041574 5$: TYPE ,MSG50 ;SWITCH AC BACK ON
3945 015314 104401 043672 TYPE ,MSG70 ;VERIFY HEADS LOAD
3946 015320 104401 043615 TYPE ,MSG69 ;PRESS SPACE AFTER READY ON
3947 015324 004737 030452 JSR PC,GETSP ;GET SPACE
3948
3949 015330 004737 026436 JSR PC,SUBCLR
3950 015334 104024 ERROR 24 ;CERR AFTER SCLR
3951
3952 015336 032737 000200 005366 BIT #D.DRDY,HMR2 ;SEE IF LOADED
3953 015344 001001 BNE 65 ;BR IN YES
3954 015346 104141 ERROR 141 ;DRV NOT RDY AFTER AC UP
3955
3956 015350 005037 005322 6$: CLR UNLD ;FOR VALID HALT
3957 015354 032737 000100 005370 BIT #D.ACLO,HMR3
3958 015362 001401 BEQ 75
3959 015364 104152 ERROR 152 ;ACLO NOT RESET AFTER POWER UP
3960
3961 015366 032737 000100 005366 7$: BIT #D.VV,HMR2
3962 015374 001401 BEQ 85
3963 015376 104153 ERROR 153 ;VV NOT RESET AFTER POWER UP
3964
3965 015400 8$:
3966 015400 012765 100000 000000 MOV #CCLR,RKCS1(R5)
3967 015406 013765 001222 000010 MOV #UNIT,RKCS2(R5) ;DRIVE #
3968 015414 012737 000003 005340 MOV #PACK,HCS1
3969 015422 004737 024446 JSR PC,DOCMD ;DO PACK CMD & GET CONTR RDY
3970 015426 104116 ERROR 116 ;CONTR NOT RDY
3971
3972 015430 032737 000100 005366 BIT #D.VV,HMR2
3973 015436 001001 BNE 645
3974 015440 104027 ERROR 27 ;VOLUME VALID NOT SET AFTER PACK CMD
3975
3976 015442 64$:
3977 015442 10$:
3978
3979 ;*****
3980 ;*TEST 14 CHECK NXF LOGIC
3981 ;*
3982 ;* THIS TEST VERIFIES NON-EXECUTABLE FUNCTION (NXF) IS DETECTED
3983 ;* AS A RESULT OF DOING A SEEK WITH VOLUME VALID RESET.
3984 ;*
3985 ;*****
015442 000004 †ST14: SCOPE

```

```

3986 015444 012737 000001 001174 MOV #1,STIMES ;DO 1 ITERATION
3987 015452 012706 001100 MOV #STACK,SP ;RESTORE STACK PTR
3988
3989 015456 004737 026436 JSR PC,SUBCLR
3990 015462 104024 ERROR 24 ;CERR AFTER SCLR
3991
3992 015464 032737 010000 005366 BIT #D.SPIN,HMR2 ;SEE IF SPINDLE ALREADY ON
3993 015472 001020 BNE 64$ ;BR IF YES
3994 015474 104401 040032 TYPE ,MSG29 ;PLEASE WAIT, HEADS BEING LOADED
3995
3996 015500 012737 000011 005340 MOV #SRTSPL,HCS1
3997 015506 004737 024446 JSR PC,DOCMD ;DO START SPINDLE CMD & GET CONTR RDY
3998 015512 104143 ERROR 143 ;CONTR RDY NOT SET AFTER CMD
3999
4000 015514 013737 001426 005400 MOV T100,TEMP2
4001 015522 004737 025056 JSR PC,FATT1 ;FIND ATTN
4002 015526 104144 ERROR 144 ;NO ATTN AFTER CMD
4003
4004 015530 005037 005322 CLR UNLD
4005 015534 64$: CLR UNLD ;FOR VALID HALT
4006 015534 005037 005322
4007
4008 015540 004737 026436 JSR PC,SUBCLR
4009 015544 104024 ERROR 24 ;CERR AFTER SCLR
4010
4011 015546 104401 037572 TYPE ,MSG22 ;NXF TEST
4012 015552 104401 041765 TYPE ,MSG52 ;PRESS CONT-E TO EXIT OR SPACE TO CONTINUE
4013 015556 004737 030412 JSR PC,CCSP ;INPUT CONT-E OR SPACE
4014 015562 000137 016070 JMP 7$ ;RETURN HERE FOR CONT-E
4015
4016 015566 104401 041644 TYPE ,MSG51 ;REMOVE UNIT SELECT PLUG TO CLEAR VV
4017 015572 104401 040573 TYPE ,MSG37 ;PRESS SPACE WHEN DONE
4018 015576 004737 030452 JSR PC,GETSP ;GET SPACE
4019
4020 015602 004737 026106 JSR PC,GSTAT
4021 015606 032737 000100 005366 BIT #D.VV,HMR2
4022 015614 001403 BEQ 1$
4023 015616 104177 ERROR 177 ;VV NOT CLEARED AFTER UNIT SEL PLUG
4024 015620 000137 016070 JMP 7$ ;EXIT TEST
4025
4026 015624 032737 000100 005366 1$: BIT #D.VV,HMR2
4027 015632 001406 BEQ 2$
4028 015634 104155 ERROR 155 ;VV SET AFTER HEADS LOADED
4029 015636 000137 016070 JMP 7$ ;EXIT TEST
4030
4031 015642 004737 026436 JSR PC,SUBCLR
4032 015646 104024 ERROR 24 ;CERR AFTER SCLR
4033
4034 015650 012765 000001 000020 2$: MOV #1,RKDC(R5)
4035 015656 012737 000017 005340 MOV #SEEK,HCS1
4036 015664 004737 024446 JSR PC,DOCMD ;DO SEEK CMD & GET CONTR READY
4037 015670 104131 ERROR 131 ;NO RDY AFTER SEEK CMD
4038
4039 015672 013737 001432 005376 MOV T50000,TEMP1
4040 015700 004737 025152 JSR PC,FATT2 ;FIND ATTN
4041 015704 104132 ERROR 132 ;NO ATTN AFTER SEEK CMD

```

```

4042
4043 015706 032737 000400 005370 BIT #D.NXF,HMR3
4044 015714 001003 BNE 3$
4045 015716 104156 ERROR 156 ;NXF NOT SET AFTER SEEK WITH VV=0
4046 015720 000137 016070 JMP 7$ ;EXIT TEST
4047
4048 015724 032737 000200 005370 3$: BIT #D.FLT,HMR3
4049 015732 001003 BNE 4$
4050 015734 104172 ERROR 172 ;NXF DID NOT SET FAULT
4051 015736 000137 016070 JMP 7$ ;EXIT TEST
4052
4053 015742 012737 050240 005430 4$: MOV #<D.DSC!D.SPIN!D.DRDY!D.DRA>,E.A0 ;EXPECTED A0
4054 015750 012737 000600 005432 MOV #<D.NXF!D.FLT>,E.B0
4055 015756 012737 001720 005434 MOV #<D.SPOK!D.CART!D.DOOR!D.BRHM!D.SSP>,E.A1
4056 015764 012737 000001 005436 MOV #1,E.B1
4057
4058 015772 004737 025264 JSR PC,CHKMSG ;CHECK MSGS A0, B0, A1, B1
4059 015776 000003 .WORD T.A2!T.B2!0 ;& MSGS SPECIFIED HERE
4060 016000 104160 ERROR 160 ;MSG A0 ERROR AFTER SEEK WITH VV=0
4061 016002 104161 ERROR 161 ;MSG B0 ERROR
4062 016004 104162 ERROR 162 ;MSG A1 ERROR
4063 016006 104163 ERROR 163 ;MSG B1 ERROR
4064 016010 005737 001400 TST CYLADD
4065 016014 001401 BEQ 5$
4066 016016 104157 ERROR 157 ;HEADS MOVED WITH SEEK & DXF
4067
4068 016020 004737 026436 5$: JSR PC,SUBCLR
4069 016024 104024 ERROR 24 ;CERR AFTER SCLR
4070
4071 016026 012765 100000 000000 MOV #CCLR,RKCS1(R5)
4072 016034 013765 001222 000010 MOV $UNIT,RKCS2(R5) ;DRIVE #
4073 016042 012737 000003 005340 MOV #PACK,HCS1
4074 016050 004737 024446 JSR PC,DOCMD ;DO PACK CMD & GET CONTR RDY
4075 016054 104116 ERROR 116 ;CONTR NOT RDY
4076
4077 016056 032737 000100 005366 BIT #D.VV,HMR2
4078 016064 001001 BNE 65$
4079 016066 104027 ERROR 27 ;VOLUME VALID NOT SET AFTER PACK CMD
4080
4081 016070 65$:
4082 7$:

```

```

4083 *****
4084 *TEST 15 READ & SAVE BAD SECTOR INFO & TYPE PACK SERIAL *
4085 *
4086 * THIS TEST VERIFIES THAT CYL 632 (1456 FOR RK07), TRACK 2 CAN BE READ.
4087 * THIS AREA CONTAINS BAD SECTOR INFOR WHICH IS WRITTEN BY THE
4088 * FACTORY DURING MANF. ALL BAD SECTOR INFOR (BSE) WILL BE STORED
4089 * AT THIS TIME TO MASK FUTURE READ HEADER OR DATA ERROR PRINTOUTS.
4090 *
4091 * SECTORS 0,2,4,6,8 CONTAIN IDENTICAL INFO FOR 22 SECTOR HARDWARE DETECTED BAD SEC
4092 * SECTORS 10,12,14,16,18,20 CONTAIN IDENTICAL INFO FOR 22 SECTOR SOFTWARE DETECTED
4093 *
4094 * IF BSE INFO CANNOT BE READ, OR IF AFTER READING THE BSE INFO
4095 * IT IS DETERMINED THAT AN ALIGNMENT CARTRIDGE IS USED,
4096 * A MESSAGE WILL BE TYPED INDICATING THAT ALL
4097 * FUTURE FORMAT AND READ-WRITE TESTS WILL BE BYPASSED.

```



```

4154 016344 013765 005404 000004 5$: MOV TEMP4,RKBA(R5) ;RESTORE TABLE ADDR
4155 016352 062737 000002 005406 ADD #2,TEMPS ;SETUP TO READ 2 SECTORS FROM LAST
4156 016360 013765 005406 000006 MOV TEMPS,RKDA(R5)
4157 016366 000671 BR 1$
4158
4159 016370 005237 001474 3$: INC BSERR ;SET BSE FLAG
4160 016374 000501 BR TST16 ;GO TO NEXT TEST
4161 016376 005737 002330 8$: TST BSE22H+6 ;TEST CARTRIDGE TYPE
4162 016402 001405 BEQ 9$ ;BRANCH IF DATA CARTRIDGE
4163 016404 104401 042326 TYPE ,MSG56 ;ALIGNMENT CARTRIDGE USED
4164 016410 005237 001474 INC BSERR ;SET BSE ERROR FLAG
4165 016414 000426 BR 10$
4166
4167 016416 005237 005402 9$: INC TEMP3
4168 016422 023727 005402 000001 CMP TEMP3,#1
4169 016430 001020 BNE 10$
4170 016432 005037 005400 CLR TEMP2
4171 016436 012737 003322 005404 MOV #BSE22S,TEMP4 ;STORE 22 SECTOR SOFTWARE BSE ADDR
4172 016444 013765 005404 000004 MOV TEMP4,RKBA(R5)
4173 016452 012737 001012 005406 MOV #1012,TEMPS ;TRACK 2, SECTOR 12
4174 016460 013765 005406 000006 MOV TEMPS,RKDA(R5)
4175 016466 000137 016152 JMP 1$ ;REPEAT
4176
4177 016472 005737 001216 10$: TST $PASS
4178
4179
4180 016476 001014 BNE 13$ ;TYPE CART # ONLY ON 1'ST PASS
4181 016500 104401 037416 TYPE ,MSG17 ;CART SERIAL #
4182 016504 012746 002322 MOV #BSE22H,-(SP)
4183 016510 004737 034744 JSR PC,$DB20 ;CONVERT DBL BINARY WORD TO OCTAL
4184 016514 004737 035314 JSR PC,$SUPRS ;TYPE SERIAL #
4185 016520 104401 001205 TYPE ,SCLRF
4186 016524 104401 001205 TYPE ,SCLRF
4187
4188 016530 004737 026436 13$: JSR PC,SUBCLR
4189 016534 104024 ERROR 24 ;CERR AFTER SCLR
4190 ;RETURN TO CYL 0
4191
4192 016536 012737 000017 005340 MOV #SEEK,HCS1
4193 016544 004737 024446 JSR PC,DCCMD ;DO SEEK CMD & GET CONTR READY
4194 016550 104131 ERROR 131 ;NO RDY AFTER SEEK CMD
4195
4196 016552 013737 001432 005376 MOV T50000,TEMP1 ;SETUP TIMEOUT
4197 016560 004737 025152 JSR PC,FATT2 ;FIND ATTN
4198 016564 104132 ERROR 132 ;NO ATTN AFTER SEEK CMD
4199
4200 016566 032737 100000 005340 BIT #CERR,HCS1
4201 016574 001401 BEQ 64$
4202 016576 104210 ERROR 210 ;CERR AFTER SEEK CMD
4203
4204 016600 64$:
4205
4206
4207 ;*****
4208 ;*TEST 16 WRITE LOCK TEST
4209 ;*

```

E07

CZR6JED UNIBUS RK6 DR PRT3
CZR6JE.P11 25-JAN-78 12:08

MACY11 30A(1052) 25-JAN-78 12:16 PAGE 82
T16 WRITE LOCK TEST

SEQ 0082

```

4210      *      THIS TEST VERIFIES THAT DATA WRITTEN ON A SECTOR CANNOT BE
4211      *      ALTERED ONCE THE WRITE LOCK SWITCH HAS BEEN ENABLED.
4212      *      IT ALSO CHECKS THAT WRITE PROTECT TAKES EFFECT ONLY AT
4213      *      SECTOR BOUNDARIES WHEN DOING CONTINUOUS WRITING ON CYL 0, HEAD 0
4214      *
4215      * *****
4216 016600 000004          ST16: SCOPE
4217 016602 012737 000001 001174  MOV    #1,STIMES      ;DO 1 ITERATION
4218 016610 012706 001100      MOV    #STACK,SP     ;RESTORE STACK PTR
4219 016614 005737 001474      TST    BSERR         ;SEE IF ALIGN CRT
4220 016620 001402          BEQ    15$            ;BR IF NO
4221 016622 000137 021450      JMP    12$            ;ELSE EXIT TEST
4222
4223 016626 004737 026436      15$:  JSR    PC,SUBCLR   ;
4224 016632 104024          ERROR   24            ;CERR AFTER SCLR
4225
4226 016634 104401 037653      TYPE   ,MSG24        ;WRITE LOCK TEST
4227 016640 104401 041765      TYPE   ,MSG53        ;PRESS CONT-E TO EXIT OR SPACE TO CONTINUE
4228 016644 004737 030412      JSR    PC,CCSP       ;INPUT CONT-E OR SPACE
4229 016650 000137 021450      JMP    12$            ;RETURN HERE IF CONT-E
4230
4231 016654 004737 026106      JSR    PC,GSTAT      ;
4232 016660 032737 004000 005366  BIT    #D.WRL,HMR2   ;SEE IF WRITE LOCK IS ON
4233 016666 001416          BEQ    2$            ;BR IF NO
4234 016670 104401 042066      1$:   TYPE   ,MSG53        ;DISABLE WRITE LOCK
4235 016674 104401 040573      TYPE   ,MSG37        ;TYPE SPACE WHEN DONE
4236 016700 004737 030452      JSR    PC,GETSP      ;GET SPACE
4237
4238 016704 004737 026106      JSR    PC,GSTAT      ;
4239 016710 032737 004000 005366  BIT    #D.WRL,HMR2   ;SEE IF WRITE LOCK IS OFF
4240 016716 001402          BEQ    2$            ;
4241 016720 104110          ERROR   110          ;WRITE LOCK NOT DISABLED
4242 016722 000762          BR     1$            ;
4243
4244 016724 005037 001365      2$:   CLR    TOCYL       ;SETUP
4245 016730 005037 001402          CLR    CALADD        ;FOR
4246 016734 005037 001442          CLR    HEAD          ;FILL HEADER TABLE
4247 016740 005037 001450          CLR    FORMAT        ;ROUTINE
4248
4249 016744 004737 027420      16$:  JSR    PC,FHDTAB     ;BUILD STD 22 SECTOR HEADER TABLE
4250
4251 016750 012765 001506 000004  MOV    #HDTAB,RKBA(R5)
4252 016756 012765 177676 000002  MOV    #-66.,RKWC(R5)
4253 016764 000337 001442      SWAB   HEAD
4254 016770 013765 001442 000006  MOV    HEAD,RKDA(R5) ;HEAD ADDR
4255 016776 000337 001442      SWAB   HEAD
4256
4257
4258 017002 012737 000027 005340  MOV    #<WRHEAD>,HCS1
4259 017010 004737 024504      JSR    PC,DATCMD     ;DO DATA X FOR CMD & GET CONTR RDY
4260 017014 104200          ERROR   200          ;NO RDY AFTER WRITE HEADER CMD
4261 017016 004737 026106      JSR    PC,GSTAT      ;GET FRESH STATUS
4262 017022 032737 100000 005340  BIT    #CERR,HCS1
4263 017030 001401          BEQ    64$          ;
4264 017032 104201          ERROR   201          ;CERR AFTER WRITE HEADER CMD
4265 017034

```

4266											
4267											
4268	017034	005237	001442			INC	HEAD				
4269	017040	023727	001442	000003		CMP	HEAD, #3			; SEE IF ALL HEADS DONE	
4270	017046	001336				BNE	16\$; BR IF NO	
4271											
4272	017050	004737	026436			JSR	PC, SUBCLR				
4273	017054	104024				ERROR	24			; CERR AFTER SCLR	
4274											
4275	017056	005037	001422			CLR	SECTOR				
4276	017062	013765	001422	000006	14\$:	MOV	SECTOR, RKDA(R5)				
4277											
4278	017070	012765	001456	000004		MOV	#DATA0, RKBA(R5)			; WRITE ALL 0'S	
4279	017076	052765	000020	000010		BIS	#BAI, RKCS2(R5)				
4280	017104	012765	177400	000002		MOV	#-256., RKWC(R5)			; CYL 0, TRK 0, SECTOR 0	
4281											
4282	017112	012737	000023	005340		MOV	#(WRDATA), HCS1				
4283	017120	004737	024504			JSR	PC, DATCMD			; DO DATA X FOR CMD & GET CONTR RDY	
4284	017124	104011				ERROR	11			; NO RDY AFTER WRITE DATA CMD	
4285	017126	004737	026106			JSR	PC, GSTAT			; GET FRESH STATUS	
4286	017132	032737	100000	005340		BIT	#CERR, HCS1				
4287	017140	001465				BEQ	68\$; BR IF NO ERRORS	
4288											
4289	017142	032737	000200	005354		BIT	#BSE, HER			; SEE IF BAD SECTOR FLAG	
4290	017150	001421				BEQ	66\$; BR IF NO	
4291	017152	004737	030056			JSR	PC, TRUERR			; ELSE SEE IF SECTOR LISTED IN BSE TABLE	
4292	017156	000455				BR	67\$; RETURN HERE IF NO	
4293											
4294	017160	005237	001422			INC	SECTOR			; RETURN HERE IF YES	
4295	017164	023727	001422	000012		CMP	SECTOR, #10.			; ARE 10 CONSEC. SECTORS BAD	
4296	017172	001003				BNE	65\$; BR IF NO	
4297	017174	104040				ERROR	40			; ABORTING TEST DETECTED 10 BAD SECTORS	
4298	017176	000137	021450			JMP	12\$; BYPASS TEST	
4299											
4300	017202	012765	100000	000000	65\$:	MOV	#CCLR, RKCS1(R5)			; TRY ANOTHER SECTOR	
4301	017210	000137	017062			JMP	14\$				
4302											
4303	017214	104012			66\$:	ERROR	12			; CERR WITH WRITE DATA CMD	
4304											
4305	017216	012737	010340	005430		MOV	#(D.SPIN!D.DRDY!D.VV!D.DRA), E.A0			; EXPECTED MSG A0	
4306	017224	005037	005432			CLR	E.B0			; EXPECTED MSG B0	
4307	017230	012737	001720	005434		MOV	#(D.SPOK!D.CART!D.DOOR!D.BRHM!D.SSP), E.A1			; EXPECTED A1	
4308	017236	012737	000001	005436		MOV	#1, E.B1			; MSG ID FOR EXPECTED MSG B1	
4309	017244	005037	005440			CLR	E.A2			; EXPECTED MSG A2	
4310	017250	012737	000002	005442		MOV	#2, E.B2			; MSG ID FOR EXPECTED MSG B2	
4311	017256	012737	000003	005446		MOV	#3, E.B3			; MSG ID FOR EXPECTED MSG B3	
4312											
4313	017264	004737	025264			JSR	PC, CHKMSG			; CHECK MSGS A0, B0, A1, B1	
4314	017270	000003				.WORD	T.A2!T.B2!0			; & MSGS SPECIFIED HERE	
4315	017272	104052				ERROR	52			; MSG A0 ERROR AFTER WRITE DATA CMD	
4316	017274	104023				ERROR	23			; MSG B0 ERROR	
4317	017276	104053				ERROR	53			; MSG A1 ERROR	
4318	017300	104025				ERROR	25			; MSG B1 ERROR	
4319	017302	104401	044003			TYPE	MSG72			; ABORTING BALANCE OF TESTS	
4320	017306	000137	023734			JMP	\$EOP				
4321	017312	104047			67\$:	ERROR	47			; BAD SECTOR NOT LISTED IN TABLE	

```

4322 017314          68$:
4323
4324 017314 004737 026436 JSR PC,SUBCLR
4325 017320 104024 ERROR 24 ;CERR AFTER SCLR
4326
4327 017322 012765 001456 000004 MOV #DATA0,RKBA(R5)
4328 017330 052765 000020 000010 BIS #BAI,RKCS2(R5)
4329 017336 012765 177400 000002 MOV #-256,RKWC(R5)
4330 017344 013737 001456 001436 MOV DATA0,WD2 ;EXPECTED WORD FOR TRUERR TYPEOUT
4331 017352 013765 001422 000006 MOV SECTOR,RKDA(R5)
4332
4333 017360 012737 000031 005340 MOV #<WRTCHK>,HCS1
4334 017366 004737 024504 JSR PC,DATCMD ;DO DATA X FOR CMD & GET CONTR RDY
4335 017372 104015 ERROR 15 ;NO RDY AFTER WRITE CHECK CMD
4336 017374 004737 026106 JSR PC,GSTAT ;GET FRESH STATUS
4337 017400 032737 100000 005340 BIT #CERR,HCS1
4338 017406 001450 BEQ 70$
4339 017410 032737 040000 005342 BIT #WCE,HCS2 ;SEE IF WRITE CHECK ERROR
4340 017416 001405 BEQ 63$
4341 017420 016537 000024 001434 MOV RKDB(R5),WD1 ;ACTUAL WORD FOR PRINTOUT
4342 017426 104016 ERROR 16 ;WCE AFTER WRITE CMD
4343 017430 000437 BR 70$
4344
4345 017432 104022          69$: ERROR 22 ;CERR AFTER WRITE CHECK CMD
4346
4347 017434 012737 010340 005430 MOV #<O!D.SPIN!D.DRDY!D.VV!D.DRA>,E.A0 ;EXPECTED MSG A0
4348 017442 005037 005432 CLR E.B0 ;EXPECTED MSG B0
4349 017446 012737 001720 005434 MOV #<D.SPOK!D.CART!D.DOOR!D.BRHM!D.SSP>,E.A1 ;EXPECTED A1
4350 017454 012737 000001 005436 MOV #1,E.B1 ;MSG ID FOR EXPECTED MSG B1
4351 017462 005037 005440 CLR E.A2 ;EXPECTED MSG A2
4352 017466 012737 000002 005442 MOV #2,E.B2 ;MSG ID FOR EXPECTED MSG B2
4353 017474 012737 000003 005446 MOV #3,E.B3 ;MSG ID FOR EXPECTED MSG B3
4354
4355 017502 004737 025264 JSR PC,CHKMSG ;CHECK MSGS A0, B0, A1, B1
4356 017506 000003 .WORD T.A2!T.B2!0 ;& MSGS SPECIFIED HERE
4357 017510 104057 ERROR 57 ;MSG A0 ERROR AFTER WRITE CHECK CMD
4358 017512 104031 ERROR 31 ;MSG B0 ERROR
4359 017514 104060 ERROR 60 ;MSG A1 ERROR
4360 017516 104032 ERROR 32 ;MSG B1 ERROR
4361 017520 104401 044003 TYPE #MSG72 ;ABORTING BALANCE OF TESTS
4362 017524 000137 023374 JMP ENDRV
4363
4364 017530          70$:
4365
4366 017530 004737 026436          3$: JSR PC,SUBCLR
4367 017534 104024 ERROR 24 ;CERR AFTER SCLR
4368
4369 017536 104401 042157 TYPE #MSG54 ;ENABLE WRITE LOCK
4370 017542 104401 040573 TYPE #MSG37 ;PRESS SPACE WHEN DONE
4371 017546 004737 030452 JSR PC,GETSP ;GET SPACE
4372
4373 017552 012765 100000 000000 MOV #CCLR,RKCS1(R5)
4374 017560 004737 026106 JSR PC,GSTAT
4375 017564 032737 004000 005366 BIT #D.WRL,HMR2 ;SEE IF WRITE LOCK IS ON
4376 017572 001002 BNE 4$
4377 017574 104115 ERROR 115 ;WRITE LOCK NOT ENABLED

```

H07

CZR6JED UNIBJS RK6 DR PRT3
CZR6JE.P11 25-JAN-78 12:08

MACY11 30A(1052) 25-JAN-78 12:16 PAGE 85
T16 WRITE LOCK TEST

SEQ 0085

```

4378 017576 000754 BR 35
4379
4380 017600 012765 001462 000004 4$: MOV #DATA1,RKBA(R5) ;ATTEMPT TO WRITE ALL 1'S WITH WRITE LOCK SET
4381 017606 052765 000020 000010 BIS #BAI,RKCS2(R5)
4382 017614 012765 177400 000002 MOV #-256,RKWC(R5) ; CYLINDER 0, HEAD 0
4383 017622 013765 001422 000006 MOV SECTOR,RKDA(R5)
4384 017630 012737 000023 005340 MOV #WRDATA,HCS1
4385 017636 004737 024504 JSR PC,DATCMD ;DO WRITE DATA CMD & GET CONTR RDY
4386 017642 104116 ERROR 116 ;CONTR NOT RDY
4387
4388 017644 004737 026106 JSR PC,GSTAT ;GET FRESH STATUS
4389 017650 032737 100000 005340 BIT #CERR,HCS1
4390 017656 001001 BNE 17$
4391 017660 104207 ERROR 207 ;CERR NOT SET AFTER WRITE WITH WRL SET
4392 017662 032737 004000 005370 17$: BIT #D.WLE,HMR3 ;CHECK WRITE LOCK ERROR SET
4393 017670 001001 BNE 5$ ;BR IF SET
4394 017672 104121 ERROR 121 ;WRITE LOCK ERROR NOT SET
4395 ;AFTER WRITE DATA WITH WRITE LOCK SET
4396 017674 012737 054340 005430 5$: MOV #<D.DSC!D.SPIN!D.WRL!D.DRDY!D.VV!D.DRA>,E.A0 ;EXP A0
4397 017702 012737 004200 005432 MOV #<D.WLE!D.FLT>,E.B0
4398 017710 012737 001720 005434 MOV #<D.SPOK!D.CART!D.DOOR!D.BRHM!D.SSP>,E.A1
4399 017716 012737 000001 005436 MOV #1,E.B1
4400
4401 017724 004737 025264 JSR PC,CHKMSG ;CHECK MSGS A0, B0, A1, B1
4402 017730 000000 WORD 0:0:0 ;& MSGS SPECIFIED HERE
4403 017732 104123 ERROR 123 ;MSG A0 ERROR AFTER WRITE WITH WRITE LOCK SET
4404 017734 104125 ERROR 125 ;MSG B0 ERROR
4405 017736 104126 ERROR 126 ;MSG A1 ERROR
4406 017740 104127 ERROR 127 ;MSG B1 ERROR
4407
4408 017742 004737 026436 JSR PC,SUBCLR
4409 017746 104024 ERROR 24 ;CERR AFTER SCLR
4410
4411 017750 012765 001456 000004 MOV #DATA0,RKBA(R5) ;CHECK THAT NONE OF ORIG DATA
4412 017756 052765 000020 000010 BIS #BAI,RKCS2(R5) ;HAS CHANGED
4413 017764 012765 177400 000002 MOV #-256,RKWC(R5) ;AS RESULT OF WRITE WITH WRITE LOCK SET
4414 017772 013737 001456 001436 MOV DATA0,WD2 ;EXPECTED WORD FOR TRUERR TYPEOUT
4415 020000 013765 001422 000006 MOV SECTOR,RKDA(R5)
4416
4417 020006 012737 000031 005340 MOV #<WRTCHK>,HCS1
4418 020014 004737 024504 JSR PC,DATCMD ;DO DATA X FOR CMD & GET CONTR RDY
4419 020020 104015 ERROR 15 ;NO RDY AFTER WRITE CHECK CMD
4420 020022 004737 026106 JSR PC,GSTAT ;GET FRESH STATUS
4421 020026 032737 100000 005340 BIT #CERR,HCS1
4422 020034 001450 BEQ 72$
4423 020036 032737 040000 005342 BIT #WCE,HCS2 ;SEE IF WRITE CHECK ERROR
4424 020044 001405 BEQ 71$
4425 020046 016537 000024 001434 MOV RKDB(R5),WD1 ;ACTUAL WORD FOR PRINTOUT
4426 020054 104016 ERROR 16 ;WCE AFTER WRITE CMD
4427 020056 000437 BR 72$
4428
4429 020060 104022 71$: ERROR 22 ;CERR AFTER WRITE CHECK CMD
4430
4431 020062 012737 010340 005430 MOV #<0!D.SPIN!D.DRDY!D.VV!D.DRA>,E.A0 ;EXPECTED MSG A0
4432 020070 005037 005432 CLR E.B0 ;EXPECTED MSG B0
4433 020074 012737 001720 005434 MOV #0,SPOK!D.CART!D.DOOR!D.BRHM!D.SSP>,E.A1 ;EXPECTED A1

```

```

4434 020102 012737 000001 005436      MOV      #1,E.B1      ;MSG ID FOR EXPECTED MSG B1
4435 020110 005037 005440      CLR      E.A2        ;EXPECTED MSG A2
4436 020114 012737 000002 005442      MOV      #2,E.B2      ;MSG ID FOR EXPECTED MSG B2
4437 020122 012737 000003 005446      MOV      #3,E.B3      ;MSG ID FOR EXPECTED MSG B3
4438
4439 020130 004737 025264      JSR      PC,CHKMSG    ;CHECK MSGS A0, B0, A1, B1
4440 020134 000003          .WORD    T.A2!T.B2!0  ;& MSGS SPECIFIED HERE
4441 020136 104057      ERROR    57          ;MSG A0 ERROR AFTER WRITE CHECK CMD
4442 020140 104031      ERROR    31          ;MSG B0 ERROR
4443 020142 104060      ERROR    60          ;MSG A1 ERROR
4444 020144 104032      ERROR    32          ;MSG B1 ERROR
4445 020146 104401 044003      TYPE     ,MSG72      ;ABORTING BALANCE OF TESTS
4446 020152 000137 023374      JMP      ENDRV
4447
4448 020156          72$:
4449
4450 020156 004737 026436      6$:      JSR      PC,SUBCLR    ;CERR AFTER SCLR
4451 020162 104024      ERROR    24
4452
4453 020164 104401 042066      TYPE     ,MSG53      ;DISABLE WRITE LOCK
4454 020170 104401 040573      TYPE     ,MSG37      ;SPACE BAR WHEN DONE
4455 020174 004737 030452      JSR      PC,GETSP     ;GET SPACE
4456 020200 004737 026106      JSR      PC,GSTAT
4457 020204 032737 004000 005366      BIT      #D.WRL,HMR2 ;SEE IF WRITE LOCK OFF
4458 020212 001361      BNE     6$
4459
4460          ;TEST FOR WRITE LOCK AT SECTOR BOUNDRIES
4461 020214 104401 042157      TYPE     ,MSG54      ;FOR CONTINUOUS WRITING ON CYL 0, TRACK 0,1,2
4462 020220 013737 001430 001160      MOV      #5000,$TMP0 ;SET WRITE LOCK
4463 020226 005037 001162      CLR      $TMP1
4464 020232 005037 001366      CLR      TOCYL
4465
4466 020236 004737 026436      7$:      JSR      PC,SUBCLR    ;CERR AFTER SCLR
4467 020242 104024      ERROR    24
4468
4469 020244 032737 000001 001162      BIT      #BIT0,$TMP1 ;BR IF WRITING 1'S
4470 020252 001004      BNE     8$
4471 020254 012765 001456 000004      MOV      #DATA0,RKBA(R5) ;SETUP ALL 0'S
4472 020262 000403      BR
4473
4474 020264 012765 001462 000004 8$:      MOV      #DATA1,RKBA(R5) ;SETUP ALL 1'S
4475 020272 052765 000020 000010 9$:      BIS      #BA1,RKCS2(R5)
4476 020300 012765 140400 000002      MOV      #-63,*256.,RKWC(R5) ;WRITE TRACK 0,1,2 63 SECTORS
4477 020306 005065 000006      CLR      RKDA(R5)    ;BEGIN AT TRACK AND SEC 0
4478 020312 013765 001366 000020      MOV      TOCYL,RKDC(R5)
4479 020320 012737 000023 005340      MOV      #WRDATA,HCS1
4480 020326 004737 024504      JSR      PC,DATCMD   ;DO WRITE DATA CMD & GET CONTR RDY
4481 020332 104011      ERROR    11          ;CONTR NOT RDY
4482
4483 020334 032737 100000 005340      BIT      #CERR,HCS1
4484 020342 001017      BNE     13$
4485 020344 005337 001160      DEC      $TMP0
4486 020350 001011      BNE     10$
4487 020352 104204      ERROR    204        ;CERR NOT SET BY TIMEOUT
4488 020354 104401 042066      TYPE     ,MSG53      ;DISABLE WRITE LOCK
4489 020360 104401 040573      TYPE     ,MSG37      ;PRESS SPACE WHEN DONE

```

4490	020364	004737	030452			JSR	PC,GETSP		;INPUT SPACE
4491	020370	000137	021450			JMP	12\$;EXIT TEST
4492									
4493	020374	005237	001162		10\$:	INC	\$TMP1		
4494	020400	000716				BR	7\$;GO WRITE OPPOSITE DATA
4495	020402	032737	000200	005354	13\$:	BIT	#BSE,HER		;SEE IF BAD SECTOR
4496	020410	001411				BEQ	21\$;BR IF NO
4497	020412	005237	001366			INC	TOCYL		;ELSE TRY ANOTHER CYL
4498	020416	023727	001366	000012		CMP	TOCYL,#10.		;SEE IF 10 CONSEC CYL BAD
4499	020424	001304				BNE	7\$;BR IF NO & DO AGAIN
4500	020426	104062				ERROR	62		;10 BAD CONSEC CYLINDERS
4501	020430	000137	021450			JMP	12\$;EXIT TEST
4502									
4503	020434	032737	004000	005370	21\$:	BIT	#D.WLE,HMR3		
4504	020442	001001				BNE	11\$		
4505	020444	104205				ERROR	20\$;NO WRL BY TIMEOUT
4506									
4507	020446	104401	042066		11\$:	TYPE	,MSG53		;DISABLE WRITE LOCK
4508	020452	104401	040573			TYPE	MSG37		;TYPE SPACE WHEN DONE
4509	020456	004737	030452			JSR	PC,GETSP		
4510									
4511	020462	013737	005350	001164		MOV	HDA,\$TMP2		;STORE RKDA OF ERROR
4512	020470	013737	005350	001166		MOV	HDA,\$TMP3		;STORE FOR ERROR TYPEOUT
4513									
4514	020476	023727	001164	000001		CMP	\$TMP2,#1		;SEE IF WRL ON TRK 0, SEC 1
4515	020504	001003				BNE	22\$;BR IF NO
4516	020506	104401	044201			TYPE	MSG100		;ELSE WRL OCCURRED ON TRK 2, SEC 21
4517	020512	000621				BR	6\$		& NO NEW DATA XFER TOOK PLACE
4518									
4519	020514	005737	001164		22\$:	TST	\$TMP2		;SEE IF WRL ON TRK 0, SEC 0
4520	020520	001004				BNE	23\$;BR IN NO
4521	020522	104401	044201			TYPE	MSG100		;ELSE WRL ON TRK 2, SEC 20
4522	020526	000137	020156			JMP	6\$		& NO NEW DATA XFER TOOK PLACE
4523									
4524	020532	023727	001164	001025	23\$:	CMP	\$TMP2,#1025		;SEE IF WRL ON TRK 2, SEC 21
4525	020540	001004				BNE	24\$;BR IF NO
4526	020542	104401	044201			TYPE	MSG100		;ELSE WRL ON TRK 2, SEC 19
4527	020546	000137	020156			JMP	6\$		& NO NEW DATA XFER TOOK PLACE
4528									
4529	020552	023727	001164	001024	24\$:	CMP	\$TMP2,#1024		;SEE IF WRL ON TRK 2, SEC 20
4530	020560	001004				BNE	25\$;BR IF NO
4531	020562	104401	044201			TYPE	MSG100		;ELSE WRL ON TRK 2, SEC 18
4532	020566	000137	020156			JMP	6\$		& NO OLD DATA TO CHECK AGAINST
4533									
4534	020572	023727	001164	000400	25\$:	CMP	\$TMP2,#400		;SEE IF WRL AT TRK 1, SEC 0
4535	020600	001004				BNE	18\$;BR IF NO
4536	020602	012765	000025	000006		MOV	#21.,RKDA(R5)		;ELSE SECTOR AT WRL IS TRK 0, SEC 21
4537	020610	000415				BR	20\$		
4538									
4539	020612	023727	001164	001000	18\$:	CMP	\$TMP2,#1000		;SEE IF WRL AT TRK 2, SEC 0
4540	020620	001004				BNE	19\$;BR IF NO
4541	020622	012765	000425	000006		MOV	#425,RKDA(R5)		;ELSE SECTOR AT WRL IS TRK 1, SEC 21
4542	020630	000405				BR	20\$		
4543									
4544	020632	005337	001164		19\$:	DEC	\$TMP2		;GET SECTOR AT WRL
4545	020636	013765	001164	000006		MOV	\$TMP2,RKDA(R5)		


```

4546
4547 020644 016537 000006 001166 20$: MOV RKDA(R5), $TMP3 ;FOR ERROR PRINTOUT
4548
4549
4550
4551 020652 004737 026436 JSR PC, SUBCLR
4552 020656 104024 ERROR 24 ;CERR AFTER SCLR
4553
4554 020660 005737 001164 TST $TMP2 ;SEE IF TRK/SECTOR 0
4555 020664 001414 BEQ 80$ ;REPEAT, NO NEW DATA XFER TOOK PLACE
4556 020666 023727 001164 001023 CMP $TMP2, #1023 ;SEE IF TRK 2, SECTOR 19
4557 020674 001410 BEQ 80$ ;REPEAT, NO OLD DATA TO CHECK AGAINST
4558 020676 032737 000001 001162 BIT #BIT0, $TMP1
4559 020704 001006 BNE 73$ ;BR IF WRITING 1'S WHEN WLE OCCURRED
4560 020706 012765 001462 000004 MOV #DATA1, RKBA(R5) ;WRITING 0'S: WLE SECTOR SHOULD HAVE ALL 1'S
4561 020714 000405 BR 74$
4562
4563 020716 000137 020156 80$: JMP 6$
4564
4565 020722 012765 001456 000004 73$: MOV #DATA0, RKBA(R5) ;WRITING 1'S: WLE SECTOR SHOULD HAVE ALL 0'S
4566 020730 052765 000020 000010 74$: BIS #BA1, RKCS2(R5)
4567 020736 012765 177400 000002 MOV #-256, RKWC(R5)
4568 020744 013765 001166 000006 MOV $TMP3, RKDA(R5) ;REFRESH RKDA
4569 020752 017537 000004 001436 MOV #RKBA(R5), WD2 ;EXPECTED WORD FOR TRUERR TYPEOUT
4570 020760 013765 001366 000020 MOV TOCYL, RKDC(R5)
4571
4572 020766 012737 000031 005340 MOV #(<WATCHK>), HCS1
4573 020774 004737 024504 JSR PC, DATCMD ;DO DATA X FOR CMD & GET CONTR RDY
4574 021000 104015 ERROR 15 ;NO RDY AFTER WRITE CHECK CMD
4575 021002 004737 026106 JSR PC, GSTAT ;GET FRESH STATUS
4576 021006 032737 100000 005340 BIT #CERR, HCS1
4577 02'014 001450 BEQ 82$
4578 021016 032737 040000 005342 BIT #WCE, HCS2 ;SEE IF WRITE CHECK ERROR
4579 021024 001405 BEQ 81$
4580 021026 016537 000024 001434 MOV RKDB(R5), WD1 ;ACTUAL WORD FOR PRINTOUT
4581 021034 104134 ERROR 134 ;WCE AFTER WRITE CMD
4582 021036 000437 BR 82$
4583
4584 021040 104022 81$: ERROR 22 ;CERR AFTER WRITE CHECK CMD
4585
4586 021042 012737 010340 005430 MOV #(<O!D.SPIN!D.DRDY!D.VV!D.DRA>), E.A0 ;EXPECTED MSG A0
4587 021050 005037 005432 CLR E.B0 ;EXPECTED MSG B0
4588 021054 012737 001720 005434 MOV #(<D.SPOK!D.CART!D.DOOR!D.BRHM!D.SSP>), E.A1 ;EXPECTED A1
4589 021062 012737 000001 005436 MOV #1, E.B1 ;MSG ID FOR EXPECTED MSG B1
4590 021070 005037 005440 CLR E.A2 ;EXPECTED MSG A2
4591 021074 012737 000002 005442 MOV #2, E.B2 ;MSG ID FOR EXPECTED MSG B2
4592 021102 012737 000003 005446 MOV #3, E.B3 ;MSG ID FOR EXPECTED MSG B3
4593
4594 021110 004737 025264 JSR PC, CHKMSG ;CHECK MSGS A0, B0, A1, B1
4595 021114 000003 .WORD T.A2!T.B2!0 ; & MSGS SPECIFIED HERE
4596 021116 104057 ERROR 57 ;MSG A0 ERROR AFTER WRITE CHECK CMD
4597 021120 104031 ERROR 31 ;MSG B0 ERROR
4598 021122 104060 ERROR 60 ;MSG A1 ERROR
4599 021124 104032 ERROR 32 ;MSG B1 ERROR
4600 021126 104401 044003 TYPE #MSG72 ;ABORTING BALANCE OF TESTS
4601 021132 000137 023374 JMP ENDRV

```

```

4602
4603 021136      82$:
4604 021136 000240      NOP
4605 021140 000240      NOP
4606
4607 021142 023727 001164 000400      CMP      $TMP2,#400      ;SEE IF WRL AT TRK 1, SECTOR 0
4608 021150 001004      BNE      75$            ;BR IF NO
4609 021152 012765 000025 000006      MOV      #21.,RKDA(R5)  ;ELSE SECTOR BEFORE WRL IS TRK 0, SEC 21
4610 021160 000415      BR
4611 021162 023727 001164 001000 75$:      CMP      $TMP2,#1000   ;SEE IF WRL AT TRK 2,SECTOR 0
4612 021170 001004      BNE      76$            ;BR IF NO
4613 021172 012765 000425 000006      MOV      #425,RKDA(R5) ;ELSE SECTOR BEFORE WRL IS TRK 1, SEC 21
4614 021200 000405      BR      77$
4615
4616 021202 005337 001164      76$:      DEC      $TMP2          ;GET SECTOR BEFORE WRL
4617 021206 013765 001164 000006      MOV      $TMP2,RKDA(R5)
4618 021214 016537 000006 001166 77$:      MOV      RKDA(R5),$TMP3 ;FOR ERROR PRINTOUT
4619 021222 032737 000001 001162      BIT      #BIT0,$TMP1
4620 02.230 001004      BNE      78$            ;BR IF WRITING 1'S WHEN WLE OCCURRED
4621 021232 012765 001456 000004      MOV      #DATA0,RKBA(R5);WRITING 0'S:WLE SECTOR -1 SHOULD HAVE ALL 0'S
4622 021240 000403      BR      79$
4623
4624 021242 012765 001462 000004 78$:      MOV      #DATA1,RKBA(R5);WRITING 1'S:WLE SECTOR -1 SHOULD HAVE ALL 1'S
4625 021250 052765 000020 000010 79$:      BIS      #BA1,RKCS2(R5)
4626 021256 012765 177400 000002      MOV      #-256.,RKWC(R5)
4627 021264 017537 000004 001436      MOV      #RKBA(R5),WD2 ;EXPECTED WORD FOR TRUERR TYPEOUT
4628 021272 013765 001366 000020      MOV      TOCYL,RKDC(R5)
4629
4630 021300 012737 000031 005340      MOV      #<WRTCHK>,HCS1
4631 021306 004737 024504      JSR      PC,DATCMD     ;DO DATA X FOR CMD & GET CONTR RDY
4632 021312 104015      ERROR   15            ;NO RDY AFTER WRITE CHECK CMD
4633 021314 004737 026106      JSR      PC,GSTAT     ;GET FRESH STATUS
4634 021320 032737 100000 005340      BIT      #CERR,HCS1
4635 021326 001450      BEQ     84$
4636 021330 032737 040000 005342      BIT      #WCE,HCS2   ;SEE IF WRITE CHECK ERROR
4637 021336 001405      BEQ     83$
4638 021340 016537 000024 001434      MOV      RKDB(R5),WD1 ;ACTUAL WORD FOR PRINTOUT
4639 021346 104135      ERROR   135          ;WCE AFTER WRITE CMD
4640 021350 000437      BR      84$
4641
4642 021352 104022      83$:      ERROR   22            ;CERR AFTER WRITE CHECK CMD
4643
4644 021354 012737 010340 005430      MOV      #<0!D.SPIN!D.DRDY!D.VV!D.DRA>,E.A0 ;EXPECTED MSG A0
4645 021362 005037 005432      CLR     E.B0          ;EXPECTED MSG B0
4646 021366 012737 001720 005434      MOV      #<D.SPOK!D.CART!D.DCOR!D.BRAH!D.SSP>,E.A1 ;EXPECTED A1
4647 021374 012737 000001 005436      MOV      #1,E.B1      ;MSG ID FOR EXPECTED MSG B1
4648 021402 005037 005440      CLR     E.A2          ;EXPECTED MSG A2
4649 021406 012737 000002 005442      MOV      #2,E.B2      ;MSG ID FOR EXPECTED MSG B2
4650 021414 012737 000003 005446      MOV      #3,E.B3      ;MSG ID FOR EXPECTED MSG B3
4651
4652 021422 004737 025264      JSR      PC,CHKMSG    ;CHECK MSGS A0 B0 A1 B1
4653 021426 000033      .WORD   T.A2!T.B2!0   ;& MSGS SPECIFIED HERE
4654 021430 104057      ERROR   57            ;MSG A0 ERROR AFTER WRITE CHECK CMD
4655 021432 104031      ERROR   31            ;MSG B0 ERROR
4656 021434 104060      ERROR   60            ;MSG A1 ERROR
4657 021436 104032      ERROR   32            ;MSG B1 ERROR

```

4658 021440 104401 044003
4659 021444 000137 023374

TYPE MSG72 ;ABORTING BALANCE OF TESTS
JMP ENDRV

4660
4661 021450

84\$:

4662
4663 021450

12\$:

4664
4665

4666
4667

TEST 17 AC LOW DETECTION PART 2

4668
4669

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.

4670
4671

4672
4673

TEST17:

4674 021450 000004
4675 021452 012737 000001 001174

SCOPE
MOV #1, \$TIMES ;DO 1 ITERATION
MOV #STACK, SP ;RESTORE STK PTR

4676 021460 012706 001100

JSR PC, SUBCLR
ERROR 24 ;CERR AFTER SCLR

4677
4678 021464 004737 026436

TYPE ,MSG26 ;AC LOW-PART 2
TYPE ,MSG52 ;CONT-E TO BYPASS TEST

4679 021470 104024

JSR PC, CCSP ;OR SPACE TO CONTINUE
JMP 12\$;INPUT CONT-E OR SPACE

4680
4681 021472 104401 037677

TST BSERR ;TEST FOR ALIGN CARTRIDGE
BEQ 1\$;BR IF NOT ALIGN CART.

4682 021476 104401 041765

4683
4684 021502 004737 030412

JMP 12\$;RETURN HERE FOR CONT-E
;RETURN HERE FOR SPACE

4685 021506 000137 023374

4686
4687 021512 005737 001474

1\$:

JSR PC, GSTAT ;SEE IF WRITE LOCK
BIT #D.WRL, HMR2 ;BR IF NO
BEQ 11\$

4688 021516 001402

4689 021520 000137 022142

4690
4691 021524 004737 026106

005366

JSR PC, GSTAT ;SEE IF STILL WRITE LOCK
BIT #D.WRL, HMR2 ;BR IF NO

4692 021530 032737 004000

4693 021536 001417

BEQ 11\$;WRITE LOCK NOT DISABLED
ERROR 110 ;EXIT TEST
JMP 12\$

4694
4695 021540 104401 042066

11\$:

CLR TOCYL ;SETUP
CLR CALADD ;FOR

4696 021544 104401 040573

4697 021550 004737 030452

CLR HEAD ;FILL HEADER TABLE
CLR FORMAT ;ROUTINE

4698
4699 021554 004737 026106

13\$:

JSR PC, FHDTAB ;BUILD STD 22 SECTOR HEADER TABLE

4700 021560 032737 004000 005366

4701 021566 001403

4702 021570 104110

4703 021572 000137 023374

4704
4705 021576 005037 001366

4706 021602 005037 001402

4707 021606 005037 001442

4708 021612 005037 001450

4709
4710 021616 004737 027420

4711
4712 021622 012765 001506 000004
4713 021630 012765 177676 000002

MOV #HDTAB, RKBA(R5)
MOV #-66., RKWC(R5)

4714	021636	000337	001442		SWAB	HEAD	
4715	021642	013765	001442	000006	MOV	HEAD,RKDA(R5)	;HEAD ADDR
4716	021650	000337	001442		SWAB	HEAD	
4717							
4718	021654	012737	000027	005340	MOV	#(WRHEAD),HCS1	
4719	021662	004737	024504		JSR	PC, DATCMD	;DO DATA X FOR CMD & GET CONTR RDY
4720	021666	104200			ERROR	200	;NO RDY AFTER WRITE HEADER CMD
4721	021670	004737	026106		JSR	PC, GSTAT	;GET FRESH STATUS
4722	021674	032737	100000	005340	BIT	#CERR,HCS1	
4723	021702	001401			BEQ	64\$	
4724	021704	104201			ERROR	201	;CERR AFTER WRITE HEADER CMD
4725	021706						
4726							
4727	021706	005237	001442		INC	HEAD	
4728	021712	023727	001442	000003	CMP	HEAD,#3	;SEE IF ALL HEADS DONE
4729	021720	001336			BNE	13\$;BR IF NO
4730	021722	005037	001366		CLR	TOCYL	
4731							
4732	021726	004737	026436		JSR	PC,SUBCLR	
4733	021732	104024			ERROR	24	;CERR AFTER SCLR
4734							
4735	021734	012765	001462	000004	MOV	#DATA1,RKBA(R5)	;RETURN HERE FOR SPACE
4736	021742	052765	000020	000010	BIS	#BAI,RKCS2(R5)	;WRITE INITIAL BACKGROUND OF 1'S
4737	021750	012765	137000	000002	MOV	#-66.*256,RKWC(R5)	;TRACK 0,1,2 ALL SECTORS
4738	021756	013765	001366	000020	MOV	TOCYL,RKDC(R5)	
4739	021764	012737	000023	005340	MOV	#WRDATA,HCS1	
4740	021772	004737	024504		JSR	PC, DATCMD	;DO WRITE DATA CMD & GET CONTR RDY
4741	021776	104011			ERROR	11	;NO RDY AFTER WRITE DATA CMD
4742	022000	004737	026106		JSR	PC, GSTAT	;GET FRESH STATUS
4743	022004	032737	100000	005340	BIT	#CERR,HCS1	
4744	022012	001453			BEQ	2\$	
4745	022014	032737	000200	005354	BIT	#BSE,HER	;SEE IF BAD SECTOR
4746	022022	001034			BNE	17\$	
4747							
4748	022024	012737	010340	005430	MOV	#(D.SPIN!D.DRDY!D.VV!D.DRA),E.A0	;EXPECTED MSG A0
4749	022032	005037	005432		CLR	E.B0	;EXPECTED MSG B0
4750	022036	012737	001720	005434	MOV	#(D.SPOK!D.CART!D.DOOR!D.BRHM!D.SSP),E.A1	;EXPECTED A1
4751	022044	012737	000001	005436	MOV	#1,E.B1	;MSG ID FOR EXPECTED MSG B1
4752	022052	005037	005440		CLR	E.A2	;EXPECTED MSG A2
4753	022056	012737	000002	005442	MOV	#2,E.B2	;MSG ID FOR EXPECTED MSG B2
4754	022064	012737	000003	005446	MOV	#3,E.B3	;MSG ID FOR EXPECTED MSG B3
4755							
4756	022072	004737	025264		JSR	PC,CHKMSG	;CHECK MSGS A0, B0, A1, B1
4757	022076	000003			.WORD	T.A2!T.B2!0	; & MSGS SPECIFIED HERE
4758	022100	104052			ERROR	52	;MSG A0 ERROR AFTER WRITE DATA CMD
4759	022102	104023			ERROR	23	;MSG B0 ERROR
4760	022104	104053			ERROR	53	;MSG A1 ERROR
4761	022106	104025			ERROR	25	;MSG B1 ERROR
4762	022110	000137	023374		JMP	12\$	
4763							
4764	022114	005237	001366		INC	TOCYL	
4765	022120	023727	001366	000012	CMP	TOCYL,#10.	;TRIED 10 CYLINDERS?
4766	022126	001003			BNE	18\$;BR IF NO
4767	022130	104062			ERROR	62	;CANNOT WRITE ON 10 CONSEC CYL
4768	022132	000137	023374		JMP	12\$	
4769							

```

4770 022136 000137 021726      18$: JMP 16$
4771
4772 022142 004737 026436      2$: JSR PC,SUBCLR
4773 022146 104024                ERROR 24 ;CERR AFTER SCLR
4774
4775 022150 104401 041521                TYPE ,MSG49 ;TURN OFF AC
4776 022154 104401 042532                TYPE ,MSG57 ;VERIFY BATTERY RETRACT FUNCTIONAL
4777
4778 022160 005737 001474                TST BSERR ;SEE IF ALIGN CART USED
4779 022164 001405                BEQ 15$ ;BR IF NO
4780 022166 104401 040573                TYPE ,MSG37 ;PRESS SPACE WHEN DONE
4781 022172 004737 030452                JSR PC,GETSP ;GET SPACE
4782 022176 000523                BR 8$ ;SKIP ALL WRITING
4783
4784 022200 013737 001432 001160 15$: MOV T5000,$TMPO
4785 022206 005037 001162                CLR $TMP1 ;BIT0=0;WRITE 0'S; BIT0=1:WRITE 1'S
4786
4787 022212 004737 026436      4$: JSR PC,SUBCLR
4788 022216 104024                ERROR 24 ;CERR AFTER SCLR
4789
4790 022220 032737 000001 001162                BIT #BIT0,$TMP1
4791 022226 001004                BNE 5$ ;BR IF WRITING 1'S
4792 022230 012765 001456 000004                MOV #DATA0,RKBA(R5) ;SETUP ALL 0'S
4793 022236 000403                BR 6$
4794
4795 022240 012765 001462 000004 5$: MOV #DATA1,RKBA(R5) ;SETUP ALL 1'S
4796 022246 052765 000020 000010 6$: BIS #BA,RKCS2(R5)
4797 022254 012765 140400 000002                MOV #-63.*256.,RKWC(R5) ;TRACK 0,1,2 63 SECTORS
4798 022262 005065 000006                CLR RKDA(R5) ;BEGIN AT TRK AND SECTOR 0
4799 022266 013765 001366 000020                MOV TOCYL,RKDC(R5)
4800 022274 012737 000023 005340                MOV #WRDATA,HCS1
4801 022302 004737 024504                JSR PC,DATCMD ;DO WRITE DATA CMD & GET CONTR RDY
4802 022306 104011                ERROR 11 ;CONTR NOT RDY
4803
4804 022310 004737 026106                JSR PC,GSTAT
4805 022314 032737 100000 005340                BIT #CERR,HCS1
4806 022322 001017                BNE 3$
4807 022324 005337 001160                DEC $TMPO
4808 022330 001011                BNE 7$
4809 022332 104146                ERROR 146 ;CERR NOT SET BY TIMEOUT
4810 022334 104401 041574                TYPE ,MSG50 ;TURN AC BACK ON
4811 022340 104401 043615                TYPE ,MSG69 ;DEPRESS SPACE AFTER 'READY' LIGHT ON
4812 022344 004737 030452                JSR PC,GETSP ;GET SPACE
4813 022350 000137 023374                JMP 12$ ;EXIT TEST
4814
4815 022354 005237 001162      7$: INC $TMP1
4816 022360 000714                BR 4$ ;GO WRITE OPPOSITE DATA
4817

```

4818	022362	032737	000100	005370	3\$:	BIT	#D.ACLO,HMR3	
4819	022370	001001				BNE	10\$	
4820	022372	104147				ERROR	147	;AC LOW NOT SET
4821								
4822	022374	005237	005322		10\$:	INC	UNLD	;FOR VALID HALT
4823	022400	012737	070140	005430		MOV	#<D.DSC!D.PIP!D.SPIN!D.VV!D.DRA>,E.A0	;EXPECTED MSG A0
4824	022406	012737	010300	005432		MOV	#<D.SPLS!D.ACLO!D.FLT>,E.B0	
4825	022414	012737	044720	005434		MOV	#<D.UNLD!D.REV!D.CART!D.DOOR!D.BRHM!D.SSP>,E.A1	
4826	022422	012737	000001	005436		MOV	#1,E.B1	
4827								
4828	022430	004737	025264			JSR	PC,CHKMSG	;CHECK MSGS A0, B0, A1, B1
4829	022434	000000				WORD	0!0!0	& MSGS SPECIFIED HERE
4830	022436	104035				ERROR	35	;MSG A0 ERROR AFTER AC SWITCH OFF FROM HEADS LOADED
4831	022440	104203				ERROR	203	;MSG B0 ERROR
4832	022442	104036				ERROR	36	;MSG A1 ERROR
4833	022444	104037				ERROR	37	;MSG B1 ERROR
4834								
4835	022446	013737	005350	001164	8\$:	MOV	HDA,\$TMP2	;SAVE RKDA
4836	022454	013737	005350	001166		MOV	HDA,\$TMP3	;SAVE FOR TYPEOUT
4837								
4838	022462	104401	041574			TYPE	,MSG50	;TURN AC BACK ON
4839	022466	104401	043615			TYPE	,MSG69	;DEPRESS SPACE AFTER 'READY' LIGHT ON
4840	022472	004737	030452			JSR	PC,GETSP	;GET SPACE
4841								
4842	022476	004737	026436			JSR	PC,SUBCLR	
4843	022502	104024				ERROR	24	;CERR AFTER SCLR
4844								
4845	022504	032737	000100	005370		BIT	#D.ACLO,HMR3	
4846	022512	001401				BEQ	9\$	
4847	022514	104152				ERROR	152	;ACLO NOT RESET AFTER POWER UP
4848								
4849	022516	005037	005322		9\$:	CLR	UNLD	;FOR VALID HALT
4850	022522	012765	100000	000000		MOV	#CCLR,RKCS1(R5)	
4851	022530	013765	001222	000010		MOV	\$UNIT,RKCS2(R5)	;DRIVE #
4852	022536	012737	000003	005340		MOV	#PACK,HCS1	
4853	022544	004737	024446			JSR	PC,DOCMD	;DO PACK CMD & GET CONTR RDY
4854	022550	104116				ERROR	116	;CONTR NOT RDY
4855								
4856	022552	032737	000100	005366		BIT	#D.VV,HMR2	
4857	022560	001001				BNE	65\$	
4858	022562	104027				ERROR	27	;VOLUME VALID NOT SET AFTER PACK CMD
4859	022564				65\$:			
4860	022564	005737	001474			TST	BSERR	;SEE IF ALIGN CART USED
4861	022570	001402				BEQ	14\$;OR IF NO
4862	022572	000137	023374			JMP	12\$;ELSE EXIT TEST
4863								
4864	022576				14\$:			
4865								
4866	022576	004737	026436			JSR	PC,SUBCLR	
4867	022602	104024				ERROR	24	;CERR AFTER SCLR
4868								
4869	022604	005737	001164			TST	\$TMP2	;SEE IF TRK/SECTOR 0
4870	022610	001414				BEQ	73\$;REPEAT,NO NEW DATA XFER TOOK PLACE
4871	022612	023727	001164	001023		CMP	\$TMP2,#1023	;SEE IF TRK 2,SECTOR 19
4872	022620	001410				BEQ	73\$;REPEAT,NO OLD DATA TO CHECK AGAINST
4873	022622	032737	000001	001162		BIT	#BIT0,\$TMP1	

```

4874 022630 001006          BNE      66$      ;BR IF WRITING 1'S WHEN WLE OCCURRED
4875 022632 012765 001462 000004  MOV     #DATA1,RKBA(R5) ;WRITING 0'S:WLE SECTOR SHOULD HAVE ALL 1'S
4876 022640 000405          BR      67$
4877
4878 022642 000137 022142      73$:   JMP     2$
4879
4880 022646 012765 001456 000004 66$:   MOV     #DATA0,RKBA(R5) ;WRITING 1'S:WLE SECTOR SHOULD HAVE ALL 0'S
4881 022654 052765 000020 000010 67$:   BIS     #BA1,RKCS2(R5)
4882 022662 012765 177400 000002  MOV     #-256,RKWC(R5)
4883 022670 013765 001166 000006  MOV     $TMP3,RKDA(R5) ;REFRESH RKDA
4884 022676 017537 000004 001436  MOV     #RKBA(R5),WD2 ;EXPECTED WORD FOR TRUERR TYPEOUT
4885 022704 013765 001366 000020  MOV     TOCYL,RKDC(R5)
4886
4887 022712 012737 000031 005340  MOV     #<WATCH>,HCS1
4888 022720 004737 024504          JSR     PC,DATCMD ;DO DATA X FOR CMD & GET CONTR RDY
4889 022724 104015          ERROR  15 ;NO RDY AFTER WRITE CHECK CMD
4890 022726 004737 026106          JSR     PC,GSTAT ;GET FRESH STATUS
4891 022732 032737 100000 005340  BIT     #CERR,HCS1
4892 022740 001450          BEQ     75$
4893 022742 032737 040000 005342  BIT     #WCE,HCS2 ;SEE IF WRITE CHECK ERROR
4894 022750 001405          BEQ     74$
4895 022752 016537 000024 001434  MOV     RKDB(R5),WD1 ;ACTUAL WORD FOR PRINTOUT
4896 022760 104136          ERROR  136 ;WCE AFTER WRITE CMD
4897 022762 000437          BR      75$
4898
4899 022764 104022      74$:   ERROR  22 ;CERR AFTER WRITE CHECK CMD
4900
4901 022766 012737 010340 005430  MOV     #<O!D.SPIN!D.DRDY!D.VV!D.DRA>,E.A0 ;EXPECTED MSG A0
4902 022774 005037 005432          CLR     E.B0 ;EXPECTED MSG B0
4903 023000 012737 001720 005434  MOV     #<D.SPOK!D.CART!D.DOOR!D.BRHM!D.SSP>,E.A1 ;EXPECTED A1
4904 023006 012737 000001 005436  MOV     #1,E.B1 ;MSG ID FOR EXPECTED MSG B1
4905 023014 005037 005440          CLR     E.A2 ;EXPECTED MSG A2
4906 023020 012737 000002 005442  MOV     #2,E.B2 ;MSG ID FOR EXPECTED MSG B2
4907 023026 012737 000003 005446  MOV     #3,E.B3 ;MSG ID FOR EXPECTED MSG B3
4908
4909 023034 004737 025264          JSR     PC,CHKMSG ;CHECK MSGS A0, B0, A1, B1
4910 023040 000003          .WORD  T.A2!T.B2!0 ;& MSGS SPECIFIED HERE
4911 023042 104057          ERROR  57 ;MSG A0 ERROR AFTER WRITE CHECK CMD
4912 023044 104031          ERROR  31 ;MSG B0 ERROR
4913 023046 104060          ERROR  60 ;MSG A1 ERROR
4914 023050 104032          ERROR  32 ;MSG B1 ERROR
4915 0230' 2 104401 044003  TYPE   MSG72 ;ABORTING BALANCE OF TESTS
4916 023056 000137 023374          JMP     ENDRV
4917
4918 023062      75$:   NOP
4919 023062 000240          NOP
4920 023064 000240          NOP
4921
4922 023066 023727 001164 000400  CMP     $TMP2,#400 ;SEE IF WRL AT TRK 1, SECTOR 0
4923 023074 001004          BNE     68$ ;BR IF NO
4924 023076 012765 000025 000006  MOV     #21.,RKDA(R5) ;ELSE SECTOR BEFORE WRL IS TRK 0, SEC 21
4925 023104 000415          BR      70$
4926 023106 023727 001164 001000 68$:   CMP     $TMP2,#1000 ;SEE IF WRL AT TRK 2,SECTOR 0
4927 023114 001004          BNE     69$ ;BR IF NO
4928 023116 012765 000425 000006  MOV     #425,RKDA(R5) ;ELSE SECTOR BEFORE WRL IS TRK 1, SEC 21
4929 023124 000405          BR      70$

```

```

4930
4931 023126 005337 001164 69$: DEC $TMP2 ;GET SECTOR BEFORE WRL
4932 023132 013765 001164 000006 MOV $TMP2,RKDA(R5)
4933 023140 016537 000006 001166 70$: MOV RKDA(R5), $TMP3 ;FOR ERROR PRINTOUT
4934 023146 032737 000001 001162 BIT #BIT0,$TMP1
4935 023154 001004 BNE 71$ ;BR IF WRITING 1'S WHEN WLE OCCURRED
4936 023156 012765 001456 000004 MOV #DATA0,RKBA(R5) ;WRITING 0'S:WLE SECTOR -1 SHOULD HAVE ALL 0'S
4937 023164 000403 BR 72$
4938
4939 023166 012765 001462 000004 71$: MOV #DATA1,RKBA(R5) ;WRITING 1'S:WLE SECTOR -1 SHOULD HAVE ALL 1'S
4940 023174 052765 000020 000010 72$: BIS #BA1,RKCS2(R5)
4941 023202 012765 177400 000002 MOV #-256,RKWC(R5)
4942 023210 017537 000004 001436 MOV #RKBA(R5),WD2 ;EXPECTED WORD FOR TRUERR TYPEOUT
4943 023216 013765 001366 000020 MOV TOCYL,RKDC(R5)
4944
4945 023224 012737 000031 005340 MOV #<WRTCHK>,HCS1
4946 023232 004737 024504 JSR PC,DATCMD ;DO DATA X FOR CMD & GET CONTR RDY
4947 023236 104015 ERROR 15 ;NO RDY AFTER WRITE CHECK CMD
4948 023240 004737 026106 JSR PC,GSTAT ;GET FRESH STATUS
4949 023244 032737 100000 005340 BIT #CERR,HCS1
4950 023252 001450 BEQ 77$
4951 023254 032737 040000 005342 BIT #WCE,HCS2 ;SEE IF WRITE CHECK ERROR
4952 023262 001405 BEQ 76$
4953 023264 016537 000024 001434 MOV RKDB(R5),WD1 ;ACTUAL WORD FOR PRINTOUT
4954 023272 104137 ERROR 137 ;WCE AFTER WRITE CMD
4955 023274 000437 BR 77$
4956
4957 023276 104022 76$: ERROR 22 ;CERR AFTER WRITE CHECK CMD
4958
4959 023300 012737 010340 005430 MOV #<0!D.SPIN!D.DRDY!D.VV!D.DRA>,E.A0 ;EXPECTED MSG A0
4960 023306 005037 005432 CLR E.B0 ;EXPECTED MSG B0
4961 023312 012737 001720 005434 MOV #<D.SPOK!D.CART!D.DOOR!D.BRHM!D.SSP>,E.A1 ;EXPECTED A1
4962 023320 012737 000001 005436 MOV #1,E.B1 ;MSG ID FOR EXPECTED MSG B1
4963 023326 005037 005440 CLR E.A2 ;EXPECTED MSG A2
4964 023332 012737 000002 005442 MOV #2,E.B2 ;MSG ID FOR EXPECTED MSG B2
4965 023340 012737 000003 005446 MOV #3,E.B3 ;MSG ID FOR EXPECTED MSG B3
4966
4967 023346 004737 025264 JSR PC,CHKMSG ;CHECK MSGS A0, B0, A1, B1
4968 023352 000003 .WORD T.A2!T.B2!0 ;& MSGS SPECIFIED HERE
4969 023354 104057 ERROR 57 ;MSG A0 ERROR AFTER WRITE CHECK CMD
4970 023356 104031 ERROR 31 ;MSG B0 ERROR
4971 023360 104060 ERROR 60 ;MSG A1 ERROR
4972 023362 104032 ERROR 32 ;MSG B1 ERROR
4973 023364 104401 044003 TYPE MSG72 ;ABORTING BALANCE OF TESTS
4974 023370 000137 023374 JMP ENDRV
4975
4976 023374 77$:
4977
4978 023374 12$:
4979
4980 023374 ENDRV:
4981
4982 ;*****
4983 ;*TEST 20 END OF PROGRAM
4984 ;*
4985 ;* THIS IS NOT A TEST BUT A LINKAGE TO PERFORM ALL THE

```


F08

CZR6JEO UNIBUS RK6 DR PRT3
CZR6JE.P11 25-JAN-78 12:08

MACY11 30A(1052) 25-JAN-78 12:16 PAGE 96
T20 END OF PROGRAM

SEQ 0096

```

4986      *      ABOVE TESTS FOR THE NEXT DRIVE PRESENT.
4987      *      THE NEXT TEST IS ENTERED ONLY AFTER ALL DRIVES PRESENT
4988      *      HAVE BEEN TESTED.
4989      *      DO NOT LOOP ON THIS 'TEST'.
4990      *
4991      *      *****
4992 023374 000004 TST20: SCOPE
4993 023376 012737 000001 001174 MOV #1,STIMES ;DO 1 ITERATION
4994 023404 012706 001100 MOV #STACK,SP ;RESTORE STK PTR
4995
4996 023410 005237 001220 INC $DEVCT ;INCR COUNT FOR # OF DRIVES THAT ARE CHECKED
4997 023414 023737 005460 001220 CMP DRIVS,$DEVCT ;ARE ALL DRIVES PRESENT TESTED?
4998 023422 001402 BEQ TST21 ;GO TO NEXT TEST IF YES
4999 023424 000137 012402 JMP NUDRV ;IF NOT, GO BACK & TEST NEXT DRIVE PRESENT.
5000

```

```

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

```

```

5021      *      *****
5022 023430 000004 TST21: SCOPE
5023 023432 012737 000001 001174 MOV #1,STIMES ;DO 1 ITERATION
5024 023440 012706 001100 MOV #STACK,SP ;RESTORE STACK PTR
5025
5026 023444 005737 001216 TST $PASS
5027 023450 001402 BEQ 1$ ;DO TEST ONLY IN 1ST PASS
5028 023452 000137 023722 JMP 11$ ;ELSE EXIT TEST
5029
5030 023456 023727 005460 000001 1$: CMP DRIVS,#1
5031 023464 001004 BNE 2$ ;BR IF MORE THAN 1 DRIVE PRESENT
5032 023466 104401 043143 TYPE MSG62 ;BYPASS TEST, ONLY 1 DRIVE PRESENT
5033 023472 000137 023722 JMP 11$ ;ELSE EXIT TEST
5034
5035 023476 104401 037770 2$: TYPE ,MSG28 ;MULT DRV DETECTION TEST
5036 023502 104401 041765 TYPE ,MSG52 ;PRESS CONT-E TO EXIT OR SPACE TO CONTINUE
5037 023506 004737 030412 JSR PC,CCSP ;INPUT CONT-E OR SPACE
5038 023512 000137 023722 JMP 11$ ;RETURN HERE FOR CONT-E
5039
5040 023516 104401 042600 TYPE ,MSG58 ;LOAD HEADS ON ALL DRIVES
5041 023522 104401 040573 TYPE ,MSG37 ;PRESS SPACE WHEN DONE

```

G08

CZR6JED UNIBUS RK6 DR PRT3
CZR6JE.P11 25-JAN-78 12:08

MACY11 30A(1052) 25-JAN-78 12:16 PAGE 97
T21 MULTIPLE DRIVE DETECTION TEST

SEQ 0097

```

5042 023526 004737 030452          JSR    PC,GETSP          ;GET SPACE
5043
5044 023532 004737 026436          3$:   JSR    PC,SUBCLR
5045 023536 104024                    ERROR  24                ;CERR AFTER SCLR
5046
5047 023540 104401 042662          TYPE  ,MSG59            ;INSERT SAME UNIT SEL PLUG # IN 2 DRIVES
5048 023544 104401 040573          TYPE  ,MSG37            ;DEPRESS SPACE BAR WHEN DONE
5049 023550 004737 030452          JSR    PC,GETSP          ;GET SPACE
5050
5051 023554 005000                    CLR    RD                ;DRIVE # COUNTER
5052
5053 023556 012765 100000 000000 6$:   MOV    #CCLR,RKCS1(R5)
5054 023564 010065 007010                    MO.   RD,RKCS2(R5)      ;DRIVE #
5055 023570 012737 000001 005340    MOV    #SELDRV,HCS1
5056 023576 053737 001170 005340    BIS    $TMP4,HCS1      ;ADD CDT IF RKO?
5057 023604 013765 005340 000000    MOV    HCS1,RKCS1(R5)  ;GET STATUS
5058 023612 013737 001432 005376    MOV    T5000,TEMP1
5059 023620 004737 025232          JSR    PC,DLY           ;DO DELAY TO CATCH MDS
5060
5061 023624 032765 001000 000010    BIT    #MDS,RKCS2(R5)  ;SEE IF THAT DRIVE HAS MDS
5062 023632 001006                    RNE   7$                ;BR IF YES
5063 023634 005200                    INC   RD                ;ELSE TRY ANOTHER DRIVE
5064 023636 020027 000010    CMP    RD,#8           ;SEE IF ALL DRIVES TESTED
5065 023642 001345                    BNE   6$                ;BR IF NO
5066 023644 104176                    ERROR  176              ;CANNOT FIND MDS
5067 023646 000411                    BR    10$               ;TRY AGAIN
5068
5069 023650 104401 043075          7$:   TYPE  ,MSG61
5070 023654 010046                    MOV    RD,-(SP)        ;MULT DRIVES FOUND ON DRIVE #
5071                                     ;SAVE RD FOR TYPEOUT
5072                                     ;TYPE UNIT NO
5073 023656 104403                    TYPOS 1                 ;GO TYPE--OCTAL ASCII
5074 023660 001                                     .BYTE 1                 ;TYPE 1 DIGIT(S)
5075 023661 000                                     .BYTE 0                 ;SUPPRESS LEADING ZEROS
5076 023662 104401 043555          TYPE  ,MSG68            ;VERIFY BOTH DRIVES UNLOADED
5077 023666 005237 005322          INC   UNLD              ;FOR VALID HALT
5078 023672 104401 042765          10$:  TYPE  ,MSG60
5079 023676 104401 043237          TYPE  ,MSG63
5080 023702 004737 030412          JSR    PC,CCSP          ;INSERT CORRECT UNIT SEL PLUG & LOAD HEADS
5081 023706 000137 023722          JMP    11$              ;DEPRESS CONT-E OR SPACE BAR WHEN DONE
5082 023712 005037 005322          CLR   UNLD              ;INPUT CONT-E OR SPACE
5083 023716 000137 023532          JMP    3$               ;RETURN HERE FOR CONTROL-E (EXIT)
5084 023722 005037 005322          11$:  CLR   UNLD              ;RETURN HERE FOR SPACE (NO AGAIN)
5085 023726 004737 026436          JSR    PC,SUBCLR
5086 023732 104024                    ERROR  24                ;CERR AFTER SCLR

```

```
5087  
5088  
5089  
5090  
5091  
5092  
5093  
5094  
5095 023734  
5096 023734 000004  
5097 023736 005037 001102  
5098 023742 005037 001174  
5099 023746 005237 001216  
5100 023752 042737 100000 001216  
5101 023760 005327  
5102 023762 000001  
5103 023764 003022  
5104 023766 012737  
5105 023770 000001  
5106 023772 023762  
5107 023774 104401 024041  
5108 024000 013746 001216  
5109 024004 104405  
5110 024006 104401 024036  
5111 024012 013700 000042  
5112 024016 001405  
5113 024020 000005  
5114 024022 004710  
5115 024024 000240  
5116 024026 000240  
5117 024030 000240  
5118 024032  
5119 024032 000137  
5120 024034 024056  
5121 024036 377 377 060  
5122 024041 015 042412 042116  
5123 024046 050040 051501 020123  
5124 024054 000043  
5125 024056 122737 000001 001230  
5126 024064 001007  
5127 024066 022737 000002 001216  
5128 024074 101003  
5129 024076 005237 001102  
5130 024102 000775  
5131 024104 000137 010612
```

```
.SBTTL END OF PASS ROUTINE  
: *****  
: *INCREMENT THE PASS NUMBER ($PASS)  
: *TYPE "END PASS #XXXXX" (WHERE XXXXX IS A DECIMAL NUMBER)  
: *IF THERES A MONITOR GO TO IT  
: *IF THERE ISN'T JUMP TO STSXY  
$EOP: SCOPE  
CLR $STNM ; ZERO THE TEST NUMBER  
CLR $TIMES ; ZERO THE NUMBER OF ITERATIONS  
INC $PASS ; INCREMENT THE PASS NUMBER  
BIC #100000,$PASS ; DON'T ALLOW A NEG. NUMBER  
DEC (PC)+ ; LOOP?  
$EOPCT: .WORD 1  
BGT $DOAGN ; YES  
MOV (PC)+,(PC)+ ; RESTORE COUNTER  
$ENDCT: .WORD 1  
$EOPCT  
TYPE $SENDMG ; TYPE "END PASS #"  
MOV $PASS,-(SP) ; SAVE $PASS FOR TYPEOUT  
TYPDS ; GO TYPE--DECIMAL ASCII WITH SIGN  
TYPE $ENULL ; TYPE A NULL CHARACTER  
$GET42: MOV #42,RO ; GET MONITOR ADDRESS  
BEQ $DOAGN ; BRANCH IF NO MONITOR  
$ENDAD: RESET ; CLEAR THE WORLD  
JSR PC,(RO) ; GO TO MONITOR  
NOP ; SAVE ROOM  
NOP ; FOR  
NOP ; ACT11  
$DOAGN: JMP @ (PC)+ ; RETURN  
$RTNAD: .WORD STSXY  
$ENULL: .BYTE -1,-1,0 ; NULL CHARACTER STRING  
$SENDMG: .ASCIZ <15><12>/END PASS #/  
STSXY: CMPB #APTENV,$ENV ; RUN UNDER APT ?  
BNE 2$ ; BRANCH IF NOT  
CMP #2,$PASS ; TWO PASSES DONE ?  
BHI 2$ ; BRANCH IF NOT  
1$: INC $STNM ; CHANGE THE TEST NUMBER  
BR 1$ ; LOOPING  
2$: JMP STS ; EXIT
```

S132
S133
S134
S135
S136
S137
S138
S139
S140
S141
S142
S143
S144
S145
S146
S147
S148
S149
S150
S151
S152
S153
S154
S155
S156
S157
S158
S159
S160
S161
S162
S163
S164
S165
S166
S167
S168
S169
S170
S171
S172
S173
S174
S175
S176
S177
S178
S179
S180
S181
S182
S183
S184
S185
S186
S187

024110 012700 005450
024114 012701 177757
024120 005020
024122 005201
024124 001375
024126 000207

024130 005737 001360
024134 001024
024136 005237 001360
024142 104401 035652

024146 005737 000042
024152 001012
024154 123727 001230 000001
024162 001406
024164 023727 001140 000176
024172 001005
024174 104406
024176 000403
024200 112737 000001 001134
024206
024206 000207

024210 104411
024212 012600
024214 012701 177770
024220 112002
024222 042702 177400
024226 012703 005462
024232 012704 000060

024236 020402
024240 001415
024242 005723
024244 005204
024246 020427 000070
024252 001371
024254 005702
024256 001022
024260 020127 177770
024264 001426

```
.SBTTL SUBROUTINES
;SUBROUTINE TO CLEAR ALL FLAGS FROM DDUMP THRU DOTIM
;
CLRFLG: MOV     #DDUMP, R0
        MOV     #-17., R1
1$:     CLR     (R0)+
        INC     R1
        BNE    1$
        PTS    PC

;
;TYPE PROGRAM ID IF FTITLE=0
;
TITLE:  TST     FTITLE
        BNE    1$
        INC     FTITLE
        TYPE    MSG1
.SBTTL  GET VALUE FOR SOFTWARE SWITCH REGISTER
        TST    #42
        BNE    64$
        CMPB   $ENV, #1
        BEQ    64$
        CMP    SWR, #SWREG
        BNE    65$
        GTSWR
        BR     65$
64$:    'OVB    #1, $AUTOB
        BNE    65$
1$:     RTS     PC

;
;ROUTINE TO INPUT DRIVE NOS. TYPED IN & SET
;DRIVS, DRIVO-DRIV7 REGISTERS APPROPRIATELY
;
GDRVS:  RDLIN
        MOV     (SP)+, R0
        MOV     #-8., R1
1$:     MOVB   (R0)+, R2
        BIC    #177400, R2
        MOV     #DRIVO, R3
        MOV     #60, R4
        CMP    R4, R2
        BEQ    3$
        TST   (R3)+
        INC    R4
        CMP    R4, #70
        BNE    2$
        TST   R2
        BNE    4$
        CMP    R1, #-8.
        BEQ    5$
;GET STARTING ADDR OF ASCII STRING
;SET UP COUNT
;GET ASCII CHAR
;MASK HI BYTE
;DRIVE FLAG ADDR
;WAS TYPED CHAR 0 THRU 7?
;BRANCH IF YES
;NO, INCREMENT DR FLAG ADDR
;S/B 0-7 OR TERMINATOR
;DEFAULT ALL DRIVES
```

JOB

CZR6JED UNIBUS RK6 DR PRT3
CZR6JE.P11 25-JAN-78 12:08

MACY11 30A(1052) 25-JAN-78 12:16 PAGE 100
GET VALUE FOR SOFTWARE SWITCH REGISTER

SEQ 0100

```

5188 024266 005037 005510      7$: CLR      SIZFLG      ;BYPASS TEST 1 (SIZING)
5189 024272 000207              RTS      PC              ;FOUND TERMINATOR, EXIT
5190
5191 024274 005213              3$: INC      DR3          ;SET UP FLAG FOR THE DRIVE
5192 024276 005237 005460      INC      DRVS          ;INCREMENT TOTAL # DRIVES TO BE TESTED
5193 024302 112002              MOV      (P0)+,R2      ;GET NEXT ASCII CHAR.
5194 024304 042702 177400      BIC      #177400,R2    ;MASK
5195 024310 022702 000054      CMP      #54,R2       ;IS IT A COMMA?
5196 024314 001407              BEQ      5$            ;YES, GO TO NEXT WORD.
5197 024316 005702              TST      R2            ;NO, IS IT A TERMINATOR?
5198 024320 001001              BNE      4$            ;IF NOT, SOMETHING WRONG.
5199 024322 000761              BR       1$            ;FOUND TERMINATOR, EXIT
5200
5201 024324 104401 044350      4$: TYPE     EMI          ;ONLY 0-7 ALLOWED.
5202 024330 000137 010016      JMP      PRGSRT        ;START ALL OVER
5203
5204 024334 005201              5$: INC      R1          ;S/B NO MORE THAN 8 DIFF
5205 024336 001330              BNE      1$            ;DRIVES TYPED IN.
5206 024340 000771              BR       4$            ;IF MORE, HAVE ERROR.
5207
5208 024342 005237 005510      6$: INC      SIZFLG      ;DO TEST 1 (SIZING)
5209 024346 000207              RTS      PC              ;EXIT.
5210
5211
5212      ;ROUTINE TO INPUT RKBAS OR DEFAULT.
5213
5214
5215 024350 104412      GBA: RDOCT
5216 024352 012600      MOV      (SP)+,RO      ;GET LOW ORDER FROM STACK
5217 024354 005700      TST      RO
5218 024356 001403      BEQ      1$            ;BRANCH IF DEFAULT.
5219 024360 010037 001264      MOV      RO,$BASE
5220 024364 000207      RTS      PC
5221 024366 012737 177440 001264 1$: MOV      #177440,$BASE ;DEFAULT VALUE
5222 024374 000207      RTS      PC
5223
5224
5225      ;ROUTINE TO INPUT RKVEC OR DEFAULT
5226
5227
5228 024376 104412      GINT: RDOCT
5229 024400 012600      MOV      (SP)+,RO      ;GET LOW ORDER FROM STACK
5230 024402 005700      TST      RO
5231 024404 001405      BEQ      1$            ;BRANCH IF DEFAULT
5232 024406 010037 001334      MOV      RO,RKVEC
5233 024412 004737 024430      2$: JSR      PC,SETINT
5234 024416 000207      RTS      PC
5235 024420 012737 000210 001334 1$: MOV      #210,RKVEC ;DEFAULT VALUE
5236 024426 000771      BR       2$
5237
5238
5239      ;ROUTINE TO SETUP INTERRUPT VECTOR & PRIORITY
5240
5241
5242 024430 013700      SETINT: MOV      RKVEC,RO
5243 024434 012720      MOV      #INTER,(RO)+ ;INTER ADDR TO RKVEC

```

K08

CZR6JEO UNIBUS RK6 DR PRT3
CZR6JE.P11 25-JAN-78 12:08

MACY11 30A(1052) 25-JAN-78 12:16 PAGE 101
GET VALUE FOR SOFTWARE SWITCH REGISTER

SEQ 0101

```

5244 024440 013710 001336      MOV      RKPRI,(R0)      ;PRS TO RKVEC+2
5245 024444 000207      RTS      PC
5246
5247
5248
5249      ; THIS ROUTINE SETS CDT IN RKCS1 IF DRIVE UNDER TEST IS AN RK07.
5250      ; ENTER WITH COMMAND IN HCS1
5251
5252 024446 053737 001170 005340 DOCMD:  BIS      $TMP4,HCS1      ;ADD CDT IF RK07
5253 024454 013765 005340 000000      MOV      HCS1,RKCS1(R5)    ;DO COMMAND
5254 024462 013737 001424 005376      MOV      T10,TEMP1
5255 024470 004737 024542      JSR      PC,FRDY          ;FIND CONTR READY
5256 024474 000207      RTS      PC              ;SET HERE IF NOT RDY
5257 024476 062716 000002      ADD      #2,(SP)         ;ELSE SKIP OVER ERROR
5258 024502 000207      RTS      PC
5259
5260      ; THIS ROUTINE IS SIMILAR TO THE ABOVE BUT IS USED FOR DATA TRANSFERS
5261      ; & REQUIRES A LONGER TIMEOUT
5262
5263 024504 053737 001170 005340 DATCMD: BIS      $TMP4,HCS1      ;ADD CDT IF RK07
5264 024512 013765 005340 000000      MOV      HCS1,RKCS1(R5)    ;DO CMD
5265 024520 013737 001432 005376      MOV      T5000,TEMP1
5266 024526 004737 024542      JSR      PC,FRDY          ;FIND CONTR RDY
5267 024532 000207      RTS      PC
5268 024534 062716 000002      ADD      #2,(SP)
5269 024540 000207      RTS      PC
5270
5271
5272      ; ROUTINE TO FIND CONTROLLER READY (RDY) DURING A DELAY
5273      ; ENTER WITH A COUNT IN TEMP1
5274      ; RETURN IF RDY NOT PRESENT (ERROR CONDITION)
5275      ; RETURN +2 IF RDY PRESENT (SKIP OVER ERROR)
5276      ; STATUS IS OBTAINED BEFORE THE RETURN FOR EITHER CASE
5277
5278 024542 032765 000200 000000 FRDY:  BIT      #RDY,RKCS1(R5)
5279 024550 001010      BNE     1$
5280 024552 005337 005376      DEC     TEMP1
5281 024556 001371      BNE     FRDY
5282 024560 004737 024676      JSR     PC,HOLD          ;STORE ALL RK611 REGS IN HOLDING REGS.
5283 024564 004737 026024      JSR     PC,CKCERR        ;CHECK FOR SPECIAL CERR
5284 024570 000207      RTS     PC              ;NO RDY, EXIT
5285 024572 062716 000002 1$:  ADD     #2,(SP)         ;SKIP OVER ERROR
5286 024576 004737 024676      JSR     PC,HOLD
5287 024602 004737 026024      JSR     PC,CKCERR        ;CHECK FOR SPECIAL CERR
5288 024606 000207      RTS     PC
5289
5290      ; ROUTINE TO FIND CONTROLLER READY AND STORE DRIVE REGS ONLY
5291
5292 024610 032765 000200 000000 FRDY1: BIT      #RDY,RKCS1(R5)
5293 024616 001014      BNE     1$
5294 024620 005337 005376      DEC     TEMP1
5295 024624 001371      BNE     FRDY1
5296 024626 016537 000034 005366      MOV     RKMR2(R5),HMR2
5297 024634 016537 000036 005370      MOV     RKMR3(R5),HMR3
5298 024642 004737 026024      JSR     PC,CKCERR        ;CHECK FOR SPECIAL CERR CONDITIONS
5299 024646 000207      RTS     PC              ;NO RDY, EXIT

```

```

5300 024650 062716 000002      1S:  ADD      #2,(SP)          ;SKIP OVER ERROR
5301 024654 016537 000034      MOV      RKMR2(R5),HMR2
5302 024662 016537 000036      MOV      RKMR3(R5),HMR3
5303 024670 004737 026024      JSR      PC,CKCERR          ;CHECK FOR SPECIAL CERR CONDITIONS
5304 024674 000207      RTS      PC
5305
5306
5307      ;STORE ALL RK611 REGISTERS IN HOLDING REGS
5308
5309
5310 024676 016537 000000      HOLD:  MOV      RKCS1(R5),HCS1
5311 024704 016537 000010      MOV      RKCS2(R5),HCS2
5312 024712 016537 000002      MOV      RKWC(R5),HWC
5313 024720 016537 000004      MOV      RKBA(R5),HBA
5314 024726 016537 000006      MOV      RKDA(R5),HDA
5315 024734 016537 000012      MOV      RKDS(R5),HDS
5316 024742 016537 000014      MOV      RKER(R5),HER
5317 024750 016537 000016      MOV      RKASOF(R5),HASOF
5318 024756 016537 000020      MOV      RKDC(R5),HDC
5319 024764 016537 000026      MOV      RKMR1(R5),HMR1
5320 024772 016537 000034      MOV      RKMR2(R5),HMR2
5321 025000 016537 000036      MOV      RKMR3(R5),HMR3
5322 025006 016537 000030      MOV      RKECPS(R5),HPOS
5323 025014 016537 000032      MOV      RKECPT(R5),HPAT
5324 025022 000207      RTS      PC
5325
5326
5327      ;ROUTINE TO CHECK FOR CORRECT ATTN
5328      ;RETURN IF ATTN NOT PRESENT (ERROR CONDITION)
5329      ;RETURN +2 IF ATTN PRESENT (SKIP OVER ERROR)
5330
5331 025024 010446      †STATN: MOV      R4,-(SP)          ;SAV R4
5332 025026 013704      MOV      $UNIT,R4
5333 025032 136437 001222      BITB     ATTN(R4),HASOF+1
5334 025040 001404      BEQ      1S                ;BRANCH IF ATTN NOT PRESENT
5335 025042 012604      MOV      (SP)+,R4          ;RESTOR R4
5336 025044 062716 000002      ADD      #2,(SP)          ;INCR RET ADDR TO JUMP OVER ERROR.
5337 025050 000207      RTS      PC
5338 025052 012604      1S:     MOV      (SP)+,R4          ;RESTOR R4
5339 025054 000207      RTS      PC
5340
5341
5342
5343      ;ROUTINE TO FIND ATTN WITHIN TIMES GREATER THAN 1 SEC
5344      ;ENTER WITH TIME IN SECONDS IN TEMP2
5345      ;RETURN IF NO ATTN (ERROR CONDITION)
5346      ;RETURN +2 IF ATTN FOUND
5347      ;STATUS IS OBTAINED BEFORE THE RETURN FOR EITHER CASE
5348
5349
5350 025056 010446      FATT1:  MOV      R4,-(SP)          ;SAV R4
5351 025060 012737 177777      3S:     MOV      #-1,TEMP1
5352 025066 013704      MOV      $UNIT,R4
5353 025072 136465 005330      1S:     BITB     ATTN(R4),RKASOF+1(R5) ;FIND CORRECT ATTN
5354 025100 001014      BNE     2S
5355 025102 005337 005376      DEC     TEMP1

```

M08

CZR6JED UNIBUS RK6 DR PRT3
CZR6JE.P11 25-JAN-78 12:08

MACY11 30A(1052) 25-JAN-78 12:16 PAGE 103
GET VALUE FOR SOFTWARE SWITCH REGISTER

SEQ 0103

```

5356 025106 001371          BNE 1$
5357 025110 005337 005400  DEC  TEMP2
5358 025114 001361          BNE 3$
5359 025116 005065 000026  CLR  RKMR1(R5)      ;SELECT WORD 0
5360 025122 004737 026106  JSR  PC,GSTAT      ;GET LATEST STATUS
5361 025126 012604          MOV  (SP)+,R4      ;RESTOR R4
5362 025130 000207          RTS  PC
5363
5364 025132 005065 000026 2$: CLR  RKMR1(R5)
5365 025136 004737 026106  JSR  PC,GSTAT      ;GET STATUS AFTER ATTN SEEN
5366 025142 012604          MOV  (SP)+,R4      ;RESTOR R4
5367 025144 062716 000002  ADD  #2,(SP)       ;SKIP OVER ERROR
5368 025150 000207          RTS  PC
5369
5370
5371
5372
5373
5374
5375
5376
5377
5378 025152 010446          FATT2: MOV  R4, -(SP)      ;SAV R4
5379 025154 013704 001222 2$: MOV  $UNIT,R4
5380 025160 136465 005330 000017 BITB  ATTN(R4),RKASOF+1(R5) ;FIND CORRECT ATTN
5381 025166 001011          BNE 1$
5382 025170 005337 005376  DEC  TEMP1
5383 025174 001367          BNE 2$
5384 025176 005065 000026  CLR  RKMR1(R5)      ;SELECT WORD 0
5385 025202 004737 026106  JSR  PC,GSTAT      ;GET LATEST STATUS.
5386 025206 012604          MOV  (SP)+,R4      ;RESTOR R4
5387 025210 000207          RTS  PC
5388 025212 005065 000026 1$: CLR  RKMR1(R5)
5389 025216 004737 026106  JSR  PC,GSTAT
5390 025222 012604          MOV  (SP)+,R4      ;RESTOR R4
5391 025224 062716 000002  ADD  #2,(SP)       ;SKIP OVER ERROR
5392 025230 000207          RTS  PC
5393
5394
5395
5396
5397
5398 025232 005737 005376  DLY: TST  TEMP1      ;5.6 US
5399 025236 001403          BEQ  1$            ;2.5 US
5400 025240 005337 005376  DEC  TEMP1          ;6.8 US
5401 025244 000772          BR   DLY           ;2.5 US
5402 025246 000207          1$: RTS  PC        ;3.8 US
5403
5404
5405
5406
5407 025250 104401 037335  BYP: TYPE  MSG14      ;BYPASS DRIVE
5408 025254 010046          MOV  RD, -(SP)     ;SAVE RD FOR TYPEOUT
5409
5410 025256 104403          TYPOS
5411 025260 001          .BYTE 1          ;TYPE DR#
                    ;GO TYPE--OCTAL ASCII
                    ;TYPE 1 DIGIT(S)

```


N08

CZR6JED UNIBUS RK6 DR PRT3
CZR6JE.P11 25-JAN-78 12:08

MACY11 30A(1052) 25-JAN-78 12:16 PAGE 104
GET VALUE FOR SOFTWARE SWITCH REGISTER

SEQ 0104

					.BYTE	0		
					RTS	PC		::SUPPRESS LEADING ZEROS
5412	025261	000						
5413	025262	000207						
5414								
5415								
5416								
5417	025264	017637	000000	001504	CHKMSG: MOV	2(SP),CHKFLG		;PASS MSGS TO BE TESTED
5418	025272	062716	000002		ADD	#2,(SP)		;BUMP RETURN ADDR TO 1ST ERROR
5419	025276	004737	026142		JSR	PC,GSTAT1		;GET ALL ACTUAL DRIVE & CONTR STATUS
5420								
5421	025302	053737	001222	005430	BIS	\$UNIT,E.A0		;SET UNIT #
5422	025310	053737	001222	005434	BIS	\$UNIT,E.A1		
5423	025316	053737	001222	005440	BIS	\$UNIT,E.A2		
5424	025324	053737	001222	005444	BIS	\$UNIT,E.A3		
5425								
5426	025332	053737	012662	005430	BIS	E.DDT,E.A0		;ADD DRIVE TYPE
5427								
5428	025340	013746	005376		MOV	TEMP1,-(SP)		;SAVE TEMP1
5429								
5430	025344	013737	005430	005376	MOV	E.A0,TEMP1		
5431	025352	004737	030306		JSR	PC,SBPARG		;GET PARITY FOR MSG A0
5432	025356	013737	005376	005430	MOV	TEMP1,E.A0		
5433								
5434	025364	013737	005434	005376	MOV	E.A1,TEMP1		
5435	025372	004737	030306		JSR	PC,SBPARG		;GET PARITY FOR MSG A1
5436	025376	013737	005376	005434	MOV	TEMP1,E.A1		
5437								
5438	025404	013737	005440	005376	MOV	E.A2,TEMP1		
5439	025412	004737	030306		JSR	PC,SBPARG		;GET PARITY FOR MSG A2
5440	025416	013737	005376	005440	MOV	TEMP1,E.A2		
5441								
5442	025424	013737	005432	005376	MOV	E.B0,TEMP1		
5443	025432	004737	030306		JSR	PC,SBPARG		;GET PARITY FOR MSG B0
5444	025436	013737	005376	005432	MOV	TEMP1,E.B0		
5445								
5446	025444	013737	005436	005376	MOV	E.B1,TEMP1		
5447	025452	004737	030306		JSR	PC,SBPARG		;GET PARITY FOR MSG B1
5448	025456	013737	005376	005436	MOV	TEMP1,E.B1		
5449								
5450	025464	013737	005442	005376	MOV	E.B2,TEMP1		
5451	025472	004737	030306		JSR	PC,SBPARG		;GET PARITY FOR MSG B2
5452	025476	013737	005376	005442	MOV	TEMP1,E.B2		
5453								
5454	025504	013737	005446	005376	MOV	E.B3,TEMP1		
5455	025512	004737	030306		JSR	PC,SBPARG		;GET PARITY FOR MSG B3
5456	025516	013737	005376	005446	MOV	TEMP1,E.B3		
5457								
5458	025524	012637	005376		MOV	(SP)+,TEMP1		;RESTORE TEMP1
5459	025530	013737	001176	001172	MOV	\$ESCAPE,\$TMP5		;SAVE ESCAPE
5460								
5461	025536	023737	005410	005430	CMP	H.A0,E.A0		;TEST MSG A0
5462	025544	001411			BEQ	2\$;BR IF OK
5463	025546	012737	025560	001176	MOV	#1,\$ESCAPE		;ELSE SETUP ESCAPE
5464	025554	011646			MOV	(SP),-(SP)		;COPY RET ADDR.
5465	025556	000207			RTS	PC		; & RETURN TO MAINLINE ERROR
5466								
5467	025560	032777	001000	153352	1\$: BIT	#SW9,\$SWR		;RET HERE FROM MAINLINE ERROR

```

5468 025566 001107          BNE      20$          ;& BR IF LOOP ON ERROR
5469 025570 062716 000002 2$:      ADD      #2,(SP) ;BUMP RET ADDR TO NEXT ERROR
5470
5471 025574 023737 005412 005432      CMP      H.B0,E.B0    ;TEST MSG B0
5472 025602 001411          BEQ      5$          ;BR IF OK
5473 025604 012737 025616 001176      MOV      #4$, $ESCAPE ;ELSE SETUP ESCAPE
5474 025612 011646          MOV      (SP),-(SP)  ;COPY RET ADDR
5475 025614 000207          RTS      PC          ;& RETURN TO MAINLINE ERROR
5476
5477 025616 032777 001000 153314 4$:      BIT      #SW9, $SWR   ;RETURN HERE FROM MAINLINE ERROR
5478 025624 001070          BNE      20$          ;& BR IF LOOP ON ERROR
5479 025626 062716 000002 5$:      ADD      #2,(SP) ;BUMP RET ADDR TO NEXT ERROR
5480
5481 025632 023737 005414 005434      CMP      H.A1,E.A1    ;TEST MSG A1
5482 025640 001411          BEQ      8$          ;BR IF OK
5483 025642 012737 025654 001176      MOV      #7$, $ESCAPE
5484 025650 011646          MOV      (SP),-(SP)
5485 025652 000207          RTS      PC
5486
5487 025654 032777 001000 153256 7$:      BIT      #SW9, $SWR
5488 025662 001051          BNE      20$
5489 025664 062716 000002 8$:      ADD      #2,(SP)
5490
5491 025670 023737 005416 005436      CMP      H.B1,E.B1    ;TEST MSG B1
5492 025676 001411          BEQ      11$         ;BR IF OK
5493 025700 012737 025712 001176      MOV      #10$, $ESCAPE
5494 025706 011646          MOV      (SP),-(SP)
5495 025710 000207          RTS      PC
5496
5497 025712 032777 001000 153220 10$:     BIT      #SW9, $SWR
5498 025720 001032          BNE      20$
5499 025722 062716 000002 11$:     ADD      #2,(SP)
5500
5501 025726 032737 000001 001504 12$:     BIT      #T.A2,CHKFLG ;TEST MSG A2?
5502 025734 001402          BEQ      13$         ;BR IF NO
5503 025736 004737 026722          JSR      PC,RCYLD    ;PUT INFO CYLDIF, DO NOT CHECK
5504 025742 032737 000002 001504 13$:     BIT      #T.B2,CHKFLG ;TEST MSG B2?
5505 025750 001402          BEQ      14$         ;BR IF NO
5506 025752 004737 026774          JSR      PC,RCYLA    ;PUT INFO IN CYLADD, DO NOT CHECK
5507
5508 025756 032737 000004 001504 14$:     BIT      #T.B3,CHKFLG ;TEST MSG B3?
5509 025764 001404          BEQ      15$
5510 025766 004737 027032          JSR      PC,RSEC     ;PUT INFO IN SECTOR, DO NOT CHECK
5511 025772 004737 027070          JSR      PC,RHEAD    ;PUT INFO IN HEADA, DO NOT CHECK
5512
5513 025776 013737 001172 001176 15$:     MOV      $TMP5,$ESCAPE ;RESTORE ESCAPE
5514 026004 000207          RTS      PC
5515
5516 026006 012706 001100          MOV      #STACK,SP   ;RESET STACK PTR
5517 026012 013737 001172 001176 20$:     MOV      $TMP5,$ESCAPE ;RESTORE ESCAPE
5518 026020 000177 153064          JMP      $SLPERR
5519
5520 ; THIS ROUTINE CHECKS FOR CERTAIN ERROR CONDITIONS ONLY
5521 ; I.E.: IF NED, CTO OR MDS SET MESSAGE A & B ARE INVALID
5522
5523 026024 005737 001502      CKCERR: TST      BYPCERR

```

```

5524 026030 001025          BNE      4$
5525 026032 032737 100000 005340  BIT      #CERR,HCS1
5526 026040 001001          BNE      1$ ;BR IF CERR
5527 026042 000207          RTS      PC
5528
5529 026044 032737 004000 005340 1$:  BIT      #CTO,HCS1
5530 026052 001402          BEQ      2$ ;BR IF NOT CTO
5531 026054 104211          ERROR    211 ;CTO ERROR, MSG A & B INVALID
5532 026056 000207          RTS      PC
5533
5534 026060 032737 010000 005342 2$:  BIT      #NED,HCS2
5535 026066 001401          BEQ      3$ ;BR IF NOT NED
5536 026070 104212          ERROR    212 ;NED ERROR, MSG A & B INVALID
5537
5538 026072 032737 001000 005342 3$:  BIT      #MDS,HCS2
5539 026100 001401          BEQ      4$
5540 026102 104213          ERROR    213 ;MDS ERROR, MSG A & B INVALID
5541
5542 026104 000207          4$:  RTS      PC
5543
5544
5545 ; THIS ROUTINE DOES THE SELECT DRIVE COMMAND TO GET STATUS
5546 ; IT THEN WAITS FOR CONTROLLER READY
5547
5548 ; IF RDY NOT RECEIVED BY THE TIMEOUT, AN ERROR IS FLAGGED
5549 ;
5550
5551 026106 013746 005376          GSTAT:  MOV     TEMP1,-(SP) ;SAVE TEMP1
5552 026112 013765 001222 000010  MOV     $UNIT,RKCS2(R5) ;CURRENT DRIVE #
5553 026120 012737 000001 005340  MOV     #SELDRV,HCS1
5554 026126 004737 024446          JSR     PC,DOCMD ;DO SELDRV (STATUS) CMD & GET CONTR RDY
5555 026132 104117          ERROR    117 ;RDY NOT SET BY END OF SELECT DRIVE CMD
5556 026134 012637 005376          MOV     (SP)+,TEMP1 ;RESTOR TEMP1.
5557 026140 000207          RTS      PC
5558
5559 ; THIS ROUTINE GETS STATUS OF ALL DRIVE REGISTERS (MSG A0-A3, B0-B3)
5560 ; & ALL CONTROLLER REGISTERS.
5561
5562 026142 013746 005376          GSTAT1: MOV     TEMP1,-(SP) ;SAVE TEMP1
5563 026146 004737 024676          JSR     PC,HOLD ;GET ALL CONTR REG
5564 026152 012765 100000 000000  MOV     #CCLR,RKCS1(R5) ;CLEAR CONTR
5565 026160 013765 001222 000010  MOV     $UNIT,RKCS2(R5) ;CURRENT DRIVE #
5566 026166 012765 000003 000026  MOV     #3,RKMR1(R5) ;SELECT WORD 3
5567 026174 004737 026372          JSR     PC,GSTAT2
5568 026200 104117          ERROR    117 ;RDY NOT SET BY END OF SELECT DRV CMD
5569 026202 013737 005366 005424  MOV     HMR2,H.A3 ;STORE MSG A3
5570 026210 013737 005370 005426  MOV     HMR3,H.B3 ;STORE MSG B3
5571
5572 026216 012765 100000 000000  MOV     #CCLR,RKCS1(R5)
5573 026224 013765 001222 000010  MOV     $UNIT,RKCS2(R5)
5574 026232 012765 000002 000026  MOV     #2,RKMR1(R5) ;SELECT WORD 2
5575 026240 004737 026372          JSR     PC,GSTAT2
5576 026244 104117          ERROR    117 ;RDY NOT SET BY END OF SELECT DRV CMD
5577 026246 013737 005366 005420  MOV     HMR2,H.A2 ;STORE MSG A2
5578 026254 013737 005370 005422  MOV     HMR3,H.B2 ;STORE MSG B2
5579

```

```

5580 026262 012765 100000 000000 MOV #CCLR,RKCS1(R5)
5581 026270 013765 001222 000010 MOV $UNIT,RKCS2(R5)
5582 026276 012765 000001 000026 MOV #1,RKMR1(R5) ;SELECT WORD 1
5583 026304 004737 026372 JSR PC,GSTAT2
5584 026310 104117 ERROR 117 ;RDY NOT SET BY END OF SELECT DRV CMD
5585 026312 013737 005366 005414 MOV HMR2,H.A1 ;STORE MSG A1
5586 026320 013737 005370 005416 MOV HMR3,H.B1 ;STORE MSG B1
5587
5588 026326 012765 100000 000000 MOV #CCLR,RKCS1(R5)
5589 026334 013765 001222 000010 MOV $UNIT,RKCS2(R5)
5590 026342 004737 026372 JSR PC,GSTAT2
5591 026346 104117 ERROR 117 ;RDY NOT SET BY END OF SEL DRV CMD
5592 026350 013737 005366 005410 MOV HMR2,H.A0 ;STORE MSG A0
5593 026356 013737 005370 005412 MOV HMR3,H.B0 ;STORE MSG B0
5594
5595 026364 012637 005376 MOV (SP)+,TEMP1 ;RESTORE TEMP1
5596 026370 000207 RTS PC
5597
5598
5599 026372 012737 000001 005340 GSTAT2: MO. #SELDIV,HCS1
5600 026400 053737 001170 005340 BIS $TMP4,HCS1 ;ADD CDT IF RK07
5601 026406 013765 005340 000000 MOV HCS1,RKCS1(R5) ;GET STATUS
5602 026414 013737 001424 005376 MOV T10,TEMP1
5603 026422 004737 024610 JSR PC,FRDY1 ;FIND CONTR RDY & STORE DRIVE REGS ONLY
5604 026426 000207 RTS PC ;RET HERE IF NOT RDY
5605 026430 062716 000002 ADD #2,(SP) ;RET HERE IF OK
5606 026434 000207 RTS PC
5607
5608 ; THIS ROUTINE DOES A SUBSYSTEM CLEAR & WAITS FOR CONTROLLER READY
5609 ; IF RDY IS NOT RECEIVED BY THE END OF THE TIMEOUT, AN ERROR IS FLAGGED.
5610 ; THE ROUTINE THEN GETS CURRENT STATUS & CHECKS FOR CONTROLLER ERROR (CERR)
5611 ; RETURN IF CERR SET
5612 ; RETURN +2 IF CERR CLEAR
5613
5614 026436 012765 000040 000010 SUBCLR: MOV #SCLR,RKCS2(R5) ;SUBSYS CLEAR
5615 026444 013737 001424 005376 MOV T10,TEMP1
5616 026452 004737 024542 JSR PC,FRDY ;FIND RDY
5617 026456 104120 ERROR 120 ;RDY NOT SET BY END OF SCLR
5618 026460 013765 001222 000010 MOV $UNIT,RKCS2(R5) ;CURRENT DRIVE #
5619 026466 005065 000026 CLR RKMR1(R5) ;SELECT WORD 0
5620 026472 004737 026106 JSR PC,GSTAT ;GET STATUS
5621 026476 032737 100000 005340 BIT #CERR,HCS1 ;CHECK FOR CONT ERROR
5622 026504 001401 BEQ 1$
5623 026506 000207 RTS PC
5624 026510 062716 000002 1$: ADD #2,(SP) ;SKIP OVER ERROR
5625 026514 000207 RTS PC
5626
5627
5628 ; READ THE SECTOR COUNT IN RKMR3, RIGHT JUSTIFY IT & STORE IT IN 'SECTOR'
5629
5630 026516 012765 000003 000026 RDSEC: MOV #3,RKMR1(R5) ;WORD 3
5631 026524 004737 026106 JSR PC,GSTAT
5632 026530 013737 005370 001422 MOV HMR3,SECTOR
5633 026536 042737 177017 BIC #C<M.SECT>,SECTOR
5634 026544 006237 001422 ASR SECTOR ;RIGHT JUSTIFY
5635 026550 006237 001422 JSR SECTOR ;SECTOR

```

```

5636 026554 006237 001422      ASR      SECTOR      ;INFO
5637 026560 006237 001422      ASR      SECTOR
5638 026564 000207      RTS      PC
5639
5640      ;READ THE CYL DIFF/OFFSET IN RKMR2, RIGHT JUSTIFY IT & STORE IT IN 'CYLDIF'
5641
5642 026566 012765 000002 000026 RDCYLD: MOV      #2,RKMR1(R5) ;WORD 2
5643 026574 004737 026106      JSR      PC,GSTAT
5644 026500 013737 005366 001376      MOV      HMR2,CYLDIF
5645
5646 026606 043737 012660 001376      BIC      MASK1,CYLDIF
5647 026614 006237 001376      ASR      CYLDIF ;RIGHT JUSTIFY
5648 026620 006237 001376      ASR      CYLDIF ;CYL DIFF/OFFSET
5649 026624 006237 001376      ASR      CYLDIF ;INFO
5650 026630 006237 001376      ASR      CYLDIF
5651 026634 023737 001376 012656      CMP      CYLDIF,MASK ;CHK TO SEE IF RET IN COMPL. FORM
5652 026642 001002      BNE      IS ;BR IF NOT
5653 026644 005037 001376      CLR      CYLDIF ;CLR IF YES
5654 026650 000207      IS:      RTS      PC
5655
5656      ;READ THE CYL ADDR IN RKMR3, RIGHT JUSTIFY IT & STORE IT IN 'CYLADD'
5657
5658 026652 012765 000002 000026 RDCYLA: MOV      #2,RKMR1(R5) ;WORD 2
5659 026660 004737 026106      JSR      PC,GSTAT
5660 026664 013737 005370 001400      MOV      HMR3,CYLADD
5661 026672 043737 012660 001400      BIC      MASK1,CYLADD
5662 026700 006237 001400      ASR      CYLADD ;RIGHT JUSTIFY
5663 026704 006237 001400      ASR      CYLADD ;CYL ADDR
5664 026710 006237 001400      ASR      CYLADD ;INFO
5665 026714 006237 001400      ASR      CYLADD
5666 026720 000207      RTS      PC
5667
5668      ;READ THE CYL DIFF/OFFSET IN H.A2, RIGHT JUSTIFY IT & STORE IT IN 'CYLDIF'
5669
5670 026722 013737 005420 001376 RACYLD: MOV      H.A2,CYLDIF
5671 026730 043737 012660 001376      BIC      MASK1,CYLDIF ;CLEAR UNWANTED INFO
5672 026736 006237 001376      ASR      CYLDIF ;RIGHT JUSTIFY
5673 026742 006237 001376      ASR      CYLDIF
5674 026746 006237 001376      ASR      CYLDIF
5675 026752 006237 001376      ASR      CYLDIF
5676 026756 023737 001376 012656      CMP      CYLDIF,MASK ;CHK TO SEE IF RET IN COMPL. FORM
5677 026764 001002      BNE      IS ;BR IF NO
5678 026766 005037 001376      CLR      CYLDIF ;ELSE CLEAR
5679 026772 000207      IS:      RTS      PC
5680
5681      ;READ THE CYL ADDR IN H.B2, RIGHT JUSTIFY IT & STORE IT IN 'CYLADD'
5682
5683 026774 013737 005422 001400 RACYLA: MOV      H.B2,CYLADD
5684 027002 043737 012660 001400      BIC      MASK1,CYLADD ;CLEAR UNWANTED INFO
5685 027010 006237 001400      ASR      CYLADD ;RIGHT JUSTIFY
5686 027014 006237 001400      ASR      CYLADD
5687 027020 006237 001400      ASR      CYLADD
5688 027024 006237 001400      ASR      CYLADD
5689 027030 000207      RTS      PC
5690
5691      ;READ THE SECTOR COUNT IN H.B3, RIGHT JUSTIFY IT & STORE IT IN 'SECTOR'

```

```

5692
5693 027032 013737 005426 001422 RSEC: MOV H.B3,SECTOR
5694 027040 042737 177017 001422 BIC #1C<M.SECT>,SECTOR ;CLEAR UNWANTED INFO
5695 027046 006237 001422 ASR SECTOR ;RIGHT JUSTIFY
5696 027052 006237 001422 ASR SECTOR
5697 027056 006237 001422 ASR SECTOR
5698 027062 006237 001422 ASR SECTOR
5699 027066 000207 RTS PC
5700
5701 ;READ THE HEAD ADDR IN H.B3, RIGHT IT & STORE IT IN 'HEAD'
5702
5703 027070 013737 005426 001444 RHEAD: MOV H.B3,HEAD
5704 027076 042737 170777 001444 BIC #1C<M.HEAD>,HEAD ;CLEAR UNWANTED INFO
5705 027104 006237 001444 ASR HEAD ;RIGHT JUSTIFY IT
5706 027110 000337 001444 SWAB HEAD
5707 027114 000207 RTS PC
5708
5709 ;FIND SECTOR 17
5710 ;RETURN IF NOT FOUND
5711 ;RETURN +4 IF FOUND
5712
5713 027116 013737 001430 005376 FSEC17: MOV T5000,TEMP1 ;SETUP TIMEOUT
5714 027124 004737 026516 000021 JSR PC,RDSEC ;READ SECTOR
5715 027130 023727 001422 000021 CMP SECTOR,#17. ;TEST FOR SECTOR 17
5716 027136 001014 BNE 2$ ;BR IF NOT 17
5717
5718 027140 004737 026516 JSR PC,RDSEC
5719 027144 023727 001422 000021 CMP SECTOR,#17.
5720 027152 001412 BEQ 3$ ;BR IF READ SAME TWICE
5721 027154 004737 026516 JSR PC,RDSEC ;ELSE TRY 1 MORE TIME
5722 027160 023727 001422 000021 CMP SECTOR,#17.
5723 027166 001404 BEQ 3$ ;BR IF 17
5724
5725 027170 005337 005376 2$: DEC TEMP1
5726 027174 001353 BNE 1$ ;TRY AGAIN
5727 027176 000207 RTS PC
5728
5729 027200 062716 000004 3$: ADD #4,(SP) ;SKIP OVER ERROR
5730 027204 000207 RTS PC
5731
5732 ;FIND DESIRED CYL DIFF
5733 ;RETURN IF NOT FOUND
5734 ;RETURN+6 IF FOUND
5735
5736
5737 027206 013737 001430 005376 FCYL: MOV T5000,TEMP1 ;SETUP TIMEOUT
5738 027214 004737 026566 005400 JSR PC,RDCYLD
5739 027220 023737 001376 005400 CMP CYLDIF,TEMP2 ;TEST FOR CYL DIFF
5740 027226 001014 BNE 2$ ;BR IF NOT FOUND
5741
5742 027230 004737 026566 JSR PC,RDCYLD
5743 027234 023737 001376 005400 CMP CYLDIF,TEMP2
5744 027242 001412 BEQ 3$ ;BR IF READ SAME TWICE
5745 027244 004737 026566 JSR PC,RDCYLD ;ELSE TRY 1 MORE TIME
5746 027250 023737 001376 005400 CMP CYLDIF,TEMP2
5747 027256 001404 BEQ 3$

```

```

5748
5749 027260 005337 005376      2$:   DEC   TEMP1
5750 027264 001353                BNE   1$           ;TRY AGAIN
5751 027266 000207                RTS   PC
5752
5753 027270 062716 000006      3$:   ADD   #E,(SP)   ;SKIP OVER ERROR
5754 027274 000207                RTS   PC
5755
5756
5757 ;ROUTINE TO FIND HEADS HOME IN RKMR2 WORD 1 BEFORE SPECIFIED DELAY
5758 ;ENTER WITH TIME IN SECONDS IN TEMP2
5759 ;RETURN IF NOT FOUND
5760 ;RETURN+2 IF FOUND - SKIP OVER ERROR
5761
5762 027276 012737 177777 005376 FHDHM: MOV   #-1,TEMP1   ;ALL 1'S
5763 027304 012765 000001 000026      MOV   #1,RKMR1(R5) ;WORD 1
5764 027312 004737 026106      1$:   JSR   PC,GSTAT
5765 027316 032737 000040 005366      BIT   #D.HDHM,HMR2
5766 027324 001007                BNE   2$
5767 027326 005337 005376      DEC   TEMP1
5768 027332 001367                BNE   1$
5769
5770 027334 005337 005400      DEC   TEMP2
5771 027340 001356                BNE   FHDHM
5772 027342 000207                RTS   PC
5773 027344 062716 000002      2$:   ADD   #2,(SP)   ;SKIP OVER ERROR
5774 027350 000207                RTS   PC
5775
5776 ;ROUTINE TO FIND LOAD HEADS IN RKMR2 WORD 1 BEFORE SMS
5777 ;RETURN IF NOT FOUND
5778 ;RETURN+2 IF FOUND: SKIP OVER ERROR
5779
5780 027352 012737 000372 005376 FLOAD: MOV   #250,TEMP1
5781 027360 012765 000001 000026      MOV   #1,RKMR1(R5) ;WORD 1
5782 027366 004737 026106      1$:   JSR   PC,GSTAT
5783 027372 032737 010000 005366      BIT   #D.LOAD,HMR2
5784 027400 001004                BNE   2$
5785 027402 005337 005376      DEC   TEMP1
5786 027406 001367                BNE   1$
5787 027410 000207                RTS   PC
5788 027412 062716 000002      2$:   ADD   #2,(SP)   ;SKIP OVER ERROR
5789 027416 000207                RTS   PC
5790
5791 ;FILL HEADER TABLE WITH 66 WORDS OF VALID HEADERS
5792 ;ENTER WITH CYL # IN 'CALADD'
5793 ;ENTER WITH HEAD # IN 'HEAD'
5794 ;ENTER WITH FORMAT IN 'FORMAT'
5795
5796 027420 01C046 FHDTAB: MOV   R0,-(SP)   ;SAV R0
5797 027422 010146      MOV   R1,-(SP)   ;SAV R1
5798 027424 012700 001506      MOV   #HDTAB,R0   ;HEADER WORD TABLE ADDR
5799 027430 005001                CLR   R1           ;SECTOR COUNTER
5800 027432 013737 001442 001446      MOV   HEAD,HD1
5801 027440 006337 001446      ASL   HD1
5802 027444 006337 001446      ASL   HD1
5803 027450 006337 001446      ASL   HD1

```

```

5804 027454 006337 001446      ASL    HD1
5805 027460 006337 001446      ASL    HD1          ;SETUP HEAD # FOR WORD 2 OF HEADER
5806 027464 013737 001450 001452  MOV    FORMAT,FMT1
5807 027472 000337 001452      SWAB   FMT1
5808 027476 006337 001452      ASL    FMT1          ;SETUP FORMAT FOR WORD 2 OF HEADER
5809
5810 027502 013720 001402      1$:   MOV    CALADD,(R0)+ ;HEADER WORD 1-CYL ADDR
5811 027506 010110      MOV    R1,(R0)      ;HEADER WORD 2-SECTOR NO
5812 027510 053710 001446      BIS    HD1,(R0)     ;
5813 027514 053710 001452      BIS    FMT1,(R0)   ;
5814 027520 004737 027600      JSR    PC,SECFLG   ;GET SECTOR FLAGS
5815
5816 027524 013737 001402 005376  MOV    CALADD,TEMP1
5817 027532 011037 005400      MOV    (R0),TEMP2
5818 027536 043737 001402 005400  BIC    CALADD,TEMP2
5819 027544 042037 005376      BIC    (R0)+,TEMP1
5820 027550 053737 005400 005400  BIS    TEMP1,TEMP2
5821 027556 013720 005400      MOV    TEMP2,(R0)+ ;HEADER WORD 3-HEADER CHECK
5822
5823 027562 005201      INC    R1           ;SECTOR CTR
5824 027564 020127 000026      CMP    R1,#22.     ;ALL 22 SECTORS DONE? (66 WORDS)
5825 027570 001344      BNE    1$          ;BR IF NO
5826
5827 027572 012601      MOV    (SP)+,R1    ;RESTOR R1
5828 027574 012600      MOV    (SP)+,R0    ;RESTOR R0
5829 027576 000207      RTS    PC
5830
5831      ; THIS ROUTINE GETS INFORMATION FROM THE BAD SECTOR TABLE FILLED BY A PREVIOUS
5832      ; TEST & SETS BITS 14 & 15 APPROPRIATLY.
5833
5834 027600 010246      SECFLG: MOV    R2,-(SP)   ;SAVE R2
5835 027602 005737 001450      TST    FORMAT
5836 027606 001016      BNE    1$          ;BR IF 20 SECTOR FORMAT
5837 027610 012702 002332      MOV    #BSE22H+8.,R2
5838 027614 004737 027650      JSR    PC,FLGTST  ;GET HARDWARE DETECTED FLAG
5839 027620 052710 100000      BIS    #BIT15,(R0) ;RETURN HERE IF GOOD SECTOR
5840
5841 027624 012702 003332      MOV    #BSE22S+8.,R2 ;ELSE RETURN HERE
5842 027630 004737 027650      JSR    PC,FLGTST  ;GET SOFTWARE DETECTED FLAG
5843 027634 052710 040000      BIS    #BIT14,(R0) ;RETURN HERE IF GOOD SECTOR
5844
5845 027640 012602      MOV    (SP)+,R2   ;ELSE RETURN HERE
5846 027642 000207      RTS    PC
5847
5848 027644 012602      1$:   MOV    (SP)+,R2   ;RESTORE R2
5849 027646 000207      RTS    PC
5850
5851      ; THIS ROUTINE DOES THE ACTUAL SCANNING OF THE BAD SECTOR TABLES.
5852      ; ENTER WITH THE ADDRESS OF TABLE (BSE22H, BSE22S, ETC.) IN TEMP1
5853      ; RETURN IF NO COMPARE
5854      ; RETURN+4 IF COMPARE
5855
5856
5857 027650 010346      FLGTST: MOV    R3,-(SP) ;SAVE R3
5858
5859 027652 021227 177777      1$:   CMP    (R2),#-1   ;SEE IF ALL 1'S

```



```

5860 027656 001002          BNE      2$          ;BR IF NO
5861 027660 012603          MOV      (SP)+,R3    ;RESTORE R3
5862 027662 000207          RTS      PC
5863
5864 027664 022237 001402    2$:      CMP      (R2)+,CALADD ;SEE IF=CYL # & ADR PTR TO TRK/SECTOR WORD
5865 027670 001403          BEQ      3$
5866 027672 062702 000002    ADD      #2,R2      ;GO TO NEXT CYL WORD IN TABLE
5867 027676 000765          BR       1$
5868
5869 027700 013703 001442    3$:      MOV      HEAD,R3    ;GET HEAD # FROM FHDTAB ROUTINE
5870 027704 000303          SWAB    R3
5871 027706 050103          BIS     R1,R3      ;ADD SECTOR # FROM FHDTAB ROUTINE
5872 027710 022203          CMP     (R2)+,R3    ;SEE IF SECTOR/HEAD COMPARE
5873                                     ;& INCR PTR TO NEXT CYL WORD
5874 027712 001401          BEQ     4$          ;BR IF COMPARE
5875 027714 000756          BR      1$          ;ELSE TRY NEXT CYL
5876
5877 027716 012603          4$:      MOV     (SP)+,R3    ;RESTORE R3
5878 027720 062716 000004    ADD     #4,(SP)     ;INCREMENT RET ADDR
5879 027724 000207          RTS     PC
5880
5881                                     ;
5882                                     ; THIS ROUTINE SORTS THE RHTAB TABLE FROM WHATEVER SECTOR IT BEGINS
5883                                     ; WITH AND RE-WRITES THE INFO IN SRTTAB TABLE TO BEGIN WITH SECTOR 0
5884
5885                                     SORT:
5886 MOV     RO,-(SP)      ;SAVE RO
5887 MOV     R1,-(SP)      ;SAVE R1
5888 JSR     PC,ROSEC
5889 ADD     #1,SECTOR
5890 JSR     PC,MULT6      ;MULT SECTOR BY 6
5891
5892 MOV     #132,RO
5893 SUB     SECTOR,RO     ;RO-SECTOR TO RO = INDEX
5894 MOV     RO,SECTOR
5895 ADD     #RHTAB,SECTOR ;SAVE INDEX
5896
5897 ADD     #RHTAB,RO     ;INDEX TO BOT HALF OF RHTAB
5898 MOV     #SRTTAB,R1    ;INDEX TO TOP HALF OF SRTTAB
5899
5900 030002 012021          1$:      MOV     (RO)+,(R1)+ ;PUT BOTTOM OF RHTAB TO TOP OF SRTTAB
5901 030004 027027 002116    CMP     RO,#RHTAB+132.
5902 030010 00.374          BNE     1$
5903
5904 030012 012700 001712    2$:      MOV     #RHTAB,RO   ;PUT TOP OF RHTAB TO BOT OF SRTTAB
5905 030016 012021          MOV     (RO)+,(R1)+
5906 030020 020037 001422    CMP     RO,SECTOR
5907 030024 001374          BNE     2$
5908
5909 030026 012601          MOV     (SP)+,R1    ;RESTOR R1
5910 030030 012600          MOV     (SP)+,RO    ;RESTOR RO
5911 030032 000207          RTS     PC
5912
5913                                     ;
5914                                     ; MULT BY 6. ENTER WITH DESIRED # IN 'SECTOR'
5915                                     ;

```

5916 030034 006337 001422
5917 030040 013746 001422
5918 030044 006337 001422
5919 030050 062637 001422
5920 030054 000207

MULT6: ASL SECTOR ;2 X SECTOR
MOV SECTOR,-(SP)
ASL SECTOR ;4 X SECTOR
ADD (SP)+,SECTOR ;(4 X 5)+(2 X 5) = 6 X SECTOR
RTS PC

THIS ROUTINE IS ENTERED ONLY IF THERE IS A BSE ERROR AFTER A WRITE DATA
CMD. IT VERIFIES THAT THE BAD SECTOR IS LISTED IN THE BSE INFORMATION
CYLINDER AT CYL 410, TRACK 2.

RETURN IF SECTOR NOT LISTED IN BSE TABLE, ERROR CONDITION.
RETURN+2 IF LISTED, SKIP OVER ERROR

5930 030056 010446
5931
5932 030060 032737 010000 005340
5933 030066 001014

TRUERR: MOV R4,-(SP) ;SAVE R4
BIT #CFMT,HCS1 ;CHECK FORMAT
BNE 3\$;BR FOR 20 SECTOR FORMAT
;NOTE, 20 SECTOR FORMAT NOT
;DONE IN THIS PROGRAM

5937 030070 012704 002332
5938 030074 004737 030130
5939 030100 000407

MOV #BSE22H+8.,R4
JSR PC,TERR1 ;SEE IF ON HARDWARE DETECTED TABLE
BR 3\$;RETURN HERE IF YES

5941 030102 012704 003332
5942 030106 004737 030130
5943 030112 000402

MOV #BSE22S+8.,R4 ;ELSE RETURN HERE
JSR PC,TERR1 ;& SEE IF ON SOFTWARE DETECTED TABLE
BR 3\$;RETURN HERE IF YES

5945 030114 012604
5946 030116 000207

1\$: MOV (SP)+,R4 ;RESTORE R4
RTS PC ;RETURN WITHOUT JUMPING OVER ERROR

5949 030120 012604
5950 030122 062716 000002
5951 030126 000207

3\$: MOV (SP)+,R4 ;RESTORE R4
ADD #2,(SP) ;SKIP OVER ERROR ON RETURN
RTS PC

THIS ROUTINE DOES THE ACTUAL COMPARING OF CYLINDER, HEAD & TRACK AGAINST
THE BSE TABLE FOR THE ABOVE SUBROUTINE.
RETURN IF FOUND ON TABLE
RETURN+2 IF NOT FOUND

5959 030130 021427 177777
5960 030134 001405
5961 030136 022437 005360
5962 030142 001405
5963 030144 005724
5964 030146 000770

TERR1: CMP (R4),#-1 ;SEE IF ALL 1'S
BEQ 1\$;BR IF YES, NOT ON TABLE
CMP (R4)+,HDC ;SEE IF CYL MATCH
BEQ 2\$;BR IF YES
TST (R4)+ ;ELSE ADV TO NEXT CYL WORD
BR TERR1 ;& TRY AGAIN.

5966 030150 062716 000002
5967 030154 000207

1\$: ADD #2,(SP)
RTS PC

5969 030156 022437 005350
5970 030162 001401
5971 030164 000761

2\$: CMP (R4)+,HDA ;SEE IF SECTOR & TRACK MATCH
BEQ 3\$;BR IF YES
BR TERR1 ;OR TRY AGAIN

```

5972
5973 030166 000207          3$:   RTS   PC
5974
5975
5976
5977
5978
5979
5980
5981
5982 030170 005037 001412    CLKON: CLR   TIMUP
5983 030174 005737 005504      TST   PCLKF
5984 030200 001004          BNE   1$           ;BRANCH IF P-CLOCK PRESENT
5985 030202 012777 000100 151136  MOV   #100,@LKS    ;L-CLOCK, ENABLE INT
5986 030210 000207          RTS   PC
5987 030212 012777 177777 151122 1$:   MOV   #-1,@PKSB   ;P-CLOCK, ALL 1'S
5988 030220 012777 000135 151112  MOV   #135,@PKS   ;ENABLE INT, CT UP, REP INT
5989 030226 000207          RTS   PC           ;LINE FREQ & RUN
5990
5991
5992
5993 030230 005037 001412    ;KW11-L & KW11-P INTERRUPT HANDLER
5994 030234 005337 001406    CLOCK: CLR   TIMUP
5995 030240 001010          DEC   COUNT
5996 030242 013737 001404 001406  BNE   1$
5997 030250 005337 001410  MOV   HZ,COUNT
5998 030254 001002          DEC   SEC
5999 030256 005237 001412  BNE   1$
6000 030262 000002          INC   TIMUP       ;SORRY, TIME IS UP
6001
6002
6003
6004 030264 005737 005504    1$:   RTI
6005 030270 001003          ;ROUTINE TO TURN L OR P CLOCK INTERRUPT OFF
6006 030272 005077 151050    CLKOF: TST   PCLKF
6007 030276 000207          BNE   1$           ;BRACH IF P-CLOCK PRESENT
6008 030300 005077 151034  CLR   @LKS        ;L-CLOCK, CLEAR INTERRUPT
          RTS   PC
          CLR   @PKS ;P-CLOCK, CLEAR INTERRUPT

```

6009 030304 000207
6010
6011
6012
6013
6014
6015
6016
6017
6018
6019
6020
6021 030306 010046
6022 030310 010146
6023 030312 012700 000021
6024 030316 005001
6025 030320 000241
6026
6027 030322 006137 005376
6028 030326 103001
6029 030330 005201
6030 030332 005300
6031 030334 001372
6032
6033 030336 032701 000001
6034 030342 001003

RTS PC

```

; THIS ROUTINE GENERATES PARITY FOR THE EXPECTED MESSAGES
; ENTER WITH THE EXPECTED WORD IN TEMP1
; TEMP1 IS ROTATED LEFT 17 TIMES. EACH TIME THE CARRY BIT IS SET
; R1 IS INCREMENTED. AT THE END OF 17 ROTATES ( TEMP1 BACK TO ORIG),
; R1 BIT 0 IS EXAMINED. IF IT IS SET, INDICATING AN ODD # OF 1'S.
; THE PARITY BIT IS NOT SET IN B
; IF IT IS NOT SET, INDICATING AN EVEN # OF 1'S ,THE PARITY BIT IS
; SET IN TEMP1
    
```

```

$BPAR:  MOV    RO, -(SP)      ;SAVE RO
        MOV    R1, -(SP)      ;SAVE R1
        MOV    #17, R0        ;SHIFT COUNTER
        CLR    R1             ;COUNT # OF 1'S IN TEMP1
        CLC                    ;CLEAR CARRY

1$:    ROL    TEMP1
        BCC   2$              ;BR IF CARRY CLEAR
        INC   R1              ;COUNT # OF 1'S
        DEC   R0              ;SHIFT COUNTER
        BNE   1$

2$:    DEC   R0
        BNE   1$

        BIT   #BIT0, R1
        BNE   3$              ;BR IF ODD # IN RO

3$:
    
```

```

6035 030344 052737 100000 005376      BIS      #M.PAR,TEMP1      ;SET PARITY BIT
6036 030352 012601      JS:      MOV      (SP)+,R1      ;RESTORE R1
6037 030354 012600      MOV      (SP)+,R0      ;RESTORE R0
6038 030356 000207      RTS      PC
6039
6040
6041      ;ROUTINE TO ENABLE LOOPING ON INTERMITTANT ERRORS
6042      ;WHEN $LPERR SET BY OTHER THAN SCOPE ROUTINE
6043      ;IE: MY LOOP MACRO
6044
6045 030360 032777 001000 150552  SCOPI$: BIT      #SW9,$SWR      ;LOOP ON ERROR?
6046 030366 001406      BEQ      1$           ;BR IF NO
6047 030370 105737 001103      TSTB     $ERFLG      ;HAD ERROR?
6048 030374 001403      BEQ      1$           ;BR IF NO
6049 030376 013716 001110      MOV      SLPERR,(SP)
6050 030402 000002      RTI
6051
6052 030404 011637 001110      1$:      MOV      (SP),$LPERR      ;SET LOOP ADDR FOR TIGHT SCOPE LOOP
6053 030410 000002      RTI
6054
6055
6056      ;ROUTINE TO INPUT A 'SPACE' OR 'CONTROL-E' FROM TTY
6057      ;RETURN IF CONTROL-E
6058      ;RETURN +4 IF SPACE
6059
6060
6061
6062 030412 005777 150530      CCSP:    TST      $STKB      ;CLEA. DONE FLAG
6063 030416 104410      RDCHR      ;READ CHAR FROM TTY
6064 030420 012600      MOV      (SP)+,R0      ;GET CHAR OFF STACK
6065 030422 020027 000040      CMP      R0,$SPBAR      ;SEE IF SPACE
6066 030426 001406      BEQ      1$           ;BR IF YES.
6067
6068 030430 020027 000J05      CMP      R0,$5         ;SEE IF CONTROL-E
6069 030434 001405      BEQ      2$           ;BR IF YES
6070 030436 104401 040160      TYPE     MSG31        ;?
6071 030442 000763      BR      CCSP          ;TRY AGAIN
6072
6073 030444 062716 000004      1$:      ADD      #4,(SP)
6074 030450 000207      2$:      RTS      PC
6075
6076
6077      ;ROUTINE TO INPUT A 'SPACE' FROM TTY
6078
6079 030452 005777 150470      GETSP:   TST      $STKB      ;CLEAR DONE FLAG
6080 030456 104410      RDCHR      ;READ CHAR OFF TTY
6081 030460 012600      MOV      (SP)+,R0      ;GET CHAR OFF STACK
6082 030462 020027 000040      CMP      R0,$SPBAR      ;SEE IF SPACE
6083 030466 001403      BEQ      1$           ;EXIT IF YES
6084 030470 104401 040160      TYPE     MSG31        ;?
6085 030474 000766      BR      GETSP        ;TRY AGAIN
6086 030476 000207      1$:      RTS      PC
6087
6088
6089      ;THIS ROUTINE IS ENTERED BY TYPING A CONTROL-C.
6090      ;IT IS USED TO ALLOW THE OPERATOR TO HALT THE CPU WHILE INSURING

```

```

; THAT HEADS ARE LOADED & FORMATTING IS VALID BEFORE ACTUALLY HALTING
; THE CPU.
6091
6092
6093
6094 030500 022626          STOP:  CMP      (SP)+,(SP)+      ;RESTORE STACK FROM INTERRUPT
6095
6096 030502 004737 026436  STOP1: JSR      PC,SUBCLR
6097 030506 104024          ERROR    24      ;CERR AFTER
6098
6099 030510 005737 005322          TST      UNLD      ;SEE IF HEADS UNLOADED
6100 030514 001437          BEQ      3$      ;BR IF NO
6101 030516 005737 000042          TST      42      ;SEE IF MANUAL OR AUTO MODE
6102 030522 001403          BEQ      1$      ;BR IF MANUAL MODE
6103 030524 104401 044037          TYPE    MSG74   ;PGM ABORT PENDING
6104 030530 000402          BR       2$
6105 030532 104401 044105          1$:     TYPE    ,MSG75 ;HALT PENDING
6106 030536
6107
6108 030536 004737 026436          JSR      PC,SUBCLR
6109 030542 104024          ERROR    24      ;CERR AFTER SCLR
6110
6111 030544 032737 010000 005366          BIT      #D.SPIN,HMR2 ;SEE IF SPINDLE ALREADY ON
6112 030552 001020          BNE     64$     ;BR IF YES
6113 030554 104401 040032          TYPE    ,MSG29   ;PLEASE WAIT, HEADS BEING LOADED
6114
6115 030560 012737 000011 005340          MOV      #SRTSPL,HCS1
6116 030566 004737 024446          JSR      PC,DOCMD  ;DO START SPINDLE CMD & GET CONTR RDY
6117 030572 104143          ERROR    143     ;CONTR RDY NOT SET AFTER CMD
6118
6119 030574 013737 001426 005400          MOV      T100,TEMP2
6120 030602 004737 025056          JSR      PC,FATT1  ;FIND ATTN
6121 030606 104144          ERROR    144     ;NO ATTN AFTER CMD
6122
6123 030610 005037 005322          CLR      UNLD
6124 030614          64$:
6125
6126 030614 005737 005324          3$:     TST      BADHDR   ;SEE IF HEADERS VALID
6127 030620 001460          BEQ      4$      ;BR IF YES
6128 030622 005237 005326          INC      HPEND
6129 030626 012765 100000 000000          MOV      #CCLR,RKCS1(R5)
6130 030634 013765 001222 000010          MOV      $UNIT,RKCS2(R5)
6131 030642 012737 000013 005340          MOV      #RECAL,HCS1
6132 030650 004737 024446          JSR      PC,DOCMD  ;DO RECAL CMD & GET CONTR RDY
6133 030654 104124          ERROR    124     ;RDY NOT SET AFTER RECAL CMD
6134
6135 030656 012765 000001 000026          MOV      #1,RKMR1(R5) ;SELECT WORD 1
6136 030664 004737 026106          JSR      PC,GSTAT
6137 030670 032737 020000 005366          BIT      #D.RTZ,HMR2
6138 030676 001001          BNE     65$
6139 030700 104214          ERROR    214     ;RTZ NOT SET DURING RECAL CMD
6140 030702 013737 001424 005400          65$:   MOV      T10,TEMP2 ;SETUP TIMEOUT
6141 030710 004737 025056          JSR      PC,FATT1  ;FIND ATTN
6142 030714 104055          ERROR    55      ;NO ATTN AFTER RECAL CMD
6143
6144 030716 012765 100000 000000          MOV      #CCLR,RKCS1(R5)
6145 030724 013765 001222 000010          MOV      $UNIT,RKCS2(R5) ;DRIVE#
6146 030732 012737 000005 005340          MOV      #CLEAR,HCS1

```

```

6147 030740 004737 024446      JSR    PC,DOCMD      ;DO DRIVE CLEAR CMD & GET CONTR RDY
6148 030744 104151              ERROR  151           ;NO RDY AFTER DRIVE CLEAR CMD
6149 030746 004737 025024      JSR    PC,TSTATN    ;TEST FOR ATTN
6150 030752 000401              BR     66$          ;
6151 030754 104154              ERROR  154           ;ATTN NOT CLEARED AFTER DRIVE CLEAR CMD
6152 030756                      66$:
6153
6154
6155 030756 000137 031022      JMP    FORM          ;WRITE VALID FORMATS
6156
6157 030762 005737 000042      4$:   TST    42          ;SEE IF MANUAL OR AUTO MODE
6158 030766 001410              BEQ    5$           ;BR IF MANUAL MODE
6159 030770 104401 044142      TYPE  MSG76        ;PGM ABORTED
6160 030774 005037 023762      CLR    $EOPCT      ;SET UP EOP TO EXIT TO MONITOR
6161 031000 005037 001176      CLR    $ESCAPE
6162 031004 000137 023734      JMP    $EOP        ;ABORT PROGRAM
6163
6164 031010 104401 044164      5$:   TYPE  ,MSG77    ;CPU HALTED
6165 031014 000000              HALT
6166 031016 000137 010612      JMP    ST5         ;START OVER IF CONTINUE PRESSED
6167
6168 031022                      FORM:
6169                      .SBTTL  UNEXPECTED TIMEOUT HANDLER
6170
6171                      ;
6172                      ; THIS ROUTINE IS ENTERED IF THERE IS
6173                      ; A. NON EXISTANT MEMORY (NO SSYN)
6174                      ; B. BOUNDARY ERROR
6175                      ; C. STACK OVERFLOW
6176                      ;
6177
6178 031022 011600      9ADTMO: MOV    (SP),RO      ;SAVE PC WHERE TIMEOUT OCCURRED.
6179 031024 005740      TST    -(RO)          ;GET PC BEFORE UPDATE
6180 031026 032777 020000 150104  BIT    #SW13,@SWR    ;INHIBIT ERR TYP0UT?
6181 031034 001005      BNE    1$           ;YES, DON'T TYPE
6182 031036 104401 044514      TYPE  EM3          ;ABORT TESTS UNEXP T.O. @ PC=
6183 031042 010046      MOV    RO,-(SP)     ;SAVE RO FOR TYPEOUT
6184
6185 031044 104403      TYPOS
6186 031046 006          .BYTE  6           ;GO TYPE--OCTAL ASCII
6187 031047 000          .BYTE  0           ;TYPE 6 DIGIT(S)
6188 031050 032777 001000 150062 1$:  BIT    #SW9,@SWR    ;SUPPRESS LEADING ZEROS
6189 031056 001403      BEQ    2$           ;LOOP ON ERROR?
6190 031060 022626      CMP    (SP)+,(SP)+ ;NO BRANCH
6191 031062 000177 150020      JMP    @SLPADR     ;YES, RESTORE STACK
6192
6193 031066 032777 040000 150044 2$:  BIT    #SW14,@SWR  ;GO TO STARTING ADDR OF TEST
6194 031074 001401      BEQ    3$           ;THAT GAVE BAD TIMEOUT
6195 031076 000002      RTI              ;LOOP ON TEST?
6196
6197 031100 000000      3$:   HALT         ;NO BRANCH
6198
6199
6200
6201
6202
; UNEXPECTED TIME OUT OCCURRED
; AS INDICATED. YOU CAN LOOP ON
; ERROR, LOOP ON TEST OR INHIBIT
; ERROR TYPEOUT BY SETTING THOSE
; SWITCHES.
;

```

```

6203 031102 022626          CMP      (SP)+,(SP)+      ;RESTORE STACK
6204 031104 000137 023734  JMP      $EOP             ;ABORT TESTS
6205
6206          .SBTTL MEMORY CHECK ENABLE TRAP
6207
6208 031110 012737 031124 001176 MEMERR: MOV      #1,$SESCAPE
6209 031116 011637 001354          MOV      (SP),TRAPPC      ;STORE PC
6210 031122 104202          ERROR      202          ;UNEXP MEM PARITY ERROR
6211
6212 031124 005037 001176          CLR      $SESCAPE
6213 031130 032777 001000 150002 1$:      BIT      #SW9,$SWR      ;CHECK IF LOOP ON ERROR
6214 031136 001001          BNE      2$             ;YES, FORCE STACK AND TRY AGAIN
6215 031140 000002          RTI                     ;ELSE RETURN
6216
6217 031142 012706 001100          MOV      #STACK,$SP      ;INI. STACK
6218 031146 000177 147736          JMP      $SLPERA
6219
6220          .SBTTL RK06 INTERRUPT HANDLER
6221
6222 031152 000240          INTER: NOP
6223 031154 000240          NOP
6224 031156 000240          NOP
6225 031160 011600          MOV      (SP),RO        ;SAVE PC WHERE INT OCCURRED.
6226 031162 005740          TST      -(RO)          ;GET PC BEFORE UPDATE.
6227 031164 104401 037004          TYPE      MSG6          ;INT AT PC=
6228 031170 010046          MOV      RO,-(SP)       ;SAVE RO FOR TYPEOUT
6229
6230
6231 031172 104403          TYPOS
6232 031174          .BYTE      6          ;GO TYPE--OCTAL ASCII
6233 031176 000000          .BYTE      0          ;TYPE 6 DIGIT(S)
6234 031200 000240          HALT                    ;SUPPRESS LEADING ZEROS
6235 031202 000240          NOP
6236 031204 000002          NOP
6237          RTI
6238
6239          .SBTTL POWER DOWN AND UP ROUTINES
6240
6241          OWER DOWN ROUTINE
6242 031206 012737 031220 000024 $PWRDN: MOV      #SPWRUP,PWRVEC ;SET UP VECTOR
6243 031214 000000          HALT
6244 031216 000776          BR      -2             ;HANG UP.
6245
6246          ;POWER UP ROUTINE
6247
6248 031220 005037 031272          $PWRUP: CLR      $PWRCT      ;WAIT LOOP FOR TTY
6249 031224 005237 031272          1$:      INC      $PWRCT      ;WAIT FOR THE INCR
6250 031230 001375          BNE      1$             ;OF WORD
6251 031232 012737 031206 000024          MOV      #SPWRDN,PWRVEC ;SET POWER DOWN VECTOR
6252 031240 012737 000340 000026          MOV      #PR7,PWRVEC+2 ;PRIORITY 7
6253 031246 012737 000340 000036          MOV      #PR7,TRAPVEC+2 ;LOCKOUT ALL INTERRUPTS FOR TRAPS
6254 031254 012706 001100          MOV      #STACK,$SP      ;INITIALIZE STACK
6255 031260 104401 037200          TYPE      ,MSG11        ;REPORT POWER FAIL
6256 031264 000005          RESET
6257 031266 000137 012664          JMP      PFSRT
6258

```



```

6259 031272 000000 $PWRCT: 0 ;WAIT COUNT FOR TTY
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
6270 031274 012746 000040 M.DPID: MOV #40,-(SP) ;COUNTER FOR DIVISION CYCLES
6271 031300 010446 MOV R4,-(SP) ;HI ORDER
6272 031302 010546 MOV R5,-(SP) ;LO ORDER TO THE STACK
6273 031304 005466 000002 NEG 2(SP) ;FORM NEGATIVE
6274 031310 005416 NEG @SP ;VERSION OF DIVISOR
6275 031312 005666 000002 SBC 2(SP)
6276 031316 061601 ADD @SP,R1
6277 031320 005500 ADC R0 ;PERFORM INIT SUBT.
6278 031322 066600 000002 ADD 2(SP),R0
6279 031326 103445 BCS M.DP50 ;IF CARRY THEN OVERFLOW HAS OCCURRED
6280 031330 005046 CLR -(SP) ;THIS IS A LONGER LASTING CARRY BIT
6281 031332 006103 M.DP40: ROL R3
6282 031334 006102 ROL R2
6283 031336 006101 ROL R1
6284 031340 006100 ROL R0
6285 031342 005716 TST @SP ;TEST CARRY INDICATOR
6286 031344 001410 BEQ M.DP41 ;IF TO CARRY THEN ADD, ELSE SUBT.
6287 031346 005016 CLR @SP ;CLEAR UP FOR NEXT TIME
6288 031350 066601 000002 ADD 2(SP),R1
6289 031354 005500 ADC R0 ;ADD -(DIVISOR)
6290 031356 005516 ADC @SP ;SET CARRY
6291 031360 066600 000004 ADD 4(SP),R0
6292 031364 000404 BR M.DP42
6293
6294 031366 060501 M.DP41: ADD R5,R1
6295 031370 005500 ADC R0 ;ADD +(DIVISOR)
6296 031372 005516 ADC @SP ;SET CARRY
6297 031374 060400 ADD R4,R0
6298 031376 005516 M.DP42: ADC @SP ;SET CARRY
6299 031400 005716 TST @SP ;TEST THE UPDATE INDICATOR
6300 031402 001401 BEQ .+4 ;IF 0, FORGET IT
6301 031404 005203 INC R3 ;NO CARRY POSSIBLE HERE
6302 031406 005366 000006 DEC 6(SP) ;DECREMENT CTR
6303 031412 003347 BGT M.DP40 ;BR IF MORE TO DO
6304 031414 006003 ROR R3
6305 031416 103404 BCS M.DP44
6306 031420 060501 ADD R5,R1
6307 031422 005500 ADC R0
6308 031424 060400 ADD R4,R0
6309 031426 000241 CLC
6310
6311 031430 006103 M.DP44: ROL R3
6312 031432 062706 000010 ADD #10,SP ;ADJUST STACK BY 4 WORDS
6313 031436 000242 CLV
6314 031440 000207 RTS PC

```

CZR6JEO UNIBUS RK6 DR PRT3
CZR6JE.P11 25-JAN-78 12:08

MACY11 30A(1052) 25-JAN-78 12:16 PAGE 121
POWER DOWN AND UP ROUTINES

SEQ 0121

6315
6316 031442 062706 000006
6317 031446 000262
6318 031450 000207
6319

M.DP50: ADD #6,SP
SEV
RTS PC

.SBTTL SCOPE HANDLER ROUTINE

```

*****
*THIS ROUTINE CONTROLS THE LOOPING OF SUBTESTS. IT WILL INCREMENT
*AND LOAD THE TEST NUMBER($STNM) INTO THE DISPLAY REG.(DISPLAY<7:0>)
*AND LOAD THE ERROR FLAG ($ERFLG) INTO DISPLAY<15:08>
*THE SWITCH OPTIONS PROVIDED BY THIS ROUTINE ARE:
*SW14=1      LOOP ON TEST
*SW11=1      INHIBIT ITERATIONS
*SW09=1      LOOP ON ERROR
*SW08=1      LOOP ON TEST IN SWR<7:0>
*CALL
*          SCOPE          ;;SCOPE=IOT

```

```

$SCOPE:
1$:      CKSWR          ;;TEST FOR CHANGE IN SOFT-SWR
        BIT           $BIT14,$SWR      ;;LOOP ON PRESENT TEST?
        BNE          $OVER           ;;YES IF SW14=1
        *****START OF CODE FOR THE XOR TESTER*****
$XTSTR:  BR           6$
        MOV          2$,$ERRVEC, -(SP)  ;;IF RUNNING ON THE "XOR" TESTER CHANGE
        MOV          5$,$ERRVEC      ;;THIS INSTRUCTION TO A "NOP" (NOP=240)
        TST         2$177060        ;;SAVE THE CONTENTS OF THE ERROR VECTOR
        MOV          (SP)+, 2$ERRVEC  ;;SET FOR TIMEOUT
        BR           $SVLAD          ;;TIME OUT ON XOR?
        CMP          (SP)+, (SP)+    ;;RESTORE THE ERROR VECTOR
        MOV          (SP)+, 2$ERRVEC  ;;GO TO THE NEXT TEST
        BR           7$             ;;CLEAR THE STACK AFTER A TIME OUT
        *****END OF CODE FOR THE XOR TESTER*****
        BIT           $BIT08,$SWR      ;;LOOP ON SPEC. TEST?
        BEQ          2$             ;;BR IF NO
        CMPB        2$SWR,$STNM      ;;ON THE RIGHT TEST? SWR<7:0>
        BEQ          $OVER           ;;BR IF YES
        TSTB        $ERFLG          ;;HAS AN ERROR OCCURRED?
        BEQ          3$             ;;BR IF NO
        CMPB        $ERMAX,$ERFLG   ;;MAX. ERRORS FOR THIS TEST OCCURRED?
        BHI         3$             ;;BR IF NO
        BIT           $BIT09,$SWR      ;;LOOP ON ERROR?
        BEQ          4$             ;;BR IF NO
        MOV          $LPERR,$LPADR    ;;SET LOOP ADDRESS TO LAST SCOPE
        BR           $OVER
        CLRB        $ERFLG          ;;ZERO THE ERROR FLAG
        CLR         $TIMES          ;;CLEAR THE NUMBER OF ITERATIONS TO MAKE
        BR         1$             ;;ESCAPE TO THE NEXT TEST
        BIT           $BIT11,$SWR      ;;INHIBIT ITERATIONS?
        BNE         1$             ;;BR IF YES
        TST         $PASS          ;;IF FIRST PASS OF PROGRAM
        BEQ         1$             ;;INHIBIT ITERATIONS
        INC         $ICNT          ;;INCREMENT ITERATION COUNT
        CMP         $TIMES,$ICNT     ;;CHECK THE NUMBER OF ITERATIONS MADE
        BGE        $OVER           ;;BR IF MORE ITERATION REQUIRED
        MOV         #1,$ICNT        ;;REINITIALIZE THE ITERATION COUNTER
        MOV         $MXCNT,$TIMES   ;;SET NUMBER OF ITERATIONS TO DO
        INCB        $STNM          ;;COUNT TEST NUMBERS
        MOVB       $STNM,$TESTN     ;;SET TEST NUMBER IN APT MAILBOX

```

```

6320
6321
6322
6323
6324
6325
6326
6327
6328
6329
6330
6331
6332
6333
6334 031452
6335 031452 104407
6336 031454 032777 040000 147456
6337 031462 001114
6338
6339 031464 000416
6340
6341 031466 013746 000004
6342 031472 012737 031512 000004
6343 031500 005737 177060
6344 031504 012637 000004
6345 031510 000463
6346 031512 022626
6347 031514 012637 000004
6348 031520 000423
6349 031522
6350 031522 032777 000400 147410
6351 031530 001404
6352 031532 127737 147402 001102
6353 031540 001465
6354 031542 105737 001103
6355 031546 001421
6356 031550 123737 001115 001103
6357 031556 101015
6358 031560 032777 001000 147352
6359 031566 001404
6360 031570 013737 001110 001106
6361 031576 000446
6362 031600 105037 001103
6363 031604 005037 001174
6364 031610 000415
6365 031612 032777 004000 147320
6366 031620 001011
6367 031622 005737 001216
6368 031626 001406
6369 031630 005237 001104
6370 031634 023737 001174 001104
6371 031642 002024
6372 031644 012737 000001 001104
6373 031652 013737 031730 001174
6374 031660 105237 001102
6375 031664 113737 001102 001214

```



```

6432 032116
6433 032116 022737 024022 000042
6434 032124 001001
6435 032126 000000
6436 032130
6437 032130 000002
6438
6439
6440
6441
6442
6443
6444
6445
6446
6447
6448
6449
6450
6451
6452
6453
6454
6455 032132 105737 001157
6456 032136 100002
6457 032140 000000
6458 032142 000430
6459 032144 010046
6460 032146 017600 000002
6461 032152 122737 000001 001230
6462 032160 001011
6463 032162 132737 000100 001231
6464 032170 001405
6465 032172 010037 032202
6466 032176 004737 032646
6467 032202 000000
6468 032204 132737 000040 001231
6469 032212 001003
6470 032214 112046
6471 032216 001005
6472 032220 005726
6473 032222 012600
6474 032224 062716 000002
6475 032230 000002
6476 032232 122716 000011
6477 032236 001430
6478 032240 122716 000200
6479 032244 001006
6480 032246 005726
6481 032250 104401
6482 032252 001205
6483 032254 105037 032410
6484 032260 000755
6485 032262 004737 032344
6486 032266 123726 001156
6487 032272 001350

```

```

5$: CMP #SENDAD, @#42 ;; ACT-11 AUTO-ACCEPT?
   BNE 6$ ;; BRANCH IF NO
   HALT ;; YES
6$: RTI ;; RETURN
.SBTTL TYPE ROUTINE

*****
*ROUTINE TO TYPE ASCIZ MESSAGE. MESSAGE MUST TERMINATE WITH A 0 BYTE.
*THE ROUTINE WILL INSERT A NUMBER OF NULL CHARACTERS AFTER A LINE FEED.
*NOTE1: $NULL CONTAINS THE CHARACTER TO BE USED AS THE FILLER CHARACTER.
*NOTE2: $FILLS CONTAINS THE NUMBER OF FILLER CHARACTERS REQUIRED.
*NOTE3: $FILLC CONTAINS THE CHARACTER TO FILL AFTER.
*
*CALL:
*1) USING A TRAP INSTRUCTION
* TYPE ,MESADR ;; MESADR IS FIRST ADDRESS OF AN ASCIZ STRING
*OR
* TYPE
* MESADR
*
$TYPE: TSTB $TFPLG ;; IS THERE A TERMINAL?
        BPL 1$ ;; BR IF YES
        HALT ;; HALT HERE IF NO TERMINAL
        BR 3$ ;; LEAVE
1$: MOV RO, -(SP) ;; SAVE RO
   MOV @2(SP), RO ;; GET ADDRESS OF ASCIZ STRING
   CMPB #APTENV, $ENV ;; RUNNING IN APT MODE
   BNE 62$ ;; NO, GO CHECK FOR APT CONSOLE
   BITB #APTSPool, $ENVM ;; SPOOL MESSAGE TO APT
   BEQ 62$ ;; NO, GO CHECK FOR CONSOLE
   MOV RO, 61$ ;; SETUP MESSAGE ADDRESS FOR APT
   JSR PC, $ATY3 ;; SPOOL MESSAGE TO APT
   .WORD 0 ;; MESSAGE ADDRESS
61$: BITB #APTCSUP, $ENVM ;; APT CONSOLE SUPPRESSED
   BNE 60$ ;; YES, SKIP TYPE OUT
   MOVB (RO)+, -(SP) ;; PUSH CHARACTER TO BE TYPED ONTO STACK
   BNE 4$ ;; BR IF IT ISN'T THE TERMINATOR
   TST (SP)+ ;; IF TERMINATOR POP IT OFF THE STACK
60$: MOV (SP)+, RO ;; RESTORE RO
   ADD #2, (SP) ;; ADJUST RETURN PC
   RTI ;; RETURN
4$: CMPB #HT, (SP) ;; BRANCH IF <HT>
   BEQ 8$
   CMPB #CRLF, (SP) ;; BRANCH IF NOT <CRLF>
   BNE 5$
   ST (SP)+ ;; POP <CR><LF> EQUIV
   TYPE ;; TYPE A CR AND LF
   $CRLF
   CLRB $CHARCNT ;; CLEAR CHARACTER COUNT
   BR 2$ ;; GET NEXT CHARACTER
5$: JSR PC, $TYPEC ;; GO TYPE THIS CHARACTER
6$: CMPB $FILLC, (SP)+ ;; IS IT TIME FOR FILLER CHARS.?
   BNE 2$ ;; IF NO GO GET NEXT CHAR.

```

```

6488 032274 013746 001154          MOV     $NULL,-(SP)          ;; GET # OF FILLER CHARS. NEEDED
6489                                     ;; AND THE NULL CHAR.
6490 032300 105366 000001          7$:   DECB     1(SP)          ;; DOES A NULL NEED TO BE TYPED?
6491 032304 002770                                     BLT     6$                  ;; BR IF NO--GO POP THE NULL OFF OF STACK
6492 032306 004737 032344          JSR     PC,$TYPEC          ;; GO TYPE A NULL
6493 032312 105337 032410          DECB     $CHARCNT          ;; DO NOT COUNT AS A COUNT
6494 032316 000770                                     BR      7$                  ;; LOOP
6495
6496                                     ;HORIZONTAL TAB PROCESSOR
6497
6498 032320 112716 000040          8$:   MOVVB   #' (SP)          ;; REPLACE TAB WITH SPACE
6499 032324 004737 032344          9$:   JSR     PC,$TYPEC          ;; TYPE A SPACE
6500 032330 132737 000007 032410   BITB   #',$CHARCNT          ;; BRANCH IF NOT AT
6501 032336 001372                                     BNE     9$                  ;; TAB STOP
6502 032340 005726                                     TST    (SP)+                ;; POP SPACE OFF STACK
6503 032342 000724                                     BR      2$                  ;; GET NEXT CHARACTER
6504 032344 105777 146600          $TYPEC: TSTB   2$TPS          ;; WAIT UNTIL PRINTER IS READY
6505 032350 100375                                     BPL    $TYPEC
6506 032352 116677 000002 146572   MOVVB  2(SP),2$TPB          ;; LOAD CHAR TO BE TYPED INTO DATA REG.
6507 032360 122766 000015 000002   CMPB   #CR,2(SP)          ;; IS CHARACTER A CARRIAGE RETURN?
6508 032366 001003                                     BNE     1$                  ;; BRANCH IF NO
6509 032370 105037 032410          CLRB   $CHARCNT          ;; YES--CLEAR CHARACTER COUNT
6510 032374 000406                                     BR      $TYPEX
6511 032376 122766 000012 000002   1$:   CMPB   #LF,2(SP)          ;; IS CHARACTER A LINE FEED?
6512 032404 001402                                     BEQ     $TYPEX              ;; BRANCH IF YES
6513 032406 105227                                     INCB   (PC)+                ;; COUNT THE CHARACTER
6514 032410 000000          $CHARCNT: .WORD 0          ;; CHARACTER COUNT STORAGE
6515 032412 000207          $TYPEX: RTS     PC
6516
6517                                     .SBTTL CONVERT BINARY TO DECIMAL AND TYPE ROUTINE
6518
6519                                     ;*****
6520                                     ;THIS ROUTINE IS USED TO CHANGE A 16-BIT BINARY NUMBER TO A 5-DIGIT
6521                                     ;SIGNED DECIMAL (ASCII) NUMBER AND TYPE IT. DEPENDING ON WHETHER THE
6522                                     ;NUMBER IS POSITIVE OR NEGATIVE A SPACE OR A MINUS SIGN WILL BE TYPED
6523                                     ;BEFORE THE FIRST DIGIT OF THE NUMBER. LEADING ZEROS WILL ALWAYS BE
6524                                     ;REPLACED WITH SPACES.
6525                                     ;CALL:
6526                                     ;*
6527                                     ;*   MOV     NUM,-(SP)          ;; PUT THE BINARY NUMBER ON THE STACK
6528                                     ;*   TYPDS          ;; GO TO THE ROUTINE
6529
6530 032414          $TYPDS: MOV     R0,-(SP)          ;; PUSH R0 ON STACK
6531 032416          MOV     R1,-(SP)          ;; PUSH R1 ON STACK
6532 032420          MOV     R2,-(SP)          ;; PUSH R2 ON STACK
6533 032422          MOV     R3,-(SP)          ;; PUSH R3 ON STACK
6534 032424          MOV     R5,-(SP)          ;; PUSH R5 ON STACK
6535 032426          MOV     #20200,-(SP)      ;; SET BLANK SWITCH AND SIGN
6536 032432          MOV     20(SP),R5        ;; GET THE INPUT NUMBER
6537 032436          BPL     1$              ;; BR IF INPUT IS POS.
6538 032440          NEG     R5              ;; MAKE THE BINARY NUMBER POS.
6539 032442          MOVVB  #'-,1(SP)        ;; MAKE THE ASCII NUMBER NEG.
6540 032450          1$:   CLR     R0              ;; ZERO THE CONSTANTS INDEX
6541 032452          MOV     #DBLK,R3        ;; SETUP THE OUTPUT POINTER
6542 032456          MOVVB  #' ,(R3)+        ;; SET THE FIRST CHARACTER TO A BLANK
6543 032462          2$:   CLR     R2              ;; CLEAR THE BCD NUMBER

```

```

6544 032464 016001 032620      MOV      $DTBL(R0),R1      ;; GET THE CONSTANT
6545 032470 160105      SUB      R1,R5            ;; FORM THIS BCD DIGIT
6546 032472 002402      BLT     4$               ;; BR IF DONE
6547 032474 005202      INC     R2               ;; INCREASE THE BCD DIGIT BY 1
6548 032476 000774      BR      3$
6549 032500 060105      4$:     ADD      R1,R5            ;; ADD BACK THE CONSTANT
6550 032502 005702      TST     R2               ;; CHECK IF BCD DIGIT=0
6551 032504 001002      BNE     5$               ;; FALL THROUGH IF 0
6552 032506 105716      TSTB   (SP)              ;; STILL DOING LEADING 0'S?
6553 032510 100407      BMI     7$               ;; BR IF YES
6554 032512 106316      5$:     ASLB   (SP)              ;; MSD?
6555 032514 103003      BCC     6$               ;; BR IF NO
6556 032516 116663 000001 177777      MOVB   1(SP),-1(R3)      ;; YES--SET THE SIGN
6557 032524 052702 000060      BIS    #'0,R2            ;; MAKE THE BCD DIGIT ASCII
6558 032530 052702 000040      7$:     BIS    #' ,R2        ;; MAKE IT A SPACE IF NOT ALREADY A DIGIT
6559 032534 110223      MOVB   R2,R3)+          ;; PUT THIS CHARACTER IN THE OUTPUT BUFFER
6560 032536 005720      TST    (R0)+            ;; JUST INCREMENTING
6561 032540 020027 000010      CMP    R0,#10           ;; CHECK THE TABLE INDEX
6562 032544 002746      BLT    2$               ;; GO DO THE NEXT DIGIT
6563 032546 003002      BGT    8$               ;; GO TO EXIT
6564 032550 010502      MOV    R5,R2            ;; GET THE LSD
6565 032552 000764      BR     6$               ;; GO CHANGE TO ASCII
6566 032554 105726      8$:     TSTB   (SP)+          ;; WAS THE LSD THE FIRST NON-ZERO?
6567 032556 100003      BPL    9$               ;; BR IF NO
6568 032560 116663 177777 177776      MOVB   -1(SP),-2(R3)    ;; YES--SET THE SIGN FOR TYPING
6569 032566 105013      9$:     CLRB   (R3)          ;; SET THE TERMINATOR
6570 032570 012605      MOV    (SP)+,R5         ;; POP STACK INTO R5
6571 032572 012603      MOV    (SP)+,R3         ;; POP STACK INTO R3
6572 032574 012602      MOV    (SP)+,R2         ;; POP STACK INTO R2
6573 032576 012601      MOV    (SP)+,R1         ;; POP STACK INTO R1
6574 032600 012600      MOV    (SP)+,R0         ;; POP STACK INTO R0
6575 032602 104401 032630      TYPE   $DBLK            ;; NOW TYPE THE NUMBER
6576 032606 016666 000702 000004      MOV    2(SP),4(SP)      ;; ADJUST THE STACK
6577 032614 012616      MOV    (SP)+,(SP)
6578 032616 000002      RTI
6579 032620 023420      $DTBL: 1000.
6580 032622 001750      1000.
6581 032624 000144      100.
6582 032626 000012      10.
6583 032630 000004      $DBLK: .BLKW 4
6584      .SBTTL APT COMMUNICATIONS ROUTINE
6585
6586
6587 032640 112737 000001 033104 $ATY1: MOVB #1,$FFLG      ;; TO REPORT FATAL ERROR
6588 032646 112737 000001 033102 $ATY3: MOVB #1,$MFLG      ;; TO TYPE A MESSAGE
6589 032654 000403      BR     $ATYC
6590 032656 112737 000001 033104 $ATY4: MOVB #1,$FFLG      ;; TO ONLY REPORT FATAL ERROR
6591 032664 $ATYC:
6592 032664 010046      MOV    R0,-(SP)         ;; PUSH R0 ON STACK
6593 032666 010146      MOV    R1,-(SP)         ;; PUSH R1 ON STACK
6594 032670 105737 033102      TSTB   $MFLG            ;; SHOULD TYPE A MESSAGE?
6595 032674 001450      BEQ    5$               ;; IF NOT: BR
6596 032676 122737 000001 001230      CMPEQ  #APTENV,$ENV     ;; OPERATING UNDER APT?
6597 032704 001031      BNE    3$               ;; IF NOT: BR
6598 032706 132737 000100 001231      BITB   #APTPOOL,$ENVM   ;; SHOULD SPOOL MESSAGES?
6599 032714 001425      BEQ    3$               ;; IF NOT: BR

```

```

6600 032716 017600 000004      MOV      24(SP),R0      ;;GET MESSAGE ADDR.
6601 032722 062766 000002 000004      ADD      #2,4(SP)      ;;BUMP RETURN ADDR.
6602 032730 005737 001210      1$:     TST      $MSGTYPE      ;;SEE IF DONE W/ LAST XMISSION?
6603 032734 001375      BNE      1$            ;;IF NOT: WAIT
6604 032736 010037 001224      MOV      R0,$MSGAD      ;;PUT ADDR IN MAILBOX
6605 032742 105720      2$:     TSTB     (R0)+      ;;FIND END OF MESSAGE
6606 032744 001376      BNE      2$            ;;
6607 032746 163700 001224      SUB      $MSGAD,R0      ;;SUB START OF MESSAGE
6608 032752 006200      ASR      R0            ;;GET MESSAGE LNGTH IN WORDS
6609 032754 010037 001226      MOV      R0,$MSGGLT      ;;PUT LENGTH IN MAILBOX
6610 032760 012737 000004 001210      MOV      #4,$MSGTYPE      ;;TELL APT TO TAKE MSG.
6611 032766 000413      BR       5$            ;;
6612 032770 017637 000004 033014 3$:     MOV      24(SP),4$      ;;PUT MSG ADDR IN JSR LINKAGE
6613 032776 062766 000002 000004      ADD      #2,4(SP)      ;;BUMP RETURN ADDRESS
6614 033004 0.3746 177776      MOV      177776,-(SP)   ;;PUSH 177776 ON STACK
6615 033010 004737 032132      JSR     PC,$TYPE      ;;CALL TYPE MACRO
6616 033014 000000      4$:     .WORD    0
6617 033016      5$:
6618 033016 105737 033104      10$:    TSTB     $FFLG      ;;SHOULD REPORT FATAL EPROR?
6619 033022 001416      BEQ     12$           ;;IF NOT: BR
6620 033024 005737 001230      TST     $ENV         ;;RUNNING UNDER APT?
6621 033030 001413      BEQ     12$           ;;IF NOT: BR
6622 033032 005737 001210      11$:    TST     $MSGTYPE      ;;FINISHED LAST MESSAGE?
6623 033036 001375      BNE     11$          ;;IF NOT: WAIT
6624 033040 017637 000004 001212      MOV     24(SP),$FATAL  ;;GET ERROR #
6625 033046 062766 000002 000004      ADD     #2,4(SP)      ;;BUMP RETURN ADDR.
6626 033054 005237 001210      INC     $MSGTYPE      ;;TELL APT TO TAKE ERROR
6627 033060 105037 033104      12$:    CLRB     $FFLG      ;;CLEAR FATAL FLAG
6628 033064 105037 033103      CLRB     $LFLG      ;;CLEAR LOG FLAG
6629 033070 105037 033102      CLRB     $MFLG      ;;CLEAR MESSAGE FLAG
6630 033074 012601      MOV     (SP)+,R1      ;;POP STACK INTO R1
6631 033076 012600      MOV     (SP)+,R0      ;;POP STACK INTO R0
6632 033100 000207      RTS     PC           ;;RETURN
6633 033102 000      $MFLG: .BYTE    0      ;;MESSG. FLAG
6634 033103 000      $LFLG: .BYTE    0      ;;LOG FLAG
6635 033104 000      $FFLG: .BYTE    0      ;;FATAL FLAG
6636      .EVEN
6637      APTSIZE=200
6638      APTENV=001
6639      APTSPool=100
6640      APTCSUP=040
6641      .SBTTL  BINARY TO OCTAL (ASCII) AND TYPE
6642
6643      ;*****
6644      ;THIS ROUTINE IS USED TO CHANGE A 16-BIT BINARY NUMBER TO A 6-DIGIT
6645      ;OCTAL (ASCII) NUMBER AND TYPE IT.
6646      ;$TYPOS-- ENTER HERE TO SETUP SUPPRESS ZEROS AND NUMBER OF DIGITS TO TYPE
6647      ;CALL:
6648      ;      MOV      NUM,-(SP)      ;;NUMBER TO BE TYPED
6649      ;      TYPOS      ;;CALL FOR TYPEOUT
6650      ;      .BYTE    N            ;;N=1 TO 6 FOR NUMBER OF DIGITS TO TYPE
6651      ;      .BYTE    M            ;;=1 OR 0
6652      ;      ;;1=TYPE LEADING ZEROS
6653      ;      ;;0=SUPPRESS LEADING ZEROS
6654      ;
6655      ;$STYPON----ENTER HERE TO TYPE OUT WITH THE SAME PARAMETERS AS THE LAST

```



```

6656      ;*STYPOS OR STYPOC
6657      ;*CALL:
6658      ;*      MOV      NUM,-(SP)      ;;NUMBER TO BE TYPED
6659      ;*      TYPON      ;;CALL FOR TYPEOUT
6660      ;*
6661      ;*STYPOC---ENTER HERE FOR TYPEOUT OF A 16 BIT NUMBER
6662      ;*CALL:
6663      ;*      MOV      NUM,-(SP)      ;;NUMBER TO BE TYPED
6664      ;*      TYPOC      ;;CALL FOR TYPEOUT
6665
6666      033106 017646 000000      STYPOS: MOV      2(SP),-(SP)      ;; PICKUP THE MODE
6667      033112 116637 000001 033331      MOVVB 1(SP),$OFILL      ;; LOAD ZERO FILL SWITCH
6668      033120 112637 033333      MOVVB (SP)+,$OMODE+1      ;; NUMBER OF DIGITS TO TYPE
6669      033124 062716 000002      ADD      #2,(SP)      ;; ADJUST RETURN ADDRESS
6670      033130 000406
6671      033132 112737 000001 033331      STYPOC: MOVVB #1,$OFILL      ;; SET THE ZERO FILL SWITCH
6672      033140 112737 000006 033333      MOVVB #6,$OMODE+1      ;; SET FOR SIX(6) DIGITS
6673      033146 112737 000005 033330      STYPON: MOVVB #5,$OCNT      ;; SET THE ITERATION COUNT
6674      033154 010346      MOV      R3,-(SP)      ;; SAVE R3
6675      033156 010446      MOV      R4,-(SP)      ;; SAVE R4
6676      033160 010546      MOV      R5,-(SP)      ;; SAVE R5
6677      033162 113704 033333      MOVVB $OMODE+1,R4      ;; GET THE NUMBER OF DIGITS TO TYPE
6678      033166 005404      NEG      R4
6679      033170 062704 000006      ADD      #6,R4      ;; SUBTRACT IT FOR MAX. ALLOWED
6680      033174 110437 033332      MOVVB R4,$OMODE      ;; SAVE IT FOR USE
6681      033200 113704 033331      MOVVB $OFILL,R4      ;; GET THE ZERO FILL SWITCH
6682      033204 016605 000012      MOV      12(SP),R5      ;; PICKUP THE INPUT NUMBER
6683      033210 005003      CLR      R3      ;; CLEAR THE OUTPUT WORD
6684      033212 006105      1$: ROL      R5      ;; ROTATE MSB INTO "C"
6685      033214 000404      BR      3$      ;; GO DO MSB
6686      033216 006105      2$: ROL      R5      ;; FORM THIS DIGIT
6687      033220 006105      ROL      R5
6688      033222 006105      ROL      R5
6689      033224 010503      MOV      R5,R3
6690      033226 006103      3$: ROL      R3      ;; GET LSB OF THIS DIGIT
6691      033230 105337 033332      DECB $OMODE      ;; TYPE THIS DIGIT?
6692      033234 100016      BPL      7$      ;; BR IF NO
6693      033236 042703 177770      BIC      #177770,R3      ;; GET RID OF JUNK
6694      033242 001002      BNE      4$      ;; TEST FOR 0
6695      033244 005704      TST      R4      ;; SUPPRESS THIS 0?
6696      033246 001403      BEQ      5$      ;; BR IF YES
6697      033250 005204      4$: INC      R4      ;; DON'T SUPPRESS ANYMORE 0'S
6698      033252 052703 000060      BIS      #'0,R3      ;; MAKE THIS DIGIT ASCII
6699      033256 052703 000040      5$: BIS      #' ,R3      ;; MAKE ASCII IF NOT ALREADY
6700      033262 110337 033335      MOVVB R3,$S      ;; SAVE FOR TYPING
6701      033266 104401 033326      TYPE      8$      ;; GO TYPE THIS DIGIT
6702      033272 105337 033330      7$: DECB $OCNT      ;; COUNT BY 1
6703      033276 003347      BGT      2$      ;; BR IF MORE TO DO
6704      033300 002402      BLT      6$      ;; BR IF DONE
6705      033302 005204      INC      R4      ;; INSURE LAST DIGIT ISN'T A BLANK
6706      033304 000744      BR      2$      ;; GO DO THE LAST DIGIT
6707      033306 012605      6$: MOV      (SP)+,R5      ;; RESTORE R5
6708      033310 012604      MOV      (SP)+,R4      ;; RESTORE R4
6709      033312 012603      MOV      (SP)+,R3      ;; RESTORE R3
6710      033314 016666 000002 000004      MOV      2(SP),4(SP)      ;; SET THE STACK FOR RETURNING
6711      033322 012616      MOV      (SP)+,(SP)

```

```

6712 033324 000002
6713 033326 000
6714 033327 000
6715 033330 000
6716 033331 000
6717 033332 000000
6718
6719
6720
6721
6722 033334 000000
6723 033336 000000
6724 033340 000000
6725 033342 000001
6726 033343
6727 033344
6728
6729
6730
6731
6732
6733
6734
6735
6736
6737 033344 005037 033334
6738 033350 012737 033342 033336
6739 033356 013737 033336 033340
6740 033364 012737 033414 000060
6741 033372 012737 000200 000062
6742 033400 005777 145542
6743 033404 012777 000100 145532
6744 033412 000207
6745
6746
6747
6748
6749
6750
6751
6752
6753 033414 117746 145526
6754 033420 042716 177600
6755 033424 021627 000003
6756 033430 001007
6757 033432 104401 034542
6758 033436 004737 033344
6759 033442 005726
6760 033444 000137 030500
6761 033450 021627 000007
6762 033454 001004
6763 033456 022737 000176 001140
6764 033464 001500
6765
6766 033466
6767 033466 022737 000001 033334

```

```

RTI ;: RETURN
BS: .BYTE 0 ;: STORAGE FOR ASCII DIGIT
;: .BYTE 0 ;: TERMINATOR FOR TYPE ROUTINE
SOCNT: .BYTE 0 ;: OCTAL DIGIT COUNTER
$OFILL: .BYTE 0 ;: ZERO FILL SWITCH
$OMODE: .WORD 0 ;: NUMBER OF DIGITS TO TYPE
.SBTTL TTY INPUT ROUTINE

;: *****
ENABL LSB
$TKCNT: .WORD 0 ;: NUMBER OF ITEMS IN QUEUE
$TKQIN: .WORD 0 ;: INPUT POINTER
$TKQOUT: .WORD 0 ;: OUTPUT POINTER
$TKQSR: .BLKB 1 ;: TTY KEYBOARD QUEUE
$TKQEND=.
.EVEN

; *TK INITIALIZE ROUT
; *THIS ROUTINE WILL INITIALIZE THE TTY KEYBOARD INPUT QUEUE
; *SETUP THE INTERRUPT VECTOR AND TURN ON THE KEYBOARD INTERRUPT
; *CALL:
; * JSR RETURN PC,$TKINT
; * RETURN

$TKINT: CLR $TKCNT ;: CLEAR COUNT OF ITEMS IN QUEUE
MOV $TKQSR,$TKQIN ;: MOVE THE STARTING ADDRESS OF THE
MOV $TKQIN,$TKQOUT ;: QUEUE INTO THE INPUT & OUTPUT POINTERS.
MOV $TKSRV,$TKVEC ;: INITIALIZE THE KEYBOARD VECTOR
MOV #200,$TKVEC+2 ;: "BR" LEVEL 4
TST $TKB ;: CLEAR DONE FLAG
MOV #100,$TKS ;: ENABLE TTY KEYBOARD INTERRUPT
RTS PC ;: RETURN TO CALLER

; *TK SERVICE ROUTINE
; *THIS ROUTINE WILL SERVICE THE TTY KEYBOARD INTERRUPT
; *BY READING THE CHARACTER FROM THE INPUT BUFFER AND PUTTING
; *IT IN THE QUEUE.
; *IF THE CHARACTER IS A "CONTROL-C" (↑C) $TKINT IS CALLED AND
; *UPON RETURN EXIT IS MADE TO THE "CONTROL-C" RESTART ADDRESS (STOP)

$TKSRV: MOVB $TKB, -(SP) ;: PICKUP THE CHARACTER
BIC #↑C177, (SP) ;: STRIP THE JUNK
CMP (SP), #3 ;: IS IT A CONTROL C?
BNE 1$ ;: BRANCH IF NO
TYPE $CNTLC ;: TYPE A CONTROL-C (↑C)
JSR PC,$TKINT ;: INIT THE KEYBOARD
TST (SP)+ ;: CLEAN UP STACK
JMP * STOP ;: CONTROL C RESTART

1$: CMP (SP), #7 ;: IS IT A CONTROL G?
BNE 2$ ;: BRANCH IF NO
CMP #SWREG, SWR ;: IS SOFT-SWR SELECTED?
BEQ 6$ ;: GO TO SWR CHANGE

2$: CMP #1, $TKCNT ;: IS THE QUEUE FULL?

```

6768	033474	001004			BNE	3\$:: BRANCH IF NO
6769	033476	104401	001200		TYPE	\$BELL	:: RING THE TTY BELL
6770	033502	005726			TST	(SP)+	:: CLEAN CHARACTER OFF OF STACK
6771	033504	000451			BR	5\$:: EXIT
6772	033506	021627	000023	3\$:	CMP	(SP), #23	:: IS IT A CONTROL-S?
6773	033512	001021			BNE	32\$:: BRANCH IF NO
6774	033514	005077	145424		CLR	\$STKS	:: DISABLE TTY KEYBOARD INTERRUPTS
6775	033520	005726			TST	(SP)+	:: CLEAN CHAR OFF STACK
6776	033522	105777	145416	31\$:	TSTB	\$STKS	:: WAIT FOR A CHAR
6777	033526	100375			BPL	31\$:: LOOP UNTIL ITS THERE
6778	033530	117746	145412		MOVB	\$STKB, -(SP)	:: GET THE CHARACTER
6779	033534	042716	177600		BIC	#1C17?, (SP)	:: MAKE IT 7-BIT ASCII
6780	033540	022627	000021		CMP	(SP)+, #21	:: IS IT A CONTROL-Q?
6781	033544	001366			BNE	31\$:: BRANCH IF NO
6782	033546	012777	000100	145370	MOV	#100, \$STKS	:: REENABLE TTY KEYBOARD INTERRUPTS
6783	033554	000002			RTI		:: RETURN
6784	033556	005237	033334	32\$:	INC	\$STKCNT	:: COUNT THIS CHARACTER
6785	033562	021627	000140		CMP	(SP), #140	:: IS IT UPPER CASE?
6786	033566	002405			BLT	4\$:: BRANCH IF YES
6787	033570	021627	000175		CMP	(SP), #175	:: IS IT A SPECIAL CHAR?
6788	033574	003002			BGT	4\$:: BRANCH IF YES
6789	033576	042716	000040		BIC	#40 (SP)	:: MAKE IT UPPER CASE
6790	033602	112677	177530	4\$:	MOVB	(SP)+, \$STKQIN	:: AND PUT IT IN QUEUE
6791	033606	005237	033336		INC	\$STKQIN	:: UPDATE THE POINTER
6792	033612	023727	033336	033343	CMP	\$STKQIN, \$STKQEND	:: GO OFF THE END?
6793	033620	001003			BNE	5\$:: BRANCH IF NO
6794	033622	012737	033342	033336	MOV	\$STKQSR, \$STKQIN	:: RESET THE POINTER
6795	033630	000002			RTI		:: RETURN

6796
6797
6798
6799
6800
6801
6802
6803
6804
6805
6806
6807
6808
6809
6810
6811
6812
6813
6814
6815
6816
6817
6818
6819
6820
6821
6822
6823

```

*****
*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.
$CKSWR: CMP    $SWREG, SWR    :: IS THE SOFT-SWR SELECTED
        BNE    15$          :: EXIT IF NOT
        TSTB   $STKS        :: IS A CHAR WAITING?
        BPL    15$          :: IF NOT, EXIT
        MOVB   $STKB, -(SP)  :: YES
        BIC    #1C17?, (SP)  :: MAKE IT 7-BIT ASCII
        CMP    (SP), #7      :: IS IT A CONTROL-G?
        BNE    2$           :: IF NOT, PUT IT IN THE TTY QUEUE
                          :: AND EXIT

```

```

*****
*CONTROL IS PASSED TO THIS POINT FROM EITHER THE TTY INTERRUPT SERVICE
*ROUTINE OR FROM THE SOFTWARE SWITCH REGISTER TRAP CALL, AS A RESULT OF A
*CONTROL-G BEING TYPED, AND THE SOFTWARE SWITCH REGISTER BEING SELECTED.
6$:    CMPB   $AUTOB, #1    :: ARE WE RUNNING IN AUTO-MODE?
        BEQ    2$           :: BRANCH IF YES
        TST   (SP)+        :: CLEAR CONTROL-G OFF STACK
        JSR   PC, $TKINT    :: FLUSH THE TTY INPUT QUEUE
        CLR   $STKS        :: DISABLE TTY KEYBOARD INTERRUPTS
        MOVB  #1, $INTAG    :: SET INTERRUPT MODE INDICATOR

```

6823	033716	104401	034554		TYPE	.\$CNTLG	:: ECHO THE CONTROL-G (^G)
------	--------	--------	--------	--	------	----------	----------------------------

6824	033722	104401	034561		\$GTSWR:	TYPE	\$MSWR	::	TYPE CURRENT CONTENTS
6825	033726	013746	000176			MOV	\$WREG,-(SP)	::	SAVE SWREG FOR TYPEOUT
6826	033732	104402				TYPOC		::	GO TYPE--OCTAL ASCII(ALL DIGITS)
6827	033734	104401	034572			TYPE	,\$MNEW	::	PROMPT FOR NEW SWR
6828	033740	005046			19\$:	CLR	-(SP)	::	CLEAR COUNTER
6829	033742	005046				CLR	-(SP)	::	THE NEW SWR
6830	033744	105777	145174		7\$:	TSTB	@\$TKS	::	CHAR THERE?
6831	033750	100375				BPL	7\$::	IF NOT TRY AGAIN
6832								::	
6833	033752	117746	145170			MOVB	@\$TKB,-(SP)	::	PICK UP CHAR
6834	033756	042716	177600			BIC	#↑C177,(SP)	::	MAKE IT 7-BIT ASCII
6835								::	
6836	033762	021627	000003			CMP	(SP),#3	::	IS IT A CONTROL-C?
6837	033766	001015				BNE	9\$::	BRANCH IF NOT
6838	033770	104401	034542			TYPE	,\$CNTLC	::	YES, ECHO CONTROL-C (↑C)
6839	033774	062706	000006			ADD	#6,SP	::	CLEAN UP STACK
6840	034000	123727	001135	000001		CMPB	\$INTAG,#1	::	REENABLE TTY KEYBOARD INTERRUPTS?
6841	034006	001003				BNE	8\$::	BRANCH IF NO
6842	034010	012777	000100	145126		MOV	#100,@\$TKS	::	ALLOW TTY KEYBOARD INTERRUPTS
6843	034016	000137	030500		8\$:	JMP	STOP	::	CONTROL-C RESTART
6844								::	
6845								::	
6846	034022	021627	000025		9\$:	CMP	(SP),#25	::	IS IT A CONTROL-U?
6847	034026	001005				BNE	10\$::	BRANCH IF NOT
6848	034030	104401	034547			TYPE	,\$CNTLU	::	YES, ECHO CONTROL-U (↑U)
6849	034034	062706	000006		20\$:	ADD	#6,SP	::	IGNORE PREVIOUS INPUT
6850	034040	000737				BR	19\$::	LET'S TRY IT AGAIN
6851								::	
6852								::	
6853	034042	021627	000015		10\$:	CMP	(SP),#15	::	IS IT A <CR>?
6854	034046	001022				BNE	16\$::	BRANCH IF NO
6855	034050	005766	000004			TST	4(SP)	::	YES, IS IT THE FIRST CHAR?
6856	034054	001403				BEQ	11\$::	BRANCH IF YES
6857	034056	016677	000002	145054		MOV	2(SP),@SWR	::	SAVE NEW SWR
6858	034064	062706	000006		11\$:	ADD	#6,SP	::	CLEAN UP STACK
6859	034070	104401	001205		14\$:	TYPE	,\$CRLF	::	ECHO <CR> AND <LF>
6860	034074	123727	001135	000001		CMPB	\$INTAG,#1	::	RE-ENABLE TTY KBD INTERRUPTS?
6861	034102	001003				BNE	15\$::	BRANCH IF NOT
6862	034104	012777	000100	145032		MOV	#100,@\$TKS	::	RE-ENABLE TTY KBD INTERRUPTS
6863	034112	000002			15\$:	RTI		::	RETURN
6864	034114	004737	032344		16\$:	JSR	PC,\$TYPEC	::	ECHO CHAR
6865	034120	021627	000060			CMP	(SP),#60	::	CHAR < 0?
6866	034124	002420				BLT	18\$::	BRANCH IF YES
6867	034126	021627	000067			CMP	(SP),#67	::	CHAR > 7?
6868	034132	003015				BGT	18\$::	BRANCH IF YES
6869	034134	042726	000060			BIC	#60,(SP)+	::	STRIP-OFF ASCII
6870	034140	005766	000002			TST	2(SP)	::	IS THIS THE FIRST CHAR
6871	034144	001403				BEQ	17\$::	BRANCH IF YES
6872	034146	006316				ASL	(SP)	::	NO, SHIFT PRESENT
6873	034150	006316				ASL	(SP)	::	CHAR OVER TO MAKE
6874	034152	006316				ASL	(SP)	::	ROOM FOR NEW ONE.
6875	034154	005266	000002		17\$:	INC	2(SP)	::	KEEP COUNT OF CHAR
6876	034160	056616	177776			BIS	-2(SP),(SP)	::	SET IN NEW CHAR
6877	034164	000667				BR	7\$::	GET THE NEXT ONE
6878	034166	104401	001204		18\$:	TYPE	,\$QUES	::	TYPE ?<CR><LF>
6879	034172	000720				BR	20\$::	SIMULATE CONTROL-U

```

6880
6881
6882
6883
6884
6885
6886
6887
6888
6889
6890
6891 034174 011646
6892 034176 016666 000004 000002
6893 034204 005066 000004
6894 034210 005046
6895 034212 012746 034220
6896 034216 000002
6897 034220
6898 034220 005737 033334
6899 034224 001775
6900 034226 005337 033334
6901 034232 117766 177102 000004
6902 034240 005237 033340
6903 034244 023727 033340 033343
6904 034252 001003
6905 034254 012737 033342 033340
6906 034262 000002
6907
6908
6909
6910
6911
6912
6913
6914 034264 010346
6915 034266 005046
6916 034270 012703 034520
6917 034274 022703 034542
6918 034300 101456
6919 034302 104410
6920 034304 112613
6921 034306 122713 000177
6922 034312 001022
6923 034314 005716
6924 034316 001007
6925 034320 112737 000134 034516
6926 034326 104401 034516
6927 034332 012716 177777
6928 034336 005303
6929 034340 020327 034520
6930 034344 103434
6931 034346 111337 034516
6932 034352 104401 034516
6933 034356 000746
6934 034360 005716
6935 034362 001406

```

```

.DSABL LSB

*****
*THIS ROUTINE WILL INPUT A SINGLE CHARACTER FROM THE TTY
*CALL:
*      RDCHR          ;; GET A CHARACTER FROM THE QUEUE
*      RETURN HERE   ;; CHARACTER IS ON THE STACK
*                   ;; WITH PARITY BIT STRIPPED OFF
*****

$RDCHR: MOV      (SP), -(SP)      ;; PUSH DOWN THE PC AND
        MOV      4(SP), 2(SP)    ;; THE PS
        CLR      4(SP)          ;; GET READY FOR A CHARACTER
        CLR      -(SP)          ;; PUT NEW PS ON STACK
        MOV      #64$, -(SP)    ;; PUT NEW PC ON STACK
        RTI                    ;; POP NEW PC AND PS

64$:
1$:     TST      $STKCNT          ;; WAIT ON A CHARACTER
        BEQ      1$
        DEC      $STKCNT        ;; DECREMENT THE COUNTER
        MOVB    2$STKQOUT, 4(SP) ;; GET ONE CHARACTER
        INC     $STKQOUT        ;; UPDATE THE POINTER
        CMP     $STKQOUT, #2$STKQEND ;; DID IT GO OFF OF THE END?
        BNE    2$              ;; BRANCH IF NO
        MOV     #2$STKQRT, $STKQOUT ;; RESET THE POINTER
        RTI                    ;; RETURN
2$:
*****
*THIS ROUTINE WILL INPUT A STRING FROM THE TTY
*CALL:
*      RDLIN         ;; INPUT A STRING FROM THE TTY
*      RETURN HERE  ;; ADDRESS OF FIRST CHARACTER WILL BE ON THE STACK
*                   ;; TERMINATOR WILL BE A BYTE OF ALL 0'S
*****

$RDLIN: MOV      R3, -(SP)        ;; SAVE R3
        CLR      -(SP)          ;; CLEAR THE RUBOUT KEY
        MOV      #2$TTYIN, R3   ;; GET ADDRESS
        CMP      #2$TTYIN+22, R3 ;; BUFFER FULL?
        BLOS    4$             ;; BR IF YES
        RDCHR   4$             ;; GO READ ONE CHARACTER FROM THE TTY
        MOVB    (SP)+, (R3)     ;; GET CHARACTER
        CMPB   #177, (R3)      ;; IS IT A RUBOUT
        BNE    5$             ;; BR IF NO
        TST    (SP)           ;; IS THIS THE FIRST RUBOUT?
        BNE    6$             ;; BR IF NO
        MOVB   #' \, 9$        ;; TYPE A BACK SLASH
        TYPE   9$
        MOV    #-1, (SP)      ;; SET THE RUBOUT KEY
        DEC   R3              ;; BACKUP BY ONE
        CMP   R3, #2$TTYIN    ;; STACK EMPTY?
        BLOS 4$              ;; BR IF YES
        MOVB (R3), 9$        ;; SETUP TO TYPEOUT THE DELETED CHAR.
        TYPE 9$              ;; GO TYPE
        BR   2$              ;; GO READ ANOTHER CHAR.
        TST (SP)            ;; RUBOUT KEY SET?
        BEQ 7$              ;; BR IF NO
5$:
6$:
7$:

```

```

6936 034364 112737 000134 034516      MOVB    #' \,9$          ;;TYPE A BACK S'ASH
6937 034372 104401 034516      TYPE    (9$)
6938 034376 005016          CLR     (SP)           ;;CLEAR THE RUBOUT KEY
6939 034400 122713 000025      7$:    CMPB    #25,(R3)  ;;IS CHARACTER A CTRL U?
6940 034404 001003          BNE     8$            BR IF NO
6941 034406 104401 034547      TYPE    $CNTLU        ;;TYPE A CONTROL "U"
6942 034412 000726          BR     1$            GO START OVER
6943 034414 122713 000022      8$:    CMPB    #22,(R3)  ;;IS CHARACTER A "+R"?
6944 034420 001011          BNE     3$            BRANCH IF NO
6945 034422 105013          CLRB   (R3)           CLEAR THE CHARACTER
6946 034424 104401 001205      TYPE    $CRLF         TYPE A "CR" & "LF"
6947 034430 104401 034520      TYPE    $"YIH"        TYPE THE INPUT STRING
6948 034434 000717          BR     2$            GO PICKUP ANOTHER CHACTER
6949 034436 104401 001204      4$:    TYPE    $QUES     TYPE A '?'
6950 034442 000712          BR     1$            CLEAR THE BUFFER AND LOOP
6951 034444 111337 034516      3$:    MOVB    (R3),9$   ECHO THE CHARACTER
6952 034450 104401 034516      TYPE    9$
6953 034454 122723 000015      CMPB    #15,(R3)+    ;;CHECK FOR RETURN
6954 034460 001305          BNE     2$            LOOP IF NOT RETURN
6955 034462 105063 177777      CLRB   -1(R3)        CLEAR RETURN (THE 15)
6956 034466 104401 001206      TYPE    $LF           TYPE A LINE FEED
6957 034472 005726          TST    (SP)+         CLEAN RUBOUT KEY FROM THE STACK
6958 034474 012603          MOV    (SP)+,R3     RESTORE R3
6959 034476 011646          MOV    (SP)-,(SP)   ADJUST THE STACK AND PUT ADDRESS OF THE
6960 034500 016666 000004 000002      MOV    4(SP),2(SP)  FIRST ASCII CHARACTER ON IT
6961 034506 012766 034520 000004      MOV    #STTYIN,4(SP)
6962 034514 000002          RTI
6963 034516 000          9$:    .BYTE    0          ;;RETURN
6964 034517 000          .BYTE    0          ;;STORAGE FOR ASCII CHAR. TO TYPE
6965 034520 000022          .BLKB   22          ;;TERMINATOR
6966 034542 041536 005015 000      $TTYIN: .ASCIZ  /?C<<15><12>  ;;RESERVE 22 BYTES FOR TTY INPUT
6967 034547 0136 006525 000012      $CNTLC: .ASCIZ  /?U<<15><12>  ;;CONTROL "C"
6968 034554 043536 005015 000      $CNTLU: .ASCIZ  /?G<<15><12>  ;;CONTROL "U"
6969 034561 015 051412 051127      $CNTLG: .ASCIZ  /?G<<15><12>  ;;CONTROL "G"
6970 034566 036440 000040          $MSWR:  .ASCIZ  <15><12>/SWR = /
6971 034572 020040 042516 020127      $MNEW:  .ASCIZ  / NEW = /
6972 034600 020075 000
6973 034604 034604
6974 .EVEN
6975 .SBTTL READ AN OCTAL NUMBER FROM THE TTY
6976
6977 *****
6978 *THIS ROUTINE WILL READ AN OCTAL (ASCII) NUMBER FROM THE TTY AND
6979 *CHANGE IT TO BINARY.
6980 *THE INPUT CHARACTERS WILL BE CHECKED TO INSURED THEY ARE LEGAL
6981 *OCTAL DIGITS. IF AN ILLEGAL CHARACTER IS READ A " " WILL BE TYPED
6982 *FOLLOWED BY A CARRIAGE RETURN-LINE FEED. THE COMPLETE NUMBER MUST
6983 *THEN BE RETYPED. THE INPUT IS TERMINATED BY TYPING A CARRIAGE RETURN.
6984 *CALL:
6985 *      RDOCT          ;;READ AN OCTAL NUMBER
6986 *      RETURN HERE   ;;LOW ORDER BITS ARE ON TOP OF THE STACK
6987 *                   ;;HIGH ORDER BITS ARE IN $HIOCT
6988 $RDOCT: MOV    (SP)-,(SP)  ;;PROVIDE SPACE FOR THE
6989         MOV    4(SP),2(SP) ;;INPUT NUMBER
6990         MOV    R0,-(SP)    ;;PUSH R0 ON STACK
6991         MOV    R1,-(SP)    ;;PUSH R1 ON STACK

```

6992	034620	010246	
6993	034622	104411	
6994	034624	012600	
6995	034626	010037	034732
6996	034632	005001	
6997	034634	005002	
6998	034636	112046	
6999	034640	001420	
7000	034642	122716	000060
7001	034646	003026	
7002	034650	122716	000067
7003	034654	002423	
7004	034656	006301	
7005	034660	006102	
7006	034662	006301	
7007	034664	006102	
7008	034666	006301	
7009	034670	006102	
7010	034672	042716	177770
7011	034676	062601	
7012	034700	000756	
7013	034702	005726	
7014	034704	010166	000012
7015	034710	010237	034742
7016	034714	012602	
7017	034716	012601	
7018	034720	012600	
7019	034722	000002	
7020	034724	005726	
7021	034726	105010	
7022	034730	104401	
7023	034732	000000	
7024	034734	104401	001204
7025	034740	000730	
7026	034742	000000	
7027			
7028			
7029			
7030			
7031			
7032			
7033			
7034			
7035			
7036			
7037			
7038	034744	104413	
7039	034746	016601	000002
7040	034752	012705	035063
7041	034756	012704	000014
7042	034762	012703	177770
7043	034766	012100	
7044	034770	012101	
7045	034772	005002	
7046	034774	110245	
7047	034776	010002	

```

1$: MOV R2,-(SP) ;: PUSH R2 ON STACK
RDLIN ;: READ AN ASCIZ LINE
MOV (SP)+,R0 ;: GET ADDRESS OF 1ST CHARACTER
MOV R0,$$ ;: AND SAVE IT
CLR R1 ;: CLEAR DATA WORD
CLR R2

2$: MOVB (R0)+,-(SP) ;: PICKUP THIS CHARACTER
BEQ 3$ ;: IF ZERO GET OUT
CMPB #'0,(SP) ;: MAKE SURE THIS CHARACTER
BGT 4$ ;: IS AN OCTAL DIGIT
CMPB #'7,(SP)
BLT 4$
ASL R1 ;: *2
ROL R2
ASL R1 ;: *4
ROL R2
ASL R1 ;: *8
ROL R2
BIC #'C7,(SP) ;: STRIP THE ASCII JUNK
ADD (SP)+,R1 ;: ADD IN THIS DIGIT
BR 2$ ;: LOOP
3$: TST (SP)+ ;: CLEAN TERMINATOR FROM STACK
MOV R1,12(SP) ;: SAVE THE RESULT
MOV R2,$SHIOCT
MOV (SP)+,R2 ;: POP STACK INTO R2
MOV (SP)+,R1 ;: POP STACK INTO R1
MOV (SP)+,R0 ;: POP STACK INTO R0
RTI ;: RETURN
4$: TST (SP)+ ;: CLEAN PARTIAL FROM STACK
CLRB (R0) ;: SET A TERMINATOR
TYPE ;: TYPE UP THRU THE BAD CHAR.
5$: .WORD 0 ;: "" "CR" & "LF"
TYPE $QUES ;:
BR 1$ ;: TRY AGAIN
$SHIOCT: .WORD 0 ;: HIGH ORDER BITS GO HERE
.SBTTL DOUBLE LENGTH BINARY TO OCTAL ASCII CONVERT ROUTINE

;: *****
;: *THIS ROUTINE WILL CONVERT A 32-BIT UNSIGNED BINARY NUMBER TO AN
;: *UNSIGNED OCTAL ASCIZ NUMBER.
;: *CALL
;: * MOV #PNTR,-(SP) ;: POINTER TO LOW WORD OF BINARY NUMBER
;: * JSR PC,2#$DB20 ;: CALL THE ROUTINE
;: * RETURN ;: THE ADDRESS OF THE FIRST ASCIZ CHAR. IS ON THE STACK

$DB20: SAVREG ;: SAVE ALL REGISTERS
MOV 2(SP),R1 ;: PICKUP THE POINTER TO LOW WORD
MOV #$OCTVL+13.,R5 ;: POINTER TO DATA TABLE
MOV #12.,R4 ;: DO ELEVEN CHARACTERS
MOV #'C7,R3 ;: MASK
MOV (R1)+,R0 ;: LOWER WORD
MOV (R1)+,R1 ;: HIGH WORD
CLR R2 ;: TERMINATOR
1$: MOVB R2,-(R5) ;: PUT CHARACTER IN DATA TABLE
MOV R0,R2 ;: GET THIS DIGIT

```

```

7048 035000 005304          DEC      R4          ;; COUNT THIS CHARACTER
7049 035002 003007          BGT     3$          ;; BR IF NOT THE LAST DIGIT
7050 035004 001405          BEQ     2$          ;; BR IF IT IS THE LAST DIGIT
7051 035006 005205          INC     R5          ;; ALL DIGITS DONE-ADJUST POINTER FOR FIRST
7052 035010 010566 000002   MOV     R5,2(SP)    ;; ASCII CHAR. & PUT IT ON THE STACK
7053 035014 104414          RESREG          ;; RESTORE ALL REGISTERS
7054 035016 000207          RTS     PC          ;; RETURN TO USER
7055 035020 006203          2$: ASR     R3          ;; POSITION THE MASK FOR THE LAST DIGIT
7056 035022 006001          3$: ROR     R1          ;; POSITION THE BINARY NUMBER FOR
7057 035024 006000          ROR     R0          ;; THE NEXT OCTAL DIGIT
7058 035026 006001          ROR     R1
7059 035030 006000          ROR     R0
7060 035032 006001          ROR     R1
7061 035034 006000          ROR     R0
7062 035036 040302          BIC     R3,R2      ;; MASK OUT ALL JUNK
7063 035040 062702 000060   ADD     #'0,R2     ;; MAKE THIS CHAR. ASCII
7064 035044 000753          BR      1$          ;; GO PUT IT IN THE DATA TABLE
7065 035046 000016          $OCTVL: .BLKB 14.  ;; RESERVE DATA TABLE
7066                                     .SBTTL DOUBLE LENGTH BINARY TO DECIMAL ASCII CONVERT ROUTINE
7067
7068                                     ;*****
7069                                     ;*THIS ROUTINE WILL CONVERT A 32-BIT BINARY NUMBER TO AN UNSIGNED
7070                                     ;*DECIMAL (ASCII) NUMBER. THE SIGN OF THE BINARY NUMBER MUST BE
7071                                     ;*POSITIVE.
7072                                     ;*CALL
7073                                     ;*      MOV     #PNTR, -(SP)      ;; POINTER TO LOW WORD OF BINARY NUMBER
7074                                     ;*      JSR     PC, @#$DB2D
7075                                     ;*      RETURN
7076                                     ;; THE FIRST ADDRESS OF ASCIZ
7077                                     ;; IS ON THE STACK
7078
7079 035064 104413          $DB2D: SAVREG      ;; SAVE REGISTERS
7080 035066 016602 000002   MOV     2(SP),R2    ;; PICKUP THE DATA POINTER
7081 035072 012700 035244   MOV     #$DECVL,R0  ;; GET ADDRESS OF "$DECVL" STRING
7082 035076 010066 000002   MOV     R0,2(SP)    ;; PUT ADDRESS OF ASCIZ STRING ON STACK
7083 035102 012201          MOV     (R2)+,R1    ;; PICKUP THE BINARY NUMBER
7084 035104 012202          MOV     (R2)+,R2
7085 035106 012737 000012   MOV     #10,R4      ;; SET UP TO DO 10 CONVERSIONS
7086 035114 012704 035174   MOV     #$TNPWR,R4  ;; ADDRESS OF TEN POWER
7087 035120 012705 035176   MOV     #$TNPWR+2,R5
7088 035124 005003          1$: CLR     R3          ;; CLEAR PARTIAL
7089 035126 161401          2$: SUB     (R4),R1  ;; SUBTRACT TEN POWER
7090 035130 005602          SBC     R2
7091 035132 161502          SUB     (R5),R2
7092 035134 002402          BLT     3$          ;; BR IF TEN POWER TO LARGE
7093 035136 005203          INC     R3          ;; ADD 1 TO PARTIAL
7094 035140 000772          BR      2$          ;; LOOP
7095 035142 062401          3$: ADD     (R4)+,R1  ;; RESTORE SUBTRACTED VALUE
7096 035144 005502          ADC     R2
7097 035146 062402          ADD     (R4)+,R2
7098 035150 022525          CMP     (R5)+,(R5)+ ;; MOVE TO NEXT TEN POWER
7099 035152 052703 000060   BIS     #'0,R3      ;; CHANGE PARTIAL TO ASCII
7100 035156 110320          MOVB   R3,(R0)+    ;; SAVE IT
7101 035160 005327          DEC     (PC)+      ;; DONE?
7102 035162 000000          4$: .WORD 0
7103 035164 001357          BNE    1$          ;; BR IF NO

```



```

7104 035166 105020          CLR8      (R0)+      ;; TERMINATOR
7105 035170 104414          RESREG                     ;; RESTORE REGISTERS
7106 035172 000207          RTS      PC          ;; RETURN
7107 035174 145000          $TNPWR: 145000        ;; 1.0E09
7108 035176 035632          35632
7109 035200 160400          160400          ;; 1.0E08
7110 035202 002765          2765
7111 035204 113200          113200          ;; 1.0E07
7112 035206 000230          230
7113 035210 041100          041100          ;; 1.0E06
7114 035212 000017          17
7115 035214 103240          103240          ;; 1.0E05
7116 035216 000001          1
7117 035220 023420          23420           ;; 1.0E04
7118 035222 000000          0
7119 035224 001750          1750            ;; 1.0E03
7120 035226 000000          0
7121 035230 000144          144             ;; 1.0E02
7122 035232 000000          0
7123 035234 000012          12              ;; 1.0E01
7124 035236 000000          0
7125 035240 000001          1               ;; 1.0E00
7126 035242 000000          0
7127 035244 000014          $DECVL: .BLKB 12.  ;; RESERVE STORAGE FOR ASCIZ STRING
7128                                     .SBTTL SINGLE LENGTH BINARY TO DECIMAL ASCIZ ROUTINE
7129
7130                                     ;*****
7131                                     ;*THIS ROUTINE WILL CONVERT A 16-BIT UNSIGNED BINARY NUMBER TO AN
7132                                     ;*UNSIGNED DECIMAL ASCIZ NUMBER.
7133                                     ;*CALL
7134                                     ;*      MOV      NUMBER, -(SP)      ;; PUT BINARY NUMBER ON THE STACK
7135                                     ;*      JSR      PC, @#$SB2D      ;; CALL
7136                                     ;*      RETURN                      ;; ADDRESS OF THE 1ST ASCIZ CHAR. IS ON THE STACK
7137
7138
7139 035260 016637 000002 035310 $SB2D: MOV      2(SP), 1$      ;; SAVE BINARY NUMBER
7140 035266 012746 035310      MOV      #1$, -(SP)      ;; SET POINTER
7141 035272 004737 035064      JSR      PC, @#$SDB2D    ;; CALL DOUBLE LENGTH CONVERT
7142 035276 062716 000005      ADD      #5, (SP)        ;; ONLY ALLOW FIVE CHARACTERS
7143 035302 012666 000002      MOV      (SP)+, 2(SP)    ;; PICKUP POINTER
7144 035306 000207          RTS      PC          ;; RETURN
7145 035310 000000 000000      1$:      .WORD      0,0
7146                                     .SBTTL TYPE NUMERICAL ASCIZ STRING SUPPRESS LEADING ZEROS
7147
7148                                     ;*****
7149                                     ;*THIS ROUTINE IS USED TO TYPE AN ASCIZ NUMBER SUPPRESSING THE
7150                                     ;*LEADING NUMBERS.
7151                                     ;*CALL
7152                                     ;*      MOV      #NUMADR, -(SP)  ;; FIRST ADDRESS OF ASCIZ STRING
7153                                     ;*      JSR      PC, @#$SUPRS
7154
7155
7156 035314 010046          $SUPRS: MOV      R0, -(SP)      ;; SAVE R0
7157 035316 016600 000004      MOV      4(SP), R0      ;; PICKUP THE POINTER
7158 035322 105710          1$:      TSTB      (R0)        ;; TERMINATEOR?
7159 035324 001403          BEQ      2$            ;; BR IF YES

```

H11

CZR6JED UNIBUS RK6 DR PRT3
CZR6JE.P11 25-JAN-78 12:08

MACY11 30A(1052) 25-JAN-78 12:16 PAGE 137
TYPE NUMERICAL ASCIZ STRING SUPPRESS LEADING ZEROS

SEQ 0137

7160 035326 122720 000060
7161 035332 001773
7162 035334 005300
7163 035336 010037 035344
7164 035342 104401
7165 035344 000000
7166 035346 012600
7167 035350 012616
7168 035352 000207
7169
7170
7171
7172
7173
7174
7175
7176
7177
7178
7179
7180
7181
7182
7183 035354
7184 035354 010046
7185 035356 010046
7186 035360 010246
7187 035362 005046
7188 035364 016601 000012
7189 035370 100002
7190 035372 005216
7191 035374 005401
7192 035376 016602 000014
7193 035402 100002
7194 035404 005316
7195 035406 005402
7196 035410 012746 000021
7197 035414 005000
7198 035416 103001
7199 035420 060200
7200 035422 006000
7201 035424 006001
7202 035426 005316
7203 035430 001372
7204 035432 022616
7205 035434 001403
7206 035436 005400
7207 035440 005401
7208 035442 005600
7209 035444 005726
7210 035446 010066 000012
7211 035452 010166 000010
7212 035456 012602
7213 035460 012601
7214 035462 012600
7215 035464 000207

```

      CMPB    #'0,(RO)+      ;; IS THIS AN ASCII "0" ?
      BEQ     1$              ;; BR IF YES
2$:    DEC     RO              ;; BACKUP BY "1"
      MOV     RO,3$          ;; SAVE FOR TYPING
      TYPE
3$:    .WORD   0              ;; GO TYPE
      MOV     (SP),RO        ;; ASCIZ POINTER GOES HERE
      MOV     (SP)+,(SP)     ;; RESTORE RO
      MOV     (SP)+,(SP)     ;; RESTORE THE STACK
      RTS    PC              ;; RETURN
.SBTTL  INTEGER MULTIPLY ROUTINE
*****
*CALL
*   MOV     MULTIPLIER, -(SP)
*   MOV     MULTIPLICAND, -(SP)
*   JSR     PC, @MULT
*   RETURN  ;; PRODUCT IS ON THE STACK
*
*   STACK  PRODUCT
*   -----
*   TOP    LSB'S
*   +2     MSB'S
$MULT:
      MOV     RO, -(SP)      ;; PUSH RO ON STACK
      MOV     R1, -(SP)      ;; PUSH R1 ON STACK
      MOV     R2, -(SP)      ;; PUSH R2 ON STACK
      CLR     -(SP)          ;; CLEAR THE SIGN KEY
      MOV     12(SP), R1     ;; GET THE MULTIPLICAND
      BPL     1$             ;; BR IF PLUS
      INC     (SP)           ;; SET THE SIGN KEY
      NEG     R1              ;; MAKE THE MULTIPLICAND POSTIVE
1$:    MOV     14(SP), R2     ;; GET THE MULTIPLIER
      BPL     2$             ;; BR IF PLUS
      DEC     (SP)           ;; UPDATE THE SIGN KEY
      NEG     R2              ;; MAKE THE MULTIPLIER POSTIVE
2$:    MOV     #17, -(SP)    ;; SET THE LOOP COUNT
      CLR     RO              ;; SETUP FOR THE MULTIPLY LOOP
3$:    BCC     4$            ;; DON'T ADD IF MULTIPLICAND = 0
      ADD     R2, RO
4$:    ROR     RO              ;; POSITION THE PARTIAL PRODUCT AND
      ROR     R1              ;; THE MULTIPLICAND
      DEC     (SP)           ;; HAS ALL BITS OF THE MULTIPLICAND BEEN DONE?
      BNE     3$            ;; BR IF NO
      CMP     (SP)+, (SP)    ;; SHOULD PRODUCT BE NEGATIVE?
      BEQ     5$            ;; GO TO EXIT IF NO
      NEG     RO              ;; YES--SO MAKE IT SO
      NEG     R1
      SBC     RO
5$:    TST     (SP)+         ;; CLEAR SIGN INFO. OFF OF STACK
      MOV     RO, 12(SP)     ;; PUT THE PRODUCT ON THE STACK (MSB'S)
      MOV     R1, 10(SP)     ;; LSB'S
      MOV     (SP)+, R2     ;; POP STACK INTO R2
      MOV     (SP)+, R1     ;; POP STACK INTO R1
      MOV     (SP)+, RO     ;; POP STACK INTO RO
      RTS    PC

```

7216
7217
7218
7219
7220
7221
7222
7223
7224
7225
7226
7227
7228
7229
7230
7231
7232
7233
7234 035466
7235 035466 010046
7236 035470 010146
7237 035472 010246
7238 035474 010346
7239 035476 010446
7240 035500 010546
7241 035502 016646 000022
7242 035506 016646 000022
7243 035512 016646 000022
7244 035516 016646 000022
7245 035522 000002
7246
7247
7248
7249 035524
7250 035524 012666 000022
7251 035530 012666 000022
7252 035534 012666 000022
7253 035540 012666 000022
7254 035544 012605
7255 035546 012604
7256 035550 012603
7257 035552 012602
7258 035554 012601
7259 035556 012600
7260 035560 000002
7261
7262
7263
7264
7265
7266
7267
7268
7269 035562 010046
7270 035564 016600 000002
7271 035570 005740

.SBTTL SAVE AND RESTORE RO-R5 ROUTINES

```

;*****
;SAVE RO-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:

```

MOV RO,-(SP) ;: PUSH RO ON STACK
MOV R1,-(SP) ;: PUSH R1 ON STACK
MOV R2,-(SP) ;: PUSH R2 ON STACK
MOV R3,-(SP) ;: PUSH R3 ON STACK
MOV R4,-(SP) ;: PUSH R4 ON STACK
MOV R5,-(SP) ;: PUSH R5 ON STACK
MOV 22(SP),-(SP) ;: SAVE PS OF MAIN FLOW
MOV 22(SP),-(SP) ;: SAVE PC OF MAIN FLOW
MOV 22(SP),-(SP) ;: SAVE PS OF CALL
MOV 22(SP),-(SP) ;: SAVE PC OF CALL
RTI

```

;*RESTORE RO-R5

;*CALL:

;* RESREG

\$RESREG:

```

MOV (SP)+,22(SP) ;: RESTORE PC OF CALL
MOV (SP)+,22(SP) ;: RESTORE PS OF CALL
MOV (SP)+,22(SP) ;: RESTORE PC OF MAIN FLOW
MOV (SP)+,22(SP) ;: RESTORE PS OF MAIN FLOW
MOV (SP)+,R5 ;: POP STACK INTO R5
MOV (SP)+,R4 ;: POP STACK INTO R4
MOV (SP)+,R3 ;: POP STACK INTO R3
MOV (SP)+,R2 ;: POP STACK INTO R2
MOV (SP)+,R1 ;: POP STACK INTO R1
MOV (SP)+,R0 ;: POP STACK INTO R0
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.

```

\$TRAP:

```

MOV RO,-(SP) ;: SAVE RO
MOV 2(SP),RO ;: GET TRAP ADDRESS
TST -(RO) ;: BACKUP BY 2

```

```

7272 035572 111000          MOVB    (RO),RO          ;;GET RIGHT BYTE OF TRAP
7273 035574 006300          ASL    RO              ;;POSITION FOR INDEXING
7274 035576 016000 035616    MOV    $TRPAD(PO),RO   ;;INDEX TO TABLE
7275 035602 000200          RTS    RO              ;;GO TO ROUTINE
7276
7277
7278                               ;;THIS IS USE TO HANDLE THE "GETPRI" MACRO
7279
7280 035604 011646 000004 000002 $TRAP2: MOV    (SP),-(SP) ;;MOVE THE PC DOWN
7281 035606 016666          MOV    4(SP),2(SP)    ;;MOVE THE PSW DOWN
7282 035614 000002          RTI                    ;;RESTORE THE PSW
7283
7284 .SBTTL TRAP TABLE
7285
7286 ;*THIS TABLE CONTAINS THE STARTING ADDRESSES OF THE ROUTINES CALLED
7287 ;*BY THE "TRAP" INSTRUCTION.
7288
7289 ;
7290 ; ROUTINE
7291 ; -----
7291 035616 035604 $TRPAD: .WORD $TRAP2
7292 035620 032132 $TYPE   ;;CALL=TYPE      TRAP+1(104401) TTY TYPEOUT ROUTINE
7293 035622 033132 $TYPOC  ;;CALL=TYPOC     TRAP+2(104402) TYPE OCTAL NUMBER (WITH LEADING ZEROS)
7294 035624 033106 $TYPOS  ;;CALL=TYPOS     TRAP+3(104403) TYPE OCTAL NUMBER (NO LEADING ZEROS)
7295 035626 033146 $TYPON  ;;CALL=TYPON     TRAP+4(104404) TYPE OCTAL NUMBER (AS PER LAST CALL)
7296 035630 032414 $TYPDS  ;;CALL=TYPDS     TRAP+5(104405) TYPE DECIMAL NUMBER (WITH SIGN)
7297
7298 035632 033722 $GTSWR  ;;CALL=GTSWR      TRAP+6(104406) GET SOFT-SWR SETTING
7299
7300 035634 033632 $CKSWR  ;;CALL=CKSWR      TRAP+7(104407) TEST FOR CHANGE IN SOFT-SWR
7301 035636 034174 $RDCHR  ;;CALL=RDCHR     TRAP+10(104410) TTY TYPEIN CHARACTER ROUTINE
7302 035640 034264 $RDLIN  ;;CALL=RDLIN     TRAP+11(104411) TTY TYPEIN STRING ROUTINE
7303 035642 034604 $RDOCT  ;;CALL=RDOCT     TRAP+12(104412) READ AN OCTAL NUMBER FROM TTY
7304 035644 035466 $$SAVREG ;;CALL=SAVREG        TRAP+13(104413) SAVE R0-R5 ROUTINE
7305 035646 035524 $RESREG  ;;CALL=RESREG    TRAP+14(104414) RESTORE R0-R5 ROUTINE
7306 035650 030360 $SCOPI$ ;;CALL=SCOPI        TRAP+15(104415) INTERNAL LOOP ON ERROR
7307

```

7308
7309
7310
7311
7312
7313
7314
7315
7316
7317
7318
7319
7320
7321
7322
7323
7324
7325
7326
7327
7328
7329
7330
7331
7332
7333
7334
7335
7336
7337
7338
7339
7340
7341
7342
7343
7344
7345
7346
7347
7348
7349
7350
7351
7352
7353
7354
7355
7356
7357
7358
7359
7360
7361
7362
7363

035652 005015 047125 041111
035660 051525 051040 030113
035666 026466 045522 033460
035674 042040 044522 042526
035702 042040 040511 047107
035710 051517 044524 026503
035716 050055 051101 020124
035724 063
035725 015 041412 051132
035732 045066 030105 005015
035740 005015 025011 025052
035746 025052 041440 052501
035754 044524 047117 025040
035762 025052 025052 005015
035770 005015 044124 051511
035776 050040 047522 051107
036004 046501 051440 047510
036012 046125 020104 041040
036020 020105 040510 052114
03605 042105 047440 046116
036034 020131 054502 052040
036042 050131 047111 020107
036050 047503 052116 047522
036056 026514 103
036061 015 047412 044124
036066 051105 044527 042523
036074 041440 051101 051124
036102 042111 042507 043040
036110 051117 040515 052124
036116 047111 020107 047101
036124 026104 047440 020122
036132 044124 020105 042504
036140 044526 042503
036144 005015 040515 020131
036152 042502 046040 043105
036160 020124 047111 040440
036166 020116 047125 042504
036174 042524 046522 047111
036202 042105 051440 040524
036210 042524 005015
036214 005015 047111 052111
036222 040511 046114 026131
036230 042040 044522 042526
036236 020123 047524 041040
036244 020105 042524 052123
036252 042105 051440 047510
036260 046125 020104 040510
036266 042526 006472 012
036273 015 040412 020056
036300 042510 042101 020123
036306 040515 052516 046101
036314 054514 046040 040517

.SBTTL SERVICE MESSAGES

MSG1: .ASCII <CR><LF>/UNIBUS RK06-RK07 DRIVE DIAGNOSTIC--PART 3/

.ASCII <CR><LF>/CZR6JED/<CR><LF>
.ASCII <CR><LF>/ ***** CAUTION *****/<CR><LF>

.ASCII <CR><LF>/THIS PROGRAM SHOULD BE HALTED ONLY BY TYPING CONTROL-C

.ASCII <CR><LF>/OTHERWISE CARTRIDGE FORMATTING AND, OR THE DEVICE/

.ASCII <CR><LF>/MAY BE LEFT IN AN UNDETERMINED STATE/<CR><LF>

.ASCII <CR><LF>/INITIALLY, DRIVES TO BE TESTED SHOULD HAVE: <CR><LF>

.ASCII <CR><LF>/A. HEADS MANUALLY LOADED

7364	036322	042504	104		
7365	036325	015	041012	020056	.ASCII <CR><LF>/B. CORRECT PORT SELECTED/
7366	036332	047503	051122	041505	
7367	036340	020124	047520	052122	
7368	036346	051440	046105	041505	
7369	036354	042524	104		
7370	036357	015	041412	020056	.ASCII <CR><LF>/C. WRITE LOCK DISABLED/
7371	036364	051127	052111	020105	
7372	036372	047514	045503	042040	
7373	036400	051511	041101	042514	
7374	036406	104			
7375	036407	015	042012	020056	.ASCII <CR><LF>/D. DRIVE READY INDICATOR ON/<CR><LF>
7376	036414	051104	053111	020105	
7377	036422	042522	042101	020131	
7378	036430	047111	044504	040503	
7379	036436	047524	020122	047117	
7380	036444	005015			
7381	036446	005015	051104	053111	.ASCII <CR><LF>/DRIVES NOT TO BE TESTED MUST HAVE/
7382	036454	051505	047040	052117	
7383	036462	052040	020117	042502	
7384	036470	052040	051505	042524	
7385	036476	020104	052515	052123	
7386	036504	044040	053101	105	
7387	036511	015	041012	052117	.ASCIZ <CR><LF>/BOTH PORTS DESELECTED/<CR><LF>
7388	036516	020110	047520	052122	
7389	036524	020123	042504	042523	
7390	036532	042514	052103	042105	
7391	036540	005015	000		
7392					
7393	036543	015	041412	040510	MSG2: .ASCII <CR><LF>/CHANGE XXDP PACK/
7394	036550	043516	020105	054130	
7395	036556	050104	050040	041501	
7396	036564	113			
7397	036565	015	041412	042514	.ASCIZ <CR><LF>/CLEAR LOC 40,RESTART PROGRAM/
7398	036572	051101	046040	041517	
7399	036600	032040	026060	042522	
7400	036606	052123	051101	020124	
7401	036614	051120	043517	040522	
7402	036622	000115			
7403	036624	005015	051104	053111	MSG3: .ASCIZ <CR><LF>/DRIVE(S) TO BE TESTED: .
7404	036632	024105	024523	052040	
7405	036640	020117	042502	052040	
7406	036646	051505	042524	035104	
7407	036654	030040			
7408	036656	005015	054524	042520	MSG4: .ASCIZ <CR><LF>/TYPE BUSS ADDRESS IF NOT 177440: /
7409	036664	041040	051525	020123	
7410	036672	042101	051104	051505	
7411	036700	020123	043111	047040	
7412	036706	052117	030440	033467	
7413	036714	032064	035060	020040	
7414	036722	000			
7415	036723	015	052012	050131	MSG5: .ASCIZ <CR><LF>/TYPE CONTROLLER INTERRUPT VECTOR IF NOT 210:
7416	036730	020105	047503	052116	
7417	036736	047522	046114	051105	
7418	036744	044440	052116	051105	
7419	036752	052522	052120	053040	

7420	036760	041505	047524	020122	
7421	036766	043111	047040	052117	
7422	036774	031040	030061	020072	
7423	037002	000040			
7424	037004	005015	047111	042524	MSG6: .ASCIZ <CR>\LF>/INTERRUPT OCCURRED AT PC=/
7425	037012	051122	050125	020124	
7426	037020	041517	052503	051122	
7427	037026	042105	040440	020124	
7428	037034	041520	000075		
7429	037040	005015	051104	053111	MSG7: .ASCIZ <CR><LF>/DRIVE 0 WILL NOT BE TESTED/
7430	037046	020105	020060	044527	
7431	037054	046114	047040	052117	
7432	037062	041040	020105	042524	
7433	037070	052123	042105	000	
7434	037075	015	044412	052116	MSG8: .ASCIZ <CR><LF>/INTERLOCKS TEST/<CR><LF>
7435	037102	051105	047514	045503	
7436	037110	020123	042524	052123	
7437	037116	005015	000		
7438	037121	015	051012	046505	MSG9: .ASCIZ <CR><LF>/REMOVE UNIT SELECT PLUG/
7439	037126	053117	020105	047125	
7440	037134	052111	051440	046105	
7441	037142	041505	020124	046120	
7442	037150	043525	000		
7443	037153	015	005012	044527	MSG10: .ASCIZ <CR><LF><LF>/WILL TEST DRIVES:/
7444	037160	046114	052040	051505	
7445	037166	020124	051104	053111	
7446	037174	051505	000072		
7447	037200	005015	050012	053517	MSG11: .ASCIZ <CR><LF><LF>/POWER UP RESTART TO TEST 1/<CR><LF>
7448	037206	051105	052440	020120	
7449	037214	042522	052123	051101	
7450	037222	020124	047524	052040	
7451	037230	051505	020124	006461	
7452	037236	000012			
7453	037240	047516	042516	005015	MSG12: .ASCIZ /NONE/<CR><LF>
7454	037246	000			
7455	037247	015	047012	020117	MSG13: .ASCII <CR><LF>/NO L OR P CLOCKS PRESENT/
7456	037254	020114	051117	050040	
7457	037262	041440	047514	045503	
7458	037270	020123	051120	051505	
7459	037276	047105	124		
7460	037301	015	040412	046114	.ASCIZ <CR><LF>/ALL TIMING TESTS BYPASSED/
7461	037306	052040	046511	047111	
7462	037314	020107	042524	052123	
7463	037322	020123	054502	040520	
7464	037330	051523	042105	000	
7465	037335	015	041012	050131	MSG14: .ASCIZ <CR><LF>/BYPASSING DRIVE /
7466	037342	051501	044523	043516	
7467	037350	042040	044522	042526	
7468	037356	000040			
7469	037360	005015	042012	044522	MSG15: .ASCIZ <CR><LF><LF>/DRIVE /
7470	037366	042526	000040		
7471	037372	005015	051104	053111	MSG16: .ASCIZ <CR><LF>/DRIVE SERIAL NO. /
7472	037400	020105	042523	044522	
7473	037406	046101	047040	027117	
7474	037414	000040			
7475	037416	005015	040503	052122	MSG17: .ASCIZ <CR><LF>/CARTRIDGE SERIAL NO. /

7476	037424	044522	043504	020105	
7477	037432	042523	044522	046101	
7478	037440	047040	027117	CJ0040	
7479	037446	005015	047125	052111	MSG18: .ASCIZ <CR><LF>/UNIT SELECT PLUG TEST/<CR><LF>
7480	037454	051440	046105	041505	
7481	037462	020124	046120	043525	
7482	037470	052040	051505	006524	
7483	037476	000012			
7484	037500	005015	047520	052122	MSG19: .ASCIZ <CR><LF>/PORT SELECTION TESTS/<CR><LF>
7485	037506	051440	046105	041505	
7486	037514	044524	047117	052040	
7487	037522	051505	051524	005015	
7488	037530	000			
7489	037531	015	040412	020103	MSG21: .ASCIZ <CR><LF>/AC LOW DETECTION TEST-PART 1/<CR><LF>
7490	037536	047514	020127	042504	
7491	037544	042524	052103	047511	
7492	037552	020116	042524	052123	
7493	037560	050055	01101	020124	
7494	037566	006461	000012		
7495	037572	005015	047516	026516	MSG22: .ASCIZ <CR><LF>/NON-EXECUTABLE FUNCTION (NXF) DETECTION TEST/<CR><LF>
7496	037600	054105	041505	052125	
7497	037606	041101	042514	043040	
7498	037614	047125	052103	047511	
7499	037622	020116	047050	043130	
7500	037630	020051	042504	042524	
7501	037636	052103	047511	020116	
7502	037644	042524	052123	005015	
7503	037652	000			
7504	037653	015	053412	044522	MSG24: .ASCIZ <CR><LF>/WRITE LOCK TEST/<CR><LF>
7505	037660	042524	046040	041517	
7506	037666	020113	042524	052123	
7507	037674	005015	000		
7508	037677	015	040412	020103	MSG26: .ASCIZ <CR><LF>/AC LOW DETECTION TEST-PART 2/<CR><LF>
7509	037704	047514	020127	042504	
7510	037711	042524	052103	047511	
7511	037720	020116	042524	052123	
7512	037726	050055	051101	020124	
7513	037734	006462	000012		
7514	037740	005015	040412	046114	MSG27: .ASCIZ <CR><LF><LF>/ALL DRIVES TESTED/<CR><LF><LF>
7515	037746	042040	044522	042526	
7516	037754	020123	042524	052123	
7517	037762	042105	005015	000012	
7518	037770	005015	052515	052114	MSG28: .ASCIZ <CR><LF>/MULTIPLE DRIVE DETECTION TEST/<CR><LF>
7519	037776	050111	042514	042040	
7520	040004	044522	042526	042040	
7521	040012	052105	041505	044524	
7522	040020	047117	052040	051505	
7523	040026	006524	000012		
7524	040032	005015	046120	040505	MSG29: .ASCIZ <CR><LF>/PLEASE WAIT, HEADS BEING LOADED/<CR><LF>
7525	040040	042523	053440	044501	
7526	040046	026124	044040	040505	
7527	040054	051504	041040	044505	
7528	040062	043516	046040	040517	
7529	040070	042504	006504	000012	
7530	040076	005015	042526	044522	MSG30: .ASCIZ <CR><LF>/VERIFY DOOR CAN NOW BE OPENED & LEAVE IT OPEN/<CR><LF>
7531	040104	054506	042040	047517	

7532	040112	020122	040503	020116	
7533	040120	047516	020127	042502	
7534	040126	047440	042520	042516	
7535	040134	020104	020046	042514	
7536	040142	053101	020105	052111	
7537	040150	047440	042520	006516	
7538	040156	000012			
7539	040160	005015	006477	000012	MSG31: .ASCIZ <CR><LF>/?<CR><LF>
7540	040166	005015	042522	047515	MSG32: .ASCIZ <CR><LF>/REMOVE DISK PACK & CLOSE DOOR/
7541	040174	042526	042040	051511	
7542	040202	020113	040520	045503	
7543	040210	023040	041440	047514	
7544	040216	042523	042040	047517	
7545	040224	000122			
7546	040226	005015	042504	051120	MSG33: .ASCIZ <CR><LF>/DEPRESS 'RUN-STOP' SWITCH TO 'RUN' WHILE DOOR OPEN/
7547	040234	051505	020123	051047	
7548	040242	047125	051455	047524	
7549	040250	023520	051440	044527	
7550	040256	041524	020110	047524	
7551	040264	023440	052522	023516	
7552	040272	053440	044510	042514	
7553	040300	042040	047517	020122	
7554	040306	050117	047105	000	
7555	040313	015	053012	051105	MSG34: .ASCIZ <CR><LF>/VERIFY SPINDLE DOES NOT START & HEADS DO NOT LOAD/<CR><LF>
7556	040320	043111	020131	050123	
7557	040326	047111	046104	020105	
7558	040334	047504	051505	047040	
7559	040342	052117	051440	040524	
7560	040350	052122	023040	044040	
7561	040356	040505	051504	042040	
7562	040364	020117	047516	020124	
7563	040372	047514	042101	005015	
7564	040400	000			
7565	040401	015	042012	050105	MSG35: .ASCIZ <CR><LF>/DEPRESS 'RUN-STOP' SWITCH TO 'RUN' WHILE CARTRIDGE REMOVED/
7566	040406	042522	051523	023440	
7567	040414	052522	026516	052123	
7568	040422	050117	020047	053523	
7569	040430	052111	044103	052040	
7570	040436	020117	051047	047125	
7571	040444	020047	044127	046111	
7572	040452	020105	040503	052122	
7573	040460	044522	043504	020105	
7574	040466	042522	047515	042526	
7575	040474	000104			
7576	040476	005015	047111	042523	MSG36: .ASCIZ <CR><LF>/INSERT DISK PACK, DEPRESS 'RUN-STOP' TO 'RUN' & CLOSE DOOR/
7577	040504	052122	042040	051511	
7578	040512	020113	040520	045503	
7579	040520	020054	042504	051120	
7580	040526	051505	020123	051047	
7581	040534	047125	051455	047524	
7582	040542	023520	052040	020117	
7583	040550	051047	047125	020047	
7584	040556	020046	046103	051517	
7585	040564	020105	047504	051117	
7586	040572	000			
7587	040573	015	042012	050105	MSG37: .ASCIZ <CR><LF>/DEPRESS SPACE BAR WHEN FINISHED/<CR><LF>

7588	040600	042522	051523	051440	
7589	040606	040520	042503	041040	
7590	040614	051101	053440	042510	
7591	040622	020116	044506	044516	
7592	040630	044123	042105	005015	
7593	040636	000			
7594	040637	015	044412	051516	MSG3P .ASCII <CR><LF>/INSERT UNIT SELECT PLUGS, IN ANY ORDER/
7595	040644	051105	020124	047125	
7596	040652	052111	051440	046105	
7597	040660	041505	020124	046120	
7598	040666	043525	026123	044440	
7599	040674	020116	047101	020131	
7600	040702	051117	042504	122	
7601	040707	015	052012	042510	.ASCIZ <CR><LF>/THE PROGRAM WILL ECHO THE UNIT SELECT PLUG NUMBER/<CR><LF>
7602	040714	050040	047522	051107	
7603	040722	046501	053440	046111	
7604	040730	020114	041505	047510	
7605	040736	052040	042510	052440	
7606	040744	044516	020124	042523	
7607	040752	042514	052103	050040	
7608	040760	052514	020107	052516	
7609	040766	041115	051105	005015	
7610	040774	000			
7611	040775	015	042012	050105	MSG39: .ASCIZ <CR><LF>/DEPRESS CONTROL-E TO EXIT TEST/<CR><LF>
7612	041002	042522	051523	041440	
7613	041010	047117	051124	046117	
7614	041016	042455	052040	020117	
7615	041024	054105	052111	052040	
7616	041032	051505	006524	000012	
7617	041040	005015	047526	052514	MSG40: .ASCII <CR><LF>/VOLUME VALID NOT SET/
7618	041046	042515	053040	046101	
7619	041054	042111	047040	052117	
7620	041062	051440	052105		
7621	041066	005015	040515	042513	.ASCIZ <CR><LF>/MAKE SURE ORIGINAL UNIT SELECT PLUG IS INSERTED/<CR><LF>
7622	041074	051440	051125	020105	
7623	041102	051117	043511	047111	
7624	041110	046101	052440	044516	
7625	041116	020124	042523	042514	
7626	041124	052103	050040	052514	
7627	041132	020107	051511	044440	
7628	041140	051516	051105	042524	
7629	041146	006504	000012		
7630	041152	005015	042504	042523	MSG41: .ASCIZ <CR><LF>/DESELECT PORT IN USE & SELECT OPPOSITE PORT/<CR><LF>
7631	041160	042514	052103	050040	
7632	041166	051117	020124	047111	
7633	041174	052440	042523	023040	
7634	041202	051440	046105	041505	
7635	041210	020124	050117	047520	
7636	041216	044523	042524	050040	
7637	041224	051117	006524	000012	
7638	041232	005015	042504	042523	MSG42: .ASCIZ <CR><LF>/DESELECT WRONG PORT & SELECT CORRECT PORT/<CR><LF>
7639	041240	042514	052103	053440	
7640	041246	047522	043516	050040	
7641	041254	051117	020124	020046	
7642	041262	042523	042514	052103	
7643	041270	041440	051117	042522	

7644	041276	052103	050040	051117	
7645	041304	006524	000012		
7646	041310	005015	042504	042523	MSG43: .ASCIZ <CR><LF>/DESELECT BOTH PORTS/<CR><LF>
7647	041316	042514	052103	041040	
7648	041324	052117	020110	047520	
7649	041332	052122	006523	000012	
7650	041340	005015	042523	042514	MSG44: .ASCIZ <CR><LF>/SELECT CORRECT PORT/<CR><LF>
7651	041346	052103	041440	051117	
7652	041354	042522	052103	050040	
7653	041362	051117	006524	000012	
7654	041370	005015	047503	051122	MSG45: .ASCIZ <CR><LF>/CORRECT PORT NOT SELECTED, TRY AGAIN/<CR><LF>
7655	041376	041505	020124	047520	
7656	041404	052122	047040	052117	
7657	041412	051440	046105	041505	
7658	041420	042524	026104	052040	
7659	041426	054522	040440	040507	
7660	041434	047111	005015	000	
7661	041441	015	053012	051105	MSG47: .ASCIZ <CR><LF>/VERIFY DOOR CANNOT BE OPENED (DO NOT FORCE)/<CR><LF>
7662	041446	043111	020131	047504	
7663	041454	051117	041440	047101	
7664	041462	047516	020124	042502	
7665	041470	047440	042520	042516	
7666	041476	020104	042050	020117	
7667	041504	047516	020124	047506	
7668	041512	041522	024505	005015	
7669	041520	000			
7670	041521	015	052012	051125	MSG49: .ASCIZ <CR><LF>/TURN OFF AC POWER FROM BEHIND THE RK06/<CR><LF>
7671	041526	020116	043111	020106	
7672	041534	041501	050040	053517	
7673	041542	051105	043040	047522	
7674	041550	020115	042502	044510	
7675	041556	042116	052040	042510	
7676	041564	051040	030113	006466	
7677	041572	000012			
7678	041574	005015	040527	052111	MSG50: .ASCIZ <CR><LF>/WAIT 30 SEC & SWITCH AC POWER BACK ON/
7679	041602	031440	020060	042523	
7680	041610	020103	020046	053523	
7681	041616	052111	044103	040440	
7682	041624	020103	047520	042527	
7683	041632	020122	040502	045503	
7684	041640	047440	000116		
7685	041644	005015	047515	042515	MSG51: .ASCII <CR><LF>/MOMENTARILY REMOVE & INSERT THE SAME UNIT SELECT PLUG/
7686	041652	052116	051101	046111	
7687	041660	020131	042522	047515	
7688	041666	042526	023040	044440	
7689	041674	051516	051105	020124	
7690	041702	044124	020105	040523	
7691	041710	042515	052440	044516	
7692	041716	020124	042523	042514	
7693	041724	052103	050040	052514	
7694	041732	107			
7695	041733	015	052012	020117	.ASCIZ <CR><LF>/TO RESET VOLUME VALID/<CR><LF>
7696	041740	042522	042523	020134	
7697	041746	047526	052514	042515	
7698	041754	053040	046101	042111	
7699	041762	005015	000		

7700	04 765	015	042012	050105	MSG52: .ASCII <CR><LF>/DEPRESS SPACE TO DO TEST/
7701	04 772	042522	051523	051440	
7702	042000	040520	042503	052040	
7703	042006	020117	047504	052040	
7704	042014	051505	124		
7705	042017	015	047412	020122	.ASCIZ <CR><LF>/OR CONTROL-E TO BYPASS ENTIRE TEST/<CR><LF>
7706	042024	047503	052116	047522	
7707	042032	026514	020105	047524	
7708	042040	041040	050131	051501	
7709	042046	020123	047105	044524	
7710	042054	042522	052040	051505	
7711	042062	006524	000012		
7712	042066	005015	044504	040523	MSG53: .ASCIZ <CR><LF>/DISABLE THE WRITE LOCK SWITCH, VERIFY LIGHT GOES OFF/<CR><LF>
7713	042074	046102	020105	044124	
7714	042102	020105	051127	052111	
7715	042110	020105	047514	045503	
7716	042116	051440	044527	041524	
7717	042124	026110	053040	051105	
7718	042132	043111	020131	044514	
7719	042140	044107	020124	047507	
7720	042146	051505	047440	043106	
7721	042154	005015	000		
7722	042157	015	042412	040516	MSG54: .ASCIZ <CR><LF>/ENABLE THE WRITE LOCK SWITCH, VERIFY LIGHT GOES ON/<CR><LF>
7723	042164	046102	020105	044124	
7724	042172	020105	051127	052111	
7725	042200	020105	047514	045503	
7726	042206	051440	044527	041524	
7727	042214	026110	053040	051105	
7728	042222	043111	020131	044514	
7729	042230	044107	020124	047507	
7730	042236	051505	047440	006516	
7731	042244	000012			
7732	042246	005015	054105	052111	MSG55: .ASCIZ <CR><LF>/EXIT TEST WITH ORIGINAL UNIT SELECT PLUG NO. /
7733	042254	052040	051505	020124	
7734	042262	044527	044124	047440	
7735	042270	044522	044507	040516	
7736	042276	020114	047125	052111	
7737	042304	051440	046105	041505	
7738	042312	020124	046120	043525	
7739	042320	047040	027117	000040	
7740	042326	005015	046101	043511	MSG56: .ASCII <CR><LF>/ALIGNMENT CARTRIDGE USED/
7741	042334	046516	047105	020124	
7742	042342	040503	052122	044522	
7743	042350	043504	020105	051525	
7744	042356	042105			
7745	042360	005015	051120	043517	.ASCII <CR><LF>/PROGRAM WILL BYPASS THE WRITE LOCK TEST/
7746	042366	040522	020115	044527	
7747	042374	046114	041040	050131	
7748	042402	051501	020123	044124	
7749	042410	020105	051127	052111	
7750	042416	020105	047514	045503	
7751	042424	052040	051505	124	
7752	042431	015	040412	042116	.ASCIZ <CR><LF>/AND READ-WRITE DATA PORTION OF AC LOW DETECTION TEST-PART 2.<CR>
7753	042436	051040	040505	026504	
7754	042444	051127	052111	020105	
7755	042452	040504	040524	050040	

7756	042460	051117	044524	047117	
7757	042466	047440	020106	041501	
7758	042474	046040	053517	042040	
7759	042502	052105	041505	044524	
7760	042510	047117	052040	051505	
7761	042516	026524	040520	052122	
7762	042524	031040	075015	000012	
7763	042532	005015	07526	044522	MSG57: .ASCIZ <CR><LF>/VERIFY BATTERY RETRACT FUNCTIONAL/<CR><LF>
7764	042540	054506	041040	052101	
7765	042546	042524	054522	051040	
7766	042554	052105	040522	052103	
7767	042562	043040	047125	052103	
7768	042570	047511	040516	006514	
7769	042576	000012			
7770	042600	005015	047514	042101	MSG58: .ASCIZ <CR><LF>/LOAD HEADS ON ALL DRIVES TO BE TESTED FOR MDS/<CR><LF>
7771	042606	044040	040505	051504	
7772	042614	047440	020116	046101	
7773	042622	020114	051104	053111	
7774	042630	051505	052040	020117	
7775	042636	042502	052040	051505	
7776	042644	042524	070104	047506	
7777	042652	020122	042115	006523	
7778	042660	000012			
7779	042662	005015	047111	042523	MSG59: .ASCIZ <CR><LF>/INSERT SAME UNIT SELECT PLUG NUMBER IN ANY 2 DRIVES TO BE TESTE
7780	042670	052122	071440	046501	
7781	042676	020105	07125	052111	
7782	042704	051440	046105	041505	
7783	042712	020124	046120	043525	
7784	042720	047040	046525	042502	
7785	042726	020122	047111	040440	
7786	042734	054516	031040	042040	
7787	042742	044522	042526	020123	
7788	042750	047524	041040	020105	
7789	042756	042524	052123	042105	
7790	042764	000			
7791	042765	015	044412	051516	MSG60: .ASCII <CR><LF>/INSERT CORRECT UNIT SELECT PLUGS & LOAD HEADS/
7792	042772	051105	020124	047503	
7793	043000	051122	041505	020124	
7794	043006	047125	052111	051440	
7795	043014	046105	041505	020124	
7796	043022	046120	043525	020123	
7797	043030	020046	047514	042101	
7798	043036	044040	040505	051504	
7799	043044	005015	047117	050040	.ASCIZ <CR><LF>/ON PREVIOUS 2 DRIVES/<CR><LF>
7800	043052	042522	044526	052517	
7801	043060	020123	020062	051104	
7802	043066	053111	051505	005015	
7803	043074	000			
7804	043075	015	046412	046125	MSG61: .ASCIZ <CR><LF>/MULTIPLE DRIVES FOUND ON DRIVE NO. /
7805	043102	044524	046120	020105	
7806	043110	051104	053111	051505	
7807	043116	043040	052517	042116	
7808	043124	047440	020116	051104	
7809	043132	053111	020105	047516	
7810	043140	020056	000		
7811	043143	015	041012	050131	MSG62: .ASCII <CR><LF>/BYPASSING MULT. DRIVE SELECT TEST/

7812	043150	051501	044523	043516	
7813	043156	046440	046125	027124	
7814	043164	042040	044522	042526	
7815	043172	051440	046105	041505	
7816	043200	020124	042524	052123	
7817	043206	005015	047117	054514	.ASCIZ <CR><LF>/ONLY 1 DRIVE PRESENT/<CR><LF>
7818	043214	030440	042040	044522	
7819	043222	042526	050040	042522	
7820	043230	042523	052116	005015	
7821	043236	000			
7822	043237	015	042012	050105	MSG63: .ASCII <CR><LF>/DEPRESS SPACE BAR TO DO ANOTHER 2 DRIVES/
7823	043244	042522	051523	051440	
7824	043252	040520	042503	041040	
7825	043260	051101	052040	020117	
7826	043266	047504	040440	047516	
7827	043274	044124	051105	031040	
7828	043302	042040	044522	042526	
7829	043310	123			
7830	043311	015	047412	020122	.ASCII <CR><LF>/OR CONTROL-E TO EXIT TEST/
7831	043316	047503	052116	047522	
7832	043324	026514	020105	047524	
7833	043332	042440	044530	020124	
7834	043340	042524	052123		
7835	043344	005015	044450	051516	.ASCIZ <CR><LF>/((INSERT CORRECT UNIT SELECT PLUGS BEFORE EXITING)/<CR><LF>
7836	043352	051105	020124	047503	
7837	043360	051122	041505	020124	
7838	043366	047125	052111	051440	
7839	043374	046105	041505	020124	
7840	043402	046120	043525	020123	
7841	043410	042502	047506	042522	
7842	043416	042440	044530	044524	
7843	043424	043516	006451	000012	
7844	043432	005015	042504	051120	MSG65: .ASCIZ <CR><LF>/DEPRESS 'RUN-STOP' SWITCH TO 'STOP'/'
7845	043440	051505	020123	051047	
7846	043446	047125	051455	047524	
7847	043454	023520	051440	044527	
7848	043462	041524	020110	047524	
7849	043470	023440	052123	050117	
7850	043476	000047			
7851	043500	005015	042504	042523	MSG67: .ASCIZ <CR><LF>/DESELECT PORT SWITCH ON ALL OTHER DRIVES/<CR><LF>
7852	043506	042514	052103	050040	
7853	043514	051117	020124	053523	
7854	043522	052111	044103	047440	
7855	043530	020116	046101	020114	
7856	043536	052117	042510	020122	
7857	043544	051104	053111	051505	
7858	043552	005015	000		
7859	043555	015	053012	051105	MSG68: .ASCIZ <CR><LF>/VERIFY BOTH DRIVES UNLOADED/<CR><LF>
7860	043562	043111	020131	047502	
7861	043570	044124	042040	044522	
7862	043576	042526	020123	047125	
7863	043604	047514	042101	042105	
7864	043612	005015	000		
7865	043615	015	042012	050105	MSG69: .ASCIZ <CR><LF>/DEPRESS SPACE WHEN 'READY' LIGHT GOES ON/<CR><LF>
7866	043622	042522	051523	051440	
7867	043630	040520	042503	053440	

7868	0433636	042510	020116	051047	
7869	0433644	040505	054504	020047	
7870	0433652	044514	044107	020124	
7871	0433660	047507	051505	047440	
7872	0433666	006516	000012		
7873	0433672	005015	042526	044522	MSG70: .ASCIZ <CR><LF>/VERIFY HEADS LOAD/<CR><LF>
7874	0433700	054506	044040	040505	
7875	0433706	051504	046040	040517	
7876	0433714	006504	000012		
7877	0433720	005015	042523	042514	MSG71: .ASCIZ <CR><LF>/SELECT CORRECT PORT SWITCH ON ALL OTHER DRIVES/<CR><LF>
7878	0433726	052103	041440	051117	
7879	0433734	042522	052103	050040	
7880	0433742	051117	020124	053523	
7881	0433750	052111	044103	047440	
7882	0433756	020116	046101	020114	
7883	0433764	052117	042510	020122	
7884	0433772	051104	053111	051505	
7885	044000	005015	000		
7886	044003	015	040412	047502	MSG72: .ASCIZ <CR><LF>/ABORTING BALANCE OF TESTS/
7887	044010	052122	047111	020107	
7888	044016	040502	040514	041516	
7889	044024	020105	043117	052040	
7890	044032	051505	051524	000	
7891					
7892	044037	015	050012	047522	MSG74: .ASCIZ <CR><LF>/PROGRAM ABORT PENDING...PLEASE WAIT/
7893	044044	051107	046501	040440	
7894	044052	047502	052122	050040	
7895	044060	047105	044504	043516	
7896	044066	027056	050056	042514	
7897	044074	051501	020105	040527	
7898	044102	052111	000		
7899	044105	015	044012	046101	MSG75: .ASCIZ <CR><LF>/HALT PENDING...PLEASE WAIT/
7900	044112	020124	042520	042116	
7901	044120	047111	027107	027056	
7902	044126	046120	040505	042523	
7903	044134	053440	044501	000124	
7904	044142	005015	051120	043517	MSG76: .ASCIZ <CR><LF>/PROGRAM ABORTED/
7905	044150	040522	020115	041101	
7906	044156	051117	042524	000104	
7907	044164	005015	050103	020125	MSG77: .ASCIZ <CR><LF>/CPU HALTED/
7908	044172	040510	052114	042105	
7909	044200	000			
7910	044201	015	051412	051117	MSG100: .ASCII <CR><LF>/SORRY, WRITE LOCK SHOULD NOT BE DEPRESSED/
7911	044206	054522	020054	051127	
7912	044214	052111	020105	047514	
7913	044222	045503	051440	047510	
7914	044230	046125	020104	047516	
7915	044236	020124	042502	042040	
7916	044244	050105	042522	051523	
7917	044252	042105			
7918	044254	005015	044127	046111	.ASCII <CR><LF>/WHILE ON TRACK 2 SECTORS 19, 20 OR 21/
7919	044262	020105	047117	052040	
7920	044270	040522	045503	031040	
7921	044276	051440	041505	047524	
7922	044304	051522	030440	026071	
7923	044312	031040	020060	051117	

7924	044320	031040	061		
7925	044323	015	050012	042514	.ASCIZ <CR><LF>/PLEASE TRY AGAIN/<CR><LF>
7926	044330	051501	020105	051124	
7927	044336	020131	043501	044501	
7928	044344	006516	000012		
7929					
7930					.SBTTL ERROR MESSAGES
7931					
7932	044350	005015	051105	047522	EM1: .ASCIZ <CR><LF>/ERROR, ONLY 0 THRU 7 ALLOWED, TRY AGAIN/<CR><LF>
7933	044356	026122	047440	046116	
7934	044364	020131	020060	044124	
7935	044372	052522	033440	040440	
7936	044400	046114	053517	042105	
7937	044406	020054	051124	020131	
7938	044414	043501	044501	006516	
7939	044422	000012			
7940	044424	051104	053111	020105	EM2: .ASCIZ /DRIVE # IN RKCS2 CANNOT BE READ BACK CORRECTLY IN RKMR2/
7941	044432	020043	047111	051040	
7942	044440	041513	031123	041440	
7943	044446	047101	047516	020124	
7944	044454	042502	051040	040505	
7945	044462	020104	040502	045503	
7946	044470	041440	051117	042522	
7947	044476	052103	054514	044440	
7948	044484	020116	045522	051115	
7949	044512	000062			
7950	044514	005015	041101	051117	EM3: .ASCIZ <CR><LF>/ABORT TESTS...UNEXPECTED TIME OUT AT PC=/
7951	044522	020124	042524	052123	
7952	044530	027123	027056	047125	
7953	044536	054105	042520	052103	
7954	044544	042105	052040	046511	
7955	044552	020105	052517	020124	
7956	044560	052101	050040	036503	
7957	044566	000			
7958	044567	015	040412	047502	EM4: .ASCIZ <CR><LF>/ABORT TESTS...UNEXPECTED INTERRUPT AT PC=/
7959	044574	052122	052040	051505	
7960	044602	051524	027056	052456	
7961	044610	042516	050130	041505	
7962	044616	042524	020104	047111	
7963	044624	042524	051122	050125	
7964	044632	020124	052101	050040	
7965	044640	036503	000		
7966	044643	115	051504	051440	EM5: .ASCIZ /MDS SET IN RKCS2/
7967	044650	052105	044440	020116	
7968	044656	045522	051503	000062	
7969	044664	043125	020105	042523	EM6: .ASCIZ /UFE SET IN RKCS2/
7970	044672	020124	047111	051040	
7971	044700	041513	031123	000	
7972	044705	104	040522	044440	EM7: .ASCIZ /DRA IN RKDS & NED IN RKCS2 RESET; WRONG PORT SELECTED^/
7973	044712	020116	045522	051504	
7974	044720	023040	047040	042105	
7975	044726	044440	020116	045522	
7976	044734	051503	020062	042522	
7977	044742	042523	035524	053440	
7978	044750	047522	043516	050040	
7979	044756	051117	020124	042523	

7980	044764	042514	052103	042105	
7981	044772	0C J077			
7982	044774	051104	053111	020105	EM8: .ASCIZ /DRIVE PRESENT BUT NOT SPECIFIED BY OPERATOR/
7983	045002	051120	051505	047105	
7984	045010	020124	052502	020124	
7985	045016	047516	020124	050123	
7986	045024	041505	043111	042511	
7987	045032	020104	054502	047440	
7988	045040	042520	040522	047524	
7989	045046	000122			
7990	045050	051104	053111	020105	EM9: .ASCIZ /DRIVE NOT PRESENT BUT SPECIFIED BY OPERATOR/
7991	045056	047516	020124	051120	
7992	045064	051505	047105	020124	
7993	045072	052502	020124	050123	
7994	045100	041505	043111	042511	
7995	045106	020104	054502	047440	
7996	045114	042520	040522	047524	
7997	045122	000122			
7998	045124	041101	051117	020124	EM10: .ASCIZ /ABORT TESTS...CANNOT REFERENCE CONTROLLER REGISTER/
7999	045132	042524	052123	027123	
8000	045140	027056	040503	047116	
8001	045146	052117	051040	043105	
8002	045154	051105	047105	042503	
8003	045162	041440	047117	051124	
8004	045170	046117	042514	020122	
8005	045176	042522	044507	052123	
8006	045204	051105	000		
8007	045207	104	040522	044440	EM11: .ASCIZ /DRA IN RKDS & NED IN RKCS2 BOTH SET/
8008	045214	020116	045522	051504	
EJ09	045222	023040	047040	042105	
8010	045230	044440	020116	045522	
8011	045236	051503	020062	047502	
8012	045244	044124	051440	052105	
8013	045252	000			
8014	045253	103	047117	051124	EM12: .ASCIZ /CONTROLLER NOT READY IN RKCS1/
8015	045260	046117	042514	020122	
8016	045266	047516	020124	042522	
8017	045274	042101	020131	047111	
8018	045302	051040	041513	030523	
8019	045310	000			
8020	045311	116	020117	052101	EM13: .ASCIZ /NO ATTN IN RKASOF/
8021	045316	047124	044440	020116	
8022	045324	045522	051501	043117	
8023	045332	000			
8024	045333	127	047522	043516	EM14: .ASCIZ /WRONG ATTN IN RKASOF/
8025	045340	040440	052124	020116	
8026	045346	047111	051040	040513	
8027	045354	047523	000106		
8028	045360	051104	054504	047040	EM15: .ASCIZ /DRDY NOT CLEARED IN RKMR2/
8029	045366	052117	041440	042514	
8030	045374	051101	042105	044440	
8031	045402	020116	045522	051115	
8032	045410	000062			
8033	045412	051504	020103	047516	EM16: .ASCIZ /DSC NOT SET IN RKMR2/
8034	045420	020124	042523	020124	
8035	045426	047111	051040	046513	

8036	045424	031122	000		
8037	045437	115	051505	040523	EM17: .ASCIZ /MESSAGE A0 ERROR/
8038	045444	042507	040440	020060	
8039	045452	051105	047522	000122	
8040	045460	042515	051523	043501	EM18: .ASCIZ /MESSAGE B0 ERROR/
8041	045466	020105	030102	042440	
8042	045474	051122	051117	000	
8043	045501	115	051505	040523	EM19: .ASCIZ /MESSAGE A1 ERROR/
8044	045506	042507	040440	020061	
8045	045514	051105	047522	000122	
8046	045522	042515	051523	043501	EM20: .ASCIZ /MESSAGE B1 ERROR/
8047	045530	020105	030502	042440	
8048	045536	051122	051117	000	
8049	045543	103	051105	020122	EM21: .ASCIZ /CERR SET IN RKCS1/
8050	045550	042523	020124	047111	
8051	045556	051040	041513	030523	
8052	045564	000			
8053	045565	104	047517	020122	EM22: .ASCIZ /DOOR STATUS IN RKMR2 NOT CLEARED/
8054	045572	052123	052101	051525	
8055	045600	044440	020116	045522	
8056	045606	051115	020062	047516	
8057	045614	020124	046103	040505	
8058	045622	042522	000104		
8059	045626	050123	047111	046104	EM23: .ASCIZ /SPINDLE ON SET IN RKMR2/
8060	045634	020105	047117	051440	
8061	045642	052105	044440	020116	
8062	045650	045522	051115	000062	
8063	045656	047526	052514	042515	EM24: .ASCIZ /VOLUME VALID NOT SET IN RKMR2/
8064	045664	053040	046101	042111	
8065	045672	047040	052117	051440	
8066	045700	052105	044440	020116	
8067	045706	045522	051115	000062	
8068	045714	040503	052122	044522	EM25: .ASCIZ /CARTRIDGE STATUS IN RKMR2 NOT CLEARED/
8069	045722	043504	020105	052123	
8070	045730	052101	051525	044440	
8071	045736	020116	045522	051115	
8072	045744	020062	047516	020124	
8073	045752	046103	040505	042522	
8074	045760	000104			
8075	045762	047526	052514	042515	EM26: .ASCIZ /VOLUME VALID NOT CLEARED IN RKMR2/
8076	045770	053040	046101	042111	
8077	045776	047040	052117	041440	
8078	046004	042514	051101	042105	
8079	046012	044440	020116	045522	
8080	046020	051115	000062		
8081	046024	042516	020104	047516	EM27: .ASCIZ /MED NOT SET IN RKCS2/
8082	046032	020124	042523	020124	
8083	046040	047111	051040	041513	
8084	046046	031123	000		
8085	046051	126	046117	046525	EM28: .ASCIZ /VOLUME VALID SET IN RKMR2/
8086	046056	020105	040526	044514	
8087	046064	020104	042523	020124	
8088	046072	047111	051040	046513	
8089	046100	031122	000		
8090	046103	104	041523	047040	EM29: .ASCIZ /DSC NOT SET IN RKMR2/
8091	046110	052117	051440	052105	

8092	046116	044440	020116	045522		
8093	046124	051115	000062			
8094	046130	052101	047124	047040	EM30:	.ASCIZ /ATTN NOT RESET IN RKASOF/
8095	046136	052117	051040	051505		
8096	046144	052105	044440	020116		
8097	046152	045522	051501	043117		
8098	046160	000				
8099	046161	116	042105	047040	EM31:	.ASCIZ /NED NOT CLEARED IN RKCS2/
8100	046166	052117	041440	042514		
8101	046174	051101	042105	044440		
8102	046202	020116	045522	051503		
8103	046210	000062				
8104	046212	050123	047111	046104	EM32:	.ASCIZ /SPINDLE ON NOT SET IN RKMR2/
8105	046220	020105	047117	047040		
8106	046226	052117	051440	052105		
8107	046234	044440	020116	045522		
8108	046242	051115	000062			
8109	046246	051104	053111	020105	EM33:	.ASCIZ /DRIVE NOT READY IN RKMR2/
8110	046254	047516	020124	042522		
8111	046262	042101	020131	047111		
8112	046270	051040	046513	031122		
8113	046276	000				
8114	046277	104	047517	020122	EM34:	.ASCIZ /DOOR STATUS BIT NOT SET IN RKMR2/
8115	046304	052123	052101	051525		
8116	046312	041040	052111	047040		
8117	046320	052117	051440	052105		
8118	046326	044440	020116	045522		
8119	046334	051115	000062			
8120	046340	042510	042101	020123	EM35:	.ASCIZ /HEADS HOME NOT SET IN RKMR2/
8121	046346	047510	042515	047040		
8122	046354	052117	051440	052105		
8123	046362	044440	020116	045522		
8124	046370	051115	000062			
8125	046374	041501	046040	053517	EM37:	.ASCIZ /AC LOW NOT SET IN RKMR3/
8126	046402	047040	052117	051440		
8127	046410	052105	044440	020116		
8128	046416	045522	051115	000063		
8129	046424	042516	020104	047516	EM42:	.ASCIZ /NED NOT SET IN RKCS2/
8130	046432	020124	042523	020124		
8131	046440	047111	051040	041513		
8132	046446	031123	000			
8133	046451	101	046103	020117	EM43:	.ASCIZ /ACLO NOT CLEARED IN RKMR3/
8134	046456	047516	020124	046103		
8135	046464	040505	042522	020104		
8136	046472	047111	051040	046513		
8137	046500	031522	000			
8138	046503	126	046117	046525	EM44:	.ASCIZ /VOLUME VALID NOT CLEARED IN RKMR2/
8139	046510	020105	040526	044514		
8140	046516	020104	047516	020124		
8141	046524	046103	040505	042522		
8142	046532	020104	047111	051040		
8143	046540	046513	031122	000		
8144	046545	126	046117	046525	EM45:	.ASCIZ /VOLUME VALID SET IN RKMR2 AFTER HEADS LOADED/
8145	046552	020105	040526	044514		
8146	046560	020104	042523	020124		
8147	046566	047111	051040	046513		

0148	046574	00011123	040440	052106	
0149	046502	00011105	044040	040505	
0150	046610	00015004	046040	040517	
0151	046616	00017504	000104		
0152	046622	00015016	026516	054105	EM46: .ASCIZ /NON-EXECUTABLE FUNCTION (NXF) NOT SET IN RKMR3/
0153	046632	00015016	000104	047101	
0154	046642	00015016	000104	047125	
0155	046648	00010314	047511	020116	
0156	046652	00010314	043130	020051	
0157	046660	00017516	020124	042523	
0158	046666	00017516	020124	051040	
0159	046672	00016513	047111	000000	
0160	046678	00010314	046131	047111	EM47: .ASCIZ /CYLINDER ADDRESS CHANGED FROM 0/
0161	046706	00015004	020122	042101	
0162	046714	00011004	051505	020123	
0163	046722	00041003	047101	042507	
0164	046730	00010400	051106	046517	
0165	046736	00000400	000000		
0166	046741	00012700	044522	042524	EM48: .ASCIZ /WRITE LOCK IN RKMR2 NOT CLEARED/
0167	046746	00016040	041517	020113	
0168	046754	00047040	051040	046513	
0169	046752	00011122	047040	052117	
0170	046770	00014400	042514	051101	
0171	046776	00010300	000000		
0172	047001	00012700	044522	042524	EM49: .ASCIZ /WRITE LOCK IN RKMR2 NOT SET/
0173	047006	00016040	041517	020113	
0174	047014	00011111	051040	046513	
0175	047022	00011122	047040	052117	
0176	047030	00014400	052105	000000	
0177	047035	00012700	044522	042524	EM50: .ASCIZ /WRITE LOCK ERROR IN RKMR3 NOT SET/
0178	047042	00016040	041517	020113	
0179	047050	00011040	047523	020122	
0180	047056	00017111	051040	046513	
0181	047064	00015222	047040	052117	
0182	047072	00014400	052105	000000	
0183	047077	00014400	052105	000000	
0184	047104	00012700	044522	042524	EM51: .ASCIZ /WRITE LOCK DID NOT OCCUR AT SECTOR BOUNDRY/
0185	047112	00014504	041517	020113	
0186	047120	00012024	020104	047516	
0187	047126	00012222	041517	052503	
0188	047134	00015005	052101	051440	
0189	047142	00047502	047524	020122	
0190	047150	0000131	047125	051104	
0191	047152	0000131	0000131		
0192	047160	00012500	020123	047516	EM52: .ASCIZ /UNS NOT SET IN RKMR3/
0193	047166	00012400	042523	020124	
0194	047174	00047111	051040	046513	
0195	047174	00031522	000000		
0196	047177	00012500	046116	040517	EM53: .ASCIZ /UNLOAD NOT SET IN RKMR2/
0197	047204	00020104	047516	020124	
0198	047212	00042523	020124	047111	
0199	047220	00051040	046513	031122	
0200	047226	00000000			
0201	047227	00010300	047101	047516	EM54: .ASCIZ /CANNOT FIND MULT. DRIVE SELECT IN RKCS2/
0202	047231	00020124	044506	042116	
0203	047242	00046440	046125	027124	

8204	047250	042040	044522	042526	
8205	047256	051440	046105	041505	
8206	047264	020124	047111	051040	
8207	047272	041513	031123	000	
8208	047277	101	052124	020116	EM55: .ASCIZ /ATTN NOT CLEARED IN RKASOF/
8209	047304	047516	020124	046103	
8210	047312	040505	042522	020104	
8211	047320	047111	051040	040513	
8212	047326	047523	000106		
8213	047332	047125	054105	042520	EM56: .ASCIZ /UNEXPECTED MEMORY PARITY ERROR TRAP/
8214	047340	052103	042105	046440	
8215	047346	046505	051117	020131	
8216	047354	040520	044522	054524	
8217	047362	042440	051122	051117	
8218	047370	052040	040522	000120	
8219	047376	04503	051122	044440	EM57: .ASCIZ /CERR IN RKCS1 NOT SET/
8220	047404	020116	045522	051503	
8221	047412	020061	047516	020124	
8222	047420	042523	000124		
8223	047424	040520	044522	054524	EM58: .ASCIZ /PARITY NOT SET IN RKMR3/
8224	047432	047040	052117	051440	
8225	047440	052105	044440	020116	
8226	047446	045522	051115	000063	
8227	047454	052103	020117	042523	EM59: .ASCIZ /CTO SET IN RKCS1/
8228	047462	020124	047111	051040	
8229	047470	041513	030523	000	
8230	047475	116	043130	042040	EM61: .ASCIZ /NXF DID NOT SET FAULT/
8231	047502	042111	047040	052117	
8232	047510	051440	052105	043040	
8233	047516	052501	052114	000	
8234	047523	104	052114	051440	EM63: .ASCIZ /DLT SET IN RKCS2/
8235	047530	052105	044440	020116	
8236	047536	045522	051503	000062	
8237	047544	045522	041504	023040	EM64: .ASCII /RKDC & RKDA INDICATE THAT WRITE CHECK ERROR/
8238	047552	051040	042113	020101	
8239	047560	047111	044504	040503	
8240	047566	042524	052040	040510	
8241	047574	020124	051127	052111	
8242	047602	020105	044103	041505	
8243	047610	020113	051105	047522	
8244	047616	122			
8245	047617	015	047412	041503	.ASCIZ <CR><LF>/OCCURRED AT CYL 411, TRACK 2, SECTOR 21/
8246	047624	051125	042522	020104	
8247	047632	052101	041440	046131	
8248	047640	032040	030461	020054	
8249	047646	051124	041501	020113	
8250	047654	026062	051440	041505	
8251	047662	047524	020122	030462	
8252	047670	000			
8253	047671	116	042105	051440	EM67: .ASCIZ /NED SET IN RKCS2/
8254	047676	052105	044440	020116	
8255	047704	045522	051503	000062	
8256	047712	040503	047116	052117	EM68: .ASCIZ /CANNOT READ BAD SECTOR INFORMATION/
8257	047720	051040	040505	020104	
8258	047726	040502	020104	042523	
8259	047734	052103	051117	044440	

8260	047742	043116	051117	040515	
8261	047750	044524	047117	000	
8262	047755	116	020117	051104	EM69: .ASCII /NO DRIVES FOUND ON BUSS/
8263	047762	053111	051505	043040	
8264	047770	052517	042116	047440	
8265	047776	020116	052502	051523	
8266	050004	005015	042523	052524	.ASCIZ <CR><LF>/SETUP CORRECTLY & PRESS 'CONTINUE'/<CR><LF>
8267	050012	020120	047503	051122	
8268	050020	041505	046124	020131	
8269	050025	020046	051120	051505	
8270	050034	020123	041447	047117	
8271	050042	044524	052516	023505	
8272	050050	005015	000		
8273	050053	127	044510	042514	EM70: .ASCIZ /WHILE WAITING FOR CONTR READY OR AFTER CONTR READY REC'D/
8274	050060	053440	044501	044524	
8275	050066	043516	043040	051117	
8276	050074	041440	047117	051124	
8277	050102	051040	040505	054504	
8278	050110	047440	020122	043101	
8279	050116	042524	020122	047503	
8280	050124	052116	020122	042522	
8281	050132	042101	020131	042522	
8282	050140	023503	000104		
8283	050144	042504	042524	052103	EM71: .ASCIZ /DETECTED 10 BAD SECTORS...ABORTING TEST/
8284	050152	042105	030440	020060	
8285	050160	040502	020104	042523	
8286	050166	052103	051117	027123	
8287	050174	027056	041101	051117	
8288	050202	044524	043516	052040	
8289	050210	051505	000124		
8290	050214	042504	042524	052103	EM72: .ASCIZ /DETECTED BSE BUT NOT LISTED IN BAD SECTOR FILE/
8291	050222	042105	041040	042523	
8292	050230	041040	052125	047040	
8293	050236	052117	046040	051511	
8294	050244	042524	020104	047111	
8295	050252	041040	042101	051440	
8296	050260	041505	047524	020122	
8297	050266	044506	042514	000	
8298	050273	104	052105	041505	EM73: .ASCII /DETECTED BSE IN READ COMMAND/
8299	050300	042524	020104	051502	
8300	050306	020105	047111	051040	
8301	050314	040505	020104	047503	
8302	050322	046515	047101	104	
8303	050327	015	041012	052125	.ASCIZ <CR><LF>/BUT NOT IN PREVIOUS WRITE COMMAND TO SAME SECTOR/
8304	050334	047040	052117	044440	
8305	050342	020116	051120	053105	
8306	050350	047511	051525	053440	
8307	050356	044522	042524	041440	
8308	050364	046517	040515	042116	
8309	050372	052040	020117	040523	
8310	050400	042515	051440	041505	
8311	050406	047524	000122		
8312	050412	052122	020132	047516	EM74: .ASCIZ /RTZ NOT SET IN RKMR2/
8313	050420	020124	042523	020124	
8314	050426	047111	051040	046513	
8315	050434	031122	000		

8316	050437	015	042012	052105	EM75:	.ASCIZ	<CR><LF>/DETECTED 10 BAD CYLINDERS...ABORTING TEST/
8317	050444	041505	042524	020104			
8318	050452	030061	041040	042101			
8319	050460	041440	046131	047111			
8320	050466	042504	051522	027056			
8321	050474	040456	047502	052122			
8322	050502	047111	020107	042524			
8323	050510	052123	000				
8324	050513	116	020117	051104	EM76:	.ASCII	/NO DRIVES FOUND IN DEVICE MAP (\$DEV)/
8325	050520	053111	051505	043040			
8326	050526	052517	042116	044440			
8327	050534	020116	042504	044526			
8328	050542	042503	046440	050101			
8329	050550	024040	042044	053105			
8330	050556	024515					
8331	050560	005015	042523	052524		.ASCIZ	<CR><LF>/SETUP CORRECTLY & RESTART/<CR><LF>
8332	050566	020120	047503	051122			
8333	050574	041505	046124	020131			
8334	050602	020046	042522	052123			
8335	050610	051101	006524	000012			
8336	050616	051127	052111	020105	EM80:	.ASCIZ	/WRITE CHECK ERROR SET IN RKCS2/
8337	050624	044103	041505	020113			
8338	050632	051105	047522	020122			
8339	050640	042523	020124	047111			
8340	050646	051040	041513	031123			
8341	050654	000					
8342	050655	104	052101	020101	EM83:	.ASCIZ	/DATA CHECK ERROR SET IN RKER/
8343	050662	044103	041505	020113			
8344	050670	051105	047522	020122			
8345	050676	042523	020124	047111			
8346	050704	051040	042513	000122			
8347	050712	042522	042101	047111	EM93:	.ASCIZ	/READING WRONG CYLINDER # IN HEADER/
8348	050720	020107	051127	047117			
8349	050726	020107	054503	044514			
8350	050734	042116	051105	021440			
8351	050742	044440	020116	042510			
8352	050750	042101	051105	000			
8353							
8354					.SBTTL	DATA HEADERS	
8355							
8356	050755	124	051505	020124	DH1:	.ASCIZ	/TEST NO. PC/
8357	050762	047516	020056	050040			
8358	050770	000103					
8359	050772	045522	051115	004461	DH2:	.ASCIZ	/RKMR1 RKMR2 RKMR3 RKER RKDS RKCS1 RKCS2/
8360	051000	045522	051115	004462			
8361	051006	045522	051115	004463			
8362	051014	045522	051105	051011			
8363	051022	042113	004523	045522			
8364	051030	051503	004461	045522			
8365	051036	051503	000062				
8366	051042	045522	041527	051011	DH3:	.ASCIZ	/RKWC RKBA RKDA RKASOF RKDC RKECPS RKECPT/
8367	051050	041113	004501	045522			
8368	051056	040504	051011	040513			
8369	051064	047523	004506	045522			
8370	051072	041504	051011	042513			
8371	051100	050103	004523	045522			

8372	051106	041505	052120	000	
8373	051113	05101	052106	051105	DH4: .ASCIZ /AFTER PACK RE-INSERTED & HEADS LOADED/
8374	051120	050040	041501	020113	
8375	051126	042522	044454	051516	
8376	051134	051105	042524	020104	
8377	051142	020046	042510	042101	
8378	051150	020123	047514	042101	
8379	051156	042105	000		
8380	051161	05101	052106	051105	DH5: .ASCIZ /AFTER AC SWITCHED OFF/
8381	051166	040440	020103	053523	
8382	051174	052111	044103	042105	
8383	051202	047440	043106	000	
8384	051207	05101	052106	051105	DH8: .ASCIZ /AFTER DRIVE UNLOADED & DOOR OPENED/
8385	051214	042040	044522	042526	
8386	051222	052440	046116	040517	
8387	051230	042504	020104	020046	
8388	051236	047504	051117	047440	
8389	051244	042520	042516	000104	
8390	051252	043101	042524	020122	DH11: .ASCIZ /AFTER MANUALLY LOADING HEADS WITH DOOR OPEN/
8391	051260	040515	052516	046101	
8392	051266	054514	046040	040517	
8393	051274	044504	043516	044040	
8394	051302	040505	051504	053440	
8395	051310	052111	020110	047504	
8396	051316	051117	047440	042520	
8397	051324	000116			
8398	051326	043101	042524	020122	DH12: .ASCIZ /AFTER DISK PACK REMOVED/
8399	051334	044504	045523	050040	
8400	051342	041501	020113	042522	
8401	051350	047515	042526	000104	
8402	051356	043101	042524	020122	DH13: .ASCIZ /AFTER MANUALLY LOADING HEADS WITH DISK PACK REMOVED/
8403	051364	040515	052516	046101	
8404	051372	054514	046040	040517	
8405	051400	044504	043516	044040	
8406	051406	040505	051504	053440	
8407	051414	052111	020110	044504	
8408	051422	045523	050040	041501	
8409	051430	020113	042522	047515	
8410	051436	042526	000104		
8411	051442	044527	044124	052517	DH15: .ASCIZ /WITHOUT PACK COMMAND/
8412	051450	020124	040520	045503	
8413	051456	041440	046517	040515	
8414	051464	042116	000		
8415	051467	05101	052106	051105	DH16: .ASCIZ /AFTER UNIT SELECT PLUG REMOVED/
8416	051474	052440	044516	020124	
8417	051502	042523	042514	052103	
8418	051510	050040	052514	020107	
8419	051516	042522	047515	042526	
8420	051524	000104			
8421	051526	043101	042524	020122	DH17: .ASCIZ /AFTER RECAL COMMAND/
8422	051534	042522	040503	020114	
8423	051542	047503	046515	047101	
8424	051550	000104			
8425	051552	043101	042524	020122	DH18: .ASCIZ /AFTER UNLOAD COMMAND/
8426	051560	047125	047514	042101	
8427	051566	041440	046517	040515	

E13

CZR6JEO UNIBUS RK6 DR PRT3
CZR6JE.P11 25-JAN-78 12:08MACY11 30A(1052) 25-JAN-78 12:16 PAGE 160
DATA HEADERS

SEQ 0160

8428	051574	042116	000		
8429	051577	101	052106	051105	DH19: .ASCIZ /AFTER PACK COMMAND/
8430	051604	050040	041501	020113	
8431	051612	047503	046515	047101	
8432	051620	000104			
8433	051622	043101	042524	020122	DH20: .ASCIZ /AFTER SELECT DRIVE COMMAND/
8434	051630	042523	042514	052103	
8435	051636	042040	044522	042526	
8436	051644	041440	046517	040515	
8437	051652	042116	000		
8438	051655	101	052106	051105	DH21: .ASCIZ /AFTER SUBSYSTEM CLEAR/
8439	051662	051440	041125	054523	
8440	051670	052123	046505	041440	
8441	051676	042514	051101	000	
8442	051703	101	052106	051105	DH22: .ASCIZ /AFTER DRIVE CLEAR COMMAND/
8443	051710	042040	044522	042526	
8444	051716	041440	042514	051101	
8445	051724	041440	046517	040515	
8446	051732	042116	000		
8447	051735	101	052106	051105	DH23: .ASCIZ /AFTER WRONG PORT SELECTED/
8448	051742	053440	047522	043516	
8449	051750	050040	051117	020124	
8450	051756	042523	042514	052103	
8451	051764	042105	000		
8452	051767	101	052106	051105	DH25: .ASCIZ /AFTER SEEK COMMAND/
8453	051774	051440	042505	020113	
8454	052002	047503	046515	047101	
8455	052010	000104			
8456	052012	043101	042524	020122	DH26: .ASCIZ /AFTER READ DATA COMMAND/
8457	052020	042522	042101	042040	
8458	052026	052101	020101	047503	
8459	052034	046515	047101	000104	
8460	052042	043101	042524	020122	DH27: .ASCIZ /AFTER WRITE DATA COMMAND/
8461	052050	051127	052111	020105	
8462	052056	040504	040524	041440	
8463	052064	046517	040515	042116	
8464	052072	000			
8465	052073	101	052106	051105	DH28: .ASCIZ /AFTER BOTH PORTS DESELECTED/
8466	052100	041040	052117	020110	
8467	052106	047520	052122	020123	
8468	052114	042504	042523	042514	
8469	052122	052103	042105	000	
8470	052127	101	052106	051105	DH29: .ASCIZ /AFTER CORRECT PORT SELECTED/
8471	052134	041440	051117	042522	
8472	052142	052103	050040	051117	
8473	052150	020124	042523	042514	
8474	052156	052103	042105	000	
8475	052163	101	052106	051105	DH30: .ASCIZ /AFTER READ HEADER COMMAND/
8476	052170	051040	040505	020104	
8477	052176	042510	042101	051105	
8478	052204	041440	046517	040515	
8479	052212	042116	000		
8480	052215	101	052106	051105	DH31: .ASCIZ /AFTER DRIVE MANUALLY LOADED/
8481	052222	042040	044522	042526	
8482	052230	046440	047101	040525	
8483	052236	046114	020131	047514	

8484	052244	042101	042105	000		
8485	052251	101	052106	051105	DH32:	.ASCIZ /AFTER WRITE CHECK COMMAND/
8486	052256	053440	044522	042524		
8487	052264	041440	042510	045503		
8488	052272	041440	046517	040515		
8489	052300	042116	000			
8490	052303	101	052106	051105	DH34:	.ASCIZ /AFTER START SPINDLE COMMAND/
8491	052310	051440	040524	052122		
8492	052316	051440	044520	042116		
8493	052324	042514	041440	046517		
8494	052332	040515	042116	000		
8495	052337	101	052106	051105	DH35:	.ASCIZ /AFTER MANUALLY UNLOADING/
8496	052344	046440	047101	040525		
8497	052352	046114	020131	047125		
8498	052360	047514	042101	047111		
8499	052366	000107				
8500	052370	043101	042524	020122	DH37:	.ASCIZ /AFTER TIMEOUT TO POWER DOWN/
8501	052376	044524	042515	052517		
8502	052404	020124	047524	050040		
8503	052412	053517	051105	042040		
8504	052420	053517	000116			
8505	052424	043101	042524	020122	DH38:	.ASCIZ /AFTER AC POWERED UP/
8506	052432	041501	050040	053517		
8507	052440	051105	042105	052440		
8508	052446	070120				
8509	052450	043101	042524	020122	DH39:	.ASCIZ /AFTER WRITE HEADER COMMAND/
8510	052456	051127	052111	020105		
8511	052464	042510	042101	051105		
8512	052472	041440	046517	040515		
8513	052500	042116	000			
8514	052503	104	051125	047111	DH41:	.ASCIZ /DURING RECAL COMMAND/
8515	052510	020107	042522	040503		
8516	052516	020114	047503	046515		
8517	052524	047101	000104			
8518	052530	047117	051440	041505	DH42:	.ASCIZ /ON SECTORS 0,2,4,6 OR 8 CYL 410 TRACK 2/
8519	052536	047524	051522	030040		
8520	052544	031054	032054	033054		
8521	052552	047440	020122	020070		
8522	052560	041440	046131	032040		
8523	052566	030	052040	040522		
8524	052574	045	031040	000		
8525	052601	106	051117	040515	DH44:	.ASCIZ /FORMAT & ALL READ-WRITE TESTS WILL BE BYPASSED/
8526	052606	020124	020046	046101		
8527	052614	020114	042522	042101		
8528	052622	053455	044522	042524		
8529	052630	052040	051505	051524		
8530	052636	053440	046111	020114		
8531	052644	042502	041040	050131		
8532	052652	051501	042523	000104		
8533	052660	043101	042524	020122	DH45:	.ASCIZ /AFTER SEEK WITH VOLUME VALID=0/
8534	052666	042523	045505	053440		
8535	052674	052111	020110	047526		
8536	052702	052514	042515	053040		
8537	052710	046101	042111	030075		
8538	052716	000				
8539	052717	101	052106	051105	DH46:	.ASCIZ /AFTER WRITE DATA WITH VOLUME VALID=0/

8540	052724	053440	044522	042524	
8541	052732	042040	052101	020101	
8542	052740	044527	044124	053040	
8543	052746	046117	046525	020105	
8544	052754	040526	044514	036504	
8545	052762	000060			
8546	052764	043101	042524	020122	DH48: .ASCIZ /AFTER WRITE LOCK SWITCH DISABLED/
8547	052772	051127	052111	020105	
8548	053000	047514	045503	051440	
8549	053006	044527	041524	020110	
8550	053014	044504	040523	046102	
8551	053022	042105	000		
8552	053027	101	052106	051105	DH49: .ASCIZ /AFTER WRITE LOCK SWITCH ENABLED/
8553	053032	053440	044522	042524	
8554	053040	046040	041517	020113	
8555	053046	053523	052111	044103	
8556	053054	042440	040516	046102	
8557	053062	042105	000		
8558	053065	101	052106	051105	DH50: .ASCIZ /AFTER WRITING WITH WRITE LOCK ENABLED/
8559	053072	053440	044522	044524	
8560	053100	043516	053440	052111	
8561	053106	020110	051127	052111	
8562	053114	020105	047514	045503	
8563	053122	042440	040516	046102	
8564	053130	042105	000		
8565	053133	103	046131	021440	DH56: .ASCIZ /CYL # HEADER WORD 0/
8566	053140	044011	040505	042504	
8567	053146	020122	047527	042122	
8568	053154	030040	000		
8569	053157	101	052106	051105	DH59: .ASCIZ /AFTER DRIVE SELECT COMMAND WITH EVEN PARITY/
8570	053164	042040	044522	042526	
8571	053172	051440	046105	041505	
8572	053200	020124	047503	046515	
8573	053206	047101	020104	044527	
8574	053214	044124	042440	042526	
8575	053222	020116	040520	044522	
8576	053230	054524	000		
8577	053233	101	052106	051105	DH60: .ASCIZ /AFTER WRITE LOCK ENABLED DURING CONTINUOUS WRITING/
8578	053240	053440	044522	042524	
8579	053246	046040	041517	020113	
8580	053254	047105	041101	042514	
8581	053262	020104	052504	044522	
8582	053270	043516	041440	047117	
8583	053276	044524	052516	052517	
8584	053304	020123	051127	052111	
8585	053312	047111	000107		
8586	053316	045522	040504	053411	DH61: .ASCII /RKDA WORD EXPECTED/
8587	053324	051117	004504	054105	
8588	053332	042520	052103	042105	
8589	053340	005015	020100	051127	.ASCIZ <CR><LF>/@ WRL WAS WORD/
8590	053346	004514	040527	004523	
8591	053354	047527	042122	000	
8592	053361	122	042113	004501	DH62: .ASCII /RKDA WORD EXPECTED/
8593	053366	047527	042122	042411	
8594	053374	050130	041505	042524	
8595	053402	104			

Line	Code	Code	Code	Code	Description
8652					.SBTTL ERROR OUTPUT DATA
8653					
8654					
8655		054074			.EVEN
8656	054074	001214	001116		DT1: \$TESTN,\$ERRPC
8657	054100	005364	005366	005370	HMR1,HMR2,HMR3,HER,HDS,HCS1,HCS2
8658	054106	005354	005352	005340	
8659	054114	005342			
8660	054116	005344	005346	005350	HWC,HBA,HDA,HASOF,HDC,HPOS,HPAT
8661	054124	005356	005360	005372	
8662	054132	005374			
8663	054134	001214	001116	001166	DT3: \$TESTN,\$ERRPC,\$TMP3,WD1,WD2
8664	054142	001434	001436		
8665	054146	005364	005366	005370	HMR1,HMR2,HMR3,HER,HDS,HCS1,HCS2
8666	054154	005354	005352	005340	
8667	054162	005342			
8668	054164	005344	005346	005350	HWC,HBA,HDA,HASOF,HDC,HPOS,HPAT
8669	054172	005356	005360	005372	
8670	054200	005374			
8671	054202	001214	001354		DT6: \$TESTN,TRAPPC
8672	054206	001214	001116	001366	DT9: \$TESTN,\$ERRPC,TOCYL,RHTAB
8673	054214	001712			
8674	054216	005364	005366	005370	HMR1,HMR2,HMR3,HER,HDS,HCS1,HCS2
8675	054224	005354	005352	005340	
8676	054232	005342			
8677	054234	005344	005346	005350	HWC,HBA,HDA,HASOF,HDC,HPOS,HPAT
8678	054242	005356	005360	005372	
8679	054250	005374			
8680	054252	001214	001116	005430	DT13: \$TESTN,\$ERRPC,E.A0,E.B0,E.A1,E.B1,H.A0,H.B0,H.A1,H.B1
8681	054260	005432	005434	005436	
8682	054266	005410	005412	005414	
8683	054274	005416			
8684	054276	005364	005366	005370	HMR1,HMR2,HMR3,HER,HDS,HCS1,HCS2
8685	054304	005354	005352	005340	
8686	054312	005342			
8687	054314	005344	005346	005350	HWC,HBA,HDA,HASOF,HDC,HPOS,HPAT
8688	054322	005356	005360	005372	
8689	054330	005374			
8690					
8691	054332	001214	001116	005430	DT14: \$TESTN,\$ERRPC,E.A0,E.B0,E.A1,E.B1,E.A2,E.B2
8692	054340	005432	005434	005436	
8693	054346	005440	005442		
8694	054352	005410	005412	005414	H.A0,H.B0,H.A1,H.B1,H.A2,H.B2
8695	054360	005416	005420	005422	
8696	054366	005364	005366	005370	HMR1,HMR2,HMR3,HER,HDS,HCS1,HCS2
8697	054374	005354	005352	005340	
8698	054402	005342			
8699	054404	005344	005346	005350	HWC,HBA,HDA,HASOF,HDC,HPOS,HPAT
8700	054412	005356	005360	005372	
8701	054420	005374			
8702					
8703	054422	001214	001116	005430	DT15: \$TESTN,\$ERRPC,E.A0,E.B0,E.A1,E.B1,E.A2,E.B2,E.B3
8704	054430	005432	005434	005436	
8705	054436	005440	005442	005446	
8706	054444	005410	005412	005414	H.A0,H.B0,H.A1,H.B1,H.A2,H.B2,H.B3
8707	054452	005416	005420	005422	

8708	054460	005426			
8709	054462	005364	005366	005370	HMR1, HMR2, HMR3, HER, HDS, HCS1, HCS2
8710	054470	005354	005352	005340	
8711	054476	005342			
8712	054500	005344	005346	005350	HWC, HBA, HDA, PASOF, HDC, HPOS, HPAT
8713	054506	005356	005360	005372	
8714	054514	005374			
8715					
8716					.SB^TL ERROR DATA FORMATS
8717					DF1: 3
8718	054516	000003			.BYTE 2,0
8719	054520	002	000		DH2
8720	054522	050772			.BYTE 7,0
8721	054524	007	000		DH3
8722	054526	051042			.BYTE 7,0
8723	054530	007	000		
8724					
8725					
8726	054532	000005			DF3: 5
8727	054534	000	000		.BYTE 0,0
8728	054536	050755			DH1
8729	054540	002	000		.BYTE 2,0
8730	054542	053316			DH61
8731	054544	003	000		.BYTE 3,0
8732	054546	050772			DH2
8733	054550	007	000		.BYTE 7,0
8734	054552	051042			DH3
8735	054554	007	000		.BYTE 7,0
8736					
8737	054556	000005			DF4: 5
8738	054560	000	000		.BYTE 0,0
8739	054562	050755			DH1
8740	054564	002	000		.BYTE 2,0
8741	054566	053361			DH62
8742	054570	003	000		.BYTE 3,0
8743	054572	050772			DH2
8744	054574	007	000		.BYTE 7,0
8745	054576	051042			DH3
8746	054600	007	000		.BYTE 7,0
8747					
8748	054602	000001			DF5: 1
8749	054604	002	000		.BYTE 2,0
8750					
8751	054606	000003			DF7: 3
8752	054610	002	000		.BYTE 2,0
8753	054612	050772			DH2
8754	054614	007	000		.BYTE 7,0
8755	054616	051042			DH3
8756	054620	007	000		.BYTE 7,0
8757	054622	000004			DF10: 4
8758	054624	000	000		.BYTE 0,0
8759	054626	050755			DH1
8760	054630	002	000		.BYTE 2,0
8761	054632	050772			DH2
8762	054634	007	000		.BYTE 7,0
8763	054636	051042			DH3

8764	054640	007	000	.BYTE	7,0
8765					
8766	054642	000005		DF12: 5	
8767	054644	000	000	.BYTE	0,0
8768	054646	053733		DH73	
8769	054650	000	000	.BYTE	0,0
8770	054652	050755		DH1	
8771	054654	002	000	.BYTE	2,0
8772	054656	050772		DH2	
8773	054660	007	000	.BYTE	7,0
8774	054662	051042		DH3	
8775	054664	007	000	.BYTE	7,0
8776					
8777					
8778	054666	000004		DF15: 4	
8779	054670	000	000	.BYTE	0,0
8780	054672	050755		DH1	
8781	054674	002	000	.BYTE	2,0
8782	054676	050772		DH2	
8783	054700	007	000	.BYTE	7,0
8784	054702	051042		DH3	
8785	054704	007	000	.BYTE	7,0
8786					
8787					
8788	054706	000005		DF17: 5	
8789	054710	000	000	.BYTE	0,0
8790	054712	052601		DH44	
8791	054714	000	000	.BYTE	0,0
8792	054716	050755		DH1	
8793	054720	002	000	.BYTE	2,0
8794	054722	050772		DH2	
8795	054724	007	000	.BYTE	7,0
8796	054726	051042		DH3	
8797	054730	007	000	.BYTE	7,0
8798	054732	000005		DF20: 5	
8799	054734	000	000	.BYTE	0,0
8800	054736	050755		DH1	
8801	054740	002	000	.BYTE	2,0
8802	054742	053133		DH56	
8803	054744	002	000	.BYTE	2,0
8804	054746	050772		DH2	
8805	054750	007	000	.BYTE	7,0
8806	054752	051042		DH3	
8807	054754	007	000	.BYTE	7,0
8808					
8809	054756	000007		DF21: 7	
8810	054760	000	000	.BYTE	0,0
8811	054762	050755		DH1	
8812	054764	002	000	.BYTE	2,0
8813	054766	053666		DH69	
8814	054770	000	000	.BYTE	0,0
8815	054772	053706		DH71	
8816	054774	004	000	.BYTE	4,0
8817	054776	053676		DH70	
8818	055000	004	000	.BYTE	4,0
8819	055002	050772		DH2	

```

8820 055004 007 000
8821 055006 051042
8822 055010 007 000
8823
8824 055012 000007
8825 055014 000 000
8826 055016 050755
8827 055020 002 000
8828 055022 053666
8829 055024 000 000
8830 055026 053706
8831 055030 006 000
8832 055032 053676
8833 055034 006 000
8834 055036 050772
8835 055040 007 000
8836 055042 051042
8837 055044 007 000
8838
8839 055046 000007
8840 055050 000 000
8841 055052 050755
8842 055054 002 000
8843 055056 053666
8844 055060 000 000
8845 055062 053706
8846 055064 007 000
8847 055066 053676
8848 055070 007 000
8849 055072 050772
8850 055074 007 000
8851 055076 051042
8852 055100 007 000
8853
8854
8855
8856
8857
8858
8859
8860
8861
8862
8863 055102 104413
8864 055104 032777 020000 124026
8865 055112 001107
8866 055114 113700 001114
8867 055120 042700 177400
8868 055124 005300
8869 055126 006300
3870 055130 006300
8871 055132 006300
8872 055134 062700 005512
8873 055140 012037 055154
8874 055144 001404
8875 055146 104401 001205

```

```

.BYTE 7,0
DH3
.BYTE 7,0
DF22: 7
.BYTE 0,0
DH1
.BYTE 2,0
DH69
.BYTE 0,0
DH71
.BYTE 6,0
DH70
.BYTE 6,0
DH2
.BYTE 7,0
DH3
.BYTE 7,0
DF23: 7
.BYTE 0,0
DH1
.BYTE 2,0
DH69
.BYTE 0,0
DH71
.BYTE 7,0
DH70
.BYTE 7,0
DH2
.BYTE 7,0
DH3
.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 #SW13,0SWR ;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 ENTP
MOV (RO)+,2$ ;GET EM POINTER
BEQ 3$ ;BRANCH IF THERE ISN'T ONE
TYPE .SCLF ;TYPE CARRIAGE RETURN LINE FEED

```



```

8932
8933
8934
8935
8936
8937
8938      055372
8939      055452
8940      000000
8941      000001
8942      000002
8943      000003
8944      000004
8945      000005
8946      000006
8947      000007
8948      177776
8949
8950      000014
8951      000340
8952      000020
8953      000003
8954      000006
8955
8956
8957
8958
8959
8960
8961      177562
8962      177560
8963      177566
8964      177564
8965
8966
8967
8968
8969
8970
8971
8972
8973
8974
8975      055452  000413
8976      055454  000417
8977      055456  013737  177776  055432
8978      055464  013737  000016  177776
8979      055472  010737  055430
8980      055476  000137  056630
8981
8982      055502  012706  055412
8983      055506  010637  055426
8984      055512  0004.4
8985      055514  004037  057036
8986      055520  013777  055450  177716
8987      055526  113704  055434

```

```

; DEC-11-UODPA-A-LA
; COPYRIGHT 1969,1970,1972
; DIGITAL EQUIPMENT CORPORATION
; MAYNARD, MASSACHUSETTS 01754
; .ENABL ABS,AMA
; .EVEN
; .+.60
R0      =      %0      ; REGISTER
R1      =      %1      ; NAMING
R2      =      %2      ; CONVENTIONS
R3      =      %3
R4      =      %4
R5      =      %5
SP      =      %6
PC      =      %7
ST      =      177776      ; STATUS REGISTER
O.TVEC =      14      ; TRT VECTOR LOCATION
O.STM  =      340     ; PRIORITY MASK - STATUS REGISTER
O.TBT  =      20      ; T-BIT MASK - STATUS REGISTER
TRT    =      000003  ; TRT INSTRUCTION
RTT    =      000006  ; RTT INSTRUCTION
;
; RS IS USUALLY CONSIDERED SAFE. THE CURRENT ADDRESS WORD
; RESIDES IN IT. AFTER A BREAKPOINT, IT IS SET TO ZERO, AND SEARCH
; OPERATIONS LEAVE IT RANDOMLY FILLED. OTHERWISE, IT SHOULD NOT
; BE USED EXCEPT FOR JSR'S AND THE CURRENT ADDRESS POINTER (CAD).
O.RDB  =      177562  ; R DATA BUFFER
O.RCSR =      177560  ; R C/SR
O.TDB  =      177566  ; T DATA BUFFER
O.TCSR =      177564  ; T C/SR
;
; INITIALIZE ODT
; USE O.ODT FOR A NORMAL ENTRY
; USE O.ODT+2 TO RESTART ODT - WIPING OUT ALL BREAKPOINTS
; USE O.ODT+4 TO RE-ENTER (I.E. - FAKE A BREAKPOINT)
O.ODT: BR      O.STRT      ; NORMAL ENTRY
      BR      O.RST       ; RESTART
O.ENTR: MOV     ST O.UST    ; RE-ENTER -- SAVE STATUS
      MOV     O.TVEC+2,ST  ; SET UP LOCAL STATUS
      MOV     PC O.UPC     ; FAKE THE PC
      JMP     O.BK1
;
O.STRT: MOV     #O.URD,SP  ; SET UP STACK
      MOV     SP O.USR     ; FAKE THE SAVED STACK
      BR      O.RST1      ; CLEAR BREAKPOINT TABLES
O.RST:  JSR     O.O.SVR    ; SAVE REGISTERS
      MOV     O.UIN,BO.ADR1 ; REMOVE THE BREAKPOINT
      MOV     O.PRI,R4     ; GET ODT PRIORITY

```

```

8988 055532 106004          RORB R4          ;SHIFT
8989 055534 106004          RORB R4          ;INTO
8990 055536 106004          RORB R4          ;POSITION
8991 055540 110437 177776    MOVB R4,ST      ;STORE IN STATUS
8992 055544 000127          O.RST1: JMP (PC)+
8993 055546 000403          BR 0.45
8994 055550 012737 000002 056540 MOV #RTI,0.RTIT ;SET TO RTI IF 11/20 OR /05
8995 055556 105037 057457          O.45: CLRB 0.P   ;DISA_LOW PROCEED
8996 055562 012737 000340 000016 MOV #0.STM,0.TVEC+2 ;STATUS WORD TO TRT VECTOR + 2
8997 055570 012737 056620 000014 MOV #0.BRK,0.TVEC ;PC TO TRT VECTOR
8998 055576 000447          BR 0.RALL       ;CLEAR BREAKPOINT TABLES
8999
9000          ;
9001          ; SPECIAL NAME HANDLER
9002          ; DEPENDS UPON THE EXPLICIT ORDER OF THE TWO TABLES O.TL AND O.URD
9003 055600 004537 057260    O.REGT: JSR 5.O.GET ;SPECIAL NAME, GET ONE MORE CHARACTER
9004 055604 012704 057503          MOV #O.TL,R4    ;TABLE START ADDRESS
9005 055610 120024          O.RSP: CMPB RO (R4)+ ;IS THIS THE CORRECT CHARACTER?
9006 055612 001413          BEQ 0.SP        ;JUMP IF YES
9007 055614 022704 057511          CMP #O.TL+O.LG,R4 ;IS THE SEARCH DONE?
9008 055620 101373          BHI 0.RSP       ;BRANCH IF NOT
9009 055622 042700 177770          BIC #177770,R0 ;MASK OFF OCTAL
9010 055626 C10004          MOV R0,R4
9011 055630 006304          O.SP1: ASL R4
9012 055632 062704 055412          ADD #O.URD,R4   ;GENERATE ADDRESS
9013 055636 005202          INC R2          ;SET FOUND FLAG
9014 055640 000444          BR 0.SCAN       ;GO FIND NEXT CHARACTER
9015 055642 162704 057474          O.SP: SUB #O.TL-7,R4 ;CORRECT CONSTANT
9016 055646 000770          BR 0.SP1
9017
9018          ;
9019          ; + HANDLER - OPEN INDEXED ON THE PC
9020 055650 004737 057404    O.ORPC: JSR PC,0.TCLS
9021 055654 010502          MOV R5,R2       ;CURRENT ADDRESS IN R2
9022 055656 061202          ADD @R2,R2      ;COMPUTE
9023 055660 006202          ASR R2          ;MOVE ONE BIT TO CARRY
9024 055662 103421          BCS 0.ERR       ;ERROR IF ODD NUMBER
9025 055664 006302          ASL R2          ;RESTORE WORD
9026 055666 005722          TST (R2)+       ;AND INCREMENT BY TWO
9027 055670 010205          MOV R2,R5       ;UPDATE CAD
9028 055672 000137 056144          JMP 0.OP2       ;GO FINISH UP
9029
9030          ;
9031          ; B HANDLER - SET AND REMOVE BREAKPOINTS
9032 055676 005702          O.BKPT: TST R2   ;IF NO NUMBER TYPED
9033 055700 001406          BEQ 0.RALL      ;REMOVE BREAKPOINT
9034 055702 006204          ASR R4          ;CHECK IF ODD
9035 055704 103410          BCS 0.ERR       ;JUMP IF ODD
9036 055706 006304          ASL R4          ;RESTORE ONE BIT
9037 055710 010437 055444          MOV R4,0.ADR1  ;SET A BREAKPOINT
9038
9039          BR 0.DCD
9040
9041
9042
9043 055716 012737 057520 055444 O.RALL: MOV #0.TRTC,0.ADR1 ;CLEAR BREAKPOINT

```

```

9044 055724 000406
9045
9046
9047
9048
9049
9050
9051 055726 052705 000001
9052 055732 012700 000077
9053 055736 004537 057336
9054 055742 004537 057436
9055 055746 005004
9056 055750 005002
9057 055752 004537 057260
9058 055756 022700 000060
9059 055762 101013
9060 055764 022700 000067
9061 055770 103410
9062 055772 042700 177770
9063 055776 006304
9064 056000 006304
9065 056002 006304
9066 056004 060004
9067 056006 005202
9068 056010 000760
9069 056012 005001
9070 056014 120061 057467
9071 056020 001405
9072 056022 005201
9073 056024 020127 000014
9074 056030 103336
9075 056032 000770
9076 056034 006301
9077 056036 000171 056042
9078
9079
9080 056042 056072
9081 056044 056124
9082 056046 055600
9083 056050 056434
9084 056052 056136
9085 056054 055650
9086 056056 056170
9087 056060 056200
9088 056062 056256
9089 056064 056252
9090 056066 055676
9091 056070 056542
9092 000030
9093
9094
9095
9096 056072 005702
9097 056074 001410
9098 056076 010405
9099 056100 006205

```

```

BR 0.DCD
:
: COMMAND DECODER - ODT11
:
: REGISTERS R0-R4 MAY BE USED
: REGISTER R5 WILL BE CONSIDERED SAFE
:
0.ERR: BIS #1,R5 ;CLOSE EVERYTHING
MOV #1,R0 ;? TO BE TYPED
JSR 5,O.FTYP ;OUTPUT ?
0.DCD: JSR 5,O.CRLS ;TYPE <CR><LF>*
0.DCD1: CLR R4 ;R4 CONTAINS THE CONVERTED OCTAL
CLR R2 ;R2 IS THE NUMBER FOUND FLAG
0.SCAN: JSR 5,O.GET ;GET A CHAR, RETURN IN R0
CMP #0,R0 ;COMPARE WITH ASCII 0
BHI 0.CLGL ;CHECK LEGALITY IF NON-NUMERIC
CMP #7,R0 ;COMPARE WITH ASCII 7
BLO 0.CLGL ;CHECK LEGALITY IF NOT OCTAL
BIC #177770,R0 ;CONVERT TO BCD
ASL R4 ;MAKE ROOM
ASL R4 ;IN
ASL R4 ;R4
ADD R0,R4 ;PACK THREE BITS IN R4
INC R2 ;R2 HAS NUMERIC FLAG
BR 0.SCAN ;AND TRY AGAIN
0.CLGL: CLR R1 ;CLEAR INDEX
0.LGL1: CMPB R0,0.LGCH(R1) ;DO THE CODES MATCH?
BEQ 0.LGL2 ;JUMP IF YES
INC R1 ;SET INDEX FOR NEXT SEARCH
CMP R1,#0.CLGT ;IS THE SEARCH DONE?
BHS 0.ERR ;OOPS!
BR 0.LGL1 ;RE-LOOP
0.LGL2: ASL R1 ;MULTIPLY BY TWO
JMP @0.LGDR(R1) ;GO TO PROPER ROUTINE
:
:
0.LGDR: 0.WRD ; / OPEN WORD
0.CRET ; CARRIAGE RETURN CLOSE
0.REGT ; $ REGISTER OPS
0.GO ; G GO TO ADDRESS K
0.OP1 ; <LF> MODIFY, CLOSE, OPEN NEXT
0.ORPC ; + OPEN RELATED INDEX - PC
0.BACK ; + OPEN PREVIOUS
0.OFST ; 0 OFFSET
0.WSCH ; W SEARCH WORD
0.EFF ; E SEARCH EFFECTIVE ADDRESS
0.BKPT ; B BREAKPOINTS
0.PROC ; P PROCEED
:
0.LGL = -0.LGDR ;LGL MUST EQUAL 2X CHLGT ALWAYS
:
: PROCESS / - OPEN WORD
:
0.WRD: TST R2 ;GET VALUE IF R2 IS NON-ZERO
BEQ 0.WRDA ;SKIP OTHERWISE
MOV R4,R5 ;PUT VALUE IN CAD
0.WRD1: ASR R5 ;MOVE ONE BIT TO CARRY

```

TYPE ERROR ROUTINE

9100 056102 103711
9101 056104 006305
9102 056106 011500
9103 056110 004537 057174
9104 056114 000714
9105 056116 042705 000001
9106 056122 000766
9107
9108
9109
9110 056124 004737 057404
9111 056130 052705 000001
9112 056134 000702
9113
9114
9115
9116 056136 004737 057404
9117 056142 005725
9118 056144 004537 057430
9119 056150 010500
9120 056152 004537 057174
9121 056156 012700 000057
9122 056162 004537 057336
9123 056166 000744
9124
9125
9126
9127 056170 004737 057404
9128 056174 005745
9129 056176 000762
9130
9131
9132
9133 056200 006205
9134 056202 103737
9135 056204 006305
9136 056206 012700 000040
9137 056212 004537 057336
9138 056216 160504
9139 056220 005304
9140 056222 005304
9141 056224 010400
9142 056226 010402
9143 056230 004537 057174
9144 056234 010200
9145 056236 006200
9146 056240 103402
9147 056242 004537 057174
9148 056246 000137 055746
9149
9150
9151
9152
9153
9154
9155

```

0.ERR2: BCS      0.ERR      ; JUMP IF ODD ADDRESS
        ASL      R5          ; RESTORE THE CARRY BIT
        MOV      @R5,R0      ; GET CONTENTS OF WORD
        JSR      S,0.CADV    ; GO GET AND TYPE OUT @CAD
        BR       0.DCD1      ; GO BACK TO DECODER
J.WRDA: BIC      #1,R5      ; CLEAR CLOSED BIT
        BR       0.WRD1      ; GO BACK TO MAIN-LINE
;
; PROCESS CARRIAGE RETURN
0.CRET: JSR      PC,0.TCLS   ; CLOSE LOCATION
        BIS      #1,R5      ; CLOSE EVERYTH 3
        BR       0.DCD      ; RETURN TO DECODER
;
; PROCESS <LF>, OPEN NEXT WORD
0.OP1:  JSR      PC,0.TCLS   ; CLOSE PRESENT CELL
        TST      (R5)+      ; GENERATE NEW ADDRESS
0.OP2:  JSR      S,0.CRLF    ; <CR><LF>
        MOV      R5,R0      ; NUMBER TO TYPE
        JSR      S,0.CADV    ; TYPE OUT ADDRESS
        MOV      #1,R0      ; TYPE A /
        JSR      S,C.FTYP    ;
        BR       0.WRD1      ; GO PROCESS IT
;
; PROCESS ↑, OPEN PREVIOUS WORD
0.BACK: JSR      PC,0.TCLS   ;
        TST      -(R5)      ; GENERATE NEW ADDRESS
        BR       0.OP2      ; GO DO THE REST
;
; PROCESS 0, COMPUTE OFFSET
0.OFST: ASR      R5          ; GET LOW ORDER BIT
        BCS      0.ERR2     ; ERROR IF CLOSED
        ASL      R5          ; RESTORE WORD
        MOV      #1,R0      ; TYPE ONE BLANK
        JSR      S,0.FTYP    ; AS A SEPARATOR
        SUB      R5,R4      ; COMPUTE
        DEC      R4          ;
        DEC      R4          ; 16 BIT OFFSET
        MOV      R4,R0      ; TYPE A
        MOV      R4,R2      ; SAVE R4
        JSR      S,0.CADV    ; NUMBER IN R0 - WORD MODE
        MOV      R2,R0      ;
        ASR      R0          ; DIVIDE BY TWO
        BCS      0.OF1      ; BRANCH IF ODD
        JSR      S,0.CADV    ; NUMBER IN R0 - BYTE MODE
0.OF1:  JMP      0.DCD1     ; ALL DONE
;
; SEARCHES - $MSK HAS THE MASK
;           $MSK+2 HAS THE FWA
;           $MSK+4 HAS THE LWA
    
```



```

9195 056404 020304          CMP      R3,R4          ; IS (X)+X+2=K?
9196 056406 001747          BEQ      0.WDS3         ; BRANCH IF EQUAL
9197 056410 042700 177400      BIC      #177400,R0     ; WIPE OUT EXTRANEIOUS BITS
9198 056414 110000          MOVB     R0,R0         ; EXTEND SIGN
9199 056416 000257          CCC
9200 056420 006300          ASL      R0           ; MULTIPLY BY TWO
9201 056422 005200          INC      R0           ; ADD TWO
9202 056424 005200          INC      R0
9203 056426 060200          ADD      R2,R0        ; ADD PC
9204 056430 020004          CMP      R0,R4        ; IS THE RESULT A PROPER REL. BRANCH?
9205 056432 000735          BR      0.WDS3 ..
9206
9207          ; PROCESS G - GO
9208
9209 056434 105037 057457      0.GO:   CLRB     0.P          ; DISALLOW PROCEED
9210 056440 006204          ASR      R4           ; CHECK LOW ORDER BIT
9211 056442 103617          BCS      0.ERR2       ; ERROR IF ODD NUMBER
9212 056444 006304          ASL      R4           ; RESTORE WORD
9213 056446 010437 055430      MOV      R4,0.UPC     ; SET UP NEW PC
9214 056452 112737 000340 177776      MOVB     #0,STM,ST    ; SET HIGH PRIORITY
9215 056460 004537 057126      JSR      5,0.RSTT     ; RESTORE TELETYPE
9216 056464 105037 057456      0.TBIT: CLR      0.T          ; CLEAR BOTH
9217 056470 042737 0J0020 055432      BIC      #0,TBT,0.UST ; T-BIT FLAGS
9218 056476 017737 176742 055450      MOV      #0,ADR1,0.UIN ; SAVE INSTRUCTION
9219 056504 013777 057520 176732      MOV      0,TRTC, #0,ADR1 ; REPLACE WITH TRAP
9220 056512 012600      0.G02:  MOV      (SP)+,R0     ; RESTORE
9221 056514 012601          MOV      (SP)+,R1     ; R0
9222 056516 012602          MOV      (SP)+,R2     ; THRU
9223 056520 012603          MOV      (SP)+,R3
9224 056522 012604          MOV      (SP)+,R4
9225 056524 012605          MOV      (SP)+,R5
9226 056526 012606          MOV      (SP)+,SP     ; R5
9227 056530 013746 055432      MOV      0,UST,-(SP)  ; AND SP
9228 056534 013746 055430      MOV      0,UPC,-(SP) ; AND STATUS
9229 056540 000006      0.RTIT: RTT          ; AND PC
9230          ; CHANGED TO RTI FOR 11/20 AND /05
9231
9232          ; PROCESS P - PROCEED
9233          ; ONLY ALLOWED AFTER A BREAKPOINT
9234 056542 105737 057457      0.PROC: TSTB     0.P          ; CHECK LEGALITY OF PROCEED
9235 056546 001645          BEQ      0.ERR1       ; NOT LEGAL
9236 056550 105037 057457      CLR      0.P          ; CLEAR PROCEED FLAG
9237 056554 005702          TST      R2           ; WAS COUNT SPECIFIED?
9238 056556 001402          BEQ      0.PRI        ; NO
9239 056560 010437 055446      MOV      R4,0.CT     ; YES, PUT AWAY COUNT
9240 056564 112737 000340 177776      MOVB     #0,STM,ST    ; FORCE HIGH PRIORITY
9241 056572 004537 057126      JSR      5,0.RSTT     ; RESTORE TTY
9242 056576 112737 000340 177776      MOVB     #0,STM,ST    ; SET HIGH PRIORITY
9243 056604 105237 057456      INCB     0.T          ; SET T-BIT FLAG
9244 056610 052737 000020 055432      BIS      #0,TBT,0.UST ; SET T-BIT
9245 056616 000735          BR      0.G02
9246
9247          ; BREAKPOINT HANDLER
9248          ; A TRT BREAKPOINT CAUSES 0.BRK TO BE ENTERED, WHICH SAVES
9249          ; VARIOUS ODDS AND ENDS, FINDS OUT IF THE BREAKPOINT WAS LEGAL,
9250          ; AND GIVES CONTROL TO THE COMMAND DECODER

```

```

9251
9252 056620 012637 055430      0.BRK:  MOV      (SP)+,0.UPC      ;PRIORITY IS 7 UPON ENTRY
9253 056624 012637 055432      MOV      (SP)+,0.UST      ;SAVE STATUS AND PC
9254 056630 004C37 057736      0.BK1:  JSR      0,0.SVR      ;SAVE VARIOUS REGISTERS
9255 056634 105737 057456      TSTB    0.T              ;CHECK FOR T-BIT SET
9256 056640 001311                BNE     0.TBIT           ;JUMP IF SET
9257 056642 013777 055450 176574  MOV     0.UIN,20.ADR1    ;REMOVE BREAKPOINTS
9258 056650 105737 055434      TSTB    0.PRI           ;CHECK IF PRIORITY
9259 056654 100003                BPL     0.BK2           ;IS AS SAME AS USER PGM
9260 056656 113705 055432      MOV     0.UST,R5        ;PICK UP USER UST IF SO
9261 056662 000407                BR      0.BK3           ;AND DON'T COMPUTE THE PRIORITY
9262 056664 113705 055434      0.PK2:  MOV     0.PRI,R5        ;OTHERWISE PICK UP ACTUAL PRIORITY
9263 056670 000257                CCC     ;CLEAR CARRY
9264 056672 106005                RORB   R5              ;SHIFT LOW ORDER BITS
9265 056674 106005                RORB   R5              ;INTO
9266 056676 106005                RORB   R5              ;HIGH ORDER
9267 056700 106005                RORB   R5              ;POSITION
9268 056702 110537 177776      0.BK3:  MOV     R5,ST          ;PUT THE STATUS AWAY WHERE IT BELONGS
9269 056706 013705 055430      MOV     0.UPC,R5        ;GET PC, IT POINTS TO THE TRT
9270 056712 005745                TST    -(R5)           ;SUBTRACT TWO
9271 056714 010537 055430      MOV     R5,C.UPC        ;FROM THE USER'S PC
9272 056720 020537 055444      CMP     R5,0.ADR1      ;COMPARE WITH LIST
9273 056724 001417                BEQ    0.B2            ;JUMP IF FOUND
9274 056726 004537 057074      JSR     5,0.SVTT        ;SAVE TELETYPE STATUS
9275 056732 001537 057430      JSR     5,0.CRLF        ;
9276 056736 001704 057462      MOV     #0,BD,R4        ;ERROR, NOTHING FOUND
9277 056742 012703 057463      MOV     #0,BD+1,R3      ;
9278 056746 004537 057322      JSR     5,0.TYPE        ;OUTPUT "BE" FOR BAD ENTRY
9279 056752 010500                MOV    R5,R0           ;
9280 056754 042737 000020 055432  BIC    #0.TBT,0.UST    ;CLEAR OUT ANY POSSIBLE FAKE T-BIT
9281 056762 000420                BR     0.B3            ;AND CONTINUE
9282 056764 005337 055446      0.B2:  DEC     0.CT           ;
9283 056770 003302                BGT    0.C1            ;JUMP IF REPEAT
9284 056772 012737 000001 055446  MOV     #1,0.CT        ;RESET COUNT TO 1
9285 057000 105237 057457      INCB   0.P             ;ALLOW PROCEED
9286 057004 004537 057074      JSR     5,0.SVTT        ;SAVE TELETYPE STATUS, R4 IS SAFE
9287 057010 012700 000102      MOV     #1,B,R0        ;
9288 057014 004537 057336      JSR     5,0.FTYP        ;TYPE "B"
9289 057020 013700 055444      MOV     0.ADR1,R0      ;GET ADDRESS OF BREAK
9290 057024 004537 057174      0.B3:  JSR     5,0.CADV        ;TYPE ADDRESS
9291 057030 005005                CLR    R5              ;CLEAR CAD
9292 057032 000137 055742      JMP     0.DCD          ;GO TO DECODER
9293
9294      ; SAVE REGISTERS R0-R6 IN INTERNAL STACK
9295
9296 057036 012637 057454      0.SVR:  MOV     (SP)+,0.XXX    ;PICK REGISTER FROM STACK AND SAVE
9297 057042 010637 055426      MOV     SP,0.USP        ;SAVE USER STACK ADDRESS
9298 057046 012706 055426      MOV     #0,USP,SP      ;SET TO INTERNAL STACK
9299 057052 010546                MOV    R5,-(SP)        ;SAVE
9300 057054 010446                MOV    R4,-(SP)        ;REGISTERS
9301 057056 010346                MOV    R3,-(SP)        ;
9302 057060 010246                MOV    R2,-(SP)        ;1
9303 057062 010146                MOV    R1,-(SP)        ;THRU
9304 057064 013746 057454      MOV    0.XXX,-(SP)    ;5
9305 057070 005746                TST    -(SP)           ;PUT SAVED REGISTER ON STACK
9306 057072 000200                RTS     R0

```



```

9307
9308
9309
9310 057074 113737 177560 057460
9311 057102 113737 177564 057461
9312 057110 105037 177560
9313 057114 105037 177564
9314 057120 004537 057430
9315 057124 000205
9316
9317
9318
9319 057126 004537 057430
9320 057132 105737 177564
9321 057136 100375
9322 057140 032737 004000 177560
9323 057146 001403
9324 057150 105737 177560
9325 057154 100375
9326 057156 113737 057460 177560
9327 057164 113737 057461 177564
9328 057172 000205
9329
9330
9331
9332
9333
9334 057174 010246
9335 057176 012704 057517
9336 057202 012746 000060
9337 057206 010002
9338 057210 042702 177770
9339 057214 061602
9340 057216 110244
9341 057220 006200
9342 057222 006200
9343 057224 006200
9344 057226 020427 057512
9345 057232 101365
9346 057234 042700 177776
9347 057240 062600
9348 057242 110044
9349 057244 012703 057517
9350 057250 004537 057322
9351 057254 012602
9352 057256 000205
9353
9354
9355
9356 057260 105737 177560
9357 057264 100375
9358 057266 113700 177562
9359 057272 004537 057336
9360 057276 042700 177600
9361 057302 001766
9362 057304 122700 000040

: SAVE TELETYPE STATUS
O.SVTT: MOV 0.RCSR,0.CSR1 ;SAVE R C/SR
        MOV 0.TCSR,0.CSR2 ;SAVE T C/SR
        CLRB 0.RCSR ;CLEAR ENABLE AND MAINTENANCE
        CLRB 0.TCSR ;BITS IN BOTH C/SR
        JSR 5,0.CRLF ;TYPE <CR,LF>
        RTS R5

: RESTORE TELETYPE STATUS
O.RSTT: JSR 5,0.CRLF ;<CR,LF> BEFORE RESTORING
        TSTB 0.TCSR ;WAIT READY ON PRINTER
        BPL -4
        BIT #4000,0.RCSR ;CHECK BUSY FLAG ON READER
        BEQ 0.RSE1 ;SKIP READY LOOP IF NOT BUSY
        TSTB 0.RCSR ;WAIT READY
        BPL -4 ;ON READER
O.RSE1: MOV 0.CSR1,0.RCSR ;RESTORE
        MOV 0.CSR2,0.TCSR ;THE STATUS REGISTERS
        RTS R5

: TYPE OUT CONTENTS OF WORD OR BYTE WITH ONE TRAILING SPACE
: WORD IS IN R0
O.CADV: MOV R2,-(SP) ;SAVE R2
        MOV #0,BUF+6,R4 ;BUFFER START ADDRESS
        MOV #0,-(SP) ;CONSTANT ASCII 0
O.SPC: MOV R0,R2 ;GET
        BIC #177770,R2 ;OCTAL CHARACTER
        ADD @SP,R2 ;CONVERT TO ASCII
        MOV R2,-(R4) ;STORE IN BUFFER
        ASR R0 ;SHIFT THIS MESS
        ASR R0 ;RIGHT
        ASR R0 ;THREE WHOLE PLACES
        CMP R4,#0.BUF+1 ;DONE?
        BHI 0.SPC ;NO
        BIC #177776,R0 ;GET LAST BIT
        ADD (SP)+,R0 ;CONVERT TO ASCII
        MOV R0,-(R4) ;AND PUT IT AWAY
        MOV #0,BUF+6,R3 ;LWA
        JSR 5,0.TYPE ;TYPE WHOLE STRING OF CHARACTERS
        MOV (SP)+,R2 ;RESTORE R2
        RTS R5

: GENERAL CHARACTER INPUT ROUTINE
: CHARACTER INPUT GOES TO R0
O.GET: TSTB 0.RCSR ;WAIT FOR
        BPL -4 ;INPUT FROM KEYBOARD
        MOV 0.RDB,R0 ;GET A CHARACTER
        JSR 5,0.F1YP ;ECHO CHARACTER
        BIC #177600,R0 ;STRIP OFF PARITY FROM CHARACTER
        BEQ 0.GET ;IGNORE NULLS
        CMPB #40,R0 ;CHECK FOR SPACES
    
```

9363	057310	001763		BEQ	0.GET	; IGNORE NULLS
9364	057312	122700	000073	CMPB	#.RO	; CHECK FOR SEMI-COLON
9365	057316	001760		BEQ	0.GET	; IGNORE THEM IF FOUND
9366	057320	000205		RTS	RS	
9367						
9368						
9369						
9370						
9371						
9372	057322	020304		0.TYPE: CMP	R3,R4	; CHECK FOR COMPLETION
9373	057324	103426		BLO	0.TYP1	; EXIT WHEN DONE
9374	057326	112400		MOVB	(R4)+,R0	; GET A CHARACTER
9375	057330	004537	057336	JSR	5,0.FTYP	; TYPE ONE CHARACTER
9376	057334	000772		BR	0.TYPE	; LOOP UNTIL DONE
9377						
9378						
9379						
9380	057336	105737	177564	0.FTYP: TSTB	0.TCSR	; CHECK STATUS
9381	057342	100375		BPL	-4	; WAIT UNTIL READY
9382	057344	110037	177566	MOVB	R0,0.TDB	; TYPE ONE CHARACTER
9383	057350	120037	000045	CMPB	R0,#45	; IS CHAR TO BE FILLED?
9384	057354	001012		BNE	0.TYP1	; NO
9385	057356	113746	000044	MOVB	#44,-(SP)	; YES, INIT THE COUNT
9386	057362	105737	177564	0.TYP2: TSTB	0.TCSR	
9387	057366	100375		BPL	0.TYP2	
9388	057370	105037	177566	CLRB	0.TDB	; GENERATE NULL FILLER
9389	057374	105316		DECB	#SP	
9390	057376	003371		BGT	0.TYP2	
9391	057400	005726		TST	(SP)+	; POP STACK
9392	057402	000205		0.TYP1: RTS	RS	
9393						
9394						
9395						
9396						
9397	057404	006205		0.TCLS: ASR	RS	; GET LOW ORDER BIT
9398	057406	103405		BCS	0.TC	; JUMP IF ALREADY CLOSED
9399	057410	006305		ASL	RS	
9400	057412	005702		TST	R2	; IF NO NUMBER WAS TYPED THERE IS
9401	057414	001401		BEQ	0.CLS1	; NO CHANGE TO THE OPEN CELL
9402	057416	010415		MOV	R4,#RS	; STORE WORD
9403	057420	000207		0.CLS1: RTS	PC	
9404	057422	005746		0.TC: TST	-(SP)	; POP EXTRA CELL FROM STACK
9405	057424	000137	055726	JMP	0.ERR	; AND SCREAM BLOODY MURDER
9406						
9407						
9408						
9409						
9410	057430	012703	057465	0.CRLF: MOV	#0.CR+1,R3	; LWA <CR,LF>
9411	057434	000402		BR	0.CRS	
9412	057436	012703	057466	0.CRLS: MOV	#0.CR+2,R3	; LWA <CR,LF>*
9413	057442	012704	057464	0.CRS: MOV	#0.CR,R4	; FWA
9414	057446	004537	057322	JSR	5,0.TYPE	; TYPE SOMETHING
9415	057452	000205		RTS	RS	
9416						
9417	057454	000000		0.XXX: .WORD	0	; TEMPORARY STORAGE
9418	057456	000		0.T: .BYTE	0	; T-BIT FLAG

```

9419 057457 000 O.P: .BYTE 0 ;PROCEED FLAG = 0 IF PROCEED NOT ALLOWED
9420 057460 000 O.CSR1: .BYTE 0 ; = 1 IF PROCEED ALLOWED
9421 057461 000 O.CSR2: .BYTE 0 ;SAVE CELL - R C/SR
9422 057462 000 ;SAVE CELL - T C/SR
9423
9424
9425 057462 042502 O.BD: .EVEN
9426 .WORD "BE
9427 057464 015 O.CR: .BYTE 015 ; (CR)
9428 057465 012 .BYTE 012 ; (LF)
9429 057466 052 .BYTE * ; *
9430
9431 057467 057 O.LGCH: .BYTE / ;
9432 057470 015 .BYTE 015 ; CARRIAGE RETURN
9433 057471 044 .BYTE 4 ;
9434 057472 107 .BYTE 7 ;
9435 057473 012 .BYTE 012 ; (LF)
9436 057474 137 .BYTE 7 ;
9437 057475 136 .BYTE 6 ;
9438 057476 117 .BYTE 7 ;
9439 057477 127 .BYTE 7 ;
9440 057500 105 .BYTE 5 ;
9441 057501 102 .BYTE 2 ;
9442 057502 120 .BYTE 0 ;
9443 000014 O.CLGT = .-O.LGCH ;TABLE LENGTH
9444
9445 057503 123 O.TL: .BYTE 3 ;DO
9446 057504 120 .BYTE 0 ;NOT
9447 057505 115 .BYTE 5 ;CHANGE
9448 057506 000 .BYTE 0 ;THE
9449 057507 000 .BYTE 0 ;ORDER
9450 057510 102 .BYTE 2 ;HERE 6
9451 000006 O.LG = .-O.TL
9452
9453 057511 O.BUF: ;6 CHAR. BUFFER WITH
9454 057517 057517 = ;+6 ;TRAILING BLANK
9455 057517 040 .BYTE
9456 .EVEN
9457
9458 057520 000003 O.TRTC: TRT ;TRACE TRAP PROTOTYPE
9459
9460 ;THE ORDER OF THE FOLLOWING ENTRIES IS CRITICAL
9461
9462 055412 O.URD: = 0 O.ODT-40
9463 055412 000000 ;USER R0
9464 055414 000000 ; R1
9465 055416 000000 ; R2
9466 055420 000000 ; R3
9467 055422 000000 ; R4
9468 055424 000000 ; R5
9469 055426 000000 O.USP: ;USER SP
9470 055430 000000 O.UPC: ;USER PC
9471 055432 000000 O.UST: ;USER ST
9472 055434 000007 O.PRI: ;ODT PRIORITY
9473 055436 000000 O.MSK: ;MASK
9474 055440 000000 ;LOW LIMIT

```

C2R6JED UNIBUS RK6 DR PRT3
C2F6JE.P11 25-JAN-78 12:08

MACY11 30A(1052) 25-JAN-78 12:16 PAGE 179
TYPE ERROR ROUTINE

SEQ 0179

9475 055442 000000
9476
9477
9478
9479
9480 055444 000000
9481 055446 000000
9482 055450 000000
9483 000001

0 ;HIGH LIMIT
: BREAK POINT LISTS, ADP1 = ADDRESS OF BREAKPOINT, CT = COUNT,
: UIN = CONTENTS
: O.ADR1: 0
: O.CT: 0
: O.UIN: 0
.END

ABASE = 177440	1817	1858	1882*		
ACDW1 = 000000	1817	1860			
ACDW2 = 000000	1817	1861			
ACLO = 000010	1180*				
ACPUOP = 000000	1817	1832			
ACT11 = 005454	2044*	2991*			
ADDW0 = 000000	1817	1862			
ADDW1 = 000000	1817	1863			
ADDW10 = 000000	1817	1872			
ADDW11 = 000000	1817	1873			
ADDW12 = 000000	1817	1874			
ADDW13 = 000000	1817	1875			
ADDW14 = 000000	1817	1876			
ADDW15 = 000000	1817	1877			
ADDW2 = 000000	1817	1864			
ADDW3 = 000000	1817	1865			
ADDW4 = 000000	1817	1866			
ADDW5 = 000000	1817	1867			
ADDW6 = 000000	1817	1868			
ADDW7 = 000000	1817	1869			
ADDW8 = 000000	1817	1870			
ADDW9 = 000000	1817	1871			
ADEVCT = 000000	1817	1823			
ADEVN = 000000	1817	1859			
AENV = 000000	1817	1828			
AENVN = 000000	1817	1829			
AFATAL = 000000	1817	1820			
AMADR1 = 000000	1817	1845			
AMADR2 = 000000	1817	1849			
AMADR3 = 000000	1817	1852			
AMADR4 = 000000	1817	1855			
AMAMS1 = 000000	1817	1839			
AMAMS2 = 000000	1817	1847			
AMAMS3 = 000000	1817	1850			
AMAMS4 = 000000	1817	1853			
AMSGAD = 000000	1817	1825			
AMSQLG = 000000	1817	1826			
AMSGTY = 000000	1817	1819			
AMTYP1 = 000000	1817	1840			
AMTYP2 = 000000	1817	1848			
AMTYP3 = 000000	1817	1851			
AMTYP4 = 000000	1817	1854			
APASS = 000000	1817	1822			
APRIOR = 000000	1817				
APTCSU = 000040	6468	6640*	6461	6596	6638*
APTENV = 000001	5125	6415			
APTSIZ = 000200	2918	6637*			
APTSPO = 000100	6463	6598	6639*		
ASWREG = 000000	1817	1830			
ATESTN = 000000	1817	1821			
ATTN = 005330	1979*	5333	5353	5380	
AUNIT = 000000	1817	1824			
AUSWR = 000000	1817	1831			
AVECT1 = 000000	1817	1856			
AVECT2 = 000000	1817	1857			
BADHDR = 005324	1969*	2999*	6126		

O.BKPT	055676	9032#	9090						
O.BK1	056630	8980	9254#						
O.BK2	056664	9259	9262#						
O.BK3	056702	9261	9268#						
O.BRK	056620	8997	9252#						
O.BUF	057511	9334	9343	9348	9453#				
O.B2	056764	9273	9282#						
O.B3	057024	9281	9290#						
O.CADV	057174	9103	9120	9143	9147	9181	9185	9290	9333#
O.CLGL	056012	9059	9061	9069#					
O.CLGT=	000014	9073	9443#						
O.CLS1	057420	9401	9403#						
O.CR	057464	9410	9412	9413	9427#				
O.CRET	056124	9081	9110#						
O.CRLF	057430	9110	9179	9275	9314	9319	9410#		
O.CRLS	057436	9054	9412#						
O.CRS	057442	9411	9413#						
O.CSR1	057460	9310#	9326	9421#					
O.CSR2	057461	9311#	9327	9422#					
O.CT	055446	9239#	9282*	9284*	9481#				
O.C1	056576	9242#	9283						
O.DCD	055742	9039	9044	9054#	9112	9167	9292		
O.DCD1	055746	9055#	9104	9148					
O.EFF	056252	9089	9158#						
O.EFF1	056370	9170	9189#						
O.ENTR	055456	8977#							
O.ERR	055726	9024	9035	9051#	9074	9100	9162	9405	
O.ERR1	056262	9162#	9235						
O.ERR2	056102	9100#	9134	9211					
O.FTYP	057336	9053	9122	9137	9183	9288	9359	9375	9380#
O.GET	057260	9003	9057	9356#	9361	9363	9365		
O.GO	056434	9083	9209#						
O.GO2	056512	9220#	9245						
O.LG =	000006	9007	9451#						
O.LGCH	057467	9070	9431#	9443					
O.LGDR	056042	9077	9080#	9092					
O.LGL =	000030	9092#							
O.LGL1	056014	9070#	9075						
O.LGL2	056034	9071	9076#						
O.MSK	055436	9163	9164	9166	9473#				
O.OOT	055452	1289	8975#	9462					
O.OFST	056200	9087	9133#						
O.OF1	056246	9146	9148#						
O.OPI	056136	9084	9116#						
O.OP2	056144	9028	9118#	9129					
O.ORPC	055650	9020#	9085						
O.P	057457	8995#	9209*	9234	9236*	9285*	9419#		
O.PRI	055434	8987	9258	9262	9472#				
O.PROC	056542	9091	9234#						
O.PRI	056564	9238	9240#						
O.RALL	055716	8998	9033	9043#					
O.RCSR=	177560	8962#	9310	9312*	9322	9324	9326*	9356	
O.ROB =	177562	8961#	9358						
O.REGT	055600	9003#	9082						
O.RSE1	057156	9323	9326#						
O.RSP	055610	9005#	9008						

G16

CZR6JEO UNIBUS RK6 DR PRT3
CZR6JE.P11 25-JAN-78 12:08

MACY11 30A(1052) 25-JAN-78 12:16 PAGE 203
CROSS REFERENCE TABLE -- MACRO NAMES

SEQ 0201

CALIB	1410#	6129													
CHECK	1367#	3567	3592	3646	4057	4135	4312	4354	4400	4438	4593	4651	4755	4827	4908
	4966														
COMMEN	1087#														
CWD2	1382#														
DRCLR	1395#	6143													
ENDCOM	1087#														
ERROR	981#	3078	3136	3149	3164	3173	3196	3205	3224	3228	3239	3243	3277	3286	3304
	3331	3335	3339	3343	3442	3480	3488	3492	3510	3514	3516	3540	3561	3570	3571
	3572	3573	3576	3586	3595	3596	3597	3598	3601	3612	3616	3626	3640	3649	3650
	3651	3652	3658	3662	3667	3671	3673	3693	3704	3708	3713	3723	3729	3769	3784
	3798	3812	3840	3854	3862	3866	3874	3882	3886	3915	3934	3942	3950	3954	3959
	3963	3970	3974	3990	3998	4002	4009	4023	4028	4032	4037	4041	4045	4050	4060
	4061	4062	4063	4066	4069	4075	4079	4108	4122	4126	4138	4139	4140	4141	4143
	4149	4151	4189	4194	4198	4202	4224	4241	4260	4264	4273	4284	4297	4303	4315
	4316	4317	4318	4321	4325	4335	4342	4345	4357	4358	4359	4360	4367	4377	4386
	4391	4394	4403	4404	4405	4406	4409	4419	4426	4429	4441	4442	4443	4444	4451
	4467	4481	4487	4500	4505	4552	4574	4581	4584	4596	4597	4599	4599	4632	4639
	4642	4654	4655	4656	4657	4679	4702	4720	4724	4733	4741	4758	4759	4760	4761
	4767	4773	4788	4802	4809	4820	4830	4831	4832	4833	4843	4847	4854	4858	4867
	4889	4896	4899	4911	4912	4913	4914	4947	4954	4957	4969	4970	4971	4972	5045
	5066	5086	5531	5536	5540	5555	5568	5576	5584	5591	5617	6097	6109	6117	6121
	6133	6139	6142	6148	6151	6210									
ESCAPE	1087#														
F. EAB	1347#	4127	4304	4346	4430	4595	4643	4747	4900	4958					
GETPRI	1087#														
GETSWR	1087#	5152													
XDCHK3	1484#														
XDTBL	1497#														
LOOP	1331#	3157	3270												
MSG	3036#	3038	3082#	3084	3250#	3252	3359#	3361	3428#	3430	3469#	3471	3496#	3498	3521#
	3523	3676#	3678	3827#	3829	3898#	3900	3978#	3980	4083#	4085	4207#	4209	4665#	4667
	4982#	4984	5001#	5003											
MULT	1087#														
NEWST	1087#	3036	3082	3250	3359	3428	3469	3496	3521	3676	3827	3898	3978	4083	4207
	4665	4982	5001												
OWNTAG	1325#	1882													
POP	1087#	6570	6630	6631	7016	7212	7254								
PUSH	1087#	6529	6591	6593	6614	6990	7183	7234							
QKPACK	1731#	3484	3654	3965	4071	4850									
QKSEEK	1509#	4191													
QKSRT	1707#	3988	6106												
QKUNLD	1747#	3508	3665												
RDATA	1578#														
ROHDR	1455#														
REPORT	1087#														
SBOUND	1652#	4550	4864												
SCOPE	982#	3046	3097	3265	3368	3435	3475	3503	3535	3688	3835	3910	3985	4103	4216
	4674	4992	5022	5096											
SETPRI	1087#	6894													
SETTRA	7284#	7293	7294	7295	7296	7298	7300	7301	7302	7303	7304	7305	7306		
SETUP	1087#	2879													
SKIP	1087#	3024	3075	3301	3414	3420	3440	3483	3885	3891	4160	4998			
SLASH	1087#														
SPACE	1087#														
STARS	1087#	1293	1304	1306	1313	1766	1813	1816	3036	3045	3082	3096	3250	3264	3359

H16

CZR6JEO UNIBUS RK6 DR PRT3
CZR6JE.P11 25-JAN-78 12:08

MACY11 30A(1052) 25-JAN-78 12:16 PAGE 204
CROSS REFERENCE TABLE -- MACRO NAMES

SEQ 0202

	3367	3428	3434	3469	3474	3496	3502	3521	3534	3676	3687	3827	3834	3898	3909
	3978	3984	4082	4102	4207	4215	4665	4673	4982	4991	5001	5021	5089	5322	5386
	6440	6519	6586	6643	6720	6797	6812	6883	6907	6976	7029	7068	7130	7148	7171
	7218	7263	8853	8862											
SWRSU	1087#	2901#													
TRMTRP	7284#														
TYPBIN	1087#														
TYPDEC	1087#	5108													
TYPNAM	1087#														
TYPNUM	1087#														
TYPOCS	1087#	3118	3188	3234	3400	3739	3785	3803	5070	5408	6183	6228			
TYPOCT	1087#	6825													
TYPTXT	1087#														
WDATA	1536#	4281													
WRCHK	1619#	4332	4416	4571	4629	4886	4944								
WRHDR	1434#	4257	4717												
SSCMRE	1764#														
SSCMTM	1764#	1801	1802	1803	1804	1805	1806								
SSESCA	1087#														
SSNEWT	1087#	3036	3082	3250	3359	3428	3469	3496	3521	3676	3827	3898	3978	4083	4207
	4665	4982	5001												
SSSET	7284#	7293	7294	7295	7296	7298	7300	7301	7302	7303	7304	7305	7306		
SSSETM	2917#														
SSSKIP	1087#	3024	3075	3301	3414	3420	3440	3483	3885	3891	4160	4998			
.EQUAT	942#	977													
.HEADE	942#	946													
.SETUP	942#	2850													
.SWRHI	942#	956													
.SWRLO	942#	968#													
.SACT1	942#	1291													
.SAPT8	1814#														
.SAPTH	942#	1302													
.SAPTY	942#	6584													
.SCATC	942#	127													
.SCMTA	942#	1764													
.SOB2D	942#	7066													
.SOB2O	942#	7027													
.SEOP	942#	5087													
.SERRO	942#	6384													
.SMULT	942#	7169													
.SROOC	942#	6974													
.SREAD	942#	6718													
.SSAVE	942#	7216													
.SSB2D	942#	7128													
.SSCOP	942#	7320													
.SSUPR	942#	16													
.STRAP	942#	7261													
.STYPD	942#	6517													
.STYPE	942#	6438													
.STYPO	942#	6641													

. ABS. 057522 000

ERRORS DETECTED: 0

CZR6JEO UNIBUS RK6 DR PRT3
CZR6JE.P11 25-JAN-78 12:08

MACY11 30A(1052) 25-JAN-78 12:16 PAGE 205
CROSS REFERENCE TABLE -- MACRO NAMES

SEQ 0203

DSKZ:CZR6JE, DSKZ:CZR6JE, SEQ/SOL/CRF/NL: TOC/DOC=DSKM:CZR6JE.P11
RUN-TIME: 29 23 3 SECONDS
RUN-TIME RATIO: 336/55=6.0
CORE USEC: 30K (59 PAGES)

DOCUMENT PAGES: 203

EOF1CZR6JESEQ 00010000 780223 PDP10 411 S