

RK06

DISK DRIVE DIAG. PART 2
MD-11-DZR6I-C

EP-DZR6I-C-DL-B
COPYRIGHT © 1976
FICHE 1 OF 2

DEC 1976
digital
MADE IN USA

The main body of the document is a large grid of 100 small technical diagrams or data tables, arranged in 10 rows and 10 columns. Each cell contains a small, dense technical drawing or data set, likely related to disk drive diagnostics. The diagrams are too small to read individually but appear to be organized into a structured format, possibly representing different components or test procedures for the MD-11-DZR6I-C disk drive.

RK06

DISK DRIVE DIAG. PART 2
MD-11-DZR6I-C

EP-DZR6I-C-DL-B

COPYRIGHT © 1976

FICHE 2 OF 2

DEC 1976

digital

MADE IN USA

UNIBUS RK06 DRIVE DIAGNOSTIC PART 2
DZR6I.P11 07-OCT-76 13:50

.REM %

IDENTIFICATION

PRODUCT CODE:	MAINDEC-11-DZR6I-C-D
PRODUCT NAME:	UNIBUS RK06 DISK DRIVE DIAGNOSTIC: PART 2
DATE:	DECEMBER 1976
MAINTAINER:	DIAGNOSTIC GROUP
AUTHOR:	GARY PAPAZIAN

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

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

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

COPYRIGHT (C) 1976 BY DIGITAL EQUIPMENT CORPORATION

UNIBUS DRIVE DIAGNOSTIC PART 2
DZRGIC.P11 07-OCT-76 13:50

TABLE OF CONTENTS

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

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 2 OF THE DRIVE DIAGNOSTICS TO INSURE THAT THE DISK IS CAPABLE OF PERFORMING READ AND WRITE DATA OPERATIONS IN BOTH 20 AND 22 SECTOR FORMATS. WORST CASE PATTERNS, SPIRAL WRITING AND READING, AND ALL OFFSET OPERATIONS ARE PERFORMED. ALSO, UNLOADING AND LOADING TIMES ARE REPORTED ALONG WITH ROTATIONAL MIN, MAX, 137 CYLINDER & MAX VELOCITY TIMES. ERROR DETECTION LOGIC IS CHECKED BY SOFTWARE ERROR FORCING.

AFTER A SUCCESSFUL RUN (WITH NO ERRORS) OF PART 2, THE DRIVE IS READY FOR PART 3 OF THE DRIVE DIAGNOSTICS.

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.

*****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 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 DIAGNOSTIC- PART 1.

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

3.0 PROGRAM CONSIDERATIONS

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 CAN BE CHAINED BY XXDP & WILL NOT OVERLAY THE
LOADER.

CHAIN MODE OPERATION (MONITOR)

1. THE INPUT DIALOGUE IS BYPASSED.
2. THE BUSS ADDRESS & CONTROLLER INTERRUPT VECTOR IS
DEFAULTED.
3. DRIVE 0 WILL NOT BE TESTED.
4. ALL OTHER DRIVES IN THE 'DRIVE PRESENT' CONDITION WILL
BE TESTED.

NOTE: THE DRIVE PRESENT CONDITION IS:

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

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. IT IS APT
COMPATIBLE TO THE EXTENT THAT APT HOOKS WILL BE IN THE
PROGRAM & WILL WORK THRU THE 'UPTON INTERFACE'.

FOR OTHER INTERFACES, APT MAY ONLY LOAD & START THE PROGRAM.
I.E. LOAD & DUMP MODE.

AUTOMATIC MODE (MONITOR)

1. THE INPUT DIALOGUE IS BYPASSED.

2. THE BUSS ADDRESS & CONTROLLER INTERRUPT VECTOR IS DEFAULTED.
3. ALL DRIVES IN THE 'DRIVE PRESENT' CONDITION WILL BE TESTED.

NOTE: THE DRIVE PRESENT CONDITION IS:

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

DUMP MODE (MANUAL): INPUT DIALOGUE IF STARTED FROM 22C.

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: BYTE
 - 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
 - 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:

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

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

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
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

THE EXECUTION TIMES SHOWN BELOW ARE BASED ON THE PDP 11-50.

TOTAL TIME: 2 MIN, 40 SEC

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

A BREAKDOWN OF THE MORE LENGTHY TESTS ARE SHOWN BELOW:

TEST 20	DRIVE OFF TRACK	45 SEC
TEST 21	UNLOAD HEADS	10 SEC
TEST 22	LOAD HEADS	20 SEC
TEST 23	ROTATIONAL TIMING	15 SEC
TEST 24	MAX SEEK TIMES	15 SEC
TEST 26	137 CYL SEEK TIMES	10 SEC
TEST 27	MAX VELOCITY TIMES	15 SEC

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 BUSS ADDRESS	1264	177440
CONTROLLER INTERRUPT VECTOR	1314	210
CONTROLLER PRIORITY	1316	240
P-CLOCK STATUS REG	1320	172540
P-CLOCK SET BUFFER	1322	172542
P-CLOCK READ BUFFER	1324	172544
L-CLOCK STATUS REG	1326	177546
L-CLOCK INTERRUPT VECTOR	1330	100
P-CLOCK INTERRUPT VECTOR	1332	104
TTY KB STATUS REG	1144	177560
TTY KB BUFFER	1146	177562
TTY P-INTER 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.

I01

UNIBUS RK06 DRIVE DIAGNOSTIC PART 2
DZR6IC.P11 07-OCT-76 13:50

MACY11 27(1006) 07-OCT-76 14:14 PAGE 8

SEG 0008

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

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 WAS 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:

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

LOCATION 204 - SAME AS 200 START BUT BYPASS WRITE TESTS.

LOCATION 214 - SAME AS 200 START BUT BYPASS TIMING TESTS.

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

1ST PASS.

LOCATION 224 - SAME AS 220 START BUT BYPASS WRITE TESTS.

LOCATION 230 - SAME AS 220 START BUT BYPASS TIMING TESTS.

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

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
8	LOOP ON TEST IN SW<07:00>

4.3.1 SW<15>

THE PROGRAM HALTS ON ENCOUNTERING AN ERROR, AFTER TYPING OUT
THE ERROR MESSAGE AND PERTINENT INFORMATION, IF SW13=0.
PRESSING "CONTINUE" RESTORES 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>

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

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

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

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.

4.3.8 SW<08>

THIS SWITCH IS USED TO SELECT A PARTICULAR TEST (AS PER SW<00-7>) FOR EXECUTION AND SUBSEQUENT LOOPING. THUS IF TEST 15 IS TO BE SELECTED THE SWITCH SETTING WOULD BE 000415. IT SHOULD BE NOTED THAT BEFORE SELECTING TEST 15, ALL THE PREVIOUS TESTS (1-14) WILL BE EXECUTED.

4.4 'SOFTWARE' SWITCH REGISTER

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

SWR = NNNNNN NEW =

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

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

4.5 INPUT DIALOGUE

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

THE DIALOGUE WILL BE DONE INTE-ACTIVELY. THE PROGRAM WILL REQUEST A PARAMETER BY CONSOLE TYPEOUT. THE PARAMETER MAY THEN BE ENTERED AS SPECIFIED BELOW OR ALLOWED TO DEFAULT 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 DRIVE DIAGNOSTIC
PART 2

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

MAINDEC-11-DZR6I-C-PB

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 DRIVE DIAGNOSTIC
PART 2
MAINDEC-11-DZR6I-C-PB

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 SERIAL NO. AAA
CARTRIDGE SERIAL NO. BBB

DRIVE 1

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
703DRIVE SERIAL NO. CCC
CARTRIDGE SERIAL NO. DDD

END PASS # 2

(ETC)

THE ABOVE ASSUMES NO ERRORS DETECTED.
THE NUMBER OF PASSES IS DETERMINED BY ACT/APT/XXDPIMPORTANT: FOR VARIATIONS OF THE ABOVE, SEE XXDP, 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 (↑) ON THE CONSOLE.

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

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

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

1. ECHO ↑
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.

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

5.0 DRIVE DIAGNOSTIC FUNCTIONAL DESCRIPTION

5.1 GENERAL

A. WRITE TESTS

THESE TESTS CHECK THE ABILITY OF THE DRIVE TO WRITE & READ
WORSE CASE PATTERNS; PERFORM ALL OFFSETS & PERFORM ALL
SPIRAL WRITING.

B. SERVO & SPINDLE TIMING TESTS

THESE TESTS CHECK & TYPE HEAD LOAD, UNLOAD & INDEX TIMING,
ALSO MIN, MAX, AND AVERAGE SEEK TIMES, AND MAX VELOCITY
OF THE HEADS ARE MEASURED & TYPED.

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. 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.

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. IT WILL THEN CHECK
TO SEE THAT THE DRIVE WAS INPUTTED FOR TESTING. IF NOT, IT WILL
BE AN ERROR. IF CERR WAS SET, THAT DRIVE WILL BE BYPASSED
ONLY IF THE ERROR WAS A RESULT OF MDS OR UFE SET OR BOTH

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

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.

TEST 4 FIND NEXT DRIVE TO BE TESTED

THIS TEST FINDS THE NEXT DRIVE PRESENT & PUTS THAT
ADDRESS IN 'DRVAD'.
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 READ & SAVE BAD SECTOR INFO & TYPE PACK SERIAL #

THIS TEST VERIFIES THAT CYL 410, TRACK 2 CAN BE READ.
THIS AREA CONTAINS BAD SECTOR INFO WHICH IS WRITTEN BY THE
FACTORY DURING MANF. ALL BAD SECTOR INFO (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 INFO OR AN ALIGNMENT PACK BY WRITI
THE PACK SERIAL # IS TYPED IN OCTAL & FOR THE FIRST PASS ONLY.

WRITE TESTS

TEST 10 BASIC WRITE DATA TEST; 1 WORD

THIS TEST VERIFIES THE ABILITY OF THE DRIVE TO WRITE JUST ONE WORD.
ALL SECTORS ON CYL 0 ARE GIVEN IDENTICAL HEADERS &
A WRITE COMMAND IS ISSUED. READ & WRITE CHECK COMMANDS ARE NOT
PERFORMED. THIS TEST PROVIDES THE TIGHTEST POSSIBLE SCOPE LOOP
FOR A WRITE ERROR.

TEST 11 BASIC WRITE DATA TEST; FULL SECTOR

THIS TEST VERIFIES THE ABILITY OF THE DRIVE TO WRITE

016
 017
 018
 019
 020
 021
 022
 023
 024
 025
 026
 027
 028
 029
 030
 031
 032
 033
 034
 035
 036
 037
 038
 039
 040
 041
 042
 043
 044
 045
 046
 047
 048
 049
 050
 051
 052
 053
 054
 055
 056
 057
 058
 059
 060
 061
 062
 063
 064
 065
 066
 067
 068
 069
 070
 071

A FULL SECTOR. ALL ZEROS ARE WRITTEN BY THE WRITE DATA COMMAND
 & CHECKED BY A RD DATA COMMAND. A FURTHER CHECK IS PERFORMED
 BY THE WRT CHK COMMAND.
 THE ABOVE IS REPEATED FOR AN ALL ONES PATTERN.

TEST 12 20 SECTOR FORMAT TEST

DATA IS WRITTEN ON A FULL TRACK IN 20 SECTOR FORMAT.
 MSG 80,81 ARE CHECKED FOR ANY ERROR CONDITION

TEST 13 TEST OFFSET & RTC LOGIC

THE HEADS ARE FIRST OFFSET BY OFFSET COMMANDS.
 THIS TEST CHECKS THE RTC LOGIC BY VERIFYING THAT THE
 'OFFSET ON' BIT (MSG A,00) RESETS AND THE OFFSET REG

BECOMES THE CYL DIFF INFO WHEN A SEEK CMD TO A
 DIFFERENT CYLINDER IS ISSUED
 IT ALSO TESTS THAT DRIVE CLEAR & SEEK TO SELF WILL NOT
 CLEAR THE 'OFFSET ON' BIT OR THE OFFSET REG.
 ALL OFFSET POSITIONS IN BOTH DIRECTIONS ARE CHECKED

TEST 14 TEST READ DATA AT ALL HEAD OFFSET POSITIONS

THIS TEST VERIFIES THAT THE HEAD OFFSET LOGIC IS OPERATIONAL BY
 WRITING ALL 1'S PATTERNS ON CYLINDER 0, HEAD 0. THEN
 PERFORMING READ DATA FROM CENTERLINE AND MOVING OUT + AND - OFFSET
 POSITIONS UNTIL A FAILURE OCCURES. THE FAILING OFFSET POSITIONS
 ARE PRINTED OUT IF LESS THAN THE OFFSET TOLERANCE TO BE SPECIFIED.
 OFFSET CODES ARE ALSO VERIFIED BY READING MSG A, STATUS 00 & 10.
 ALL HEADS ARE TESTED AT CYL 0.

IF THERE ARE NO FAILURES AT ALL, THIS INDICATES THAT

OR A. HEADS DID NOT MOVE AT ALL
 B. THE COMBINATION OF DISC SURFACE, HEADS, R/W AMP
 ARE EXCEPTIONALLY GOOD.

AN APPROPRIATE MESSAGE WILL BE TYPED.

TEST 15 WRITE WITH HEADS OFFSET

THIS TEST VERIFIES THAT WHEN ATTEMPTING TO
 WRITE WITH HEADS OFFSET THAT THE OFFSET WILL CLEAR
 & THE DRIVE WILL WRITE
 SINCE THE WRITE COMMAND HAS AN IMPLIED RTC.
 THIS TEST IS PERFORMED FOR MAX POS & NEG OFFSETS ONLY

TEST 16 TEST CURRENT CROSS-OVER CYLINDERS

THIS TEST VERIFIES THAT THE DRIVE CAN WRITE & READ OFF

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

CURRENT CHANGE CYLINDERS X & Y IN THE FOLLOWING WAY:

SPIRAL WRITING IS PERFORMED FROM CYLINDER X TO CYLINDER Y WITH A DATA PATTERN FILLING THE ENTIRE 2 CYLINDERS.

A WRITE CHECK IS THEN PERFORMED TO VERIFY DATA WAS PROPERLY WRITTEN. THIS TEST IS PERFORMED FOR ALL 3 HEADS.

CYLINDER X: 63 127 191 255 319 383
CYLINDER Y: 64 128 192 256 320 384

TEST 17 TEST HEAD SWITCHING TIME

TESTS THE ABILITY TO SWITCH HEADS IN LESS THEN 10MS WHEN HEADS SPIRAL.

1. SECTOR 17 IS FIRST LOCATED AND A WRITE DATA COMMAND OF 512 WORDS TO SECTOR 21 IS ISSUED.
2. THE PROGRAM NOW KNOWS THAT THE DRIVE WILL NOT HAVE TO TRAVEL A FULL REVOLUTION BEFORE FINDING SECTOR 21.
3. SINCE EACH SECTOR TAKES APPROX. 1.2MS, THE TIME BETWEEN THE START OF THE WRITE COMMAND (FROM SECTOR 21, HEAD 0; TO SECTOR 0, HEAD 1) AND CONTROLLER READY SHOULD BE APPROX 6MS

THE ABOVE IS REPEATED FOR HEAD SWITCHING BETWEEN 1 TO 2

THIS TEST IS BYPASSED IF NEITHER L OR P CLOCK IS PRESENT

TEST 20 DRIVE OFF TRACK TEST

THIS TEST CHECKS FOR SERVO OSCILLATIONS DURING SETTLING TIME BEYOND THE ALLOTTED 3MS.

1. INITIALLY, EVERY CYLINDER IS FORMATTED WITH IDENTICAL HEADERS (UNIQUE TO EACH CYLINDER)
2. A FULL SECTOR WRITE COMMAND IS ISSUED BY A SINGLE CYL SEEK FROM 0 TO AS HEADERS ARE IDENTICAL, THE NEXT SECTOR TO COME UNDER THE HEADS WILL IMMEDIATELY BE WRITTEN.
3. IF THERE IS OSCILLATION SENSED BY READING THE TRIBITS, DRIVE OFF TRACK ERROR WILL SET.

IN THIS MANNER OSCILLATING SEEKS ARE PERFORMED BETWEEN ALL MAJOR CYLINDER 100 OSCILLATIONS ARE PERFORMED AT EACH MAJOR CYLINDER BEFORE DOING THE NEXT CYLINDER

SERVO & SPINDLE TIMING TESTS.

TEST 21 TIME BETWEEN OUTER LIMIT TO HEADS HOME DURING UNLOAD

TIME IS MEASURED FROM ATTN ASSERTING (APPROXIMATES REV & OUTER LIMIT.
TO HEADS HOME ASSERTING. EXPECTED TIME APPROX 500MS

ALL TIMING TESTS ARE BYPASSED IF NEITHER
L OR P CLOCK IS PRESENT & WILL BE INDICATED BY A MESSAGE

TEST 22 TEST LOW VELOCITY TIMES DURING LOADING

THIS TEST ISSUES A START SPINDLE COMMAND
AFTER 'HEADS HOME' HAS BEEN DETECTED FROM THE PREVIOUS TEST.
THE FOLLOWING "LOW VELOCITY" TIMES ARE CHECKED AGAINST
LIMITS TO BE DEFINED:

TIME 1: TIME BETWEEN HEADS HOME NEGATING & SERVO SIG PRES ASSERTING
EXPECTED TIME APPROX 500 MS

TIME 2: TIME BETWEEN OUTER LIMIT & INNER LIMIT
TIME IS MEASURED FROM SERVO SIG PRES (APPROX OUTER LIMIT)
TO REV (APPROX INNER LIMIT)
EXPECTED TIME APPROX 2 SEC

TIME 3: TIME BETWEEN INNER LIMIT & OUTER LIMIT
TIME IS MEASURED FROM REV ASSERTING (FROM ABOVE,
TO REV NEGATING (APPROX OUTER LIMIT)
EXPECTED TIME APPROX 2 SEC

TEST 23 MEASURE ROTATIONAL SPEED

THIS TEST MEASURES INDEX TIMING BY:

- A. CHANGE FORMAT TO 20 SECTOR & READ HEADER.
CONTROLLER RDY STARTS THE TIMER
- B. CHANGE FORMAT TO 22 SECTOR & READ HEADER.
CONTROLLER RDY ENDS THE TIMER.

INDEX TIMING IS THE TIME BETWEEN THE 2 CONT. RDY TIMES.
THIS IS BECAUSE A CHANGE OF FORMAT INHIBITS SECTOR PULSES
UNTIL THE NEXT INDEX APPEARS-THUS KEEPING A GIVEN
FORMAT COMPLETE THROUGHOUT AN ENTIRE CYLINDER

THE TIME IS THE AVERAGE OF 100 READINGS.

TEST 24 MEASURE MAX SEEK TIME

THIS TEST MEASURES THE MAX SEEK TIME BETWEEN CYLINDERS 0 & 410
THE AVERAGE TIME OF 100 SEEKS IN BOTH DIRECTIONS ARE PRINTED
IF NOT WITHIN LIMITS TO BE SUPPLIED.
MAX SEEK TIME SHOULD BE LESS THAN 70MS.

TEST 25 MEASURE MIN SEEK TIME

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

984
985
986
987
988
989
990
991
992
993
994
995
996
997
998
999
1000
1001
1002
1003
1004
1005
1006
1007
1008
1009
1010
1011
1012
1013
1014
1015
1016
1017
1018
1019
1020
1021
1022
1023
1024
1025
1026
1027
1028
1029
1030
1031
1032
1033
1034
1035
1036
1037
1038
1039

THIS TEST MEASURES THE MIN SEEK TIME BETWEEN CYLINDER 0 & 1
THE AVERAGE TIME OF 100 SEEKS IN BOTH DIRECTIONS ARE PRINTED
IF NOT WITHIN LIMITS TO BE SUPPLIED.
MIN SEEK TIME SHOULD BE LESS THAN 10MS.

TEST 26 MEASURE 137 CYLINDER SEEK TIME

THIS TEST MEASURES THE AVERAGE SEEK TIME BETWEEN CYLINDERS 0 & 137
THE AVERAGE TIME OF 100 SEEKS IN BOTH DIRECTIONS ARE PRINTED

IF NOT WITHIN LIMITS TO BE SUPPLIED.
AVERAGE SEEK TIME SHOULD BE LESS THAN 40MS

TEST 27 MEASURE MAX VELOCITY OF HEADS

THIS TESTS MAX VELOCITY BY DOING SEEKS BETWEEN
CYL 0 & 383 AND MEASURING THE TIME BETWEEN CYLINDERS
128 & 256. SINCE THE DISTANCE BETWEEN CYL 128 & 256 IS KNOWN,
THE AVERAGE VELOCITY OF 100 SEEKS IS CALCULATED & TYPED
IF NOT WITHIN THE SPECIFIED LIMITS TO BE SUPPLIED.

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

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

6.0 ERROR REPORTING

6.1 ERROR INTERPRETATION

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

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

EXAMPLE #1

MESSAGE AO ERROR
AFTER START SPINDLE CMD & FWD SET

TEST NO.		PC				
EXPECT	EXPECT	EXPECT	EXPECT	EXPECT	EXPECT	EXPECT
AO	BO	A1	B1	A2	B2	B3
030144	100000	013704	000001			
ACTUAL	ACTUAL	ACTUAL	ACTUAL	ACTUAL	ACTUAL	ACTUAL
AO	BO	A1	B1	A2	B2	B3
140144	100000	101744	000001			
RKCS1	RKCS2	RKASOF	RKER	RKDS	RKDC	
040200	000100	010000	000000	000000	000000	

THE ABOVE EXAMPLE SHOWS EXPECTED & ACTUAL DATA FOR
MESSAGE REGISTERS AO, BO, 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.

EXAMPLE #2:

NO ATTN IN RKASOF
AFTER UNLOAD COMMAND

TEST NO.		PC				
RKMR2	RKMR3	RKER	RKDS	RKCS1	RKCS2	RKASOF
000144	100000	000000	100101	000206	000104	000000

[END OF DOCUMENT]

1040
1041
1042
1043
1044
1045
1046
1047
1048
1049
1050
1051
1052
1053
1054
1055
1056
1057
1058
1059
1060
1061
1062
1063
1064
1065
1066
1067
1068
1069
1070
1071
1072
1073
1074
1075
1076
1077
1078
1079
1080
1081
1082
1083
1084
1085
1086
1087
1088
1089

%

1090
1091
1092
1093
1094
1095
1096
1097
1098
1099
1100
1101
1102
1103
1104
1105
1106
1107
1108
1109
1110
1111
1112
1113
1114
1115
1116
1117
1118
1119
1120
1121
1122
1123
1124
1125
1126
1127
1128
1129
1130
1131
1132
1133
1134
1135
1136
1137
1138
1139

167400
000001

```
.NLIST CND,MC,MD
.LIST ME
.ENABL ABS,AMA

:DEFINE SYSMAC MACROS

SSWR= 167400 ;DEFINE SWITCHES 15,14,13,11,10,9,8
STN= 1 ;SET FIRST TEST NO. TO 1
```

```
.TITLE UNIBUS RK06 DRIVE DIAGNOSTIC PART 2
:*COPYRIGHT (C) 1976
:*DIGITAL EQUIPMENT CORP.
:*MAYNARD, MASS. 01754
:*
:*PROGRAM BY GARY PAPAIZIAN
:*
:*THIS PROGRAM WAS ASSEMBLED USING THE PDP-11 MAINDEC SYSMAC
:*PACKAGE (MAINDEC-11-DZQAC-C2), SEPT 14, 1976.
:*
```

```
.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
:* 204 DEFAULT PARAMETERS & BYPASS WRITE TESTS
:* 214 DEFAULT PARAMETERS & BYPASS TIMING TESTS
:* 220 INPUT PARAMETERS
:* 224 INPUT PARAMETERS & BYPASS WRITE TESTS
:* 230 INPUT PARAMETERS & BYPASS TIMING TESTS
:* 240 ODT11
:*
```

```

1140 .SETTL BASIC DEFINITIONS
1141
1142 ;*INITIAL ADDRESS OF THE STACK POINTER *** 1100 ***
1143 STACK= 1100
1144 .EQUIV EMT,ERROR ;:BASIC DEFINITION OF ERROR CALL
1145 .EQUIV IOT,SCOPE ;:BASIC DEFINITION OF SCOPE CALL
1146
1147 ;*MISCELLANEOUS DEFINITIONS
1148 HT= 11 ;:CODE FOR HORIZONTAL TAB
1149 LF= 12 ;:CODE FOR LINE FEED
1150 CR= 15 ;:CODE FOR CARRIAGE RETURN
1151 CRLF= 200 ;:CODE FOR CARRIAGE RETURN-LINE FEED
1152 PS= 177776 ;:PROCESSOR STATUS WORD
1153 .EQUIV PS,PSW
1154 STKLMT= 177774 ;:STACK LIMIT REGISTER
1155 PIRQ= 177772 ;:PROGRAM INTERRUPT REQUEST REGISTER
1156 DSWR= 177570 ;:HARDWARE SWITCH REGISTER
1157 DDISP= 177570 ;:HARDWARE DISPLAY REGISTER
1158
1159 ;*GENERAL PURPOSE REGISTER DEFINITIONS
1160 R0= %0 ;:GENERAL REGISTER
1161 R1= %1 ;:GENERAL REGISTER
1162 R2= %2 ;:GENERAL REGISTER
1163 R3= %3 ;:GENERAL REGISTER
1164 R4= %4 ;:GENERAL REGISTER
1165 R5= %5 ;:GENERAL REGISTER
1166 R6= %6 ;:GENERAL REGISTER
1167 R7= %7 ;:GENERAL REGISTER
1168 SP= %6 ;:STACK POINTER
1169 PC= %7 ;:PROGRAM COUNTER
1170
1171 ;*PRIORITY LEVEL DEFINITIONS
1172 PR0= 0 ;:PRIORITY LEVEL 0
1173 PR1= 40 ;:PRIORITY LEVEL 1
1174 PR2= 100 ;:PRIORITY LEVEL 2
1175 PR3= 140 ;:PRIORITY LEVEL 3
1176 PR4= 200 ;:PRIORITY LEVEL 4
1177 PR5= 240 ;:PRIORITY LEVEL 5
1178 PR6= 300 ;:PRIORITY LEVEL 6
1179 PR7= 340 ;:PRIORITY LEVEL 7
1180
1181 ;*"SWITCH REGISTER" SWITCH DEFINITIONS
1182 SW15= 100000
1183 SW14= 40000
1184 SW13= 20000
1185 SW12= 10000
1186 SW11= 4000
1187 SW10= 2000
1188 SW09= 1000
1189 SW08= 400
1190 SW07= 200
1191 SW06= 100
1192 SW05= 40
1193 SW04= 20
1194 SW03= 10
1195 SW02= 4
  
```

1196 000002
 1197 000001
 1198
 1199
 1200
 1201
 1202
 1203
 1204
 1205
 1206
 1207
 1208
 1209
 1210 100000
 1211 040000
 1212 020000
 1213 010000
 1214 004000
 1215 002000
 1216 001000
 1217 000400
 1218 000200
 1219 000100
 1220 000040
 1221 000020
 1222 000010
 1223 000004
 1224 000002
 1225 000001
 1226
 1227
 1228
 1229
 1230
 1231
 1232
 1233
 1234
 1235
 1236
 1237
 1238 000004
 1239 000010
 1240 000014
 1241 000014
 1242 000014
 1243 000020
 1244 000024
 1245 000030
 1246 000034
 1247 000060
 1248 000064
 1249 000240
 1250
 1251

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 SW0J,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**
 PWRVEC= 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

```

1252
1253 ;          $BASE=177440
1254
1255 000000 RKCS1= 0 ; CONTROL AND STATUS REGISTER 1
1256 000002 RKWC= 2 ; WORD COUNT REGISTER
1257 000004 RKBA= 4 ; BUS ADDRESS REGISTER
1258 000006 RKDA= 6 ; DESIRED TRACK SECTOR REGISTER
1259 000010 RKCS2= 10 ; CONTROL AND STATUS REGISTER 2
1260 000012 RKDS= 12 ; DRIVE STATUS REGISTER
1261 000014 RKER= 14 ; ERROR REGISTER
1262 000016 RKASOF= 16 ; ATTENTION SUMMARY AND OFFSET REGISTER
1263 000020 RKDC= 20 ; DESIRED CYLINDER REGISTER
1264 000024 RKDB= 24 ; DATA BUFFER
1265 000026 RKMR1= 26 ; MAINTENANCE REGISTER 1
1266 000034 RKMR2= 34 ; MAINTENANCE REGISTER 2 (MESSAGE LINE A)
1267 000036 RKMR3= 36 ; MAINTENANCE REGISTER 3 (MESSAGE LINE B)
1268 000030 RKECPS= 30 ; ECC POSITION INFORMATION
1269 000032 RKECPT= 32 ; ECC PATTERN INFORMATION
1270
1271 .SBTTL CONTROL AND STATUS REGISTER 1 BITS (RKCS1:0)
1272
1273 ;          DRIVE COMMANDS
1274
1275 000001 SELDRV= 1 ; SELECT DRIVE (GET STATUS)
1276 000003 PACK= 3 ; PACK ACKNOWLEDGE
1277 000005 CLEAR= 5 ; DRIVE CLEAR
1278 000007 UNLOAD= 7 ; UNLOAD
1279 000011 SRTSPL= 11 ; START SPINDLE
1280 000013 RECAL= 13 ; RECALIBRATE
1281 000015 OFFSET= 15 ; OFFSET
1282 000017 SEEK= 17 ; SEEK
1283 000021 RDATA= 21 ; READ DATA
1284 000023 WRDATA= 23 ; WRITE DATA
1285 000025 RDHEAD= 25 ; READ HEADER
1286 000027 WRHEAD= 27 ; WRITE HEADER AND DATA
1287 000031 WRTCHK= 31 ; WRITE CHECK
1288
1289 000001 GO= BIT0 ; GO BIT
1290 000100 IE= BIT6 ; INTERRUPT ENABLE
1291 000200 RDY= BIT7 ; CONTROLLER READY
1292 000400 BA16= BIT8 ; BUS ADDRESS BIT 16
1293 001000 BA17= BIT9 ; BUS ADDRESS BIT 17
1294 002000 CDT= BIT10 ; CONTROLLER DRIVE TYPE (0=RK06)
1295 004000 CTO= BIT11 ; CONTROLLER TIMEOUT
1296 010000 CFMT= BIT12 ; CONTROLLER DRIVE FORMAT (0=22 SECTOR, 1=20 SECTOR)
1297 020000 DCPAR= BIT13 ; SERCON PARITY ERROR DETECTED BY CONTROLLER
1298 040000 DI= BIT14 ; DRIVE INTERRUPT
1299 100000 CERR= BIT15 ; CONTROLLER ERROR
1300 100000 CCLR= BIT15 ; CONTROLLER CLEAR
1301
1302 .SBTTL CONTROL AND STATUS REGISTER 2 BITS (RKCS2:10)
1303
1304 000007 DRVMSK= 7 ; MASK FOR DRIVE SELECTION CODE
1305 000010 RLS= BIT3 ; DESELECT OR RELEASE DRIVE IN BITS 0-2
1306 000020 BAI= BIT4 ; BUS ADDRESS INCREMENT INHIBIT
1307 000040 SCLR= BIT5 ; SUBSYSTEM CLEAR CONTROLLER AND ALL DRIVES

```

1308	000100	IR=	BIT6	; INPUT READY
1309	000200	OR=	BIT7	; OUTPUT READY
1310	000400	UFE=	BIT8	; UNIT FIELD ERROR
1311	001000	MDS=	BIT9	; MULTIPLE DRIVE SELECT
1312	002000	PGE=	BIT10	; PROGRAMMING ERROR
1313	004000	NEM=	BIT11	; NON-EXISTENT MEMORY
1314	010000	NED=	BIT12	; NON-EXISTENT DRIVE
1315	020000	UPE=	BIT13	; UNIBUS PARITY ERROR
1316	040000	WCE=	BIT14	; WRITE CHECK ERROR
1317	000000	DLT=	BIT15	; DATA LATE ERROR
1318				
1319		.SBTTL	ERROR REGISTER BIT DEFINITION (RKER:14)	
1320				
1321	000001	ILF=	BIT0	; ILLEGAL FUNCTION CODE
1322	000002	SKI=	BIT1	; SEEK INCOMPLETE
1323	000004	NXF=	BIT2	; NON-EXECUTABLE FUNCTION
1324	000010	DRPAR=	BIT3	; DRIVE DETECTED SERCON PARITY ERROR
1325	000020	FMTE=	BIT4	; FORMAT ERROR
1326	000040	DTYPE=	BIT5	; DRIVE TYPE ERROR
1327	000100	ECH=	BIT6	; ECC HARD
1328	000200	BSE=	BIT7	; BAD SECTOR ERROR
1329	000400	HVRC=	BIT8	; HEADER VRC ERROR
1330	001000	COE=	BIT9	; CYLINDER ADDRESS OVERFLOW ERROR
1331	002000	IDAE=	BIT10	; INVALID DISK ADDRESS ERROR: HEAD/CYL
1332	004000	WLE=	BIT11	; WRITE LOCK ERROR
1333	010000	DTE=	BIT12	; DRIVE TIMING ERROR
1334	020000	OPI=	BIT13	; OPERATION (SEARCH) INCOMPLETE
1335	040000	UNS=	BIT14	; DRIVE UNSAFE
1336	100000	DCK=	BIT15	; DATA CHECK
1337				
1338		.SBTTL	STATUS REGISTER BIT DEFINITION (RKDS:12)	
1339				
1340	000001	DRA=	BIT0	; DRIVE AVAILABLE (CONTROLLER IS SET IF ; THIS BIT IS RESET)
1341				
1342	000004	OFST=	BIT2	; DRIVE OFFSET
1343	000010	ACLO=	BIT3	; AC LOW
1344	000020	DCLO=	BIT4	; DC LOW
1345	000040	DROT=	BIT5	; DRIVE OFF TRACK
1346	000100	VV=	BIT6	; VOLUME VALID
1347	000200	DRDY=	BIT7	; DRIVE READY
1348	000400	DDT=	BIT8	; DRIVE TYPE (0=RK06)
1349	004000	WRL=	BIT11	; WRITE LOCK
1350	020000	PIP=	BIT13	; POSITIONING IN PROGRESS
1351	040000	DSC=	BIT14	; DRIVE STATUS CHANGE
1352	100000	SVAL=	BIT15	; STATUS VALID
1353				
1354		.SBTTL	MAINTENANCE REGISTER 1 BIT DEFINITION (RKMR1:22)	
1355				
1356	000017	MESMSK=	17	; MESSAGE MASK
1357	000020	PAT=	BIT4	; FORCE EVEN PARITY ON SERCON MESSAGE LINES
1358	000040	DMD=	BIT5	; DIAGNOSTIC MODE
1359	000100	MSP=	BIT6	; MAINTENANCE SECTOR PULSE
1360	000200	MIND=	BIT7	; MAINTENANCE INDEX
1361	000400	MCLK=	BIT8	; MAINTENANCE CLOCK
1362	001000	MERD=	BIT9	; MAINTENANCE ENCODED READ DATA
1363	002000	MEWD=	BIT10	; MAINTENANCE ENCODED WRITE DATA

1364	004000	PCA= BIT11	; PRECOMPENSATION ADVANCE
1365	010000	PCD= BIT12	; PRECOMPENSATION DELAY
1366	020000	ECCW= BIT13	; ECC WORD IS BEING READ OR WRITTEN
1367	040000	WRTGAT= BIT14	; WRITE GATE
1368	100000	RDGATE= BIT15	; READ GATE
1369			
1370		.SBTTL	DEFINITION OF DRIVE STATUS BYTE 00 MESSAGE A (RKMR2:34)
1371			
1372	000040	D.DRA= BITS	; DRIVE AVAILABLE
1373	000100	D.VV= BIT6	; VOLUME VALID
1374	000200	D.DRDY= BIT7	; DRIVE READY
1375	000400	D.DDT= BIT8	; DRIVE TYPE (0=RK06)
1376	001000	D.FORM= BIT9	; DRIVE FORMAT
1377	002000	D.OFF= BIT10	; OFFSET ON
1378	004000	D.WRL= BIT11	; WRITE LOCK
1379	010000	D.SPIN= BIT12	; SPINDLE ON
1380	020000	D.PIP= BIT13	; POSITIONING IN PROGRESS
1381	040000	D.DSC= BIT14	; DRIVE STATUS CHANGE
1382			
1383		.SBTTL	DEFINITION OF DRIVE STATUS BYTE 01 MESSAGE A (RKMR2:34)
1384			
1385	000020	D.SSP= BIT4	; SERVO SIG PRES
1386	000040	D.HOHM= BIT5	; HEADS HOME
1387	000100	D.BRHM= BIT6	; BRUSHES HOME
1388	000200	D.DOOR= BIT7	; DOOR INTERLOCKED
1389	000400	D.CART= BIT8	; CARTRIDGE INTERLOCK
1390	001000	D.SPOK= BIT9	; SPEED OK
1391	002000	D.FWD= BIT10	; FORWARD
1392	004000	D.REV= BIT11	; REVERSE
1393	010000	D.LOAD= BIT12	; HEADS LOADING
1394	020000	D.RTZ= BIT13	; RETURN TO ZERO
1395	040000	D.UNLD= BIT14	; HEADS UNLOADING
1396			
1397		.SBTTL	DEFINITION OF DRIVE STATUS BYTE 00 MESSAGE B (RKMR3:36)
1398			
1399	000040	D.IDAE= BITS	; INVALID DISK ADDRESS ERROR:HEAD/CYL
1400	000100	D.ACLO= BIT6	; AC LO
1401	000200	D.FLT= BIT7	; DRIVE FAULT
1402	000400	D.ILF= BIT8	; ILLEGAL FUNCTION CODE
1403	001000	D.PAR= BIT9	; DRIVE DETECTED SERCON PARITY ERROR
1404	002000	D.SKI= BIT10	; SEEK INCOMPLETE
1405	004000	D.WLE= BIT11	; WRITE LOCK ERROR
1406	010000	D.SPLS= BIT12	; SPEED LOSS
1407	020000	D.DROT= BIT13	; DRIVE OFF TRACK
1408	040000	D.UNS= BIT14	; R/W UNSAFE
1409			
1410		.SBTTL	DEFINITION OF DRIVE STATUS BYTE 01 MESSAGE B (RKMR3:36)
1411			
1412	000020	D.SECT= BIT4	; SECTOR ERROR
1413	000040	D.WCUR= BITS	; WRITE CURRENT AND NO WRITE GATE
1414	000100	D.WGAT= BIT6	; WRITE GATE AND NO TRANSISTIONS
1415	000200	D.HDFL= BIT7	; HEAD FAULT
1416	000400	D.MHD= BIT8	; MULTIPLE HEAD SELECT
1417	001000	D.XERR= BIT9	; INDEX ERROR
1418	002000	D.TIB= BIT10	; TRIBIT ERROR
1419	004000	D.PLO= BIT11	; PLO ERROR

1420 010000
1421 020000
1422 040000
1423
1424
1425
1426
1427
1428
1429
1430
1431
1432
1433
1434
1435
1436
1437
1438
1439
1440
1441
1442
1443
1444
1445
1446
1447
1448
1449
1450
1451
1452
1453
1454
1455
1456
1457
1458
1459
1460
1461
1462
1463
1464
1465
1466
1467
1468
1469
1470
1471
1472
1473
1474
1475
1476
1477
1478
1479
1480
1481
1482
1483
1484
1485
1486
1487
1488
1489
1490
1491
1492
1493
1494
1495
1496
1497
1498
1499
1500

D.NMOV= BIT12 :SEEK AND NO MOTION
D.LIMD= BIT13 :LIMIT DETECT ON SEEK
D.SUNS= BIT14 :SERVO UNSAFE

.SBTTL COMMON MASKS AND OTHER BITS: MESSAGE A (RKMR2:34)

M.DRV= 7 :DRIVE CODE, ALL BYTES
M.CDIF= 17760 :CYLINDER DIFF, BYTE 10
M.OFST= 17760 :OFFSET VALUE, BYTE 10
M.SER= 77770 :DRIVE SERIAL #, BYTE 11

.SBTTL COMMON MASKS AND OTHER BITS: MESSAGE B (RKMR3:36)

M.ID= 3 :BYTE ID, ALL BYTES
M.CADD= 17760 :CYLINDER ADDRESS, BYTE 10
M.ALGN= BIT14 :ALIGN SIGN, BYTE 10
M.SECT= 760 :SECTOR COUNT, BYTE 11
M.HEAD= 7000 :HEAD DECODE, BYTE 11
M.PAR= BIT15 :PARITY, MESS A/B, ALL BYTES

1439
1440
1441
1442
1443
1444
1445
1446
1447
1448
1449
1450
1451
1452
1453
1454
1455
1456
1457
1458
1459
1460
1461
1462
1463
1464
1465
1466
1467
1468
1469
1470
1471
1472
1473
1474
1475
1476
1477
1478
1479
1480
1481
1482
1483
1484
1485
1486
1487
1488
1489
1490
1491
1492
1493
1494

000000
000174 000000
000176 000000
000200 000137 012726
000204 000137 012622
000214 000137 012642
000220 000137 012632
000224 000137 012662
000230 000137 012704
000240 000137 065250
000046 036150
000052 100000
001000
000024 000200
000044 000044
001000 001000
001000 000000
001002 001210
001004 000430

```
.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
.=204
JMP BYWRT
.=214
JMP BYTIM
.=220
JMP PARSRT ;INPUT ALL PARAMETERS & START TESTING
.=224
JMP BYWRTA
.=230
JMP BYTIMA
.=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 100000 ;;2)SET LOC.52 TO 100000
.=55VPC
;; 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.

$APTHD:
$HIBTS: .WORD 0 ;;TWO HIGH BITS OF 18 BIT MAILBOX ADDR.
$MBADR: .WORD $MAIL ;;ADDRESS OF APT MAILBOX (BITS 0-15)
$STIM: .WORD 280. ;;RUN TIM OF LONGEST TEST
```

1495 DC1006 DC1130
 1496 DC1010 DC1130
 1497 DC1012 000042
 1498
 1499
 1500
 1501
 1502
 1503
 1504
 1505
 1506
 1507
 1508
 1509
 1510
 1511
 1512
 1513
 1514
 1515
 1516
 1517
 1518
 1519
 1520
 1521
 1522
 1523
 1524
 1525
 1526
 1527
 1528
 1529
 1530
 1531
 1532
 1533
 1534
 1535
 1536
 1537
 1538
 1539
 1540
 1541
 1542
 1543
 1544
 1545
 1546
 1547
 1548
 1549
 1550

\$PASTM: .WORD 600. ;;RUN TIME IN SECS. OF 1ST PASS ON 1 UNIT (GLICK VERIFY)
 \$JUNITM: .WORD 600. ;;ADDITIONAL RUN TIME (SECS) OF A PASS FOR EACH ADDITIONAL UNIT
 .WORD SETEND-\$MAIL/2 ;;LENGTH MAILBOX-ETABLE(WORDS)

.LIST MD

USE LOOP X TO OMIT SUBCLR

THIS MACRO FILLS EXPECTED MSG A0, B0, A1, B1, A2, B2 & B3 WITH STANDARD BITS
 A=D.DSC AFTER ATTN OR D AFTER DRIVE CLEAR OR ANY IMPLIED SEEKS
 NOTE: A CAN BE ANY BIT COMBINATION DESIRED.

THIS MACRO ASSUMES DRIVE MSG A0, B0, A1, B1 WILL ALWAYS BE TESTED
 USE A,C,D,E FOR MSG A0, B0, A1, B1 ERROR NUMBERS RESP.
 USE G=T.A2 TO READ MSG A2 & PUT INFO INTO 'CYLDIF'
 H=T.B2 TO READ MSG B2 & PUT INFO INTO 'CYLADD'
 I=T.B3 TO READ MSG B3 & PUT INFO INTO 'SECTOR' & 'HEADA'

USE F=<ERROR DESCRIPTION>

A=CYL DIFF/OFFSET ERROR *
 B=CYL ADDR ERROR *
 C=<ERROR DESCRIPTION>

USE CALIB X TO OMIT CHECKING MSGS A0, B0, A1, B1, A2 & B2

QUICK START SPINDLE

A=WRHEAD/<CFMT!WRHEAD>
 USE WRHDR <A>,X TO OMIT CHECKING A0, B0, A1 & B1

A=RDHEAD/<CFMT!RDHEAD>
 USE RDHDR <A>,X TO OMIT CHECKING A0, B0, A1, B1

```

1551
1552
1553
1554
1555
1556
1557
1558
1559
1560
1561
1562
1563
1564
1565
1566
1567
1568
1569
1570
1571
1572
1573
1574
1575
1576
1577
1578
1579
1580
1581
1582
1583
1584
1585
1586
1587
1588
1589
1590
1591
1592
1593
1594
1595
1596
1597
1598
1599
1600
1601
1602
1603
1604

```

```

:
: A=TOCYL·FRCYL , B=HEAD#, C = 0 FOR 22 SECTOR, 1 FOR 20 SECTOR
:
:
:
: QUICK SEEK. ENTER WITH CYL# IN RKDC
:
:
:
: A=WRDATA/<CFMT!WRDATA>
: C=ADDR TO JMP TO ATTEMPT TO WRITE ON ANOTHER SECTOR
: D=ADDR TO JMP TO BYPASS TEST
: E: IF BLANK WILL CHECK A0, B0, A1 & B1 AT THE END OF WRITING
: E: IF NON BLANK WILL OMIT CHECKING A0 THRU B1
:
:
:
: A=RDDATA/<CFMT!RDDATA>
: USE RDATA <A>,X TO OMIT CHECKING A0, B0, A1 & B1
:
:
:
: A=WRTCHK/<CFMT!WRTCHK>
: C=EXPECTED DATA FOR TYPEOUT
: USE WRCHK <A>,DATA0,X TO OMIT CHECKING A0, B0, A1 & B1
:
:
:
: A=CYL # TO GO TO
: B=FWD DIRECTION MSG
: C=REV DIRECTION MSG
:
:
:
: A= ERROR #
: B = ERROR CONDITION
:
:
:
: A&B=ERROR # & CYL #
: C&D=ERROR # & CYL #
:
:
: .NLIST MD

```

```

1605
1606
1607
1608
1609
1610
1611 001100
1612 001100 001100
1613 001100 000000
1614 001102 000
1615 001103 000
1616 001104 000000
1617 001106 000000
1618 001110 000000
1619 001112 000000
1620 001114 000
1621 001115 001
1622 001116 000000
1623 001120 000000
1624 001122 000000
1625 001124 000000
1626 001126 000000
1627 001130 000000
1628 001132 000000
1629 001134 000
1630 001135 000
1631 001136 000000
1632 001140 177570
1633 001142 177570
1634 001144 177560
1635 001146 177562
1636 001150 177564
1637 001152 177566
1638 001154 000
1639 001155 002
1640 001156 012
1641 001157 000
1642 001160 000000
1643 001162 000000
1644 001164 000000
1645 001166 000000
1646 001170 000000
1647 001172 000000
1648 001174 000000
1649 001176 000000
1650 001200 177607 000377
1651 001204 077
1652 001205 015
1653 001206 000012
1654
1655
1656
1657
1658
1659 001210
1660 001210 000000

```

```

.SBTTL COMMON TAGS
;*****
;THIS TABLE CONTAINS VARIOUS COMMON STORAGE LOCATIONS
;USED IN THE PROGRAM.
.=110C
$CMTAG: .WORD 0 ;;START OF COMMON TAGS
$STNM: .BYTE 0 ;;CONTAINS THE TEST NUMBER
$ERFLG: .BYTE 0 ;;CONTAINS ERROR FLAG
$ICNT: .WORD 0 ;;CONTAINS SUBTEST ITERATION COUNT
$LPADR: .WORD 0 ;;CONTAINS SCOPE LOOP ADDRESS
$LPERR: .WORD 0 ;;CONTAINS SCOPE RETURN FOR ERRORS
$ERTTL: .WORD 0 ;;CONTAINS TOTAL ERRORS DETECTED
$ITEMB: .BYTE 0 ;;CONTAINS ITEM CONTROL BYTE
$ERMAX: .BYTE 1 ;;CONTAINS MAX. ERRORS PER TEST
$ERRPC: .WORD 0 ;;CONTAINS PC OF LAST ERROR INSTRUCTION
$GDADR: .WORD 0 ;;CONTAINS ADDRESS OF 'GOOD' DATA
$BDADR: .WORD 0 ;;CONTAINS ADDRESS OF 'BAD' DATA
$GDDAT: .WORD 0 ;;CONTAINS 'GOOD' DATA
$BDDAT: .WORD 0 ;;CONTAINS 'BAD' DATA
;RESERVED--NOT TO BE USED
$AUTOB: .BYTE 0 ;;AUTOMATIC MODE INDICATOR
$INTAG: .BYTE 0 ;;INTERRUPT MODE INDICATOR
$SWR: .WORD 0 ;;ADDRESS OF SWITCH REGISTER
$DISP: .WORD 0 ;;ADDRESS OF DISPLAY REGISTER
$TKS: 177560 ;;TTY KBD STATUS
$TKB: 177562 ;;TTY KBD BUFFER
$TPS: 177564 ;;TTY PRINTER STATUS REG. ADDRESS
$TPB: 177566 ;;TTY PRINTER BUFFER REG. ADDRESS
$NULL: .BYTE 0 ;;CONTAINS NULL CHARACTER FOR FILLS
$FILLS: .BYTE 2 ;;CONTAINS # OF FILLER CHARACTERS REQUIRED
$FILLC: .BYTE 12 ;;INSERT FILL CHARS. AFTER A "LINE FEED"
$TPFLG: .BYTE 0 ;;"TERMINAL AVAILABLE" FLAG (BIT<07>=0=YES)
$TMP0: .WORD 0 ;;USER DEFINED
$TMP1: .WORD 0 ;;USER DEFINED
$TMP2: .WORD 0 ;;USER DEFINED
$TMP3: .WORD 0 ;;USER DEFINED
$TMP4: .WORD 0 ;;USER DEFINED
$TMP5: .WORD 0 ;;USER DEFINED
$TIMES: 0 ;;MAX. NUMBER OF ITERATIONS
$ESCAPE: 0 ;;ESCAPE ON ERROR ADDRESS
$BELL: .ASCIZ <207><377><377> ;;CODE FOR BELL
$QUES: .ASCII /?/ ;;QUESTION MARK
$CRLF: .ASCII <15> ;;CARRIAGE RETURN
$LF: .ASCIZ <12> ;;LINE FEED
;*****
.SBTTL APT MAILBOX-ETABLE
;*****
.EVEN
$MAIL: ;;APT MAILBOX
$MSGTY: .WORD 0 ;;MESSAGE TYPE CODE

```

1661	001212	000000	\$FATAL: .WORD	AFATAL	:: FATAL ERROR NUMBER
1662	001214	000000	\$TESTN: .WORD	ATESTN	:: TEST NUMBER
1663	001216	000000	\$PASS: .WORD	APASS	:: PASS COUNT
1664	001220	000000	\$DEVCT: .WORD	ADEVCT	:: DEVICE COUNT
1665	001222	000000	\$UNIT: .WORD	AUNIT	:: I/O UNIT NUMBER
1666	001224	000000	\$MSGAD: .WORD	AMSGAD	:: MESSAGE ADDRESS
1667	001226	000000	\$MSGLG: .WORD	AMSGLG	:: MESSAGE LENGTH
1668	001230		\$ETABLE:		:: APT ENVIRONMENT TABLE
1669	001230	000	\$ENV: .BYTE	AENV	:: ENVIRONMENT BYTE
1670	001231	000	\$ENVM: .BYTE	AENVM	:: ENVIRONMENT MODE BITS
1671	001232	000000	\$SWREG: .WORD	ASWREG	:: APT SWITCH REGISTER
1672	001234	000000	\$USWR: .WORD	AUSWR	:: USER SWITCHES
1673	001236	000000	\$CPUOP: .WORD	ACPUOP	:: CPU TYPE, OPTIONS
1674			*		BITS 15-11=CPU TYPE
1675			*		11/04=01, 11/05=02, 11/20=03, 11/40=04, 11/45=05
1676			*		11/70=06, PDQ=07, Q=10
1677			*		BIT 10=REAL TIME CLOCK
1678			*		BIT 9=FLOATING POINT PROCESSOR
1679			*		BIT 8=MEMORY MANAGEMENT
1680	001240	000	\$MAMS1: .BYTE	AMAMS1	:: HIGH ADDRESS, M.S. BYTE
1681	001241	000	\$MTYP1: .BYTE	AMTYP1	:: MEM. TYPE, BLK#1
1682			*		MEM. TYPE BYTE -- (HIGH BYTE)
1683			*		900 NSEC CORE=001
1684			*		300 NSEC BIPOLAR=002
1685			*		500 NSEC MOS=003
1686	001242	000000	\$MADR1: .WORD	AMADR1	:: HIGH ADDRESS, BLK#1
1687			*		MEM. LAST ADDR.=3 BYTES, THIS WORD AND LOW OF "TYPE" ABOVE
1688	001244	000	\$MAMS2: .BYTE	AMAMS2	:: HIGH ADDRESS, M.S. BYTE
1689	001245	000	\$MTYP2: .BYTE	AMTYP2	:: MEM. TYPE, BLK#2
1690	001246	000000	\$MADR2: .WORD	AMADR2	:: MEM. LAST ADDRESS, BLK#2
1691	001250	000	\$MAMS3: .BYTE	AMAMS3	:: HIGH ADDRESS, M.S. BYTE
1692	001251	000	\$MTYP3: .BYTE	AMTYP3	:: MEM. TYPE, BLK#3
1693	001252	000000	\$MADR3: .WORD	AMADR3	:: MEM. LAST ADDRESS, BLK#3
1694	001254	000	\$MAMS4: .BYTE	AMAMS4	:: HIGH ADDRESS, M.S. BYTE
1695	001255	000	\$MTYP4: .BYTE	AMTYP4	:: MEM. TYPE, BLK#4
1696	001256	000000	\$MADR4: .WORD	AMADR4	:: MEM. LAST ADDRESS, BLK#4
1697	001260	000000	\$VECT1: .WORD	AVECT1	:: INTERRUPT VECTOR#1, BUS PRIORITY#1
1698	001262	000000	\$VECT2: .WORD	AVECT2	:: INTERRUPT VECTOR#2, BUS PRIORITY#2
1699	001264	177440	\$BASE: .WORD	ABASE	:: BASE ADDRESS OF EQUIPMENT UNDER TEST
1700	001266	000000	\$DEVN: .WORD	ADEVN	:: DEVICE MAP
1701	001270	000000	\$CDW1: .WORD	ACDW1	:: CONTROLLER DESCRIPTION WORD#1
1702	001272	000000	\$CDW2: .WORD	ACDW2	:: CONTROLLER DESCRIPTION WORD#2
1703	001274	000000	\$DDW0: .WORD	ADDW0	:: DEVICE DESCRIPTOR WORD#0
1704	001276	000000	\$DDW1: .WORD	ADDW1	:: DEVICE DESCRIPTOR WORD#1
1705	001300	000000	\$DDW2: .WORD	ADDW2	:: DEVICE DESCRIPTOR WORD#2
1706	001302	000000	\$DDW3: .WORD	ADDW3	:: DEVICE DESCRIPTOR WORD#3
1707	001304	000000	\$DDW4: .WORD	ADDW4	:: DEVICE DESCRIPTOR WORD#4
1708	001306	000000	\$DDW5: .WORD	ADDW5	:: DEVICE DESCRIPTOR WORD#5
1709	001310	000000	\$DDW6: .WORD	ADDW6	:: DEVICE DESCRIPTOR WORD#6
1710	001312	000000	\$DDW7: .WORD	ADDW7	:: DEVICE DESCRIPTOR WORD#7
1711	001314		\$ETEND:		
1712			. MEXIT		
1713		177440	ABASE=	177440	:: DEFAULT BUSS ADDRESS
1714	001314	000210	RKVEC:	210	:: DEFAULT CONTROLLER INTERRUPT VECTOR
1715	001316	000240	RKPRI:	PR5	:: PRIORITY
1716	001320	172540	PKS:	172540	:: P-CLOCK STATUS REG

1717	001322	172542	PKSB:	172542	;P-CLOCK SET BUFFER
1718	00132	172544	PKRB:	172544	;P-CLOCK READ BUFFER
1719	00132b	177546	LKS:	177546	;L-CLOCK STATUS REG.
1720					
1721	001330	000100	LCVEC:	100	;L-CLOCK INTERRUPT VECTOR
1722	001332	000104	PCVEC:	104	;P-CLOCK INTERRUPT VECTOR.
1723					
1724		000114	MEMVEC=	114	;MEMORY PARITY VECTOR
1725		172100	MEMBAS=	172100	;MEMORY PARITY OPTION CSR START ADDR
1726	001334	000000	TRAPPC:	0	;PC FOR MEM CHECK ENABLE TRAP
1727					
1728	001336	000000	PARAM:	0	;1 FOR 220 START, NO DEFAULT
1729	001340	000000	BYPWRT:	0	;1 FOR 204 OR 224 START
1730	001342	000000	BYPTIM:	0	;1 FOR 210 OR 230 START
1731	001344	000000	FTITLE:	0	;FLAG FOR PRINTING OUT 1ST PROGRAM TITLE
1732					
1733	001346	000000	DRVPTX:	0	;CONTAINS THE POINTER TO THE DRIVE FLAG
1734					; (DRIV0-DRIV7) OF THE DRIVE TO BE CHECKED NEXT.
1735	001350	000000	FRCYL:	0	;FROM CYLINDER
1736	001352	000000	TOCYL:	0	;TO CYLINDER
1737	001354	000000	CCYL:	0	;CURRENT CYL, USED IN N SQUARE TEST
1738	001356	000000	PCYL:	0	;PREV CYL, USED IN N SQUARE TEST
1739	001360	000000	CALDIF:	0	;CALC CYL DIFF USED IN N SQUARE TEST
1740	001362	000000	CYLDIF:	0	;CYL DIFF, RIGHT JUSTIFIED FROM RKMR3
1741	001364	000000	CYLADD:	0	;CYL ADDR, RIGHT JUSTIFIED FROM RKMR3
1742	001366	000000	CALADD:	0	;CYL ADDR USED IN FHD TAB ROUTINE
1743					
1744	001370	000074	HZ:	60.	;60 FOR 60 CPS
1745					;50 FOR 50 CPS
1746	001372	000000	COUNT:	0	;LOADED TO 50 OR 60 TO COUNT TO 1 SEC
1747					;OR ANY OTHER NUMBER TO COUNT OFF FRACTIONAL SECOND
1748	001374	000000	SEC:	0	;SECOND COUNTER
1749	001376	000000	TIMUP:	0	;FLAG TO INDICATE TIME IS UP
1750	001400	000000	SECNT:	0	;SECTOR COUNT
1751	001402	000000	PSEC:	0	;PREVIOUS SECTOR
1752	001404	000000	ESEC:	0	;EXPECTED SECTOR
1753	001406	000000	SECTOR:	0	;SECTOR COUNT, RIGHT JUSTIFIED FROM RKMR3
1754					
1755	001410	000001	T1:	1	;TIMEOUT CONSTANTS
1756	001412	000012	T10:	10.	
1757	001414	000062	T50:	50.	
1758	001416	000764	T500:	500.	
1759	001420	000144	T100:	100.	
1760	001422	011610	T5000:	5000.	
1761	001424	141520	T50000:	50000.	
1762					
1763	001426	000077	CYL:	63.	;CYLINDER NUMBERS USED IN
1764	001430	000177		127.	;CURRENT CROSSOVER TEST
1765	001432	000277		191.	
1766	001434	000377		255.	
1767	001436	000477		319.	
1768	001440	000577		383.	
1769					
1770	001442	000000	TIM1:	0	;USED IN TIMING TESTS
1771	001444	000000	TIM2:	0	
1772	001446	000000	TIM3:	0	

1773	001450	000000	TIM4:	0	
1774					
1775	001452	000000	LPCNT:	0	; LOOP CTR USED IN CALC.LK
1776	001454	000000	LPTIM:	0	; LOOP TIME IN USEC
1777					
1778	001456	000000	SUM:	0	; LO ORDER FOR TIMING TESTS
1779	001460	000000		0	; HI ORDER FOR TIMING TESTS
1780	001462	000000	SUM1:	0	; LO ORDER FOR TIMING TESTS
1781	001464	000000		0	; HI ORDER FOR TIMING TESTS
1782					
1783	001466	000000	WD1:	0	; ACTUAL HEADER/DATA WORD
1784	001470	000000	WD2:	0	; EXPECTED DATA WORD
1785					
1786	001472	000000	OFFERR:	0	; SET WHEN WRITE CHECK ERROR ON OFFSET
1787					
1788					
1789	001474	000000	HEAD:	0	; HEAD NUMBER
1790	001476	000000	HEAD#:	0	; HEAD # FROM H.E3. RT. JUSTIFIED
1791	001500	000000	HD1:	0	; SHIFTED HEAD# FOR FORMATTER ROUTINE
1792	001502	000000	FORMAT:	0	; FORMAT TYPE
1793	001504	000000	FMT1:	0	; SHIFTED FORMAT FOR FORMATTER ROUTINE
1794	001506	000000	WDCNT:	0	; WORD COUNT
1795					
1796	001510	000000	DATA0:	0	; ALL 0'S
1797	001512	052525	DATA01:	52525	; 0101 PATT
1798	001514	177777	DATA1:	177777	; ALL 1'S
1799	001516	133467	DPAT1:	133467	
1800	001520	070627	DPAT2:	70627	
1801					
1802	001522	000000	WORD:	0	; HEADER/DATA WORD
1803	001524	000000	HDWD:	0	; HEADER WORD FROM RKDB
1804					
1805	001526	000000	BSERR:	0	; CANNOT READ BSE INFO WHEN SET
1806	001530	000000	LIMERR:	0	; LIMIT DETECT ERROR FLAG
1807	001532	000000	BYPCERR:	0	; SET TO 1 TO BYPASS CKCERR IN 'GSTAT1'
1808	001534	000000	CHKFLG:	0	; WORDS TO BE TESTED
1809					
1810	001536	000102	HDTAB:	.BLKW 66.	; CALCULATED HEADER WORD TABLE
1811	001742	000102	RHTAB:	.BLKW 66.	; FILLED AFTER READ HEADER CMD
1812	002146	000102	SRTTAB:	.BLKW 66.	; ABOVE RHTAB SORTED STARTING FORM
1813					; SECTOR 0 BY SORT ROUTINE
1814	002352	000400	BSE20H:	.BLKW 256.	; 20 SECTOR HARDWARE BSE INFO
1815	003352	000400	BSE22H:	.BLKW 256.	; 22 SECTOR HARDWARE BSE INFO.
1816	004352	000400	BSE20S:	.BLKW 256.	; 20 SECTOR SOFTWARE BSE INFO.
1817	005352	000400	BSE22S:	.BLKW 256.	; 22 SECTOR SOFTWARE BSE INFO.
1818	006352	000400	ROTAB:	.BLKW 256.	; FILLED AFTER READ DATA CMD
1819					
1820	007352	000000	UNLD:	0	; SET TO 0 IF HEADS ARE LOADED
1821					; SET TO 1 IF HEADS UNLOADED
1822	007354	000000	BADHDR:	0	; SET TO 0 IF FORMATTING OK
1823					; SET TO 1 IF FORMATTING ALTERED
1824	007356	000000	HPENC:	0	; SET TO 0 IF HALT NOT PENDING
1825					; SET TO 1 IF HALT PENDING
1826					
1827					; THE ABOVE 3 FLAGS ARE USED
1828					; BY 'STOP' ROUTINE TO BRING

;THE CPU TO A VALID HALT.

1829						
1830						
1831						
1832	007360	001	002	004	ATTN: .BYTE 1,2,4,10,20,40,100,200	:ATN 0-7 RESP.
1833	007363	010	020	040		
1834	007366	100	200			
1835					.EVEN	

;THE FOLLOWING ARE HOLDING REGISTERS FOR THE RK611 REGISTERS
;THEY ARE LOADED AFTER RDY IS REC'D FROM WRDY ROUTINE.

1842	007370	000000	HCS1:	0	;HOLD RKCS1
1843	007372	000000	HCS2:	0	;HOLD RKCS2
1844	007374	000000	HWC:	0	;HOLD RKWC
1845	007376	000000	HBA:	0	;ETC.
1846	007400	000000	HDA:	0	
1847	007402	000000	HDS:	0	
1848	007404	000000	HER:	0	
1849	007406	000000	HASOF:	0	
1850	007410	000000	HDC:	0	
1851	007412	000000	HDB:	0	
1852	007414	000000	HMR1:	0	
1853	007416	000000	HMR2:	0	
1854	007420	000000	HMR3:	0	
1855	007422	000000	HPOS:	0	
1856	007424	000000	HPAT:	0	

;TEMPORARY STORAGE.

1859	007426	000000	TEMP1:	0
1860	007430	000000	TEMP2:	0
1861	007432	000000	TEMP3:	0
1862	007434	000000	TEMP4:	0
1863	007436	000000	TEMP5:	0

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

1867	007440	000000	H.A0:	0
1868	007442	000000	H.B0:	0
1869	007444	000000	H.A1:	0
1870	007446	000000	H.B1:	0
1871	007450	000000	H.A2:	0
1872	007452	000000	H.B2:	0
1873	007454	000000	H.A3:	0
1874	007456	000000	H.B3:	0

;THE FOLLOWING ARE 'EXPECTED' REGISTER FOR THE ABOVE.

1878	007460	000000	E.A0:	0
1879	007462	000000	E.B0:	0
1880	007464	000000	E.A1:	0
1881	007466	000000	E.B1:	0
1882	007470	000000	E.A2:	0
1883	007472	000000	E.B2:	0
1884	007474	000000	E.A3:	0

1885 007476 000000
1886
1887
1888
1889 000001
1890 000002
1891 000004
1892
1893
1894
1895
1896
1897 007500 000000
1898 007502 000000
1899 007504 000000
1900 007506 000000
1901 007510 000000
1902
1903
1904
1905
1906 007512 000000
1907 007514 000000
1908 007516 000000
1909 007520 000000
1910 007522 000000
1911 007524 000000
1912 007526 000000
1913 007530 000000
1914
1915 007532 000000
1916 007534 000000
1917 007536 000000
1918 007540 000000

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
; ACT11: 0 ; FLAG - SET WHEN IN ACT11 MODE OF OPERATION
; PPTP: 0 ; FLAG - SET WHEN PROGRAM LOADED BY PAPER TAPE
; DRIV5: 0 ; 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 ; FLAG SET TO 1 WHEN DRIVE 0 PRESENT
DRIV1: 0 ; FOR DRIVE 1
DRIV2: 0 ; FOR DRIVE 2
DRIV3: 0 ; FOR DRIVE 3
DRIV4: 0 ; FOR DRIVE 4
DRIV5: 0 ; FOR DRIVE 5
DRIV6: 0 ; FOR DRIVE 6
DRIV7: 0 ; FOR DRIVE 7
LCLKF: 0 ; L-CLOCK FLAG PRESENT FLAG
PCLKF: 0 ; P-CLOCK FLAG PRESENT FLAG
DOTM: 0 ; SET IF EITHER CLOCK PRESENT FOR TIMING TESTS.
SIZFLG: 0 ; SET IF DEFAULT DO SIZING IN TEST 1

1919
1920
1921
1922
1923
1924
1925
1926
1927
1928
1929
1930
1931
1932
1933 007542
1934
1935
1936 007542 055115
1937 007544 061123
1938 007546 064002
1939 007550 064374
1940
1941
1942 007552 055334
1943 007554 061123
1944 007556 064002
1945 007560 064374
1946
1947
1948 007562 055355
1949 007564 061123
1950 007566 064002
1951 007570 064374
1952
1953
1954 007572 055376
1955 007574 061123
1956 007576 064002
1957 007600 064374
1958

.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 THEN 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

```

\$ERRTAB:

```

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

;ERROR 2
      EM5      ;DETECTED MDS
      DH1
      DT1
      DF1

;ERROR 3
      EM6      ;DETECTED UFE
      DH1
      DT1
      DF1

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

;ERROR 5

```

1959	007602	055465	EM8	;DR PRESENT BUT NOT SPECIFIED BY OPERATOR
1960	007604	061123	DH1	
1961	007606	064002	DT1	
1962	007610	064374	DF1	
1963				
1964			:ERROR 6	
1965	007612	055541	EM9	;DR NOT PRESENT BUT SPECIFIED BY OPERATOR
1966	007614	061123	DH1	
1967	007616	064002	DT1	
1968	007620	064374	DF1	
1969				
1970			:ERROR 7	
1971	007622	055615	EM10	;ABORT TEST, COULD NOT REFERENCE CONTROLLER REGISTER
1972	007624	061123	DH1	
1973	007626	064002	DT1	
1974	007630	064374	DF1	

1975				
1976			;ERROR 10	
1977	007632	055700	EM11	;DRA & NED BOTH SET
1978	007634	061123	DH1	
1979	007636	064002	DT1	
1980	007640	064374	DF1	
1981			;ERR 11	
1982	007642	055744	EM12	;NO RDY
1983	007644	062322	DH27	;AFTER WRITE DATA CMD
1984	007646	064002	DT1	
1985	007650	064470	DF10	
1986			;ERR 12	
1987	007652	056340	EM21	;CERR SET
1988	007654	062322	DH27	
1989	007656	064002	DT1	
1990	007660	064470	DF10	
1991			;ERR 13	
1992	007662	055744	EM12	;NO RDY
1993	007664	062272	DH26	;AFTER READ DATA CMD
1994	007666	064002	DT1	
1995	007670	064470	DF10	
1996			;ERR 14	
1997	007672	056340	EM21	;CERR SET
1998	007674	062272	DH26	
1999	007676	064002	DT1	
2000	007700	064470	DF10	
2001			;ERR 15	
2002	007702	055744	EM12	;NO RDY
2003	007704	062574	DH32	;AFTER WRITE CHECK CMD
2004	007706	064002	DT1	
2005	007710	064470	DF10	
2006			;ERR 16	
2007	007712	057640	EM80	;WRITE CHECK ERROR SET
2008	007714	062574	DH32	;AFTER WRITE CHECK CMD
2009	007716	064072	DT6	
2010	007720	064410	DF3	
2011			;ERR 17	
2012	007722	057677	EM81	;WRITE CHECK CMD NOT FUNCTIONING
2013	007724	063514	DH52	;WITH INTENTIONAL MISCOMPARE
2014	007726	064002	DT1	
2015	007730	064470	DF10	
2016			;ERR 20	
2017	007732	057743	EM82	;READ DATA NOT COMPARE WITH WRITE DATA
2018	007734	062272	DH26	;AFTER READ DATA CMD
2019	007736	064072	DT6	

2020	007740	064410
2021		
2022	007742	060015
2023	007744	062272
2024	007746	064002

:ERR 21

DF3
EM83
DH26
DT1

:DATA CHECK ERROR

2025	007750	064470		
2026			:ERR 22	
2027	007752	056340		:CERR SET
2028	007754	062574		:AFTER WRITE CHECK CMD
2029	007756	064002		
2030	007760	064470		
2031			:ERR 23	
2032	007762	056255		:MSG B0 ERROR
2033	007764	062322		:AFTER WRITE DATA CMD
2034	007766	064210		
2035	007770	064564		
2036			:ERROR 24	
2037	007772	056340		:CERR SET
2038	007774	062142		:AFTER SCLR
2039	007776	064002		
2040	010000	064470		
2041			:ERR 25	
2042	010002	056317		:MSG B1 ERROR
2043	010004	062322		
2044	010006	064210		
2045	010010	064564		
2046			:ERR 26	
2047	010012	056255		
2048	010014	062272		:AFTER READ DATA CMD
2049	010016	064210		
2050	010020	064564		
2051			:ERROR 27	
2052	010022	056567		:VOL VALID NOT SET
2053	010024	062064		:AFTER PACK CMD
2054	010026	064002		
2055	010030	064470		
2056			:ERR 30	
2057	010032	056317		:MSG B1 ERROR
2058	010034	062272		:AFTER READ DATA CMD.
2059	010036	064210		
2060	010040	064564		
2061			:ERR 31	
2062	010042	056255		:MSG B0 ERROR
2063	010044	062574		:AFTER WRITE CHECK CMD
2064	010046	064210		
2065	010050	064564		
2066			:ERR 32	
2067	010052	056317		:MSG B1 ERROR
2068	010054	062574		
2069	010056	064210		
2070	010060	064564		
2071			:ERR 33	
2072	010062	055744		:CONTR NOT READY
2073	010064	062222		:AFTER OFFSET CMD
2074	010066	064002		
2075	010070	064470		
2076			:ERR 34	
2077	010072	056002		:NO ATTN
2078	010074	062222		
2079	010076	064002		
2080				

2081	010100	064470		DF10	
2082			:ERR 35	EM17	:MSG A0 ERROR
2083	010102	056234		DH53	:DURING OFFSET COMMAND
2084	010104	063550		DT13	
2085	010106	064210		DF21	
2086	010110	064564	:ERR 36		:MSG A1 ERROR
2087				EM19	
2088	010112	056276		DH53	
2089	010114	063550		DT13	
2090	010116	064210		DF21	
2091	010120	064564	:ERR 37		:MSG A1 ERROR
2092				EM19	:AFTER OFFSET CMD
2093	010122	056276		DH24	
2094	010124	062222		DT13	
2095	010126	064210		DF21	
2096	010130	064564	:ERR 40		:MSG B1 ERROR
2097				EM20	
2098	010132	056317		DH24	
2099	010134	062222		DT13	
2100	010136	064210		DF21	
2101	010140	064564	:ERR 41		:UNEXP MEM PCRTY TRAP
2102				EM14	:TEST #, TRAP PC
2103	010142	056024		DH8	
2104	010144	061411		DT3	
2105	010146	064024		DF2	
2106	010150	064404	:ERR 42		:CYL ADDR IN B2 DID NOT REMAIN CLEARED
2107				EM41	
2108	010152	057242		DH24	
2109	010154	062222		DT14	
2110	010156	064250		DF22	
2111	010160	064620	:ERR 43		
2112				EM85	
2113	010162	060143		DH22	
2114	010164	062170		DT1	
2115	010166	064002		DF10	
2116	010170	064470	:ERR 44		:WCE AT CYL 411, TRK 2, SEC 21
2117				EM15	
2118	010172	056062		DH1	
2119	010174	061123		DT10	
2120	010176	064170		DF4	
2121	010200	064424	:ERR 45		:OFFSET BIT IN RKMR2 CLEARED
2122				EM85	:AFTER SEEK TO SELF
2123	010202	060143		DH51	
2124	010204	063461		DT1	
2125	010206	064002		DF10	
2126	010210	064470	:ERR 46		:DETECTED 10 BAD SECTORS
2127				EM25	:AFTER WRITE DATA CMD.
2128	010212	056622		DH27	
2129	010214	062322		DT1	
2130	010216	064002		DF10	
2131	010220	064470	:ERROR 47		:CYL DIFF/OFFSET IN RKMR2 NOT CLEARED
2132				EM39	:AFTER RECAL CMD
2133	010222	057135		DH17	
2134	010224	062013		DT14	
2135	010226	064250		DF22	
2136	010230	064620			

2137			:ERROR 50		
2138	010232	057204		EM40	:CYL ADDR IN RKMR3 NOT CLEARED
2139	010234	062013		DH17	:AFTER RECAL COMD
2140	010236	064250		DT14	
2141	010240	064620		DF22	
2142			:ERR 51		
2143	010242	060463		EM93	:WRONG CYL# IN HEADER WORD (MISPOSITION)
2144	010244	062247		DH25	:AFTER SEEK CMD
2145	010246	064152		DT9	
2146	010250	064550		DF20	
2147			:ERR 52		
2148	010252	056234		EM17	:MSG A0 ERROR
2149	010254	062322		DH27	:AFTER WRITE DATA CMD
2150	010256	064210		DT13	
2151	010260	064564		DF21	
2152			:ERR 53		
2153	010262	056276		EM19	:MSG A1 ERROR
2154	010264	062322		DH27	
2155	010266	064210		DT13	
2156	010270	064564		DF21	
2157			:ERR 54		
2158	010272	056234		EM17	:MSG A0 ERROR
2159	010274	062272		DH26	:AFTER READ DATA CMD
2160	010276	064210		DT13	
2161	010300	064564		DF21	
2162			:ERROR 55		
2163	010302	056002		EM13	:NO ATTN
2164	010304	062013		DH17	:AFTER RECAL CMD
2165	010306	064002		DT1	
2166	010310	064470		DF10	
2167			:ERR 56		
2168	010312	056276		EM19	:MSG A1 ERROR
2169	010314	062272		DH26	
2170	010316	064210		DT13	
2171	010320	064564		DF21	
2172			:ERR 57		
2173	010322	056234		EM17	:MSG A0 ERROR
2174	010324	062574		DH32	:AFTER WRITE CHECK CMD
2175	010326	064210		DT13	
2176	010330	064564		DF21	
2177			:ERR 60		
2178	010332	056276		EM19	:MSG A1 ERROR
2179	010334	062574		DH32	
2180	010336	064210		DT13	
2181	010340	064564		DF21	
2182			:ERR 61		
2183	010342	056255		EM18	:MSG B0 ERROR
2184	010344	063550		DH53	:DURING OFFSET CMD
2185	010346	064210		DT13	
2186	010350	064564		DF21	
2187			:ERR 62		
2188	010352	056317		EM20	:MSG B1 ERROR
2189	010354	063550		DH53	
2190	010356	064210		DT13	
2191	010360	064564		DF21	
2192			:ERR 63		

2193	010362	056674	EM26	:BSE ERROR IN WRITE CMD NOT ON BSE TABLE
2194	010364	062322	DH27	:AFTER WRITE DATA CMD
2195	010366	064002	DT1	
2196	010370	064470	DF10	
2197			:ERR 64	
2198	010372	060244	EM88	:DID NOT FIND SECTOR 0 FROM INDEX
2199	010374	063576	DH54	:AFTER FORMAT CHANGE AND READY REC'D
2200	010376	064002	DT1	
2201	010400	064470	DF10	
2202			:ERR 65	
2203	010402	056753	EM27	:DETECTED BSE IN READ BUT NOT IN WRITE CMD.
2204	010404	061123	DH1	
2205	010406	064002	DT1	
2206	010410	064374	DF1	
2207			:ERR 66	
2208	010412	057343	EM60	:NO HEAD HOME
2209	010414	062037	DH18	:AFTER UNLOAD CMD
2210	010416	064002	DT1	
2211	010420	064470	DF10	
2212			:ERR 67	
2213	010422	060305	EM89	:HDS HOME NOT CLEARED
2214	010424	061432	DH9	:DURING START SPIN CMD
2215	010426	064002	DT1	
2216	010430	064470	DF10	
2217			:ERROR 70	
2218	010432	060345	EM90	:SERVO SIG PRES NOT SET
2219	010434	061432	DH9	
2220	010436	064002	DT1	
2221	010440	064470	DF10	
2222			:ERR 71	
2223	010442	060405	EM91	:REV NOT SET
2224	010444	061432	DH9	
2225	010446	064002	DT1	
2226	010450	064470	DF10	
2227			:ERROR 72	
2228	010452	060432	EM92	:REV NOT CLEARED
2229	010454	061432	DH9	
2230	010456	064002	DT1	
2231	010460	064470	DF10	
2232			:ERR 73	
2233	010462	056002	EM13	:NO ATTN
2234	010464	062037	DH18	:AFTER UNLOAD CMD
2235	010466	064002	DT1	
2236	010470	064470	DF10	
2237			:ERR 74	
2238	010472	056002	EM13	:NO ATTN
2239	010474	061505	DH10	:AT END OF HEAD LOADING
2240	010476	064002	DT1	
2241	010500	064470	DF10	
2242			:ERR 75	
2243	010502	056362	EM22	:NO DRIVS IN \$DEVN
2244	010504	061123	DH1	
2245	010506	064002	DT1	
2246	010510	064374	DF1	
2247			:ERR 76	
2248	010512	056467	EM23	:NO DRIVS ON BUSS

2249	010514	061123	DH1	
2250	010516	064002	DT1	
2251	010520	064374	DF1	
2252			:ERR 77	
2253	010522	000000	0	
2254	010524	000000	0	
2255	010526	000000	0	
2256	010530	000000	0	
2257			:ERR 100	
2258	010532	000000	0	
2259	010534	000000	0	
2260	010536	000000	0	
2261	010540	000000	0	
2262			:ERROR 101	
2263	010542	060544	EM94	:OFFSET NOT CLEARED
2264	010544	063334	DH47	:AFTER READ HEADER WITH MOVEMENT
2265	010546	064250	DT14	
2266	010550	064620	DF22	
2267			:ERROR 102	
2268	010552	060600	EM95	:FORMAT NOT SET
2269	010554	062322	DH27	:AFTER WRITE DATA CMD
2270	010556	064002	DT1	
2271	010560	064470	DF10	
2272			:ERR 103	
2273	010562	060600	EM95	
2274	010564	062574	DH32	:AFTER WRITE CHECK CMD
2275	010566	064002	DT1	
2276	010570	064470	DF10	
2277			:ERR 104	
2278	010572	057135	EM39	:OFFSET NOT RESET
2279	010574	063715	DH57	:AFTER WRITE CMD WITH OFFSET
2280	010576	064250	DT14	
2281	010600	064620	DF22	
2282			:ERR 105	
2283	010602	057204	EM40	:CYL ADDR NOT 0
2284	010604	063715	DH57	
2285	010606	064250	DT14	
2286	010610	064620	DF22	
2287			:ERR 106	
2288	010612	060634	EM96	:CANNOT FIND SECTOR 2318
2289	010614	061123	DH1	
2290	010616	064002	DT1	
2291	010620	064374	DF1	
2292			:ERR 107	
2293	010622	060665	EM97	:HEAD SWITCHING TOO LONG
2294	010624	062322	DH27	:AFTER WRITE DTA CMD
2295	010626	064050	DT5	
2296	010630	064514	DF15	
2297			:ERR 110	
2298	010632	060752	EM98	:CANNOT FIND CYL 128
2299	010634	062247	DH25	:AFTER SEEK CMD
2300	010636	064002	DT1	
2301	010640	064470	DF10	
2302			:ERR 111	
2303	010642	061003	EM99	:CANNOT FIND CYL 256
2304	010644	062247	DH25	

2305	010646	064002	DT1	
2306	010650	064470	DF10	
2307			:ERR 112	
2308	010652	061034	EM100	:DRIVE OFF TRACK SET
2309	010654	062322	DH27	:AFTER WRITE DATA CMD
2310	010656	064030	DT4	
2311	010660	064454	DF6	
2312			:ERR 113	
2313	010662	057072	EM36	:CYL ADDR IN RKMR3 INCORRECT
2314	010664	062322	DH27	
2315	010666	064030	DT4	
2316	010670	064454	DF6	
2317			:ERROR 114	
2318	010672	060206	EM86	:OFFSET IN A2 NOT = RKASOF
2319	010674	062222	DH24	:AFTER OFFSET CMD
2320	010676	064250	DT14	
2321	010700	064620	DF22	
2322			:ERR 115	
2323	010702	060206	EM86	
2324	010704	062170	DH22	:AFTER DRIVE CLEAR CMD
2325	010706	064250	DT14	
2326	010710	064620	DF22	
2327			:ERROR 116	
2328	010712	055744	EM12	:CONT NOT RDY
2329	010714	062064	DH19	:AFTER PACK CMD
2330	010716	064002	DT1	
2331	010720	064470	DF10	
2332			:ERROR 117	
2333	010722	055744	EM12	:CONT NOT RDY
2334	010724	062107	DH20	:AFTER SEL DR CMD
2335	010726	064002	DT1	
2336	010730	064470	DF10	
2337			:ERROR 120	
2338	010732	055744	EM12	
2339	010734	062142	DH21	:AFTER SUBSYS CLEAR
2340	010736	064002	DT1	
2341	010740	064470	DF10	
2342			:ERROR 121	
2343	010742	055744	EM12	
2344	010744	061432	DH9	:AFTER START SPINDLE CMD
2345	010746	064002	DT1	
2346	010750	064470	DF10	
2347			:ERROR 122	
2348	010752	061071	EM101	:DID NOT GO TO CYL 10
2349	010754	062515	DH30	:AFTER READ HEADER CMD
2350	010756	064250	DT14	
2351	010760	064620	DF22	
2352			:ERROR 123	
2353	010762	060206	EM86	:A2 OFFSET NOT = RKASOF
2354	010764	063461	DH51	:AFTER SEEK TO SELF
2355	010766	064250	DT14	
2356	010770	064620	DF22	
2357			:ERROR 124	
2358	010772	055744	EM12	
2359	010774	062013	DH17	:AFTER RECAL CMD
2360	010776	064002	DT1	

2361	011000	064470		
2362			DF10	
2363	011002	057551	;ERR 125	
2364	011004	060052	EM73	;CTO SET
2365	011006	064002	EM84	;WHILE WAITING FOR OR REC'D CONTR RDY. MSG A&B BAD
2366	011010	064434	DT1	
2367			DF5	
2368	011012	057617	;ERR 126	
2369	011014	060052	EM79	;NED SET
2370	011016	064002	EM84	
2371	011020	064050	DT1	
2372			DT5	
2373	011022	055334	;ERR 127	
2374	011024	060052	EM5	;MDS SET
2375	011026	064002	EM84	
2376	011030	064434	DT1	
2377			DF5	
2378	011032	000000	;ERROR 130	
2379	011034	000000	0	
2380	011036	000000	0	
2381	011040	000000	0	
2382			;ERROR 131	
2383	011042	055744	EM12	;NO RDY
2384	011044	062247	DH25	;AFTER SEEK CMD
2385	011046	064002	DT1	
2386	011050	064470	DF10	
2387			;ERROR 132	
2388	011052	056002	EM13	;NO ATTN
2389	011054	062247	DH25	
2390	011056	064002	DT1	
2391	011060	064470	DF10	
2392			;ERROR 133	
2393	011062	000000	0	
2394	011064	000000	0	
2395	011066	000000	0	
2396	011070	000000	0	
2397			;ERROR 134	
2399	011072	000000	0	
2400	011074	000000	0	
2401	011076	000000	0	
2402			;ERROR 135	
2403	011102	000000	0	
2404	011104	000000	0	
2405	011106	000000	0	
2406	011110	000000	0	
2407			;ERROR 136	
2408	011112	000000	0	
2409	011114	000000	0	
2410	011116	000000	0	
2411	011120	000000	0	
2412			;ERROR 137	
2413	011122	057135	EM39	;CYL DIFF/OFFSET IN RKMR2 NOT CLEARED
2414	011124	062247	DH25	
2415	011126	064002	DT1	
2416	011130	064470	DF10	

2417				
2418	011132	056234	;ERR 140	
2419	011134	063461	EM17	;MSG AD ERROR
2420	011136	064210	DH51	;AFTER SEEK TO SELF
2421	011140	064564	DT13	
2422			DF21	
2423	011142	056255	;ERR 141	
2424	011144	063461	EM18	
2425	011146	064210	DH51	
2426	011150	064564	DT13	
2427			DF21	
2428	011152	056276	;ERR 142	
2429	011154	063461	EM19	
2430	011156	064210	DH51	
2431	011160	064564	DT13	
2432			DF21	
2433	011162	056317	;ERR 143	
2434	011164	063461	EM20	
2435	011166	064210	DH51	
2436	011170	064564	DT13	
2437			DF21	
2438	011172	000000	;ERROR 144	
2439	011174	000000	0	
2440	011176	000000	0	
2441	011200	000000	0	
2442			;ERROR 145	
2443	011202	000000	0	
2444	011204	000000	0	
2445	011206	000000	0	
2446	011210	000000	0	
2447			;ERROR 146	
2448	011212	000000	0	
2449	011214	000000	0	
2450	011216	000000	0	
2451	011220	000000	0	
2452			;ERROR 147	
2453	011222	000000	0	
2454	011224	000000	0	
2455	011226	000000	0	
2456	011230	000000	0	
2457			;ERROR 150	
2458	011232	000000	0	
2459	011234	000000	0	
2460	011236	000000	0	
2461	011240	000000	0	
2462			;ERROR 151	
2463	011242	055744	EM12	;NO RDY
2464	011244	062170	DH22	;AFTER CLEAR CMD
2465	011246	064002	DT1	
2466	011250	064470	DF10	
2467			;ERROR 152	
2468	011252	000000	0	
2469	011254	000000	0	
2470	011256	000000	0	
2471	011260	000000	0	
2472			;ERROR 153	

K04

UNIBUS RK06 DRIVE DIAGNOSTIC PART 2
DZR6IC.P11 07-OCT-76 13:50

MACY11 27(1006) 07-OCT-76 14:14 PAGE 49
ERROR POINTER TABLE

SEQ 0049

2473 011262 000000
 2474 011264 000000
 2475 011266 000000
 2476 011270 000000
 2477
 2478 011272 057310
 2479 011274 062170
 2480 011276 064002
 2481 011300 064470
 2482
 2483 011302 000000
 2484 011304 000000
 2485 011306 000000
 2486 011310 000000
 2487
 2488 011312 000000
 2489 011314 000000
 2490 011316 000000
 2491 011320 000000
 2492
 2493 011322 000000
 2494 011324 000000
 2495 011326 000000
 2496 011330 000000
 2497
 2498 011332 000000
 2499 011334 000000
 2500 011336 000000
 2501 011340 000000
 2502
 2503 011342 000000
 2504 011344 000000
 2505 011346 000000
 2506 011350 000000
 2507
 2508 011352 000000
 2509 011354 000000
 2510 011356 000000
 2511 011360 000000
 2512
 2513 011362 000000
 2514 011364 000000
 2515 011366 000000
 2516
 2517
 2518
 2519
 2520
 2521
 2522
 2523
 2524
 2525
 2526
 2527
 2528

0
 0
 0
 0
 ;ERROR 154
 EM55
 DH22
 DT1
 DF10
 ;ERROR 155
 0
 0
 0
 0
 ;ERROR 156
 0
 0
 0
 0
 ;ERROR 157
 0
 0
 0
 0
 ;ERROR 160
 0
 0
 0
 0
 ;ERROR 161
 0
 0
 0
 0
 ;ERROR 162
 0
 0
 0
 0
 ;ERROR 163
 0
 0
 0
 0

;ATTN NOT CLEARED



2529				
2530				
2531				
2532				
2533				
2534				
2535				
2536				
2537	011370	000000	0	
2538			:ERROR 164	
2539	011372	000000	0	
2540	011374	000000	0	
2541	011376	000000	0	
2542	011400	000000	0	
2543			:ERROR 165	
2544	011402	000000	0	
2545	011404	000000	0	
2546	011406	000000	0	
2547	011410	000000	0	
2548			:ERROR 166	
2549	011412	000000	0	
2550	011414	000000	0	
2551	011416	000000	0	
2552	011420	000000	0	
2553			:ERROR 167	
2554	011422	000000	0	
2555	011424	000000	0	
2556	011426	000000	0	
2557	011430	000000	0	
2558			:ERROR 170	
2559	011432	000000	0	
2560	011434	000000	0	
2561	011436	000000	0	
2562	011440	000000	0	
2563			:ERROR 171	
2564	011442	055744	EM12	;NO RDY
2565	011444	062515	DH30	;AFTER READ HEADER CMD
2566	011446	064002	DT1	
2567	011450	064470	DF10	
2568			:ERROR 172	
2569	011452	000000	0	
2570	011454	000000	0	
2571	011456	000000	0	
2572	011460	000000	0	
2573			:ERROR 173	
2574	011462	057421	EM63	;DLT SET
2575	011464	062515	DH30	
2576	011466	064050	DT5	
2577	011470	064514	DF15	
2578			:ERROR 174	
2579	011472	056340	EM21	;CERR SET
2580	011474	062515	DH30	
2581	011476	064050	DT5	
2582	011500	064514	DF15	
2583			:ERROR 175	
2584	011502	057135	EM39	;CYL DIFF NOT CLEARED

2585	011504	061505	DH10	;AT END OF HEAD LOADING
2586	011506	064002	DT1	
2587	011510	064470	DF10	
2588				
2589	011512	057204	;ERROR 176	
2590	011514	061505	EM40	;CYL ADDR NOT CLEARED.
2591	011516	064002	DH10	
2592	011520	064470	DT1	
2593			DF10	
2594	011522	000000	;ERROR 177	
2595	011524	000000	0	
2596	011526	000000	0	
2597	011530	000000	0	
2598			0	
2599	011532	055744	;ERROR 200	
2600	011534	062776	EM12	;NO RDY
2601	011536	064050	DH39	;AFTER WRITE HEADER CMD
2602	011540	064514	DT5	
2603			DF15	
2604	011542	056340	;ERROR 201	
2605	011544	062776	EM21	;CERR SET
2606	011546	064050	DH39	
2607	011550	064514	DT5	
2608			DF15	
2609	011552	057442	;ERROR 202	
2610	011554	061123	EM65	;READ HEADER ERROR
2611	011556	064112	DH1	
2612	011560	064504	DT7	
2613			DF14	
2614	011562	000000	;ERROR 203	
2615	011564	000000	0	
2616	011566	000000	0	
2617	011570	000000	0	
2618			0	
2619	011572	000000	;ERROR 204	
2620	011574	000000	0	
2621	011576	000000	0	
2622	011600	000000	0	
2623			0	
2624	011602	000000	;ERROR 205	
2625	011604	000000	0	
2626	011606	000000	0	
2627	011610	000000	0	
2628			0	
2629	011612	000000	;ERROR 206	
2630	011614	000000	0	
2631	011616	000000	0	
2632	011620	000000	0	
2633			0	
2634	011622	057072	;ERROR 207	
2635	011624	062247	EM36	;CYL ADDR IN RKMR3 INCORRECT
2636	011626	064030	DH25	;AFTER SEEK CMD
2637	011630	064454	DT4	
2638			DF6	
2639	011632	056340	;ERROR 210	
2640	011634	062247	EM21	;CERR SET
			DH25	

2641	011636	064002		
2642	011640	064470		
2643			;ERROR 211	DT1
2644	011642	000000		DF10
2645	011644	000000		0
2646	011646	000000		0
2647	011650	000000		0
2648			;ERROR 212	0
2649	011652	000000		0
2650	011654	000000		0
2651	011656	000000		0
2652	011660	000000		0
2653			;ERROR 213	0
2654	011662	000000		0
2655	011664	000000		0
2656	011666	000000		0
2657	011670	000000		0
2658			;ERROR 214	0
2659	011672	000000		0
2660	011674	000000		0
2661	011676	000000		0
2662	011700	000000		0
2663			;ERROR 215	0
2664	011702	000000		0
2665	011704	000000		0
2666	011706	000000		0
2667	011710	000000		0
2668			;ERROR 216	0
2669	011712	000000		0
2670	011714	000000		0
2671	011716	000000		0
2672	011720	000000		0
2673			;ERROR 217	0
2674	011722	000000		0
2675	011724	000000		0
2676	011726	000000		0
2677	011730	000000		0
2678			;ERROR 220	0
2679	011732	000000		0
2680	011734	000000		0
2681	011736	000000		0
2682	011740	000000		0
2683			;ERROR 221	EM17
2684	011742	056234		DH17
2685	011744	062013		DT13
2686	011746	064210		DF21
2687	011750	064564		0
2688			;ERROR 222	EM19
2689	011752	056276		DH17
2690	011754	062013		DT13
2691	011756	064210		DF21
2692	011760	064564		0
2693			;ERROR 223	0
2694	011762	000000		0
2695	011764	000000		0
2696	011766	000000		0

;MSG A0 ERROR

;MSG A1 ERROR

2697	011770	000000	0	
2698			:ERROR 224	
2699	011772	000000	0	
2700	011774	000000	0	
2701	011776	000000	0	
2702	012000	000000	0	
2703			:ERROR 225	
2704	012002	000000	0	
2705	012004	000000	0	
2706	012006	000000	0	
2707	012010	000000	0	
2708			:ERROR 226	
2709	012012	055744	EM12	:NO RDY
2710	012014	062272	DH26	:AFTER READ DATA CMD
2711	012016	064002	DT1	
2712	012020	064470	DF10	
2713			:ERROR 227	
2714	012022	056340	EM21	:CERR SET
2715	012024	062272	DH26	
2716	012026	064050	DT5	
2717	012030	064514	DF15	
2718			:ERROR 230	
2719	012032	056171	EM16	:CANNOT READ BSE INFO
2720	012034	061643	DH13	:ON SEC 10, 12, 14, 16, 18, 20
2721	012036	064002	DT1	
2722	012040	064530	DF17	
2723			:ERROR 231	
2724	012042	056171	EM16	
2725	012044	061727	DH14	:ON SEC 11, 13, 15, 17, 19, 21
2726	012046	064002	DT1	
2727	012050	064530	DF17	
2728			:ERROR 232	
2729	012052	000000	0	
2730	012054	000000	0	
2731	012056	000000	0	
2732	012060	000000	0	
2733			:ERROR 233	
2734	012062	056171	EM16	:CANNOT READ BSE INFO
2735	012064	063133	DH42	:ON SECT 0,2,4,6,8
2736	012066	064002	DT1	
2737	012070	064530	DF17	
2738			:ERROR 234	
2739	012072	056171	EM16	
2740	012074	063204	DH43	:ON SECT 1,3,5,7,9
2741	012076	064002	DT1	
2742	012100	064530	DF17	
2743			:ERROR 235	
2744	012102	057520	EM69	:ALIGN CARTRIDGE USED
2745	012104	063255	DH44	:WILL BYPASS FORMAT & ALL R/W TESTS
2746	012106	064002	DT1	
2747	012110	064470	DF10	
2748			:ERRCR 236	
2749	012112	000000	0	
2750	012114	000000	0	
2751	012116	000000	0	
2752	012120	000000	0	

2753			:ERROR 237	
2754	012122	000000		0
2755	012124	000000		0
2756	012126	000000		0
2757	012130	000000		0
2758			:ERROR 240	
2759	012132	000000		0
2760	012134	000000		0
2761	012136	000000		0
2762	012140	000000		0
2763			:ERROR 241	
2764	012142	000000		0
2765	012144	000000		0
2766	012146	000000		0
2767	012150	000000		0
2768			:ERROR 242	
2769	012152	000000		0
2770	012154	000000		0
2771	012156	000000		0
2772	012160	000000		0
2773			:ERROR 243	
2774				EM36
2775	012162	057072		DH25
2776	012164	062247		DT8
2777	012166	064132		DF6
2778	012170	064454		
2779			:ERR 244	
2780	012172	057572		EM74
2781	012174	063106		DH41
2782	012176	064002		DT1
2783	012200	064470		DF10
2784			:ERR 245	
2785	012202	000000		0
2786	012204	000000		0
2787	012206	000000		0
2788	012210	000000		0
2789			:ERR 246	
2790	012212	000000		0
2791	012214	000000		0
2792	012216	000000		0
2793	012220	000000		0
2794			:ERR 247	
2795	012222	000000		0
2796	012224	000000		0
2797	012226	000000		0
2798	012230	000000		0
2799			:ERR 250	
2800	012232	000000		0
2801	012234	000000		0
2802	012236	000000		0
2803	012240	000000		0
2804			:ERR 251	
2805	012242	000000		0
2806	012244	000000		0
2807	012246	000000		0
2808	012250	000000		0

:CYL ADDR IN RKM3 INCORRECT
:AFTER SEEK CMD

:RTZ NOT SET
:DURING RECAL CMD

2809			:ERR 252	
2810	012252	000000		0
2811	012254	000000		0
2812	012256	000000		0
2813	012260	000000		0
2814			:ERR 253	
2815	012262	000000		0
2816	012264	000000		0
2817	012266	000000		0
2818	012270	000000		0
2819			:ERR 254	
2820	012272	000000		0
2821	012274	000000		0
2822	012276	000000		0
2823	012300	000000		0
2824			:ERR 255	
2825	012302	000000		0
2826	012304	000000		0
2827	012306	000000		0
2828	012310	000000		0
2829			:ERR 256	
2830	012312	000000		0
2831	012314	000000		0
2832	012316	000000		0
2833	012320	000000		0
2834			:ERR 257	
2835	012322	000000		0
2836	012324	000000		0
2837	012326	000000		0
2838	012330	000000		0
2839			:ERR 260	
2840	012332	056234		EM17
2841	012334	062222		DH24
2842	012336	064210		DT13
2843	012340	064564		DF21
2844			:ERR 261	
2845	012342	056255		EM18
2846	012344	062222		DH24
2847	012346	064210		DT13
2848	012350	064564		DF21
2849			:ERR 262	
2850	012352	000000		0
2851	012354	000000		0
2852	012356	000000		0
2853	012360	000000		0
2854			:ERR 263	
2855	012362	000000		0
2856	012364	000000		0
2857	012366	000000		0
2858	012370	000000		0
2859			:ERR 264	
2860	012372	000000		0
2861	012374	000000		0
2862	012376	000000		0
2863	012400	000000		0
2864			:ERR 265	

:MSG A0 ERROR
:AFTER OFFSET CMD

:MSG B0 ERROR

E05

UNIBUS RKO6 DRIVE DIAGNOSTIC PART 2
 DERRIC.P11 07-OCT-76 13:50

MACY11 27(1006) 07-OCT-76 14:14 PAGE 56
 ERROR POINTER TABLE

SEQ 0056

2865	012402	056255	EM18	:MSG B0 ERROR
2866	012404	062170	DH22	:AFTER DRIVE CLEAR CMD
2867	012406	064210	DT13	
2868	012410	064564	DF21	
2869			:ERR 266	
2870	012412	056317	EM20	:MSG B1 ERROR
2871	012414	062170	DH22	
2872	012416	064210	DT13	
2873	012420	064564	DF21	
2874			:ERR 267	
2875	012422	056255	EM18	:MSG B0 ERROR
2876	012424	062776	DH39	:AFTER WRITE HEADER CMD
2877	012426	064210	DT13	
2878	012430	064564	DF21	
2879			:ERR 270	
2880	012432	056317	EM20	:MSG B1 ERROR
2881	012434	062776	DH39	
2882	012436	064210	DT13	
2883	012440	064564	DF21	
2884			:ERR 271	
2885	012442	056255	EM18	
2886	012444	062515	DH30	:AFTER RD. HDR. CMD.
2887	012446	064210	DT13	
2888	012450	064564	DF21	
2889			:ERR 272	
2890	012452	056317	EM20	
2891	012454	062515	DH30	
2892	012456	064210	DT13	
2893	012460	064564	DF21	
2894			:ERR 273	
2895	012462	056234	EM17	:MSG A0 ERROR
2896	012464	062170	DH22	:AFTER DRV CLR CMD
2897	012466	064210	DT13	
2898	012470	064564	DF21	
2899			:ERR 274	
2900	012472	056276	EM19	:MSG A1 ERROR
2901	012474	062170	DH22	
2902	012476	064210	DT13	
2903	012500	064564	DF21	
2904			:ERR 275	
2905	012502	056255	EM18	:MSG B0 ERROR
2906	012504	062013	DH17	:AFTER RECAL CMD
2907	012506	064210	DT13	
2908	012510	064564	DF21	
2909			:ERR 276	
2910	012512	056317	EM20	:MSG B1 ERROR
2911	012514	062013	DH17	
2912	012516	064210	DT13	
2913	012520	064564	DF21	
2914			:ERR 277	
2915	012522	056234	EM17	:MSG A0 ERROR
2916	012524	062776	DH39	:AFTER WRITE HEADER CMD
2917	012526	064210	DT13	
2918	012530	064564	DF21	
2919			:ERR 300	
2920	012532	056276	EM19	:MSG A1 ERROR

2921	012534	062776	DH39	
2922	012536	064210	DT13	
2923	012540	064564	DF21	
2924			:ERR 301	
2925	012542	056234	EM17	
2926	012544	062515	DH30	:AFT RD HDR. CMD
2927	012546	064210	DT13	
2928	012550	064564	DF21	
2929			:ERR 302	
2930	012552	056276	EM19	
2931	012554	062515	DH30	
2932	012556	064210	DT13	
2933	012560	064564	DF21	
2934			:ERR 303	
2935	012562	000000	0	
2936	012564	000000	0	
2937	012566	000000	0	
2938	012570	000000	0	
2939			:ERR 304	
2940	012572	000000	0	
2941	012574	000000	0	
2942	012576	000000	0	
2943	012600	000000	0	
2944			0	

```

2945
2946      .SBTTL PROGRAM SETUP
2947
2948 012602 012737 000001 001336 PARSR: MOV    #1,PARAM      ;SET FLAG FOR 220 START
2949 012610 005037 001340          CLR    BYPWRT
2950 012614 005037 001342          CLR    BYPTIM
2951 012620 000450          BR     PRGSRT      ;START PROGRAM
2952
2953 012622 105037 001336          CLRB   PARAM
2954 012626 012737 000001 001340 BYWRT: MOV    #1,BYPWRT    ;BYPASS WRITE TESTS
2955 012634 105037 001342          CLRB   BYPTIM
2956 012640 000440          BR     PRGSRT
2957
2958 012642 105037 001336          CLRB   PARAM
2959 012646 105037 001340          CLRB   BYPWRT
2960 012652 012737 000001 001342 BYTIM: MOV    #1,BYPTIM    ;BYPASS TIMING TESTS
2961 012660 000430          BR     PRGSRT
2962
2963 012662 012737 000001 001336 BYWRTA: MOV   #1,PARAM
2964 012670 012737 000001 001340        MOV   #1,BYPWRT
2965 012676 105037 001342          CLRB   BYPTIM
2966 012702 000417          BR     PRGSRT
2967
2968 012704 012737 000001 001336 BYTIMA: MOV   #1,PARAM
2969 012712 105037 001340          CLRB   BYPWRT
2970 012716 012737 000001 001342        MOV   #1,BYPTIM
2971 012724 000406          BR     PRGSRT
2972
2973 012726 105037 001336          CLRB   PARAM      ;CLEAR FOR 200 START
2974 012732 005037 001340          CLR    BYPWRT
2975 012736 005037 001342          CLR    BYPTIM
2976 012742 000005          PRGSRT: RESET      ;CLEAR ALL INT ENABLE & INIT
2977 012744 012706 001100          MOV    #STACK,SP    ;SETUP STACK POINTER
2978 012750 012746 000000          MOV    #PRO,-(SP)   ;PSW LOADED TO BE
2979 012754 012746 012762          MOV    #IS,-(SP)   ;LSI-11 COMPATABLE
2980 012760 000002          RTI      ;ENABLE ALL INTERRUPTS
2981
2982 012762 004737 046322          IS:   JSR    PC,STKINT ;SETUP KB VECTOR ADDR, PRIORITY 4
2983                                     ;& TURN ON KB INTERRUPT
2984
2985
2986      ;*** CPU PRIORITY LEVEL NOW AT 0 ***
2987      ;*** ANY DEVICE WHICH SETS ITS    ***
2988      ;*** INTERRUPT ENABLE BIT WILL   ***
2989      ;*** SERVICED.                    ***
2990
2991      ;CLOCK INTERRUPTS WILL CHANGE CPU PRIORITY TO LEVEL 6 (IN 'STS')
2992      ;RK06 CONTROLLER INTERRUPTS WILL CHANGE CPU PRIORITY TO LEVEL 5 IN 'SETINT')
2993      ;KEYBOARD INTERRUPTS WILL CHANGE CPU PRIORITY TO LEVEL 4 (SEE ABOVE)
2994
2995      ;ALL 'SYSMAC' TRAPS WILL CHANGE CPU PRIORITY TO LEVEL 7 (SEE BELOW)
2996
2997                                     ;SYSMAC 'SETUP'
2998      .SBTTL INITIALIZE THE COMMON TAGS
2999      ;;CLEAR THE COMMON TAGS ($CMTAG) AREA
3000 012766 012706 001100          MOV    #CMTAG,R6    ;;FIRST LOCATION TO BE CLEARED

```

H05

JNIBUS RK06 DRIVE DIAGNOSTIC PART 2
DZR6IC.P11 07-OCT-76 13:50

MACY11 27(1006) 07-OCT-76 14:14 PAGE 59
INITIALIZE THE COMMON TAGS

SEQ 0059

```

3001 012772 005026          CLR      (R6)+          ;;CLEAR MEMORY LOCATION
3002 012774 022706 001140  CMP      #SWR,R6 ;;DONE?
3003 013000 001374          BNE     -6            ;;LOOP BACK IF NO
3004 013002 012706 001100  MOV     #STACK,SP    ;;SETUP THE STACK POINTER
3005                                ;;INITIALIZE A FEW VECTORS
3006 013006 012737 044430 000020  MOV     #SSCOPE,#IOTVEC ;;IOT VECTOR FOR SCOPE ROUTINE
3007 013014 012737 000340 000022  MOV     #340,#IOTVEC+2 ;;LEVEL 7
3008 013022 012737 044710 000030  MOV     #SEERR,#EMTVEC ;;EMT VECTOR FOR ERROR ROUTINE
3009 013030 012737 000340 000032  MOV     #340,#EMTVEC+2 ;;LEVEL 7
3010 013036 012737 050540 000034  MOV     #STRAP,#TRAPVEC ;;TRAP VECTOR FOR TRAP CALLS
3011 013044 012737 000340 000036  MOV     #340,#TRAPVEC+2 ;;LEVEL 7
3012 013052 012737 044164 000024  MOV     #SPWRDN,#PWRVEC ;;POWER FAILURE VECTOR
3013 013060 012737 000340 000026  MOV     #340,#PWRVEC+2 ;;LEVEL 7
3014 013066 013737 036116 036110  MOV     SENDCT,SEOPCT ;;SETUP END-OF-PROGRAM COUNTER
3015 013074 005037 001174          CLR     $TIMES        ;;INITIALIZE NUMBER OF ITERATIONS
3016 013100 005037 001176          CLR     $ESCAPE      ;;CLEAR THE ESCAPE ON ERROR ADDRESS
3017 013104 112737 000001 001115  MOVB   #1,$SERMAX    ;;ALLOW ONE ERROR PER TEST
3018 013112 012737 013112 001106  MOV     #,$SLPADR    ;;INITIALIZE THE LOOP ADDRESS FOR SCOPE
3019 013120 012737 013120 001110  MOV     #,$SLPERR    ;;SETUP THE ERROR LOOP ADDRESS
3020                                ;;SIZE FOR A HARDWARE SWITCH REGISTER. IF NOT FOUND OR IT IS
3021                                ;;EQUAL TO A "-1" SETUP FOR A SOFTWARE SWITCH REGISTER.
3022 013126 013746 000004          MOV     #ERRVEC,-(SP) ;;SAVE ERROR VECTOR
3023 013132 012737 013166 000004  MOV     #64$,#ERRVEC ;;SET UP ERROR VECTOR
3024 013140 012737 177570 001140  MOV     #DSWR,$SWR   ;;SETUP FOR A HARDWARE SWICH REGISTER
3025 013146 012737 177570 001142  MOV     #DDISP,$DISPLAY ;;AND A HARDWARE DISPLAY REGISTER
3026 013154 022777 177777 165756  CMP     #-1,$SWR    ;;TRY TO REFERENCE HARDWARE SWR
3027 013162 001012          BNE     66$         ;;BRANCH IF NO TIMEOUT TRAP OCCURRED
3028                                ;;AND THE HARDWARE SWR IS NOT = -1
3029 013164 000403          BR     65$         ;;BRANCH IF NO TIMEOUT
3030 013166 012716 013174          64$: MOV     #65$,(SP)  ;;SET UP FOR TRAP RETURN
3031 013172 000002          RTI
3032 013174 012737 000176 001140  65$: MOV     #SWREG,$SWR ;;POINT TO SOFTWARE SWR
3033 013202 012737 000174 001142  MOV     #DISPREG,$DISPLAY
3034 013210 012637 000004          66$: MOV     (SP)+,#ERRVEC ;;RESTORE ERROR VECTOR
3035
3036 013214 005037 001216          CLR     $PASS       ;;CLEAR PASS COUNT
3037 013220 132737 000200 001231  BITB   #APTSIZE,$ENVM ;;TEST USER SIZE UNDER APT
3038 013226 001403          BEQ    67$         ;;YES,USE NON-APT SWITCH
3039 013230 012737 001232 001140  MOV     #SSWREG,$SWR ;;NO,USE APT SWITCH REGISTER
3040 013236          67$:
3041 013236 012737 013302 000004  MEMPAR: MOV     #1$,$ERRVEC ;;SETUP TIMEOUT VECTOR
3042 013244 012737 000340 000006  MOV     #PR7,$ERRVEC+2
3043
3044 013252 012701 172100          MOV     #MEMBAS,$R1 ;;ADDR OF MEM CSR
3045 013256 005011          3$: CLR     (R1)      ;;SEE IF CAN REFERENCE
3046 013260 012711 000001          MOV     #1,(R1)    ;;SET ENABLE BIT IF YES
3047 013264 012737 044066 000114  MOV     #MEMERR,$MEMVEC ;;LOAD VECTOR IF NO TIMEOUT
3048 013272 012737 000340 000116  MOV     #PR7,$MEMVEC+2
3049 013300 000401          BR     2$
3050
3051 013302 022626          1$: CMP     (SP)+,(SP)+ ;;ADJ STACK
3052 013304 062701 000002          2$: ADD     #2,$R1   ;;TRY NEXT CSR
3053 013310 020127 172140          CMP     $R1,#MEMBAS+40 ;;SEE IF TRIED ALL
3054 013314 001360          BNE    3$         ;;BR IF NO
3055 013316 012737 000006 000004  MOV     #ERRVEC+2,$ERRVEC ;;RESTORE TRAP CATCHER
3056 013324 005037 000006          CLR     ERRVEC+2

```

JNIBJS RK06 DRIVE DIAGNOSTIC PART 2
DZR6IC.P11 07-OCT-76 13:50

MACY11 27(1006) 07-OCT-76 14:14 PAGE 60
INITIALIZE THE COMMON T4GS

SEQ 0060

3057
3058 013330 004737 036204
3059 013334 005037 001220
3060 013340 005037 001222
3061
3062
3063

JSR PC,CLRF,G ;CLEAR DDUMP THRU SIZFLG
CLR \$DEVCT
CLR \$UNIT

:FIND OUT IF XXDP, ACT, APT; CHAIN OR DUMP MODE

```

3064 ;
3065 ;
3066 013344 005737 000042 START1: TST 42
3067 013350 001014 BNE 1$ ;BR IF AUTO
3068 013352 004737 036224 YSR PC,TITLE ;MANUAL, TYPE PROG ID
3069 013356 123727 000041 000013 CMPB 41,#13 ;13=LOADED BY XXDP
3070 013364 001010 BNE 2$
3071 013366 005237 007500 INC DUMP ;SET RK06 DUMP MODE FLAG
3072 013372 104401 051531 TYPE MSG2 ;REPLACE DRO PACK W/SCRATCH & DO<CR>
3073 013376 000137 013412 JMP ST2
3074 013402 000137 013456 1$: JMP ST3
3075 013406 005237 007506 2$: INC PPTP ;SET ACT/APT/PTP DUMP MODE FLAG
3076 ;
3077 ;
3078 ;CHECK IF ALL PARAMETERS DEFAULTED. IF NOT, BEGIN INPUT DIALOGUE
3079 ;WITH OPERATOR. THE REPLY TO 'DRIVES TO BE TESTED' SHOULD BE

```

K05

UNIBUS Rk06 DRIVE DIAGNOSTIC PART 2
DZR6IC.P11 07-OCT-76 13:50

MACY11 27(1006) 07-OCT-76 14:14 PAGE 62
INITIALIZE THE COMMON TAGS

SEQ 0052

3080
3081
3082
3083

;DRIVE NOS. SEPERATED BY COMMAS & TERMINATED BY <CR>
: EX: DRIVES TO BE TESTED: 1,2,4<CR>
:

```

3084 013412 005737 001336 ST2: TST PARAM
3085 013416 001002 BNE 1$ ;BR IF 220 START
3086 013420 000137 013510 JMP ST4 ;200 START, DEFAULT & SIZE THE BUSS
3087 013424 104401 051602 1$: TYPE MSG3 ;DRIVES TO BE TESTED
3088 013430 004737 036304 JSR PC,DRVS ;GET DR NOS.
3089 013434 104401 051634 TYPE MSG4 ;BUSS ADDR
3090 013440 004737 036444 JSR PC,GBA ;GET BA
3091 013444 104401 051701 TYPE MSG5 ;CONT INT VECTOR
3092 013450 004737 036472 JSR PC,GINT ;GET INT VECTOR
3093 013454 000427 BR ST5
3094
3095
3096 ;AUTO MODE
3097 ;CHECK IF LOADED BY XXDP OR OTHER. SET FLAGS & NO INPUT DIALOGUE.
3098 ;DEFAULT ALL PARAMETERS. TEST ONLY THOSE DRIVES THAT ARE READY
3099 ;ON THE BUSS
3100
3101
3102 013456 123727 000041 000013 ST3: CMPB 41,#13 ;13=LOADED BY XXDP
3103 013464 001007 BNE 1$
3104 013466 005237 007502 INC DDPCH ;SET RK06 CHAIN MODE FLAG
3105 013472 004737 036224 JSR PC,TITLE
3106 013476 104401 052016 TYPE MSG7 ;DRO NOT TSTD
3107 013502 000402 BR ST4
3108 013504 005237 007504 1$: INC ACT11 ;SET ACT AUTO FLAG.
3109
3110 013510 012737 177440 001264 ST4: MOV #177440,$BASE ;DEFAULT VALUE
3111 013516 012737 000210 001314 MOV #210,RKVEC ;DEFAULT VALUE
3112 013524 004737 036524 JSR PC,SETINT
3113 013530 005237 007540 INC SI2FLG ;DO "SIZE THE BUSS" TEST
3114
3115 013534 005037 007352 ST5: CLR UNLD ;INITIALIZE FLAGS
3116 013540 005037 007354 CLR BADHDR ;USED IN 'STOP ROUTINE
3117 013544 005037 007356 CLR HPEND ;FOR VALID PROGRAM HALTS
3118 013550 012737 007512 001346 MOV #DRIVO,DRVPTR ;SETUP
3119 013556 005037 001220 CLR $DEVCT ;NO. OF DRVS DONE
3120 013562 005037 001222 CLR $UNIT ;CURRENT DR. UNDER TEST
3121 013566 012737 013634 000004 MOV #1$,ERRVEC ;SETUP TIMEOUT ERROR VECTOR
3122 013574 005777 165526 TST @LK$ ;SEE IF L-CLOCK THERE
3123 013600 005237 007532 INC LCLKF ;PRESENT, SET FLAG.
3124 013604 013700 001330 MOV LCVEC,RO ;VECTOR ADDR
3125 013610 .2737 013676 000004 MOV #2$,ERRVEC
3126 013616 005777 165476 TST @PK$ ;SEE IF P-CLOCK THERE
3127 013622 005237 007534 INC PCLKF ;PRESENT, SET FLAG
3128 013626 013700 001332 MOV PCVEC,RO ;VECTOR ADDR
3129 013632 000412 BR 3$
3130
3131 013634 022626 013702 000004 1$: CMP (SP)+,(SP)+ ;L-CLOCK NOT THERE, CLEAR STACK
3132 013636 012737 013702 000004 MOV #4$,ERRVEC
3133 013644 005777 165450 TST @PK$ ;SEE IF P-CLOCK THERE
3134 013650 005237 007534 INC PCLKF ;PRESENT, SET FLAG
3135 013654 013700 001332 MOV PCVEC,RO ;VECTOR ADDR
3136 013660 005237 007536 3$: INC DOTIM ;INDICATES TIMING TESTS CAN BE DONE
3137 013664 012720 043276 MOV #CLOCK,(RO)+ ;SERVICE ROUTINE FOR CLOCKS
3138 013670 012710 000300 MOV #PR6,(RO)
3139 013674 000407 BR TST1 ;;GO TO NEXT TEST

```

3140					
3141	013676	022626		2S:	CMP (SP)+,(SP)+ ;P-CLOCK NOT THERE, CLEAR STACK
3142	013700	000767			BR 3S
3143					
3144	013702	022626		4S:	CMP (SP)+,(SP)+ ;NEITHER CLOCK THERE, CLEAR STACK
3145	013704	005037	007536		CLR DOTIM ;TIMING TESTS CANNOT BE DONE.
3146	013710	104401	052212		TYPE ,MSG13 ;ALL TIMING TESTS BYPASSED
3147					
3148					

.SBTTL BASIC CONTROLLER TESTS, SIZING & SETUP

3149
3150
3151
3152
3153
3154
3155
3156
3157
3158
3159
3160
3161
3162
3163
3164
3165
3166
3167
3168
3169
3170
3171
3172
3173
3174
3175
3176
3177
3178
3179
3180
3181
3182
3183
3184
3185
3186
3187
3188
3189
3190
3191
3192
3193
3194
3195
3196
3197
3198
3199
3200
3201
3202
3203
3204

:TEST 1 REFERENCE ALL CONTROLLER REGISTERS

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

TST1: SCOPE
MOV #1,STIMES ;:DO 1 ITERATION
MOV #STACK,SP ;:RESTORE STK PTR

MOV #PRO,-(SP) ;:RESET PSW TO PRIORITY 0
MOV #SS,-(SP) ;:& MAKE IT LSI COMPATABLE
RTI

SS:

MOV #IS,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
BR TST2 ;:GO TO NEXT TEST

IS: CMP (SP)+,(SP)+ ;:RESTORE STACK POINTER
ERROR 7 ;:ABORT-COULD NOT REFERENCE CONTROLLER REGISTER
JMP \$E'JP1

: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. IF SET, THE PROGRAM WILL BYPASS

```

3205
3206
3207
3208
3209
3210 014070 000004
3211 014072 012737 000001 001174
3212 014100 012706 00110C
3213
3214 014104 005237 001532
3215
3216
3217 014110 132737 000200 001231
3218 014116 001302
3219 014120 000137 014234
3220
3221 014124 104401 052125 148:
3222 014130 005037 007510
3223 014134 005000
3224 014136 012701 007512
3225 014142 013702 001266
3226
3227 014146 032702 000001 158:
3228 014152 001410
3229 014154 005237 007510
3230 014160 005211
3231 014162 104401 001205
3232 014166 010046
3233
3234 014170 104403
3235 014172 001
3236 014173 000
3237
3238 014174 005721 168:
3239 014176 005200
3240 014200 022700 000010
3241 014204 001402
3242
3243 014206 006002
3244 014210 000756
3245
3246 014212 005737 007510 178:
3247 014216 001402
3248 014220 000137 015154
3249
3250 014224 104075 188:
3251 014226 000000
3252 014230 000137 013534
3253
3254 014234 012765 000040 000010 128:
3255 014242 013737 001412 007426
3256 014250 004737 036542
3257 014254 104120
3258 014256 005737 007540
3259 014262 001562
3260 014264 104401 052125

```

: * TESTING THAT DRIVE ONLY IF THE ERROR WAS A RESULT OF
: * MDS, LFE OR NED BEING SET; OR BOTH NED & DRA RESET IN-
: * DICATING THE OTHER PORT IS ACCESSED.
: *
: *****
↑ST2: SCOPE
MOV #1,STIMES ;:DO 1 ITERATION
MOV #STACK,SP ;:RESTORE STK PTR
INC BYPCERR ;:DO NOT TEST CERR IN 'FRDY'
BITB #BIT7,\$ENVM ;:SEE IF USE APT SELECTED DRIVES
BNE 148 ;:BR IF YES
JMP 128 ;:ELSE DO NORM SIZING OR VERIFY
148: TYPE ,MSG10 ;:WILL TEST DRIVES
CLR DRIVS ;:# OF DRIVES PRESENT
CLR RD ;:DRV ADDR
MOV #DRIV0,R1 ;:DRV FLAG
MOV \$DEV0,R2 ;:APT DEVICE MAP
158: BIT #BIT0,R2 ;:SEE IF DRV IN DEVICE MAP
BEQ 168 ;:BR IF NO
INC DRIVS ;:ELSE INCR DRIVE C JNT
INC (R1) ;:& SET DRIVE PRESENT FLAG
TYPE ,SCLF ;:SAVE RD FOR TYPEOUT
MOV RD,-(SP) ;:TYPE DRIVE #
;:GO TYPE--OCTAL ASCII
;:TYPE 1 DIGIT(S)
;:SUPPRESS LEADING ZEROS
168: TST (R1)+ ;:ADV POINTER TO NEXT FLAG
INC RD ;:INC DRIVE #
CMP #8,RD ;:ALL B TESTED?
BEQ 178 ;:BR IF YES
ROR R2 ;:ELSE GET NEXT BIT OFF DEVICE MAP
BR 158 ;:& TRY AGAIN
178: TST DRIVS ;:SEE IF MORE DRIVES PRESENT
BEQ 188 ;:BR IF NO
JMP NUDRV ;:ELSE EXIT TEST
188: ERROR 75 ;:NO DRIVES FOUND IN \$DEV0
HALT ;:SETUP CORRECTLY & PRESS 'CONTINUE'
JMP ST5 ;:TO TRY AGAIN
128: MOV #SCLR,RKCS2(R5) ;:SUBSYSTEM CLEAR
MOV T10,TEMP1 ;:SET TIMEOUT
JSR PC,FRDY ;:FIND RDY
ERROR 120 ;:RDY NOT SET BY END OF SCLR
TST SIZFLG
BEQ TST3 ;:DO NOT SIZE, GOTO NEXT TEST
TYPE ,MSG10 ;:WILL TEST DRIVES

3261	014270	005037	007510		CLR	DRIVS	:# OF DRIVES PRESENT
3262	014274	005000			CLR	RO	:DRV ADDR
3263	014276	012701	007512		MOV	#DRIV0,R1	:DRV FLAG
3264	014302			15:			
3265	014302	104415			SCOPI		
3266	014304	012706	001100		MOV	#STACK,SP	:RESTORE STK PTR
3267							
3268	014310	012765	00004C	000010	MOV	#SCLR,RKCS2(R5)	:SUBSYSTEM CLEAR
3269	014316	013737	001412	007426	MOV	T10,TEMP1	:SET TIMEOUT
3270	014324	004737	036542		JSR	PC,FRDY	:FIND RDY
3271	014330	104120			ERROR	120	:RDY NOT SET BY END OF SCLR
3272	014332	010065	000010		MOV	RO,RKCS2(R5)	:SELECT THE DRIVE ADDR
3273	014336	012765	000001	000000	MOV	#SELDRV,RKCS1(R5)	:SELECT DRIVE CMD
3274	014344	013737	001412	007426	MOV	T10,TEMP1	
3275	014352	004737	036542		JSR	PC,FRDY	:FIND RDY
3276	014356	104117			ERROR	117	:NO RDY AFTER SELECT DRIVE CMD.
3277	014360	032737	100000	007370	BIT	#CERR,HCS1	
3278	014366	001046			BNE	25	
3279	014370	013737	007416	007426	MOV	HMR2,TEMP1	
3280	014376	042737	177770	007426	BIC	#1<DRVMSK>,TEMP1	
3281	014404	020037	007426		CMP	RO,TEMP1	:S/B SAME
3282	014410	001016			BNE	35	
3283	014412	005700			TST	RO	
3284	014414	001003			BNE	45	
3285	014416	005737	007502		TST	DDPCH	:SEE IF XXDP CHAIN MODE
3286	014422	001014			BNE	55	
3287	014424	005237	007510		INC	DRIVS	:INC DRIVE COUNT.
3288	014430	005211		45:	INC	(R1)	:SET DRIVE PRESENT FLAG
3289	014432	104401	001205		TYPE	#SCLF	
3290	014436	010046			MOV	RO,-(SP)	:SAVE RO FOR TYPEOUT
3291							:TYPE DR #
3292	014440	104403			TYPOS		:GO TYPE--OCTAL ASCII
3293	014442	001			.BYTE	1	:TYPE 1 DIGIT(S)
3294	014443	000			.BYTE	0	:SUPPRESS LEADING ZEROS
3295	014444	000403			BR	55	
3296							
3297	014446	004737	037250		JSR	PC,BYP	:TYPE BYPASS DR #
3298	014452	104001		35:	ERROR	1	:WRITTEN DR # DOES NOT MATCH RKMR2 DR #
3299							
3300	014454	005721		55:	TST	(R1)+	:SHIFT PTR TO NEXT DR. FLAG
3301	014456	005200			INC	RO	:INC DR #
3302	014460	022700	000010		CMP	#8.,RO	
3303	014464	001306			BNE	15	:MORE LEFT.
3304	014466	005737	007510		TST	DRIVS	
3305	014472	001054			BNE	105	
3306	014474	104076			ERROR	76	:NO DRIVES FOUND ON BUSS
3307	014476	000000			HALT		:SETUP CORRECTLY
3308	014500	000137	013534		JMP	575	:AND PRESS 'CONTINUE'
3309							
3310	014504	032737	001000	007372	25:	BIT	#MDS,HCS2
3311	014512	001015			BNE	65	
3312	014514	032737	000400	007372	BIT	#UFE,HCS2	
3313	014522	001015			BNE	75	
3314	014524	032737	000001	007402	BIT	#DRA,HDS	
3315	014532	001015			BNE	85	
3316	014534	032737	010000	007372	BIT	#NED,HCS2	

```

3317 014542 001424          BEQ      95
3318 014544 000743          BR       55
3319
3320 014546 004737 037250    65:     JSR      PC,BYP          :TYPE BYP DR #
3321 014552 104002          ERROR    2          :MDS DETECTED
3322 014554 000737          BR       55
3323
3324 014556 004737 037250    75:     JSR      PC,BYP
3325 014562 104003          ERROR    3          :LFE DETECTED
3326 014564 000733          BR       55
3327
3328 014566 032737 010000 007372 85:     BIT      #NED,MCS2
3329 014574 001713          BEQ      45
3330 014576 104401 052323          TYPE    MSG15          :DRV#
3331 014602 010046          MOV     RC,-(SP)        :SAVE RD FOR TYPEOUT
3332
3333 014604 104403          TYPOS
3334 014606          .BYTE  1          :TYPE DR#
3335 014607          .BYTE  0          :GO TYPE--OCTAL ASCII
3336 014610 104010          ERROR   10          :TYPE 1 DIGIT(S)
3337 014612 000720          BR       55          :SUPPRESS LEADING ZEROS
3338
3339 014614 004737 037250    95:     JSR      PC,BYP
3340 014620 104004          ERROR    4          :NO DRA & NO NED = OTHER PORT SELECTED
3341 014622 000714          BR       55
3342 014624 000127 015154   105:    JMP      NUDRV

```

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. IT WILL THEN CHECK TO SEE THAT THE DRIVE WAS INPUTTED FOR TESTING. IF NOT IT WILL BE AN ERROR. IF CERR WAS SET THAT DRIVE WILL BE BYPASSED ONLY IF THE ERROR WAS A RESULT OF MDS OR LFE 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.

```

3359 014630 000004          SCOPE
3360 014632 012737 000001 001174  ST3:   MOV     #1,STIMES          :DO 1 ITERATION
3361 014640 012706 001100          MOV     #STACK,SP        :RESTORE STK PTR
3362 014644 005000          CLR     RC                :DRIVE ADDR
3363 014646 012701 007512          MOV     #DRIVE,R1        :DRIVE FLAG
3364 014652
3365 014652 104415          SCOPI
3366 014654 012706 001100          MOV     #STACK,SP        :RESTORE STK PTR
3367
3368 014660 012765 000040 000010  MOV     #SCLR,RKCS2,RS.   :SUBSYSTEM CLEAR
3369 014666 013737 001412 007426  MOV     T10,TEMP1        :SET TIME OUT
3370 014674 004737 036542          JSR     PC,FRDY          :FIND RDY
3371 014700 104120          ERROR   120           :NO RDY AFTER SCLR
3372 014702 010065 000010          MOV     RC,RKCS2,RS.     :DRY ADDR

```


F06

UNIBUS RKO6 DRIVE DIAGNOSTIC PART 2
DZR6IC.P11 07-007-76 13:50

MACY11 27,1006) 07-OCT-76 14:14 PAGE 70
T3 VERIFY OPERATOR DRIVE SELECTIONS

SEG 0070

3429 015142 004737 037250
3430 015146 104004
3431 015150 000752
3432
3433
3434
3435
3436
3437
3438
3439
3440
3441
3442
3443
3444
3445
3446
3447
3448
3449
3450
3451
3452
3453
3454
3455
3456
3457
3458
3459
3460
3461
3462
3463
3464
3465
3466
3467
3468
3469
3470
3471
3472
3473
3474
3475
3476
3477
3478
3479
3480
3481
3482
3483
3484

95: JSR PC,BYP ;
ERROR 4 ;DRA & NED RESET - OTHER PORT SELECTED
BR 85 ;
BNE 15 ;BRANCH IF MORE LEFT.

;; THIS PART OF THE PROGRAM WILL BE REPEATED FOR EACH
;; DRIVE PRESENT

;; 'SUNIT' CONTAINS THE ADDRESS OF THE DRIVE CURRENTLY
;; UNDER TEST

015154 005037 001532

NUDRV: CLR BYPCERR ;ENTER HERE FROM LAST TEST
;ALLOW CHECKING CERR IN 'FRDY'

;TEST 4 FIND NEXT DRIVE TO BE TESTED

;; THIS TEST FINDS THE NEXT DRIVE PRESENT & PUTS THAT
;; ADDRESS IN 'SUNIT'.
;; THROUGHOUT THE FOLLOWING TESTS, THE DRIVE TESTED IS
;; THE DRIVE WHOSE ADDRESS IS IN 'SUNIT'.

015160 000004
015162 012737 000001 001174
015170 012706 001100
015174 012737 000004 001214
015202 012737 000004 001102

ST4: SCOPE ;
MOV #1,STIMES ;DO 1 ITERATION
MOV #STACK,SP ;RESTORE STK PTR
MOV #STN-1,STSTN
MOV #STN-1,STSTNM

015210 005737 007510
015214 001004
015216 104401 053237
015222 000137 036062

TST DRIVS ;ANY DRIVES PRESENT?
BNE 45 ;YES BRANCH
TYPE ,MSG27 ;ALL DRIVES TESTED
JMP \$EOP1 ;NO, GO TO END

015226 013701 00.346
015232 005737 001220
015236 001402
015240 005237 001222
015244 005721
015246 001774
015250 005737 007502
015254 001403
015256 005737 001222
015262 001766
015264 010137 001346
015270 104401 052323
015274 013700 001222
015300 010046

45: MOV DRVPTR,R1 ;ADDR OF NEXT DRIVE FLAG
TST \$DEVCT ;IS FIRST DRIVE BEING CHECKED
BEQ 25 ;YES, BRANCH
15: INC SUNIT ;INCR DRIVE ADDR TO NEXT DRIVE
25: TST (R1)+ ;IS DRIVE PRESENT?
BEQ 15 ;NO, FIND NEXT DRIVE PRESENT
TST DDPCH ;DDP CHAIN MODE?
BEQ 35 ;NO, BRANCH
TST SUNIT ;YES, IS IT DRIVE 0?
BEQ 15 ;IF YES, DON'T TEST DR 0
35: MOV R1,DRVPTR ;STORE POINTER TO THE NEXT DR. FLAG
TYPE ,MSG15 ;"DRIVE"
MOV SUNIT,R0
MOV R0,-(SP) ;SAVE R0 FOR TYPEOUT

015302 104403
015304 001
015305 000

TYPOS ;
;DRIVE #
;GO TYPE--OCTAL ASCII
;TYPE 1 DIGIT(S)
;SUPPRESS LEADING ZEROS

G06

UNIBUS Rk06 DRIVE DIAGNOSTIC PART 2
DZR6IC.P11 07-OCT-76 13:50

MACY11 27(1006) 07-OCT-76 14:14 PAGE 71
T4 FIND NEXT DRIVE TO BE TESTED

SEQ 0071

3485	015306	104401	001205	
3486				
3487	015312			
3488				
3489				
3490				
3491				
3492				
3493				
3494				
3495	015312	000004		
3496	015314	012737	000001	001174
3497	015322	012706	001100	
3498				
3499	015326	005737	001216	
3500	015332	001046		
3501	015334	004737	040452	
3502	015340	104024		
3503				
3504	015342	104401	052335	
3505	015346	012765	000003	000026
3506	015354	004737	040100	
3507	015360	013701	007416	
3508	015364	012704	050024	
3509	015370	010446		
3510	015372	012703	000003	
3511	015376	006101		
3512	015400	006101		
3513	015402	006101		
3514	015404	006101		
3515	015406	006101		
3516	015410	006101		
3517	015412	010100		
3518	015414	042700	177760	
3519	015420	052700	000060	
3520	015424	110024		
3521	015426	005303		
3522	015430	001364		
3523	015432	105014		
3524				
3525	015434	004737	050272	
3526	015440	104401	001205	
3527	015444	104401	001205	
3528				
3529				
3530				
3531				
3532				
3533				
3534				
3535	015450	000004		
3536	015452	012737	000001	001174
3537	015460	012706	001100	
3538				
3539	015464	004737	040452	
3540	015470	104024		

```

TYPE ,SCLF
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
;*****
TST5: SCOPE
MOV #1,$TIMES ;:DO 1 ITERATION
MOV #STACK,SP ;:RESTORE STK PTR
TST $PASS
BNE TST6 ;:GO TO NEXT IF NOT FIRST PASS
JSR PC,SJBCLR ;:DO SUBSYS CLEAR
ERROR 24 ;:CERR AFTER SCLR
TYPE ,MSG16 ;:DRIVE SERIAL NO.
MOV #3,RKMR1(R5) ;:SELECT BYTE 3
JSR PC,GSTAT ;:GET STATUS
MOV HMR2,R1 ;:GET SERIAL #
MOV #SOCTVL,R4 ;:GET ADDR CHAR BUFF
MOV R4,-(SP) ;:STORE ON STACK FOR $SUPRS
MOV #3,R3 ;:SETUP CHAR COOUNT
ROL R1 ;:INITIALIZE BIT POSITIONS
ROL R1
15: ROL R1 ;:GET NEXT 4 BITS
ROL R1
ROL R1
MOV R1,R0 ;:GET WORKING COPY
BIC #177760,R0 ;:CLEAR ALL BUT LOW 4 BITS
BIS #60,R0 ;:CONVERT TO ASCII DIGIT
MOVB R0,(R4)+ ;:PUT ASCII DIGIT INTO CHAR BUFF
DEC R3
BNE 15 ;:BR IF ALL 3 CHARS NOT DONE
CLRB (R4) ;:ELSE INSERT NULL TERMINATOR
JSR PC,$SUPRS ;:TYPE
TYPE ,SCLF
TYPE ,SCLF
;*****
;TEST 6 SET VV WITH PACK COMMAND
;
; IF VV IS RESET, THE PACK COMMAND IS USED TO SET IT.
;*****
TST6: SCOPE
MOV #1,$TIMES ;:DO 1 ITERATION
MOV #STACK,SP ;:RESTORE STK PTR
JSR PC,SUBCLR
ERROR 24 ;:CERR AFTER SCLR

```

9

```

3541
3542 015472 032737 000100 007416 BIT #D.VV,HMR2
3543 015500 001024 BNE TST7 ;;GO TO NEXT TEST IF VV SET
3544
3545 015502 104415 SCOPI
3546 015504 012706 001100 MOV #STACK,SP ;RESTORE STK PTR
3547
3548 015510 004737 040452 JSR PC,SUBCLR
3549 015514 104024 ERROR 24 ;CERR AFTER SCLR
3550
3551 015516 012765 000003 000000 MOV #PACK,RKCS1(R5) ;CMD TO SET VV
3552 015524 012737 000010 007426 MOV #10,TEMP1
3553 015532 004737 036542 JSR PC,FRDY ;FIND RDY
3554 015536 104116 ERROR 116 ;RDY NOT SET AFTER PACK CMD
3555
3556 015540 032737 000100 007416 BIT #D.VV,HMR2
3557 015546 001001 BNE TST7 ;;GO TO NEXT TEST IF VV NOW SET
3558 015550 104027 ERROR 27 ;PACK DID NOT SET V.V.
3559
3560
3561
3562
3563
3564
3565
3566
3567
3568
3569
3570
3571
3572
3573
3574
3575
3576
3577
3578
3579
3580
3581
3582
3583
3584

```

```

*****
*TEST 7 READ & SAVE BAD SECTOR INFO & TYPE PACK SERIAL *

```

```

* THIS TEST VERIFIES THAT CYL 410, TRACK 2 CAN BE READ.
* THIS AREA CONTAINS BAD SECTOR INFO WHICH IS WRITTEN BY THE
* FACTORY DURING MANF. ALL BAD SECTOR INFO (BSE) WILL BE STORED
* AT THIS TIME TO MASK FUTURE READ HEADER OR DATA ERROR PRINTOUTS.

```

```

* SECTORS 0,2,4,6,8 CONTAIN IDENTICAL INFO FOR 22 SECTOR HARDWARE DETECTED BAD SEC
* SECTORS 10,12,14,16,18,20 CONTAIN IDENTICAL INFO FOR 22 SECTOR SOFTWARE DETECTED

```

```

* SECTORS 1,3,5,7,9 CONTAIN IDENTICAL INFO FOR 20 SECTOR HARDWARE DETECTED BAD SEC
* SECTORS 11,13,15,17,19,21 CONTAIN IDENTICAL INFO FOR 20 SECTOR SOFTWARE DETECTED

```

```

* 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 INFO OR AN ALIGNMENT PACK BY WRITING

```

```

* THE PACK SERIAL # IS TYPED IN OCTAL & FOR THE FIRST PASS ONLY.

```

```

*****
*ST7:

```

```

3585 015552 000004 SCOPE
3586 015554 012737 000001 001174 MOV #1,$TIMES ;;DO 1 ITERATION
3587 015562 012706 001100 MOV #STACK,SP ;RESTORE STK PTR
3588
3589 015566 004737 040452 JSR PC,SUBCLR
3590 015572 104024 ERROR 24 ;CERR AFTER SCLR
3591
3592
3593 015574 012765 100000 000000 MOV #CCLR,RKCS1(R5)
3594 015602 013765 001222 000010 MOV $UNIT,RKCS2(R5)
3595 015610 012765 000013 000000 MOV #RECAL,RKCS1(R5) ;RECAL CMD
3596 ;RESET CYL DIFF/OFFSET & CYL ADDR REG

```


JOB

UNIBUS RK06 DRIVE DIAGNOSTIC PART 2
DZR6IC.P11 07-OCT-76 13:50

MACY11 27(1006) 07-OCT-76 14:14 PAGE 74
T7 READ & SAVE BAD SECTOR INFO & TYPE PACK SERIAL #

SEQ 0074

3653	016114	012737	000002	007472		MOV	#2,E.B2	;MSG ID FOR EXPECTED MSG B2
3654	016122	012737	000003	007476		MOV	#3,E.B3	;MSG ID FOR EXPECTED MSG B3
3655								
3656	016130	004737	037264			JSR	PC,CHKMSG	;CHECK MSGS A0, B0, A1, B1
3657	016134	000000				.WORD	0!0!0	; & MSGS SPECIFIED HERE
3658	016136	104054				ERROR	54	;MSG A0 ERROR AFTER READ DATA CMD
3659	016140	104026				ERROR	26	;MSH B0 ERROR
3660	016142	104056				ERROR	56	;MSG A1 ERROR
3661	016144	104030				ERROR	30	;MSG B1 ERROR
3662								
3663	016146	004737	040452			JSR	PC,SUBCLR	
3664	016152	104024				ERROR	24	;CERR AFTER SUBCLR
3665								
3666	016154	005237	007430			INC	TEMP2	
3667	016160	023727	007430	000005		CMP	TEMP2,#5	;READ ALL 5 SECTORS?
3668	016166	001023				BNE	5\$	
3669	016170	005737	007432			TST	TEMP3	
3670	016174	001002				BNE	2\$	
3671	016176	104233				ERROR	233	;CANT READ SECTORS 0,2,4,6,8
3672	016200	000430				BR	3\$	
3673								
3674	016202	023727	007432	000001	2\$:	CMP	TEMP3,#1	
3675	016210	001002				BNE	4\$	
3676	016212	104230				ERROR	230	;CANT READ SECTORS 10,12...
3677	016214	000422				BR	3\$	
3678								
3679	016216	023727	007432	000002	4\$:	CMP	TEMP3,#2	
3680	016224	001002				BNE	6\$	
3681	016226	104234				ERROR	234	;CANT READ SECTORS 1,3,5 ...
3682	016230	000414				BR	3\$	
3683								
3684	016232	104231			6\$:	ERROR	231	;CANT READ SECTORS 11,13,15 ...
3685	016234	000412				BR	3\$	
3686								
3687	016236	013765	007434	000004	5\$:	MOV	TEMP4,RKBA(R5)	;RESTORE TABLE ADDR
3688	016244	062737	000002	007436		ADD	#2,TEMP5	;READ 2 SECTORS FROM LAST
3689	016252	013765	007436	000006		MOV	TEMP5,RKDA(R5)	
3690	016260	000652				BR	1\$	
3691								
3692	016262	005237	001526		3\$:	INC	BSERR	;SET BSE FLAG
3693	016266	000553				BR	TST10	::GO TO NEXT TEST
3694	016270	005737	003360		8\$:	TST	BSE22H+6	;TEST CARTRIDGE TYPE
3695	016274	001404				BEQ	9\$;BRANCH IF DATA CARTRIDGE
3696	016276	104235				ERROR	235	;ALIGNMENT CARTRIDGE USED
3697	016300	005237	001526			INC	BSERR	;SET BSE ERROR FLAG
3698	016304	000476				BR	10\$	
3699								
3700	016306	005237	007432		9\$:	INC	TEMP3	
3701	016312	023727	007432	000001		CMP	TEMP3,#1	
3702	016320	001020				BNE	11\$	
3703	016322	005037	007430			CLR	TEMP2	;SECTOR CTR
3704	016326	012737	005352	007434		MOV	#BSE22S,TEMP4	;STORE 22 SECTOR SOFTWARE BSE ADDR
3705	016334	013765	007434	000004		MOV	TEMP4,RKBA(R5)	
3706	016342	012737	001012	007436		MOV	#1012,TEMP5	;TRACK 2, SECTOR 12(8)
3707	016350	013765	007436	000006		MOV	TEMP5,RKDA(R5)	
3708	016356	000137	016006			JMP	1\$;REPEAT

```

3709
3710 016362 023727 007432 000002 11$: CMP TEMP3,#2
3711 016370 001020 BNE 12$
3712 016372 005037 007430 CLR TEMP2 ;SECTOR CTR
3713 016376 012737 002352 007434 MOV #BSE20H,TEMP4 ;STORE 20 SECTOR HARDWARE BSE ADDR.
3714 016404 013765 007434 000004 MOV TEMP4,RKBA(R5)
3715 016412 012737 001001 007436 MOV #1001,TEMP5 ;TRACK 2, SECTOR 1
3716 016420 013765 007436 000006 MOV TEMP5,RKDA(R5)
3717 016426 000137 016006 JMP 1$ ;REPEAT
3718
3719 016432 023727 007432 000003 12$: CMP TEMP3,#3
3720 016440 001020 BNE 10$
3721 016442 005037 007430 CLR TEMP2 ;SECTOR CTR
3722 016446 012737 004352 007434 MOV #BSE20S,TEMP4 ;STORE 20 SECTOR SOFTWARE BSE ADDR
3723 016454 013765 007434 000004 MOV TEMP4,RKBA(R5)
3724 016462 012737 001013 007436 MOV #1013,TEMP5 ;TRACK 2, SECTOR 13(8)
3725 016470 013765 007436 000006 MOV TEMP5,RKDA(R5)
3726 016476 000137 016006 JMP 1$ ;REPEAT
3727
3728 016502 005737 001216 10$: TST $PASS
3729 016506 001043 BNE TST10 ;GO TO NEXT TST IF NOT 1'ST PASS
3730 016510 104401 052361 TYPE MSG17 ;CART SERIAL *
3731 016514 012746 003352 MOV #BSE22H,-(SP)
3732 016520 004737 047722 JSR PC,$DB20 ;CONVERT DBL BINARY WORD TO OCTAL
3733 016524 004737 050272 JSR PC,$SUPRS ;TYPE SERIAL *
3734 016530 104401 001205 TYPE $CRLF
3735 016534 104401 001205 TYPE $CRLF
3736
3737 016540 004737 040452 JSR PC,SUBCLR
3738 016544 104024 ERROR 24 ;CERR AFTER SCLR
3739 ;GO BACK TO CYL 0
3740
3741 016546 012765 000017 000000 MOV #SEEK,RKCS1(R5) ;SEEK CMD
3742 016554 013737 001414 007426 MOV T50,TEMP1 ;SETUP TIMEOUT
3743 016562 004737 036542 JSR PC,FRDY ;FIND RDY
3744 016566 104131 ERROR 131 ;NO RDY AFTER SEEK CMD
3745
3746 016570 013737 001424 007426 MOV T50000,TEMP1 ;SETUP TIMEOUT
3747 016576 004737 037152 JSR PC,FATT2 ;FIND ATTN
3748 016602 104132 ERROR 132 ;NO ATTN AFTER SEEK CMD
3749
3750 016604 032737 100000 007370 BIT #CERR,HCS1
3751 016612 001401 BEQ 66$
3752 016614 104210 ERROR 210 ;CERR AFTER SEEK CMD
3753
3754 016616 66$:

```

```

3755
3756
3757
3758 .SBTTL WRITE TESTS
3759
3760
3761 ;*****
3762 ;*TEST 10 BASIC WRITE DATA TEST; 1 WORD
3763 ;*
3764 ;* THIS TEST VERIFIES THE ABILITY OF THE DRIVE TO WRITE JUST ONE WORD,

```

L06

UNIBUS RK06 DRIVE DIAGNOSTIC PART 2
DZR6IC.P11 07-OCT-76 13:50

MACY11 27(1006) 07-OCT-76 14:14 PAGE 76
T10 BASIC WRITE DATA TEST; 1 WORD

SEG 0076

```

3765 ;* ALL SECTORS ON CYL 0 ARE GIVEN IDENTICAL HEADERS &
3766 ;* A WRITE COMMAND IS ISSUED. READ & WRITE CHECK COMMANDS ARE NOT
3767 ;* PERFORMED. THIS TEST PROVIDES THE TIGHTEST POSSIBLE SCOPE LOOP
3768 ;* FOR A WRITE ERROR.
3769 ;*
3770 ;*****
3771 016616 000004          †ST10: SCOPE
3772 016620 012737 000001 001174  MOV      #1,STIMES      ;:DO 1 ITERATION
3773 016626 012706 001100      MOV      #STACK,SP      ;:RESTORE STK PTR
3774
3775 016632 005737 001340      TST      BYPWRT
3776 016636 001404          BEQ      DOWRT
3777 016640 104401 052504      TYPE     MSG19          ;:BYPASSING WRITE TESTS
3778 016644 000137 031706      JMP      TIMING
3779 016650
3780
3781 016650 005737 001526      TST      BSERR          ;:SEE IF ALIGN CART
3782 016654 001406          BEQ      2$             ;:BR IF NO
3783 016656 104401 054343      TYPE     ,MSG40         ;:BSE OR ALIGN CART USED
3784 016662 104401 053155      TYPE     ,MSG26         ;:ABORTING DATA TESTS
3785 016666 000137 031706      JMP      TIMING
3786
3787 016672 004737 040452      2$:     JSR      PC,SUBCLR ;:CERR AFTER SCLR
3788 016676 104024          ERROR    24
3789
3790 016700 005237 007354      INC      BADHDR        ;:USED FOR VALID HALT
3791
3792 016704 012700 001536      MOV      #HDTAB,RO     ;:MAKE ALL CYL 0 HEADERS IDENTICAL
3793
3794 016710 005020          1$:     CLR      (RO)+         ;:HEADER WORD 0: CYL 0
3795 016712 012720 140000      MOV      #140000,(RO)+ ;:HEADER WORD 1: SECTOR 0
3796 016716 012720 140000      MOV      #140000,(RO)+ ;:HEADER WORD 2: XOR OF 1 & 2
3797
3798 016722 020027 001742      CMP      RO,#HDTAB+132. ;:ALL HEADERS DONE? (22X6=132)
3799 016726 001370          BNE     1$             ;:BR IF NO
3800
3801 016730 012765 001536 000004  MOV      #HDTAB,RKBA(R5) ;:HEADER TABLE
3802 016736 012765 177676 000002  MOV      #-66.,RKWC(R5) ;:WORD COUNT
3803
3804 016744 012765 000027 000000  MOV      #<WRHEAD>,RKCS1(R5) ;:WRITE HEADER CMD
3805 016752 013737 001424 007426  MOV      T5000,TEMP1     ;:SETUP TIMEOUT
3806 016760 004737 036542          JSR      PC,FRDY        ;:FIND RDY
3807 016764 104200          ERROR    200           ;:NO RDY AFTER WRITE HEADER CMD
3808 016766 004737 040100          JSR      PC,GSTAT       ;:GET FRESH STATUS
3809 016772 032737 100000 007370  BIT      #CERR,HCS1
3810 017000 001405          BEQ      64$
3811 017002 104201          ERROR    201           ;:CERR AFTER WRITE HEADER CMD
3812 017004 104401 053155      TYPE     ,MSG26         ;:ABORTING DATA TESTS TO DO TIMING TESTS
3813 017010 000137 031706      JMP      TIMING
3814 017014          64$:
3815
3816
3817 017014 104415          SCOP1
3818 017016 012706 001100      MOV      #STACK,SP      ;:RESTORE STK PTR
3819
3820 017022 004737 040452      JSR      PC,SUBCLR

```

M06

UNIBUS RK06 DRIVE DIAGNOSTIC PART 2
DZR6IC.P11 07-OCT-76 13:50

MACY11 27(1006) 07-OCT-76 14:14 PAGE 77
T10 BASIC WRITE DATA TEST; 1 WORD

SEG 0077

3821	017026	104024				ERROR	24		;CERR AFTER SCLR
3822									
3823	017030	005037	001406			CLR	SECTOR		
3824	017034	013765	001406	000006	35:	MOV	SECTOR,RKDA(R5)	; TRACK/SECTOR #	
3825	017042	012765	001514	000004		MOV	#DATA1,RKBA(R5)	; DATA TO BE ALL 1'S	
3826	017050	012765	177777	000002		MOV	#-1,RKWC(R5)	; WORD COUNT=1	
3827									
3828									
3829	017056	012765	000023	000000		MOV	#(WRDATA),RKCS1(R5)	; WRITE DATA CMD	
3830	017064	013737	001424	007426		MOV	T50000,TEMP1	; SETUP TIMEOUT	
3831	017072	004737	036542			JSR	PC,FRDY	; FIND RDY	
3832	017076	104011				ERROR	11	; NO RDY AFTER WRITE DATA CMD	
3833	017100	004737	040100			JSR	PC,GSTAT	; GET FRESH STATUS	
3834	017104	032737	100000	007370		BIT	#CERR,HCS1		
3835	017112	001465				BEQ	68\$; BR IF NO ERRORS	
3836									
3837	017114	032737	000200	007404		BIT	#BSE,HER	; SEE IF BAD SECTOR FLAG	
3838	017122	001421				BEQ	66\$; BR IF NO	
3839	017124	004737	042122			JSR	PC,TRUERR	; ELSE SEE IF SECTOR LISTED IN BSE TABLE	
3840	017130	000455				BR	67\$; RETURN HERE IF NO	
3841									
3842	017132	005237	001406			INC	SECTOR	; RETURN HERE IF YES	
3843	017136	023727	001406	000012		CMP	SECTOR,#10.	; ARE 10 CONSEC. SECTORS BAD	
3844	017144	001003				BNE	65\$; BR IF NO	
3845	017146	104046				ERROR	46	; ABORTING TEST DETECTED 10 BAD SECTORS	
3846	017150	000137	017352			JMP	5\$; BYPASS TEST	
3847	017154	012765	100000	000000	65\$:	MOV	#CCLR,RKCS1(R5)	; TRY ANOTHER SECTOR	
3848	017162	000137	017034			JMP	3\$		
3849	017166	104012			66\$:	ERROR	12	; CERR WITH WRITE DATA CMD	
3850									
3851	017170	012737	010340	007460		MOV	#(0!D.SPIN!D.DRDY!D.VV!D.DRA),E.A0	; EXPECTED MSG A0	
3852	017176	005037	007462			CLR	E.B0	; EXPECTED MSG B0	
3853	017202	012737	001720	007464		MOV	#(D.SPOK!D.CART!D.DOOR!D.BRHM!D.SSP),E.A1	; EXPECTED A1	
3854	017210	012737	000001	007466		MOV	#1,E.B1	; MSG ID FOR EXPECTED MSG B1	
3855	017216	005037	007470			CLR	E.A2	; EXPECTED MSG A2	
3856	017222	012737	000002	007472		MOV	#2,E.B2	; MSG ID FOR EXPECTED MSG B2	
3857	017230	012737	000003	007476		MOV	#3,E.B3	; MSG ID FOR EXPECTED MSG B3	
3858									
3859	017236	004737	037264			JSR	PC,CHKMSG	; CHECK MSGS A0, B0, A1, B1	
3860	017242	000003				.WORD	T.A2!T.B2!0	; & MSGS SPECIFIED HERE	
3861	017244	104052				ERROR	52	; MSG A0 ERROR AFTER WRITE DATA CMD	
3862	017246	104023				ERROR	23	; MSH B0 ERROR	
3863	017250	104053				ERROR	53	; MSG A1 ERROR	
3864	017252	104025				ERROR	25	; MSG B1 ERROR	
3865	017254	104401	053155			TYPE	MSG26	; ABORTING DATA TESTS TO DO TIMING	
3866	017260	000137	031706			JMP	TIMING		
3867	017264	104063			67\$:	ERROR	63	; BAD SECTOR NOT LISTED IN TABLE	
3868	017266				68\$:				
3869									
3870	017266	012737	010340	007460		MOV	#(0!D.SPIN!D.DRDY!D.VV!D.DRA),E.A0	; EXPECTED MSG A0	
3871	017274	005037	007462			CLR	E.B0	; EXPECTED MSG B0	
3872	017300	012737	001720	007464		MOV	#(D.SPOK!D.CART!D.DOOR!D.BRHM!D.SSP),E.A1	; EXPECTED A1	
3873	017306	012737	000001	007466		MOV	#1,E.B1	; MSG ID FOR EXPECTED MSG B1	
3874	017314	005037	007470			CLR	E.A2	; EXPECTED MSG A2	
3875	017320	012737	000002	007472		MOV	#2,E.B2	; MSG ID FOR EXPECTED MSG B2	
3876	017326	012737	000003	007476		MOV	#3,E.B3	; MSG ID FOR EXPECTED MSG B3	

```

3877
3878 017334 004737 037264
3879 017340 000003
3880 017342 104052
3881 017344 104023
3882 017346 104053
3883 017350 104025
3884 017352
3885
3886
3887
3888
3889
3890
3891
3892
3893
3894
3895 017352 000004
3896 017354 012737 000001 001174
3897 017362 012706 001100
3898
3899 017366 004737 040452
3900 017372 104024
3901
3902 017374 012765 001536 000004
3903 017402 012765 177676 000002
3904 017410 005037 001352
3905
3906 017414 013737 001352 001366
3907 017422 012737 000000 001474
3908 017430 012737 000000 001502
3909 017436 004737 041434
3910
3911
3912 017442 012765 000027 000000
3913 017450 013737 001424 007426
3914 017456 004737 036542
3915 017462 104200
3916 017464 004737 040100
3917 017470 032737 100000 007370
3918 017476 001405
3919 017500 104201
3920 017502 104401 053155
3921 017506 000137 031706
3922 017512
3923
3924 017512 005037 007354
3925 017516 104415
3926 017520 012706 001100
3927
3928 017524 004737 040452
3929 017530 104024
3930
3931 017532 005037 001406
3932 017536 013765 001406 000006

```

```

JSR PC,CHKMSG ;CHECK MSGS A0, B0, A1, B1
.WORD T.A2!T.B2!0 ;& MSGS SPECIFIED HERE
ERROR 52 ;MSG A0 ERROR AFTER WRITE DATA CMD
ERROR 23 ;MSH B0 ERROR
ERROR 53 ;MSG A1 ERROR
ERROR 25 ;MSG B1 ERROR
5$:
*****
*TEST 11 BASIC WRITE DATA TEST; FULL SECTOR
*
* THIS TEST VERIFIES THE ABILITY OF THE DRIVE TO WRITE
* A FULL SECTOR. ALL ZEROS ARE WRITTEN BY THE WRITE DATA COMMAND
* & CHECKED BY A RD DATA COMMAND. A FURTHER CHECK IS PERFORMED
* BY THE WRT CHK COMMAND.
* THE ABOVE IS REPEATED FOR AN ALL ONES PATTERN.
*****
TST11: SCOPE
MOV #1,STIMES ;DO 1 ITERATION
MOV #STACK,SP ;RESTORE STK PTR
JSR PC,SUBCLR
ERROR 24 ;CERR AFTER SCLR
MOV #HDTAB,RKBA(R5) ;RESTORE TO 22 SECTOR
MOV #-66.,RKWC(R5) ;STANDARD FORMAT
CLR TOCYL
MOV TOCYL,CALADD ;SETUP
MOV #0,HEAD ;TO FILL
MOV #0,FORMAT ;HEADER
JSR PC,FHDTAB ;TABLE
MOV #<WRHEAD>,RKCS1(R5) ;WRITE HEADER CMD
MOV T5000,TEMP1 ;SETUP TIMEOUT
JSR PC,FRDY ;FIND RDY
ERROR 200 ;NO RDY AFTER WRITE HEADER CMD
JSR PC,GSTAT ;GET FRESH STATUS
BIT #CERR,HCS1
BEQ 64$
ERROR 201 ;CERR AFTER WRITE HEADER CMD
TYPE MSG26 ;ABORTING DATA TESTS TO DO TIMING TESTS
JMP TIMING
64$:
CLR BADHDR ;USED FOR VALID HALT
SCOPI
MOV #STACK,SP ;RESTORE STK PTR
JSR PC,SUBCLR
ERROR 24 ;CERR AFTER SCLR
CLR SECTOR
MOV SECTOR,RKDA(R5) ;SETUP SECTOR
8$:

```

3933	017544	012765	001510	000004	MOV	#DATA0,RKBA(R5)	:WRITE ALL 0'S
3934	017552	013700	001510		MOV	DATA0,R0	
3935	017556	052765	000020	000010	BIS	#BA1,RKCS2(R5)	15:
3936	017564	012765	177400	000002	MOV	#-256.,RKWC(R5)	
3938	017572	012765	000023	000000	MOV	#(WRDATA),RKCS1(R5)	:WRITE DATA CMD
3939	017600	013737	001424	007426	MOV	T5000,TEMP1	:SETUP TIMEOUT
3940	017606	004737	036542		JSR	PC,FRDY	:FIND RDY
3941	017612	104011			ERROR	11	:NO RDY AFTER WRITE DATA CMD
3942	017614	004737	040100		JSR	PC,STAT	:GET FRESH STATUS
3943	017620	032737	100000	007370	BIT	#CERR,HCS1	
3944	017626	001465			BEQ	68\$:BR IF NO ERRORS
3945	017630	032737	000200	007404	BIT	#BSE,HER	:SEE IF BAD SECTOR FLAG
3946	017636	001421			BEQ	66\$:BR IF NO
3947	017640	004737	042122		JSR	PC,TRUERR	:ELSE SEE IF SECTOR LISTED IN BSE TABLE
3948	017644	000455			BR	67\$:RETURN HERE IF NO
3950	017646	005237	001406		INC	SECTOR	:RETURN HERE IF YES
3951	017652	023727	001406	000012	CMP	SECTOR,#10.	:ARE 10 CONSEC. SECTORS BAD
3952	017660	001003			BNE	65\$:BR IF NO
3953	017662	104046			ERROR	46	:ABORTING TEST DETECTED 10 BAD SECTORS
3954	017664	000137	021152		JMP	7\$:BYPASS TEST
3955	017670	012765	100000	000000	MOV	#CCLR,RKCS1(R5)	:TRY ANOTHER SECTOR
3956	017676	000137	017536		JMP	8\$	
3957	017702	104012			ERROR	12	:CERR WITH WRITE DATA CMD
3958	017702	104012			ERROR	12	:CERR WITH WRITE DATA CMD
3959	017704	012737	010340	007460	MOV	#(O!D.SPIN!D.DRDY!D.VV!D.DRA),E.A0	:EXPECTED MSG A0
3960	017712	005037	007462		CLR	E.B0	:EXPECTED MSG B0
3961	017716	012737	001720	007464	MOV	#(D.SPOK!D.CART!D.DOOR!D.BRHM!D.SSP),E.A1	:EXPECTED A1
3962	017724	012737	000001	007466	MOV	#1,E.B1	:MSG ID FOR EXPECTED MSG B1
3963	017732	005037	007470		CLR	E.A2	:EXPECTED MSG A2
3964	017736	012737	000002	007472	MOV	#2,E.B2	:MSG ID FOR EXPECTED MSG B2
3965	017744	012737	000003	007476	MOV	#3,E.B3	:MSG ID FOR EXPECTED MSG B3
3966	017744	012737	000003	007476	MOV	#3,E.B3	:MSG ID FOR EXPECTED MSG B3
3967	017752	004737	037264		JSR	PC,CHKMSG	:CHECK MSGS A0, B0, A1, B1
3968	017752	004737	037264		JSR	PC,CHKMSG	:CHECK MSGS A0, B0, A1, B1
3969	017756	000003			.WORD	T.A2!T.B2!0	:& MSGS SPECIFIED HERE
3970	017760	104052			ERROR	52	:MSG A0 ERROR AFTER WRITE DATA CMD
3971	017762	104023			ERROR	23	:MSG B0 ERROR
3972	017764	104053			ERROR	53	:MSG A1 ERROR
3973	017766	104025			ERROR	25	:MSG B1 ERROR
3974	017770	104401	053155		TYPE	MSG26	:ABORTING DATA TESTS TO DO TIMING
3975	017774	000137	031706		JMP	TIMING	
3976	020000	104063			ERROR	63	:BAD SECTOR NOT LISTED IN TABLE
3977	020002	104063			ERROR	63	:BAD SECTOR NOT LISTED IN TABLE
3978	020002	104063			ERROR	63	:BAD SECTOR NOT LISTED IN TABLE
3979	020002	012737	010340	007460	MOV	#(O!D.SPIN!D.DRDY!D.VV!D.DRA),E.A0	:EXPECTED MSG A0
3980	020010	005037	007462		CLR	E.B0	:EXPECTED MSG B0
3981	020014	012737	001720	007464	MOV	#(D.SPOK!D.CART!D.DOOR!D.BRHM!D.SSP),E.A1	:EXPECTED A1
3982	020022	012737	000001	007466	MOV	#1,E.B1	:MSG ID FOR EXPECTED MSG B1
3983	020030	005037	007470		CLR	E.A2	:EXPECTED MSG A2
3984	020034	012737	000002	007472	MOV	#2,E.B2	:MSG ID FOR EXPECTED MSG B2
3985	020042	012737	000003	007476	MOV	#3,E.B3	:MSG ID FOR EXPECTED MSG B3
3986	020042	012737	000003	007476	MOV	#3,E.B3	:MSG ID FOR EXPECTED MSG B3
3987	020050	004737	037264		JSR	PC,CHKMSG	:CHECK MSGS A0, B0, A1, B1
3988	020054	000003			.WORD	T.A2!T.B2!0	:& MSGS SPECIFIED HERE

3999	020056	104052				ERROR	52		;MSG A0 ERROR AFTER WRITE DATA CMD
3990	020060	104023				ERROR	23		;MSH B0 ERROR
3991	020062	104053				ERROR	53		;MSG A1 ERROR
3992	020064	104025				ERROR	25		;MSG B1 ERROR
3993	020066	104415				SCOPI			
3994	020070	012706	001100			MOV	#STACK,SP		;RESTORE STK PTR
3995									
3996	020074	004737	040452			JSR	PC,SUBCLR		
3997	020100	104024				ERROR	24		;CERR AFTER SCLR
3998									
3999	020102	013765	001406	000006		MOV	SECTOR,RKDA(R5)		;SETUP SECTOR
4000	020110	012765	006352	000004		MOV	#RDTAB,RKBA(R5)		
4001	020116	012765	177400	000002		MOV	#-256.,RKWC(R5)		
4002									
4003									
4004	020124	012765	000021	000000		MOV	#(RDATA),RKCS1(R5)		;READ DATA CMD
4005	020132	013737	001424	007426		MOV	T5000,TEMP1		;SETUP TIMEOUT
4006	020140	004737	036542			JSR	PC,FRDY		;FIND RDY
4007	020144	104013				ERROR	13		;NO RDY AFTER READ DATA CMD
4008	020146	004737	040100			JSR	PC,GSTAT		;GET FRESH STATUS
4009	020152	032737	100000	007370		BIT	#CERR,HCS1		
4010	020160	001454				BEQ	72\$		
4011	020162	032737	000200	007404		BIT	#BSE,HER		;SEE IF BAD SECTOR
4012	020170	001406				BEQ	70\$		
4013	020172	104065				ERROR	65		;DETECTED BSE IN READ BUT NOT IN WRITE CMD.
4014	020174	000413				BR	73\$		
4015	020176	104401	053155		69\$:	TYPE	MSG26		;ABORTING DATA TESTS
4016	020202	000137	031706			JMP	TIMING		
4017									
4018	020206	032737	100000	007404	70\$:	BIT	#DCK,HER		;SEE IF DATA CHECK ERROR
4019	020214	001402				BEQ	71\$		
4020	020216	104021				ERROR	21		;DATA CHECK ERROR AFTER READ CMD (ECC)
4021	020220	000401				BR	73\$		
4022									
4023	020222	104014			71\$:	ERROR	14		;CERR AFTER READ DATA CMD.
4024									
4025	020224				73\$:				
4026									
4027	020224	012737	010340	007460		MOV	#(O!D.SPIN!D.DRDY!D.VV!D.DRA),E.A0		;EXPECTED MSG A0
4028	020232	005037	007462			CLR	E.B0		;EXPECTED MSG B0
4029	020236	012737	001720	007464		MOV	#(D.SPOK!D.CART!D.DOOR!D.BRHM!D.SSP),E.A1		;EXPECTED A1
4030	020244	012737	000001	007466		MOV	#1,E.B1		;MSG ID FOR EXPECTED MSG B1
4031	020252	005037	007470			CLR	E.A2		;EXPECTED MSG A2
4032	020256	012737	000002	007472		MOV	#2,E.B2		;MSG ID FOR EXPECTED MSG B2
4033	020264	012737	000003	007476		MOV	#3,E.B3		;MSG ID FOR EXPECTED MSG B3
4034									
4035	020272	004737	037264			JSR	PC,CHKMSG		;CHECK MSGS A0, B0, A1, B1
4036	020276	000003				.WORD	T.A2!T.B2!0		; & MSGS SPECIFIED HERE
4037	020300	104054				ERROR	54		;MSG A0 ERROR AFTER READ DATA CMD
4038	020302	104026				ERROR	26		;MSH B0 ERROR
4039	020304	104056				ERROR	56		;MSG A1 ERROR
4040	020306	104030				ERROR	30		;MSG B1 ERROR
4041	020310	000732				BR	69\$		
4042	020312				72\$:				
4043									
4044	020312	012737	010340	007460		MOV	#(O!D.SPIN!D.DRDY!D.VV!D.DRA),E.A0		;EXPECTED MSG A0

000000	000000	000000	007462		CLR	E,BC	:EXPECTED MSG BC
000000	000000	000000	007464		MOV	8,0.SPOK!D.CARY!	:C.DCOR!C.BRHM!C.SSP,E.A1 :EXPECTED A1
000000	000000	000000	007466		MOV	81,E.B1	:MSG ID FOR EXPECTED MSG B1
000000	000000	000000	007472		CLR	E,A2	:EXPECTED MSG A2
000000	000000	000000	007476		MOV	82,E.B2	:MSG ID FOR EXPECTED MSG B2
000000	000000	000000	007476		MOV	83,E.B3	:MSG ID FOR EXPECTED MSG B3
000000	000000	000000	037264		JSR	PC,CHKMSG	:CHECK MSGS A0 BC A1 B1
000000	000000	000000			WORD	T,A2!T,B2!0	:8 MSGS SPECIFIED HERE
000000	000000	000000			ERROR	54	:MSG A0 ERROR AFTER REAC DATA CMD
000000	000000	000000			ERROR	56	:MSG B0 ERROR
000000	000000	000000			ERROR	58	:MSG A1 ERROR
000000	000000	000000			ERROR	30	:MSG B1 ERROR
000000	000000	000000	006352		MOV	8RD*AB,R1	
000000	000000	000000	006352	25:	MOV	(R1,WD1	:ACTUAL WORD FOR TYPEOUT
000000	000000	000000	006352		MOV	RD,WD2	:EXPECTED DATA FOR TYPEOUT
000000	000000	000000			MOV	RD,(R1 +	:COMPARE READ DATA TABLE AGAINST
000000	000000	000000			BEG	38	:THE 'SHOULD BE' VALUE
000000	000000	000000			ERROR	20	:REAC DATA DID NOT COMPARE WITH WRITE DATA
000000	000000	000000	007352	35:	MOV	R1,8RD*AB+5!2.	
000000	000000	000000	007352		BNE	25	
000000	000000	000000	001510		MOV	RD,DATA0	
000000	000000	000000	001510		BEG	45	
000000	000000	000000	001510		BR	55	
000000	000000	000000	001514	45:	MOV	8DATA1,RKBA,RS	:WRITE ALL 1'S
000000	000000	000000	001514		MOV	DATA1,RO	
000000	000000	000000	001406	000006	MOV	SECTOR,RKDA,RS	
000000	000000	000000	017556		JMP	15	
000000	000000	000000		55:	SCOP1		
000000	000000	000000	001100		MOV	8STACK,SP	:RESTORE STK PTR
000000	000000	000000	040452		JSR	PC,SUBCLR	
000000	000000	000000	040452		ERROR	24	:CERR AFTER SCUR
000000	000000	000000	000020	000010	BIS	8BA1,RKCS2,RS	:THIS PORTION OF THE TEST CHECKS
000000	000000	000000	001514	000004	MOV	8DATA1,RKBA,RS	:OUT THE WRITE CHECK CMD
000000	000000	000000	177400	000002	MOV	8-256,RKWC,RS	:ALL 1'S WERE PREVIOUSLY WRITTEN
000000	000000	000000	001406	000006	MOV	SECTOR,RKDA,RS	
000000	000000	000000	000031	000000	MOV	8(WRTCHK),RKCS1,RS	:WRITE CHECK CMD
000000	000000	000000	001424	007426	MOV	T5000,TEMP1	:SETUP TIMEOUT
000000	000000	000000	004737	036542	JSR	PC,FRDY	:FIND ROY
000000	000000	000000	104015		ERROR	15	:NO ROY AFTER WRITE CHECK CMD
000000	000000	000000	004737	040100	JSR	PC,GSTAT	:GET FRESH STATUS
000000	000000	000000	032737	100000	BIT	8CERR,HCS1	
000000	000000	000000	001453		BEG	755	
000000	000000	000000	032737	040000	BEG	8WCE,HCS2	:SEE IF WRITE CHECK ERROR
000000	000000	000000	001410	007372	BEG	745	
000000	000000	000000	016537	000024	MOV	RKDB,RS,WD1	:ACTUAL WORD FOR PRINTOUT

E07

JMBUS RK06 DRIVE DIAGNOSTIC PART 2
 0206:PII 07-OCT-76 13:50

MAC11 27.1006) 07-OCT-76 14:14 PAGE 82
 T11 BASIC WRITE DATA TEST: FULL SECTOR

SEG 0082

4.4.1	020602	013737	001514	001470	MOV	DATA1.W02	:EXPECTED WORD FOR TYPEOUT
4.4.2	020610	024016			ERROR	16	:WCE AFTER WRITE CMD
4.4.3	020612	000437			BR	758	
4.4.4	020614	104022			758: ERROR	22	:CERR AFTER WRITE CHECK CMD
4.4.5	020616	012737	010340	007460	MOV	#(0!D.SPIN!D.DRDY!D.VV!D.DRA),E.A0	:EXPECTED MSG A0
4.4.6	020624	005037	001720	007464	CLR	E.B0	:EXPECTED MSG B0
4.4.7	020630	012737	000001	007466	MOV	#(D.SPOK!D.CART!D.DOOR!D.BRM!D.SSP),E.A1	:EXPECTED A1
4.4.8	020636	012737	000001	007466	MOV	B1,E.B1	:MSG ID FOR EXPECTED MSG B1
4.4.9	020644	005037	007470		CLR	E.A2	:EXPECTED MSG A2
4.4.10	020650	012737	000002	007472	MOV	B2,E.B2	:MSG ID FOR EXPECTED MSG B2
4.4.11	020656	012737	000003	007476	MOV	B3,E.B3	:MSG ID FOR EXPECTED MSG B3
4.4.12	020664	004737	037264		JSR	PC,CHKMSG	:CHECK MSGS A0, B0, A1, B1
4.4.13	020670	000003			.WORD	T.A2!T.B2!0	:8 MSGS SPECIFIED HERE
4.4.14	020672	104057			ERROR	57	:MSG A0 ERROR AFTER WRITE CHECK CMD
4.4.15	020674	104031			ERROR	31	:MSG B0 ERROR
4.4.16	020676	104060			ERROR	60	:MSG A1 ERROR
4.4.17	020700	104032			ERROR	32	:MSG B1 ERROR
4.4.18	020702	104401	053155		TYPE	MSG26	:ABORTING DATA TESTS
4.4.19	020706	000137	031706		JMP	TIMING	
4.4.20	020712				758:		
4.4.21	020712	012737	010340	007460	MOV	#(0!D.SPIN!D.DRDY!D.VV!D.DRA),E.A0	:EXPECTED MSG A0
4.4.22	020720	005037	001720	007464	CLR	E.B0	:EXPECTED MSG B0
4.4.23	020724	012737	000001	007466	MOV	#(D.SPOK!D.CART!D.DOOR!D.BRM!D.SSP),E.A1	:EXPECTED A1
4.4.24	020732	012737	000001	007466	MOV	B1,E.B1	:MSG ID FOR EXPECTED MSG B1
4.4.25	020740	005037	007470		CLR	E.A2	:EXPECTED MSG A2
4.4.26	020744	012737	000002	007472	MOV	B2,E.B2	:MSG ID FOR EXPECTED MSG B2
4.4.27	020752	012737	000003	007476	MOV	B3,E.B3	:MSG ID FOR EXPECTED MSG B3
4.4.28	020760	004737	037264		JSR	PC,CHKMSG	:CHECK MSGS A0, B0, A1, B1
4.4.29	020764	000003			.WORD	T.A2!T.B2!0	:8 MSGS SPECIFIED HERE
4.4.30	020766	104057			ERROR	57	:MSG A0 ERROR AFTER WRITE CHECK CMD
4.4.31	020770	104031			ERROR	31	:MSG B0 ERROR
4.4.32	020772	104060			ERROR	60	:MSG A1 ERROR
4.4.33	020774	104032			ERROR	32	:MSG B1 ERROR
4.4.34	020776	104415			SCOPI		
4.4.35	021000	012706	001100		MOV	#STACK,SP	:RESTORE STK PTR
4.4.36	021004	004737	040452		JSR	PC,SUBCLR	
4.4.37	021010	104024			ERROR	24	:CERR AFTER SCLR
4.4.38	021012	012765	001510	000004	MOV	#DATA0,RKBA(R5)	:SETUP TO CHECK AGAINST WRONG DATA
4.4.39	021020	012765	177400	000002	MOV	#-256.,RKWC(R5)	
4.4.40	021026	012765	000031	000000	MOV	#WATCH,RKCS1(R5)	
4.4.41	021034	012737	141520	007426	MOV	#5000.,TEMP1	
4.4.42	021042	004737	036542		JSR	PC,FRDY	
4.4.43	021046	104015			ERROR	15	:NO RDY AFTER WRITE CHECK CMD
4.4.44	021050	004737	040100		JSR	PC,GSTAT	:GET FRESH STATUS
4.4.45	021054	032737	040000	007372	BIT	#WCE,HCS2	:EXPECT MISCOMPARE
4.4.46	021062	001001			BNE	68	

F07

UNIBUS RK06 DRIVE DIAGNOSTIC PART 2
 DZR6IC.P11 07-OCT-76 13:50

MACY11 27(1006) 07-OCT-76 14:14 PAGE 83
 T11 BASIC WRITE DATA TEST; FULL SECTOR

SEG 0083

```

4157 021064 104017          ERROR 17          ;WRITE CHECK CMD NOT FUNCTIONING
4158                                     ;WITH INTENTIONAL MISCOMPARE
4159 021066          65:
4160
4161 021066 012737 010340 007460      MOV      #<0!D.SPIN!D.DRDY!D.VV!D.DRA>,E.A0 ;EXPECTED MSG A0
4162 021074 005037 007462          CLR      E.B0          ;EXPECTED MSG B0
4163 021100 012737 001720 007464      MOV      #<0.D.SPOK!D.CART!0.D.DOOR!D.BRHM!D.SSP>,E.A1 ;EXPECTED A1
4164 021106 012737 000001 007466      MOV      #1,E.B1          ;MSG ID FOR EXPECTED MSG B1
4165 021114 005037 007470          CLR      E.A2          ;EXPECTED MSG A2
4166 021120 012737 000002 007472      MOV      #2,E.B2          ;MSG ID FOR EXPECTED MSG B2
4167 021126 012737 000003 007476      MOV      #3,E.B3          ;MSG ID FOR EXPECTED MSG B3
4168
4169 021134 004737 037264          JSR      PC,CHKMSG      ;CHECK MSGS A0, B0, A1, B1
4170 021140 000000          .WORD   0!0!0          ;8 MSGS SPECIFIED HERE
4171 021142 104057          ERROR   57          ;MSG A0 ERROR AFT WRT CHK CMD
4172 021144 104031          ERROR   31          ;MSG B0 ERROR
4173 021146 104060          ERROR   60          ;MSG A1 ERROR
4174 021150 104032          ERROR   32          ;MSG B1 ERROR
4175 021152          75:
4176
4177          ;*****
4178          ;*TEST 12      20 SECTOR FORMAT TEST
4179          ;*
4180          ;*      ALL 1'S ARE WRITTEN ON A FULL SECTOR IN 20 SECTOR FORMAT.
4181          ;*      MSG B0,B1 ARE CHECKED FOR ANY ERROR CONDITION
4182          ;*
4183          ;*****
4184          ;*ST12: SCOPE
4185 021154 012737 000001 001174      MOV      #1,STIMES      ;:DO 1 ITERATION
4186 021162 012706 001100          MOV      #STACK,SP      ;RESTORE STK PTR
4187
4188 021166 004737 040452          JSR      PC,SUBCLR      ;CERR AFTER SCLR
4189 021172 104024          ERROR   24
4190
4191 021174 012765 001536 000004      MOV      #HDTAB,RKBA(R5) ;HEADER WORD TABLE
4192 021202 012765 177704 000002      MOV      #-60.,RKWC(R5) ;WORD COUNT FOR 20 SECTOR FMT
4193 021210 005037 001352          CLR      TOCYL
4194 021214 005237 007354          INC      BADHDP          ;USED FOR VALID HALT
4195
4196
4197 021220 013737 001352 001366      MOV      TOCYL,CALADD   ;SETUP
4198 021226 012737 000000 001474      MOV      #0,HEAD        ;TO FILL
4199 021234 012737 000001 001502      MOV      #1,FORMAT      ;HEADER
4200 021242 004737 041434          JSR      PC,FHDTAB      ;TABLE
4201
4202
4203 021246 012765 010027 000000      MOV      #<CFMT!WRHEAD>,RKCS1(R5) ;WRITE HEADER CMD
4204 021254 013737 001424 007426      MOV      T5000,TEMP1    ;SETUP TIMEOUT
4205 021262 004737 036542          JSR      PC,FRDY        ;FIND RDY
4206 021266 104200          ERROR   200          ;NO RDY AFTER WRITE HEADER CMD
4207 021270 004737 040100          JSR      PC,GSTAT      ;GET FRESH STATUS
4208 021274 032737 100000 007370      BIT      #CERR,HCS1
4209 021302 001405          BEQ     64$
4210 021304 104201          ERROR   201          ;CERR AFTER WRITE HEADER CMD
4211 021306 104401 053155          TYPE    MSG26          ;ABORTING DATA TESTS TO DO TIMING TESTS
4212 021312 000137 031706          JMP     TIMING

```

```

4213 021316          64$:
4214
4215 021316 104415          SCOP1
4216 021320 012706 001100  MOV      #STACK,SP      ;RESTORE STK PTR
4217
4218 021324 004737 040452  JSR      PC,SUBCLR
4219 021330 104024          ERROR    24      ;CERR AFTER SCLR
4220
4221 021332 005037 001406          CLR      SECTOR
4222 021336 013765 001406 000006 4$:  MOV      SECTOR,RKDA(R5)
4223 021344 012765 001514 000004  MOV      #DATA1,RKBA(R5) ;WRITE ALL 1'S
4224 021352 052765 000020 000010  BIS      #BA1,RKCS2(R5) ;BUSS ADDR INCR INHIBIT
4225 021360 012765 177400 000002  MOV      #-256.,RKWC(R5) ;DO FULL SECTOR
4226
4227
4228 021366 012765 010023 000000  MOV      #<CFMT!WRDATA>,RKCS1(R5) ;WRITE DATA CMD
4229 021374 013737 001424 007426  MOV      T5000,TEMP1 ;SETUP TIMEOUT
4230 021402 004737 036542          JSR      PC,FRDY ;FIND RDY
4231 021406 104011          ERROR    11      ;NO RDY AFTER WRITE DATA CMD
4232 021410 004737 040100          JSR      PC,GSTAT ;GET FRESH STATUS
4233 021414 032737 100000 007370  BIT      #CERR,HCS1
4234 021422 001465          BEQ     68$      ;BR IF NO ERRORS
4235
4236 021424 032737 000200 007404  BIT      #BSE,HER ;SEE IF BAD SECTOR FLAG
4237 021432 001421          BEQ     66$      ;BR IF NO
4238 021434 004737 042122          JSR      PC,TRUERR ;ELSE SEE IF SECTOR LISTED IN BSE TABLE
4239 021440 000455          BR      67$      ;RETURN HERE IF NO
4240
4241 021442 005237 001406          INC     SECTOR ;RETURN HERE IF YES
4242 021446 023727 001406 000012  CMP     SECTOR,#10. ;ARE 10 CONSEC. SECTORS BAD
4243 021454 001003          BNE     65$      ;BR IF NO
4244 021456 104046          ERROR    46      ;ABORTING TEST DETECTED 10 BAD SECTORS
4245 021460 000137 022404          JMP     3$      ;BYPASS TEST
4246 021464 012765 100000 000000 65$:  MOV     #CCLR,RKCS1(R5) ;TRY ANOTHER SECTOR
4247 021472 000137 021336          JMP     4$
4248 021476 104012          66$:  ERROR    12      ;CERR WITH WRITE DATA CMD
4249
4250 021500 012737 010340 007460  MOV     #<0!D.SPIN!D.DRDY!D.VV!D.DRA>,E.A0 ;EXPECTED MSG A0
4251 021506 005037 007462          CLR     E.B0 ;EXPECTED MSG B0
4252 021512 012737 001720 007464  MOV     #<D.SPOK!D.CART!D.DOOR!D.BRHM!D.SSP>,E.A1 ;EXPECTED A1
4253 021520 012737 000001 007466  MOV     #1,E.B1 ;MSG ID FOR EXPECTED MSG B1
4254 021526 005037 007470          CLR     E.A2 ;EXPECTED MSG A2
4255 021532 012737 000002 007472  MOV     #2,E.B2 ;MSG ID FOR EXPECTED MSG B2
4256 021540 012737 000003 007476  MOV     #3,E.B3 ;MSG ID FOR EXPECTED MSG B3
4257
4258 021546 004737 037264          JSR     PC,CHKMSG ;CHECK MSGS A0, B0, A1, B1
4259 021552 000003          .WORD  T.A2!T.B2!0 ;& MSGS SPECIFIED HERE
4260 021554 104052          ERROR    52      ;MSG A0 ERROR AFTER WRITE DATA CMD
4261 021556 104023          ERROR    23      ;MSG B0 ERROR
4262 021560 104053          ERROR    53      ;MSG A1 ERROR
4263 021562 104025          ERROR    25      ;MSG B1 ERROR
4264 021564 104401 053155          TYPE    MSG26 ;ABORTING DATA TESTS TO DO TIMING
4265 021570 000137 031706          JMP     TIMING
4266 021574 104063          67$:  ERROR    63      ;BAD SECTOR NOT LISTED IN TABLE
4267 021576
4268 021576 012765 010001 000000 68$:  MOV     #<CFMT!SELDV>,RKCS1(R5)

```

4269	021604	013737	001412	007426	MOV	T10,TEMP1	
4270	021612	004737	036542		JSR	PC,FRDY	;FIND RDY
4271	021616	104117			ERROR	117	;RDY NOT FOUND AFTER SELDRV CMD
4272	021620	032737	001000	007416	BIT	#D.FORM,HMR2	
4273	021626	001001			BNE	15	
4274	021630	104102			ERROR	102	;FORMAT NOT SET
4275							
4276	021632						15:
4277							
4278	021632	012737	010340	007460	MOV	#<0!D.SPIN!D.DRDY!D.VV!D.DRA>,E.A0	;EXPECTED MSG A0
4279	021640	005037	007462		CLR	E.B0	;EXPECTED MSG B0
4280	021644	012737	001720	007464	MOV	#<D.SPOK!D.CART!D.DOOR!D.BRHM!D.SSP>,E.A1	;EXPECTED A1
4281	021652	012737	000001	007466	MOV	#1,E.B1	;MSG ID FOR EXPECTED MSG B1
4282	021660	005037	007470		CLR	E.A2	;EXPECTED MSG A2
4283	021664	012737	000002	007472	MOV	#2,E.B2	;MSG ID FOR EXPECTED MSG B2
4284	021672	012737	000003	007476	MOV	#3,E.B3	;MSG ID FOR EXPECTED MSG B3
4285							
4285	021700	004737	037264		JSR	PC,CHKMSG	;CHECK MSGS A0, B0, A1, B1
4287	021704	000000			.WORD	0!0!0	; & MSGS SPECIFIED HERE
4288	021706	104052			ERROR	52	;MSG A0 ERROR AFTER WRITE DATA CMD
4289	021710	104023			ERROR	57	;MSH B0 ERROR
4290	021712	104053			ERROR	53	;MSG A1 ERROR
4291	021714	104025			ERROR	25	;MSG B1 ERROR
4292	021716	012765	177400	000002	MOV	#-256.,RKWC(R5)	
4293	021724	012765	001514	000004	MOV	#DATA1,RKBA(R5)	
4294	021732	052765	000020	000010	BIS	#BA1,RKCS2(R5)	
4295	021740	013765	001406	000006	MOV	SECTOR,RKDA(R5)	
4296							
4297	021746	012765	010031	000000	MOV	#<CFMT!WRTCHK>,RKCS1(R5)	;WRITE CHECK CMD
4298	021754	013737	001424	007426	MOV	T50000,TEMP1	;SETUP TIMEOUT
4299	021762	004737	036542		JSR	PC,FRDY	;FIND RDY
4300	021766	104015			ERROR	15	;NO RDY AFTER WRITE CHECK CMD
4301	021770	004737	040100		JSR	PC,GSTAT	;GET FRESH STATUS
4302	021774	032737	100000	007370	BIT	#CERR,HCS1	
4303	022002	001453			BEQ	705	
4304	022004	032737	040000	007372	BIT	#WCE,HCS2	;SEE IF WRITE CHECK ERROR
4305	022012	001410			BEQ	695	
4306	022014	016537	000024	001466	MOV	RKDB(R5),WD1	;ACTUAL WORD FOR PRINTOUT
4307	022022	013737	001514	001470	MOV	DATA1,WD2	;EXPECTED WORD FOR TYPEOUT
4308	022030	104016			ERROR	16	;WCE AFTER WRITE CMD
4309	022032	000437			BR	705	
4310							
4311	022034	104022			ERROR	22	;CERR AFTER WRITE CHECK CMD
4312							
4313	022036	012737	010340	007460	MOV	#<0!D.SPIN!D.DRDY!D.VV!D.DRA>,E.A0	;EXPECTED MSG A0
4314	022044	005037	007462		CLR	E.B0	;EXPECTED MSG B0
4315	022050	012737	001720	007464	MOV	#<D.SPOK!D.CART!D.DOOR!D.BRHM!D.SSP>,E.A1	;EXPECTED A1
4316	022056	012737	000001	007466	MOV	#1,E.B1	;MSG ID FOR EXPECTED MSG B1
4317	022064	005037	007470		CLR	E.A2	;EXPECTED MSG A2
4318	022070	012737	000002	007472	MOV	#2,E.B2	;MSG ID FOR EXPECTED MSG B2
4319	022076	012737	000003	007476	MOV	#3,E.B3	;MSG ID FOR EXPECTED MSG B3
4320							
4321	022104	004737	037264		JSR	PC,CHKMSG	;CHECK MSGS A0, B0, A1, B1
4322	022110	000003			.WORD	T.A2!T.B2!0	; & MSGS SPECIFIED HERE
4323	022112	104057			ERROR	57	;MSG A0 ERROR AFTER WRITE CHECK CMD
4324	022114	104031			ERROR	31	;MSH B0 ERROR

4325	022116	104060			ERROR	60		;MSG A1 ERROR
4326	022120	104032			ERROR	32		;MSG B1 ERROR
4327	022122	104401	053155		TYPE	MSG26		;ABORTING DATA TESTS
4328	022126	000137	031706		JMP	TIMING		
4329								
4330	022132						70\$:	
4331	022132	012765	010001	000000	MOV	#<CFMT!SELDRV>,RKCS1(R5)		
4332	022140	013737	001412	007426	MOV	T10,TEMP1		
4333	022146	004737	036542		JSR	PC,FRDY		;FIND RDY
4334	022152	104117			ERROR	117		;NO RDY AFTER SELDRV CMD
4335	022154	032737	001000	007416	BIT	#D.FORM,HMR2		
4336	022162	001001			BNE	2\$		
4337	022164	104103			ERROR	103		;FORMAT NOT SET
4338								
4339	022166						2\$:	
4340								
4341	022166	012737	010340	007460	MOV	#<D.SPIN!D.DRDY!D.VV!D.DRA>,E.A0		;EXPECTED MSG A0
4342	022174	005037	007462		CLR	E.B0		;EXPECTED MSG B0
4343	022200	012737	001720	007464	MOV	#<D.SPOK!D.CART!D.DOOR!D.BRHM!D.SSP>,E.A1		;EXPECTED A1
4344	022206	012737	000001	007466	MOV	#1,E.B1		;MSG ID FOR EXPECTED MSG B1
4345	022214	005037	007470		CLR	E.A2		;EXPECTED MSG A2
4346	022220	012737	000002	007472	MOV	#2,E.B2		;MSG ID FOR EXPECTED MSG B2
4347	022226	012737	000003	007476	MOV	#3,E.B3		;MSG ID FOR EXPECTED MSG B3
4348								
4349	022234	004737	037264		JSR	PC,CHKMSG		;CHECK MSGS A0, B0, A1, B1
4350	022240	000000			.WORD	0!0!0		;8 MSGS SPECIFIED HERE
4351	022242	104057			ERROR	57		;MSG A0 ERROR AFTER WRITE CHECK CMD
4352	022244	104031			ERROR	31		;MSG B0 ERROR
4353	022246	104060			ERROR	60		;MSG A1 ERROR
4354	022250	104032			ERROR	32		;MSG B1 ERROR
4355	022252	104415			SCOP1			
4356	022254	012706	001100		MOV	#STACK,SP		;RESTORE STK PTR
4357								
4358	022260	004737	040452		JSR	PC,SUBCLR		
4359	022264	104024			ERROR	24		;CERR AFTER SCLR
4360								
4361	022266	012765	001536	000004	MOV	#HDTAB,RKBA(R5)		;RESTORE CYL 0 TO 22 SECTOR FMT
4362	022274	012765	177676	000002	MOV	#-66.,RKWC(R5)		
4363	022302	005037	001352		CLR	TOCYL		
4364								
4365								
4366	022306	013737	001352	001366	MOV	TOCYL,CALADD		;SETUP
4367	022314	012737	000000	001474	MOV	#0,HEAD		;TO FILL
4368	022322	012737	000000	001502	MOV	#0,FORMAT		;HEADER
4369	022330	004737	041434		JSR	PC,FHDTAB		;TABLE
4370								
4371								
4372	022334	012765	000027	000000	MOV	#<WRHEAD>,RKCS1(R5)		;WRITE HEADER CMD
4373	022342	013737	001424	007426	MOV	T5000,TEMP1		;SETUP TIMEOUT
4374	022350	004737	036542		JSR	PC,FRDY		;FIND RDY
4375	022354	104200			ERROR	200		;NO RDY AFTER WRITE HEADER CMD
4376	022356	004737	040100		JSR	PC,GSTAT		;GET FRESH STATUS
4377	022362	032737	100000	007370	BIT	#CERR,HCS1		
4378	022370	001405			BEQ	71\$		
4379	022372	104201			ERROR	201		;CERR AFTER WRITE HEADER CMD
4380	022374	104401	053155		TYPE	,MSG26		;ABORTING DATA TESTS TO DO TIMING TESTS

4381 022400 000137 031706

JMP TIMING

4382 022404

71S:

4383

4384 022404 005037 007354

3S: CLR BADHDR ;USED FOR VALID HALT

4385

4386

4387

4388

4389

4390

4391

4392

4393

4394

4395

4396

4397

4398

4399

4399 022410 000004 000001 001174

*TEST 13 TEST OFFSET & RTC LOGIC
*
* THE HEADS ARE FIRST OFFSET BY OFFSET COMMANDS.
* THIS TEST CHECKS THE RTC LOGIC BY VERIFYING THAT THE
* 'OFFSET ON' BIT (MSG A,00) RESETS AND THE OFFSET REG
* BECOMES THE CYL DIFF INFO WHEN A SEEK CMD TO A
* DIFFERENT CYLINDER IS ISSUED
* IT ALSO TESTS THAT DRIVE CLEAR & SEEK TO SELF WILL NOT
* CLEAR THE 'OFFSET ON' BIT OR THE OFFSET REG.
* ALL OFFSET POSITIONS IN BOTH DIRECTIONS ARE CHECKED

4400 022412 012737 000001 001174

TST13: SCOPE
MOV #1,STIMES ;DO 1 ITERATION
MOV #STACK,SP ;RESTORE STK PTR

4401 022420 012706 001100

4402

4403 022424 012702 000001

MOV #1,R2 ;MIN POS OFFSET

4404

4405 022430 004737 040452

1S: JSR PC,SUBCLR
ERROR 24 ;CERR AFTER SCLR

4406

4407 022434 104024

4408

4408 022436 010265 000016

MOV R2,RKASOF(R5) ;SET OFFSET

4409

4410 022442 012737 022536 001176

4411 022450 012765 000015 000000

4412 022456 013737 001420 007426

4413 022464 004737 036542

4414 022470 104033

MOV #64,\$ESCAPE
MOV #OFFSET,RKCS1(R5) ;OFFSET CMD
MOV T100,TEMP1 ;SETUP TIMEOUT
JSR PC,FRDY
ERROR 33 ;NO RDY AFTER OFFSET CMD

4415

4416 022472 012737 032140 007460

4417 022500 005037 007462

4418 022504 012737 001720 007464

4419 022512 012737 000001 007466

4420

4421 022520 004737 037264

4422 022524 000000

4423 022526 104035

4424 022530 104061

4425 022532 104036

4426 022534 104062

4427

4428 022536 005037 001176 007426

4429 022542 013737 001422 007426

4430 022550 004737 037152

4431 022554 104034

4432

4433

4434 022556 012737 052340 007460

4435 022564 005037 007462

4436 022570 012737 001720 007464

64S: CLR \$ESCAPE
MOV T500,TEMP1 ;SETUP TIMEOUT
JSR PC,FATT2 ;FIND ATTN
ERROR 34 ;NO ATTN AFTER OFFSET CMD

MOV #<D.DSC!D.OFF!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

K07

UNIBUS RK06 DRIVE DIAGNOSTIC PART 2
DZR6IC.P11 07-OCT-76 13:50MACY11 27(1006) 07-OCT-76 14:14 PAGE 88
T13 TEST OFFSET & RTC LOGIC

SEG 0088

4437	022576	012737	000001	007466	MOV	#1,E.B1	;MSG ID FOR EXPECTED MSG B1
4438	022604	005037	007470		CLR	E.A2	;EXPECTED MSG A2
4439	022610	012737	000002	007472	MOV	#2,E.B2	;MSG ID FOR EXPECTED MSG B2
4440	022616	012737	000003	007476	MOV	#3,E.B3	;MSG ID FOR EXPECTED MSG B3
4441							
4442	022624	004737	037264		JSR	PC,CHKMSG	;CHECK MSGS A0, B0, A1, B1
4443	022630	000003			.WORD	T.A2!T.B2!0	; & MSGS SPECIFIED HERE
4444	022632	104260			ERROR	260	;MSG A0 ERROR AFTER OFFSET CMD
4445	022634	104261			ERROR	261	;MSH B0 ERROR
4446	022636	104037			ERROR	37	;MSG A1 ERROR
4447	022640	104040			ERROR	40	;MSG B1 ERROR
4448							
4449	022642	005737	001364		TST	CYLADD	
4450	022646	001401			BEQ	17\$	
4451	022650	104042			ERROR	42	;CYL ADDR IN B2 WAS NOT 0
4452							;AFTER OFFSET CMD FROM CYL 0
4453	022652						
4454	022652	010265	000016		MOV	R2,RKASOF(R5)	;REFRESH RKASOF
4455							
4456	022656	032702	000200		BIT	#BIT7,R2	
4457	022662	001005			BNE	65\$;BR IF NEG OFFSET
4458							
4459	022664	020237	001362		CMP	R2,CYLDIF	;CHECK POS OFFSET
4460	022670	001406			BEQ	66\$	
4461	022672	104114			ERROR	114	;OFFSET IN A2 NOT = RKASOF
4462	022674	000404			BR	66\$;AFTER OFFSET CMD
4463							
4464	022676	020137	001362		CMP	R1,CYLDIF	;CHECK NEG OFFSET
4465	022702	001401			BEQ	66\$	
4466	022704	104114			ERROR	114	;OFFSET IN A2 NOT = RKASOF
4467							;AFTER OFFSET CMD
4468	022706						
4469							
4470	022706	012765	100000	000000	MOV	#CCLR,RKCS1(R5)	
4471	022714	013765	001222	000010	MOV	\$UNIT,RKCS2(R5)	;DRIVE#
4472	022722	012765	000005	000000	MOV	#CLEAR,RKCS1(R5)	;DRIVE CLEAR CMD
4473	022730	013737	001412	007426	MOV	T10,TEMP1	;SETUP TIMEOUT
4474	022736	004737	036542		JSR	PC,FRDY	;FIND RDY
4475	022742	104151			ERROR	151	;NO RDY AFTER DRIVE CLEAR CMD
4476	022744	004737	037024		JSR	PC,TSTATN	;TEST FOR ATTN
4477	022750	000401			BR	67\$	
4478	022752	104154			ERROR	154	;ATTN NOT CLEARED AFTER DRIVE CLEAR CMD
4479	022754						
4480							
4481	022754	012765	100000	000000	MOV	#CCLR,RKCS1(R5)	
4482	022762	004737	040100		JSR	PC,GSTAT	
4483	022766	032737	002000	007416	BIT	#D.OFF,HMR2	
4484	022774	001001			BNE	4\$	
4485	022776	104043			ERROR	43	;OFFSET BIT IN RKMR2 CLEARED
4486							;AFTER DRIVE CLEAR CMD & SELECT DRV CMD
4487	023000	012737	012340	007460	MOV	#<D.OFF!D.SPIN!D.DRDY!D.VV!D.DRA>,E.A0	;EXPECTED A0
4488	023006	005037	007462		CLR	E.B0	
4489	023012	012737	001720	007464	MOV	#<D.SPOK!D.CART!D.DOOR!D.BRHM!D.SSP>,E.A1	
4490	023020	012737	000001	007466	MOV	#1,E.B1	
4491							
4492	023026	004737	037264		JSR	PC,CHKMSG	;CHECK MSGS A0, B0, A1, B1

```

4493 023032 000003 .WORD T.A2!T.B2!0 ;& MSGS SPECIFIED HERE
4494 023034 104273 ERROR 273 ;MSG A0 ERROR AFTER DRIVE CLEAR CMD
4495 023036 104265 ERROR 265 ;MSH B0 ERROR
4496 023040 104274 ERROR 274 ;MSG A1 ERROR
4497 023042 104266 ERROR 266 ;MSG B1 ERROR
4498 023044 010265 000016 MOV R2,RKASOF(R5) ;REFRESH RKASOF
4499
4500 023050 032702 000200 BIT #BIT7,R2
4501 023054 001005 BNE 68$ ;BR IF NEG OFFSET
4502
4503 023056 020237 001362 CMP R2,CYLDIF ;CHECK POS OFFSET
4504 023062 001406 BEQ 69$
4505 023064 104115 ERROR 115 ;OFFSET IN A2 NOT = RKASOF
4506 023066 000404 BR 69$ ;AFTER DRIVE CLEAR CMD
4507
4508 023070 020137 001362 68$: CMP R1,CYLDIF ;CHECK NEG OFFSET
4509 023074 001401 BEQ 69$
4510 023076 104115 ERROR 115 ;OFFSET IN A2 NOT = RKASOF
4511 ;AFTER DRIVE CLEAR CMD
4512 023100 69$:
4513
4514 023100 012765 000017 000000 MOV #SEEK,RKCS1(R5) ;SEEK CMD
4515 023106 013737 001414 007426 MOV T50,TEMP1 ;SETUP TIMEOUT
4516 023114 004737 036542 JSR PC,FRDY ;FIND RDY
4517 023120 104131 ERROR 131 ;NO RDY AFTER SEEK CMD
4518
4519 023122 013737 001424 007426 MOV T50000,TEMP1 ;SETUP TIMEOUT
4520 023130 004737 037152 JSR PC,FATT2 ;FIND ATTN
4521 023134 104132 ERROR 132 ;NO ATTN AFTER SEEK CMD
4522
4523 023136 032737 100000 007370 BIT #CERR,HCS1
4524 023144 001401 BEQ 70$
4525 023146 104210 ERROR 210 ;CERR AFTER SEEK CMD
4526
4527 023150 70$:
4528
4529
4530 023150 032737 002000 007416 BIT #D.OFF,HMR2
4531 023156 001001 BNE 7$
4532 023160 104045 ERROR 45 ;OFFSET BIT CLEARED IN RKMR2 AFTER SEEK TO SELF.
4533
4534 023162 7$:
4535
4536 023162 012737 052340 007460 MOV #<D.DSC!D.OFF!D.SPIN!D.DRDY!D.VV!D.DRA>,E.A0 ;EXPECTED MSG A0
4537 023170 005037 007462 CLR E.B0 ;EXPECTED MSG B0
4538 023174 012737 001720 007464 MOV #<D.SPOK!D.CART!D.DOOR!D.BRHM!D.SSP>,E.A1 ;EXPECTED A1
4539 023202 012737 000001 007466 MOV #1,E.B1 ;MSG ID FOR EXPECTED MSG B1
4540 023210 005037 007470 CLR E.A2 ;EXPECTED MSG A2
4541 023214 012737 000002 007472 MOV #2,E.B2 ;MSG ID FOR EXPECTED MSG B2
4542 023222 012737 000003 007476 MOV #3,E.B3 ;MSG ID FOR EXPECTED MSG B3
4543
4544 023230 004737 037264 JSR PC,CHKMSG ;CHECK MSGS A0, B0, A1, B1
4545 023234 000003 .WORD T.A2!T.B2!0 ;& MSGS SPECIFIED HERE
4546 023236 104140 ERROR 140 ;MSG A0 ERROR AFTER SEEK TO SELF
4547 023240 104141 ERROR 141 ;MSH B0 ERROR
4548 023242 104142 ERROR 142 ;MSG A1 ERROR

```

4549	023244	104143			ERROR	143			;MSG B1 ERROR
4550	023246	010265	000016		MOV	R2,RKASOF(R5)			;REFRESH RKASOF
4551									
4552	023252	032702	000200		BIT	#BIT7,R2			
4553	023256	001005			BNE	715			;BR IF NEG OFFSET
4554									
4555	023260	020237	001362		CMP	R2,CYLDIF			;CHECK POS OFFSET
4556	023264	001406			BEQ	725			
4557	023266	104123			ERROR	123			;OFFSET IN A2 NOT = RKASOF
4558	023270	000404			BR	725			;AFTER SEEK TO SELF
4559									
4560	023272	020137	001362	715:	CMP	R1,CYLDIF			;CHECK NEG OFFSET
4561	023276	001401			BEQ	725			
4562	023300	104123			ERROR	123			;OFFSET IN A2 NOT = RKASOF
4563									;AFTER SEEK TO SELF
4564	023302			725:					
4565									
4566	023302	004737	040452		JSR	PC,SUBCLR			
4567	023306	104024			ERROR	24			;CERR AFTER SCLR
4568									
4569	023310	012737	000012	001352	MOV	#10.,TOCYL			
4570	023316	012765	000012	000020	MOV	#10.,RKDC(R5)			;SETUP CYL 10
4571									;DO ACTUAL IMPLIED SEEK TO CYL 10 TO VERIFY
4572									;OFFSET BIT IN RKMR2 CLEARED
4573									
4574									
4575	023324	012700	001742		MOV	#RHTAB,R0			
4576	023330	012765	000025	000000	MOV	#<RDHEAD>,RKCS1(R5)			;READ HEADER CMD
4577	023336	013737	001424	007426	MOV	T50000,TEMP1			;SETUP TIMEOUT
4578	023344	004737	036542		JSR	PC,FRDY			;FIND RDY
4579	023350	104171			ERROR	171			;NO RDY AFTER READ HEADER CMD
4580	023352	032737	100000	007370	BIT	#CERR,HCS1			
4581	023360	001405			BEQ	745			
4582	023362	104174			ERROR	174			;CERR AFTER READ HEADER CMD
4583	023364	104401	053155		TYPE	MSG26			;ABORTING DATA TESTS TO DO TIMING TESTS
4584	023370	000137	031706		JMP	TIMING			
4585									
4586	023374	016520	000024	745:	MOV	RKDB(R5),(R0)+			;1'ST WORD FROM SILO TO RHTAB
4587	023400	016520	000024		MOV	RKDB(R5),(R0)+			;2'ND WORD
4588	023404	016520	000024		MOV	RKDB(R5),(R0)+			;3'RD WORD
4589									
4590									
4591	023410	032765	100000	000010	BIT	#DLT,RKCS2(R5)			
4592	023416	001407			BEQ	755			
4593	023420	004737	040100		JSR	PC,GSTAT			
4594	023424	104173			ERROR	173			;DLT AFTER READ HEADER CMD
4595	023426	104401	053155		TYPE	MSG26			;ABORTING DATA TESTS TO DO TIMING TESTS
4596	023432	000137	031706		JMP	TIMING			
4597	023436			755:					
4598									
4599	023436	012737	010340	007460	MOV	#<0!D.SPIN!D.DRDY!D.VV!D.DRA>,E.A0			;EXPECTED MSG A0
4600	023444	005037	007462		CLR	E.B0			;EXPECTED MSG B0
4601	023450	012737	001720	007464	MOV	#<D.SPOK!D.CART!D.DOOR!D.BRHM!D.SSP>,E.A1			;EXPECTED A1
4602	023456	012737	000001	007466	MOV	#1,E.B1			;MSG ID FOR EXPECTED MSG B1
4603	023464	005037	007470		CLR	E.A2			;EXPECTED MSG A2
4604	023470	012737	000002	007472	MOV	#2,E.B2			;MSG ID FOR EXPECTED MSG B2

4605	023476	012737	000003	007476	MOV	#3,E.B3	;MSG ID FOR EXPECTED MSG B3
4606							
4607	023504	004737	037264		JSR	PC,CHKMSG	;CHECK MSGS A0, B0, A1, B1
4608	023510	000003			.WORD	T.A2!T.B2!0	; & MSGS SPECIFIED HERE
4609	023512	104301			ERROR	301	;MSG A0 ERROR AFTER READ HEADER CMD
4610	023514	104271			ERROR	271	;MSH B0 ERROR
4611	023516	104302			ERROR	302	;MSG A1 ERROR
4612	023520	104272			ERROR	272	;MSG B1 ERROR
4613							
4614	023522	023737	001742	001352	CMP	RHTAB,TOCYL	;CHECK WORD 0 ONLY, CYL#
4615	023530	001401			BEQ	73\$;BR IF SAME
4616	023532	104051			ERROR	51	;WRONG CYL# ON HEADER
4617	023534					73\$:	
4618							
4619							
4620	023534	032737	002000	007416	BIT	#0.OFF,HMR2	
4621	023542	001401			BEQ	9\$	
4622	023544	104101			ERROR	101	;OFFSET NOT CLEARED AFTER READ HEADER WITH MOVEMENT
4623							
4624	023546	023727	001364	000012	9\$:	CMP	CYLADD,#10.
4625	023554	001401			BEQ	10\$	
4626	023556	104122			ERROR	122	;DID NOT GO TO CYL 10
4627							
4628	023560	005737	001362		10\$:	TST	CYLDIF
4629	023564	001401			BEQ	16\$	
4630	023566	104101			ERROR	101	;OFFSET NOT CLEARED IN RKMR2
4631							
4632	023570	004737	040452		16\$:	JSR	PC,SUBCLR
4633	023574	104024			ERROR	24	;CERR AFTER SCLR
4634							
4635	023576	012765	000017	000000	MOV	#SEEK,RKCS1(R5)	;SEEK CMD
4636	023604	013737	001414	007426	MOV	T50,TEMP1	;SETUP TIMEOUT
4637	023612	004737	036542		JSR	PC,FRDY	;FIND RDY
4638	023616	104131			ERROR	131	;NO RDY AFTER SEEK CMD
4639							
4640	023620	013737	001424	007426	MOV	T50000,TEMP1	;SETUP TIMEOUT
4641	023626	004737	037152		JSR	PC,FATT2	;FIND ATTN
4642	023632	104132			ERROR	132	;NO ATTN AFTER SEEK CMD
4643							
4644	023634	032737	100000	007370	BIT	#CERR,HCS1	
4645	023642	001401			BEQ	76\$	
4646	023644	104210			ERROR	210	;CERR AFTER SEEK CMD
4647							
4648	023646					76\$:	
4649							
4650							
4651	023646	032702	000200		BIT	#BIT7,R2	;SEE IF DOING NEG OFFSETS
4652	023652	001014			BNE	18\$;BR IF YES
4653							
4654	023654	005202			INC	R2	
4655	023656	020227	000061		CMP	R2,#61	;SEE IF JUST DID MAX POS OFFSET
4656	023662	001402			BEQ	20\$;BR IF YES
4657	023664	000137	022430		JMP	1\$;ELSE DO NEXT POS OFFSET
4658							
4659	023670	012702	000201		20\$:	MOV	#201,R2
4660	023674	012701	000101		MOV	#101,R1	;SETUP NEG OFFSET OFOR MSG A

```

4661 023700 000137 022430
4662
4663 023704 005201
4664 023706 005202
4665 023710 020227 000261
4666 023714 001402
4667 023716 000137 022430
4668
4669
4670
4671
4672
4673
4674
4675
4676
4677
4678
4679
4680
4681
4682
4683
4684
4685
4686
4687
4688
4689
4690
4691
4692
4693
4694
4695
4696
4697
4698
4699
4700
4701
4702
4703
4704
4705
4706
4707
4708
4709
4710
4711
4712
4713
4714
4715
4716

```

```

JMP 18 ;DO NEG OFFSET
:BS: INC R1
      INC R2
      CMP R2,#261 ;SEE IF ALL NEG OFFSETS DONE
      BEQ TS+14 ;GO TO NEXT TST
      JMP 18 ;DO ANOTHER

```

```

*****
*TEST 14 TEST READ DATA AT ALL HEAD OFFSET POSITIONS

```

```

* THIS TEST VERIFIES THAT THE HEAD OFFSET LOGIC IS OPERATIONAL BY
* WRITING ALL 1'S PATTERNS ON CYLINDER 0, HEAD 0. THEN
* PERFORMING READ DATA FROM CENTERLINE AND MOVING OUT + AND - OFFSET
* POSITIONS UNTIL A FAILURE OCCURES. THE FAILING OFFSET POSITIONS
* ARE PRINTED OUT IF LESS THAN THE OFFSET TOLERANCE TO BE SPECIFIED.
* OFFSET CODES ARE ALSO VERIFIED BY READING MSG A, STATUS 00 & 10.

```

```

* ALL HEADS ARE TESTED AT CYLINDER 0
* IF THERE ARE NO FAILURES AT ALL, THIS INDICATES THAT

```

```

* OR
* A. HEADS DID NOT MOVE AT ALL
* B. THE COMBINATION OF DISC SURFACE, HEADS, R/W AMP
* ARE EXCEPTIONALLY GOOD.

```

```

* AN APPROPRIATE MESSAGE WILL BE TYPED.

```

```

*****

```

```

4699 023722 000004
4690 023724 012737 003001 001174
4691 023732 012706 001100
4692
4693 023736 004737 040452
4694 023742 104024
4695
4696 023744 012765 001514 000004
4697 023752 052765 000020 000010
4698 023760 012765 177400 000002
4700
4701 023766 005037 001406
4702 023772 013765 001406 000006
4703
4704 024000 012765 000023 000000
4705 024006 013737 001424 007426
4706 024014 004737 036542
4707 024020 104011
4708 024022 004737 040100
4709 024026 032737 100000 007370
4710 024034 001465
4711
4712 024036 032737 000200 007404
4713 024044 001421
4714 024046 004737 042122
4715 02-052 000455

```

```

*ST14: SCOPE
      MOV #1,STIMES ;DO 1 ITERATION
      MOV #STACK,SP ;RESTORE STK PTR
      JSR PC,SUBCLR
      ERROR 24 ;CERR AFTER SCLR
      MOV #DATA1,RKBA(R5) ;WRITE ALL 1'S
      BIS #BA1,RKCS2(R5) ;BUSS ADC INCR INHIBIT
      MOV #-256.,RKWC(R5) ;SECTOR 0 ONLY
      ;WILL DO AN IMPLIED SEEK TO CYL 0.
      ;WAS ON CYL 1 FROM LAST TEST
      CLR SECTOR
      11S: MOV SECTOR,RKDA(R5)
      MOV #(<WRDATA>,RCS1(R5) ;WRITE DATA CMD
      MOV T5000,TEMP1 ;SETUP TIMEOUT
      JSR PC,FRDY ;FIND RDY
      ERROR 11 ;NO RDY AFTER WRITE DATA CMD
      JSR PC,GSTAT ;GET FRESH STATUS
      BIT #CERR,HCS1
      BEQ 67$ ;BR IF NO ERRORS
      BIT #BSE,HER ;SEE IF BAD SECTOR FLAG
      BEQ 65$ ;BR IF NO
      JSR PC,TRUERR ;ELSE SEE IF SECTOR LISTED IN BSE TABLE
      BR 66$ ;RETURN HERE IF NO

```

4717	024054	005237	001406		INC	SECTOR	:RETURN HERE IF YES
4718	024060	023727	001406	000012	CMF	SECTOR,#10.	:ARE 10 CONSEC. SECTORS BAD
4719	024066	001003			BNE	64\$:BR IF NO
4720	024070	104046			ERROR	46	:ABORTING TEST DETECTED 10 BAD SECTORS
4721	024071	000137	025534		JMP	10\$:BYPASS TEST
4722	024076	012765	100000	000000	MOV	#CCLR,RKCS1(R5)	:TRY ANOTHER SECTOR
4723	024104	000137	023772		JMP	11\$	
4724	024110	104012		65\$:	ERROR	12	:CERR WITH WRITE DATA CMD
4725							
4726	024112	012737	010340	007460	MOV	#(0!D.SPIN!D.DRDY!D.VV!D.DRA),E.A0	:EXPECTED MSG A0
4727	024120	005037	007462		CLR	E.B0	:EXPECTED MSG B0
4728	024124	012737	001720	007464	MOV	#(D.SPOK!D.CART!D.DOOR!D.BRHM!D.SSP),E.A1	:EXPECTED A1
4729	024132	012737	000001	007466	MOV	#1,E.B1	:MSG ID FOR EXPECTED MSG B1
4730	024140	005037	007470		CLR	E.A2	:EXPECTED MSG A2
4731	024144	012737	000002	007472	MOV	#2,E.B2	:MSG ID FOR EXPECTED MSG B2
4732	024152	012737	000003	007476	MOV	#3,E.B3	:MSG ID FOR EXPECTED MSG B3
4733							
4734	024160	004737	037264		JSR	PC,CHKMSG	:CHECK MSGS A0, B0, A1, B1
4735	024164	000003			.WORD	T.A2!T.B2!0	:& MSGS SPECIFIED HERE
4736	024166	104052			ERROR	52	:MSG A0 ERROR AFTER WRITE DATA CMD
4737	024170	104023			ERROR	23	:MSG B0 ERROR
4738	024172	104053			ERROR	53	:MSG A1 ERROR
4739	024174	104025			ERROR	25	:MSG B1 ERROR
4740	024176	104001	053155		*TYPE	MSG26	:ABORTING DATA TESTS TO DO TIMING
4741	024202	000137	031706		JMP	TIMING	
4742	024206	104063		66\$:	ERROR	63	:BAD SECTOR NOT LISTED IN TABLE
4743	024210			67\$:			
4744							
4745	024210	012737	010340	007460	MOV	#(0!D.SPIN!D.DRDY!D.VV!D.DRA),E.A0	:EXPECTED MSG A0
4746	024216	005037	007462		CLR	E.B0	:EXPECTED MSG B0
4747	024222	012737	001720	007464	MOV	#(D.SP!D.CART!D.DOOR!D.BRHM!D.SSP),E.A1	:EXPECTED A1
4748	024230	012737	000001	007466	MOV	#1,E.B1	:MSG ID FOR EXPECTED MSG B1
4749	024236	005037	007470		CLR	E.A2	:EXPECTED MSG A2
4750	024242	012737	000002	007472	MOV	#2,E.B2	:MSG ID FOR EXPECTED MSG B2
4751	024250	012737	000003	007476	MOV	#3,E.B3	:MSG ID FOR EXPECTED MSG B3
4752							
4753	024256	004737	037264		JSR	PC,CHKMSG	:CHECK MSGS A0, B0, A1, B1
4754	024262	000003			.WORD	T.A2!T.B2!0	:& MSGS SPECIFIED HERE
4755	024264	104052			ERROR	52	:MSG A0 ERROR AFTER WRITE DATA CMD
4756	024266	104023			ERROR	23	:MSG B0 ERROR
4757	024270	104053			ERROR	53	:MSG A1 ERROR
4758	024272	104025			ERROR	25	:MSG B1 ERROR
4759	024274	012765	001514	000004	MOV	#DATA1,RKBA(R5)	
4760	024302	052765	000020	000010	BIS	#BA1,RKCS2(R5)	
4761	024310	012765	177400	000002	MOV	#-256.,RKWC(R5)	
4762	024316	013765	001406	000006	MOV	SECTOR,RKDA(R5)	
4763							
4764	024324	012765	000031	000000	MOV	#(WRTCHK),RKCS1(R5)	:WRITE CHECK CMD
4765	024332	013737	001424	0000426	MOV	T50000,TEMP1	:SETUP TIMEOUT
4766	024340	004737	036542		JSR	PC,FRDY	:FIND RDY
4767	024344	104015			ERROR	15	:NO RDY AFTER WRITE CHECK CMD
4768	024346	004737	040100		JSR	PC,GSTAT	:GET FRESH STATUS
4769	024352	032737	100000	007370	BIT	#CERR,HCS1	
4770	024360	001453			BEQ	69\$	
4771	024362	032737	040000	007372	BIT	#WCE,HCS2	:SEE IF WRITE CHECK ERROR
4772	024370	001410			BEQ	68\$	

024437	016537	000024	001466	MOV	RKDB(R5),WD1	:ACTUAL WORD FOR PRINTOUT
024440	013737	001514	001470	MOV	DATA1,WD2	:EXPECTED WORD FOR TYPEOUT
024440	104016			ERROR	16	:MCE AFTER WRITE CMC
024441	000437			BR	695	
024442	104022			695:	ERROR	22 :CERR AFTER WRITE CHECK CMC
024444	012737	010340	007460	MOV	#0,D.SPIN!D.DRDY!D.VV!D.DRA),E.A0	:EXPECTED MSG A0
024442	005037	007462		CLR	E.B0	:EXPECTED MSG B0
024442	012737	001720	007464	MOV	#<D.SPOK!D.CART!D.DOOR!D.BRM!D.SSP),E.A1	:EXPECTED A1
024443	012737	000001	007466	MOV	#1,E.B1	:MSG ID FOR EXPECTED MSG B1
024444	005037	007470		CLR	E.A2	:EXPECTED MSG A2
024446	012737	000002	007472	MOV	#2,E.B2	:MSG ID FOR EXPECTED MSG B2
024444	012737	000003	007476	MOV	#3,E.B3	:MSG ID FOR EXPECTED MSG B3
024462	004737	037264		JSR	PC,CHKMSG	:CHECK MSGS A0, B0, A1, B1
024466	000003			.WORD	T.A2!T.B2!0	: & MSGS SPECIFIED HERE
024470	104057			ERROR	57	:MSG A0 ERROR AFTER WRITE CHECK CMC
024472	104031			ERROR	31	:MSG B0 ERROR
024474	104060			ERROR	60	:MSG A1 ERROR
024476	104032			ERROR	32	:MSG B1 ERROR
024500	104401	053155		TYPE	MSG26	:ABORTING DATA TESTS
024504	000137	031706		JMP	*TIMING	
024510				695:		
024510	012737	010340	007460	MOV	#0,D.SPIN!D.DRDY!D.VV!D.DRA),E.A0	:EXPECTED MSG A0
024516	005037	007462		CLR	E.B0	:EXPECTED MSG B0
024522	012737	001720	007464	MOV	#<D.SPOK!D.CART!D.DOOR!D.BRM!D.SSP),E.A1	:EXPECTED A1
024530	012737	000001	007466	MOV	#1,E.B1	:MSG ID FOR EXPECTED MSG B1
024536	005037	007470		CLR	E.A2	:EXPECTED MSG A2
024542	012737	000002	007472	MOV	#2,E.B2	:MSG ID FOR EXPECTED MSG B2
024550	012737	000003	007476	MOV	#3,E.B3	:MSG ID FOR EXPECTED MSG B3
024556	004737	037264		JSR	PC,CHKMSG	:CHECK MSGS A0, B0, A1, B1
024562	000003			.WORD	T.A2!T.B2!0	: & MSGS SPECIFIED HERE
024564	104057			ERROR	57	:MSG A0 ERROR AFTER WRITE CHECK CMC
024566	104031			ERROR	31	:MSG B0 ERROR
024570	104060			ERROR	60	:MSG A1 ERROR
024572	104032			ERROR	32	:MSG B1 ERROR
024574	104401	052053		TYPE	MSG8	:READ WITH OFFSET TEST
024600	005001			CLR	A1	:HEAD #
024602	012700	000001		95:	MOV	#1,R0 :INIT OFFSET COUNTER
024606	104401	052112		TYPE	MSG9	:HEAD #
024612	010146			MOV	A1,-(SP)	::SAVE R1 FOR TYPEOUT
024614	104403			TYPOS		::TYPE HEAD #
024616	001			.BYTE	1	::GO TYPE--OCTAL ASCII
024617	000			.BYTE	0	::TYPE 1 DIGIT(S)
024620	104401	001205		TYPE	,SCLF	::SUPPRESS LEADING ZEROS
024624	005037	001472		15:	CLR	OFFERR :WRITE CHECK ERROR FLAG
024630	004737	040452		JSR	PC,SUBCLR	

E08

BUS R36 DRIVE DIAGNOSTIC PART 2
 07-07-76 13:50

MAC11 27,1006 07-07-76 14:14 PAGE 95
 T14 TEST READ DATA AT ALL HEAD OFFSET POSITIONS

SEG 0095

4839	024634	024024			ERROR	24		:CERR AFTER SCLR
4840	024636	000065	000016		MOV	R0,RKASOF(R5)		:OFFSET VALUE
4841	024642	000301			SWAB	R1		
4842	024644	010165	000006		MOV	R1,RKDA(R5)		:HEAD NO.
4843	024650	000301			SWAB	R1		
4844	024652	012737	024746	001176	MOV	#705,SESCAPE		
4845	024660	012765	000015	000000	MOV	#OFFSET,RKCS1(R5)		:OFFSET CMD
4846	024666	013737	001420	007426	MOV	T100,TEMP1		:SETUP TIMEOUT
4847	024674	004737	036542		JSR	PC,FRDY		
4848	024700	104033			ERROR	33		:NO RDY AFTER OFFSET CMD
4849	024702	012737	032140	007460	MOV	#<D.PIP!D.SPIN!D.OFF!D.VV!D.DRA>,E.A0		:EXPECTED MSG A0
4850	024710	005037	007462		CLR	E.B0		
4851	024714	012737	001720	007464	MOV	#<D.SPOK!D.CART!D.DOOR!D.BRHM!D.SSP>,E.A1		:EXPECTED A1
4852	024722	012737	000001	007466	MOV	#1,E.B1		
4853	024730	004737	037264		JSR	PC,CHKMSG		:CHECK MSGS A0, B0, A1, B1
4854	024734	000000			.WORD	0!0!0		:& MSGS SPECIFIED HERE
4855	024736	104035			ERROR	35		:MSG A0 ERROR DURING OFFSET CMD
4856	024740	104061			ERROR	61		:MSG B0 ERROR
4857	024742	104036			ERROR	36		:MSG A1 ERROR
4858	024744	104062			ERROR	62		:MSG B1 ERROR
4859	024746	005037	001176		CLR	SESCAPE		
4860	024752	013737	001422	007426	MOV	T5000,TEMP1		:SETUP TIMEOUT
4861	024760	004737	037152		JSR	PC,FA+T2		:FIND ATTN
4862	024764	104034			ERROR	34		:NO ATTN AFTER OFFSET CMD
4863	024766	012737	052340	007460	MOV	#<D.DSC!D.OFF!D.SPIN!D.DRDY!D.VV!D.DRA>,E.A0		:EXPECTED MSG A0
4864	024774	005037	007462		CLR	E.B0		:EXPECTED MSG B0
4865	025000	012737	001720	007464	MOV	#<D.SPOK!D.CART!D.DOOR!D.BRHM!D.SSP>,E.A1		:EXPECTED A1
4866	025006	012737	000001	007466	MOV	#1,E.B1		:MSG ID FOR EXPECTED MSG B1
4867	025014	005037	007470		CLR	E.A2		:EXPECTED MSG A2
4868	025020	012737	000002	007472	MOV	#2,E.B2		:MSG ID FOR EXPECTED MSG B2
4869	025026	012737	000003	007476	MOV	#3,E.B3		:MSG ID FOR EXPECTED MSG B3
4870	025034	004737	037264		JSR	PC,CHKMSG		:CHECK MSGS A0, B0, A1, B1
4871	025040	000003			.WORD	T.A2!T.B2!0		:& MSGS SPECIFIED HERE
4872	025042	104260			ERROR	260		:MSG A0 ERROR AFTER OFFSET CMD
4873	025044	104261			ERROR	261		:MSG B0 ERROR
4874	025046	104037			ERROR	37		:MSG A1 ERROR
4875	025050	104040			ERROR	40		:MSG B1 ERROR
4876	025052	012765	100000	000000	MOV	#CCLR,RKCS1(R5)		
4877	025060	013765	001222	000010	MOV	#UNIT,RKCS2(R5)		:DRIVE#
4878	025066	012765	000005	000000	MOV	#CLEAR,RKCS1(R5)		:DRIVE CLEAR CMD
4879	025074	013737	001412	007426	MOV	T10,TEMP1		:SETUP TIMEOUT
4880	025102	004737	036542		JSR	PC,FRDY		:FIND RDY
4881	025106	104151			ERROR	151		:NO RDY AFTER DRIVE CLEAR CMD
4882	025110	004737	037024		JSR	PC,TSTATN		:TEST FOR ATTN
4883	025114	000401			BR	71\$		
4884	025116	104154			ERROR	154		:ATTN NOT CLEARED AFTER DRIVE CLEAR CMD

F08

UNIBUS Rk06 DRIVE DIAGNOSTIC PART 2
 DZR6IC.P11 07-OCT-76 13:50

MACY11 27(1006) 07-OCT-76 14:14 PAGE 96
 T14 TEST READ DATA AT ALL HEAD OFFSET POSITIONS

SEQ 0096

```

4885 025120
4886
4887 025120 012765 001514 000004      MOV    #DATA1,RKBA(R5)
4888 025126 052765 000020 000010      BIS    #BA1,RKCS2(R5)
4889 025134 012765 177400 000002      MOV    #-256.,RKWC(R5)
4890 025142 013765 001406 000006      MOV    SECTOR,RKDA(R5)
4891 025150 012765 000031 000000      MOV    #WRCHK,RKCS1(R5) ;WRITE CHECK CMD
4892 025156 012737 141520 007426      MOV    #50000.,TEMP1 ;SETUP TIMEOUT
4893 025164 004737 036542      JSR    PC,FRDY ;FIND RDY
4894 025170 104015      ERROR 15 ;NO RDY AFTER WRITE CHECK CMD
4895 025172 004737 040100      JSR    PC,GSTAT ;GET FRESH STATUS
4896 025176 032737 040000 007372      BIT    #WCE,HCS2
4897 025204 001421      BEQ    2$
4898
4899 025206 016537 000024 001466      MOV    RKDB(R5),WD1 ;GET MISCOMPARED WORD
4900 025214 005237 001472      INC    OFFERR ;BAD WRITE CHK ERROR=SET ERR FLG.
4901
4902 025220 005737 001472      TST    OFFERR
4903 025224 001411      BEQ    2$
4904 025226 104401 054301      TYPE  MSG39 ;WRITE CHECK FAILURE AT OFFSET
4905 025232 010046      MOV    RD,-(SP) ;SAVE RD FOR TYPEOUT
4906 ;TYPE OFFSET VALUE
4907 025234 104403      TYPOS ;GO TYPE--OCTAL ASCII
4908 025236 006 ;TYPE 6 DIGITS
4909 025237 000 ;SUPPRESS LEADING ZEROS
4910 025240 104401 001205      TYPE  ,SCLRF
4911 025244 104401 001205      TYPE  ,SCLRF
4912
4913 025250 032700 000200 2$:      BIT    #BIT7,RO ;SEE IF OFFSET IS + OR -
4914 025254 001023      BNE    5$ ;BR IF - OFFSET
4915
4916 025256 020027 000060      CMP    RO,#60
4917 025262 001412      BEQ    4$
4918 025264 005737 001472      TST    OFFERR
4919 025270 001404      BEQ    3$
4920 025272 012700 000200 8$:      MOV    #200,RO ;SETUP FOR NEG OFFSET
4921 025276 000137 024624      JMP    1$
4922
4923 025302 005200 3$:      INC    RO
4924 025304 000137 024624      JMP    1$
4925
4926 025310 005737 001472 4$:      TST    OFFERR
4927 025314 001366      BNE    8$ ;DO NEG OFFSETS
4928 025316 104401 054143      TYPE  ,MSG37 ;NO WRITE CHECK ERROR AT MAX POS OFFSET
4929 ;NOTE! EITHER HEADS DID NOT MOVE
4930 ;OR READ/WRITE AMP IS EXCEPTIONALLY GOOD.
4931 025322 000763      BR     8$ ;DO NEG OFFSETS
4932
4933 025324 020027 000260 5$:      CMP    RO,#260
4934 025330 001404      BEQ    6$
4935 025332 005737 001472      TST    OFFERR
4936 025336 001076      BNE    TST15 ;GO TO NEXT TST
4937 025340 000760      BR     3$
4938
4939 025342 005737 001472 6$:      TST    OFFERR
4940 025346 001002      BNE    7$
  
```

```

4941 025350 104401 054221          TYPE      ,MSG38          ;NO WRITE CHECK ERROR AT MAX NEG OFFSET
4942                                     ;NOTE! EITHER HEADS DID NOT MOVE
4943                                     ;OR READ/WRITE AMP IS EXCEPTIONALLY GOOD.
4944 025354          75:
4945
4946 025354 012765 100000 000000      MOV      #CCLR,RKCS1(R5)
4947 025362 013765 001222 000010      MOV      $UNIT,RKCS2(R5)
4948 025370 012765 000013 000000      MOV      #RECAL,RKCS1(R5)          ;RECAL CMD
4949                                     ;RESET CYL DIFF/OFFSET & CYL ADDR REG
4950                                     ;IN RKMR2 & RKMR3 RESP.
4951 025376 013737 001412 007426      MOV      T10,TEMP1
4952 025404 004737 036542          JSR      PC,FRDY
4953 025410 104124          ERROR    124          ;SETUP TIMEOUT
4954                                     ;FIND RDY
4955 025412 012765 000001 000026      MOV      #1,RKMR1(R5)          ;SELECT WORD 1
4956 025420 004737 040100          JSR      PC,GSTAT
4957 025424 032737 020000 007416      BIT      #D.RTZ,HMR2
4958 025432 001001          BNE      72$
4959 025434 104244          ERROR    244          ;RTZ NOT SET DURING RECAL CMD
4960 025436 013737 001412 007430      MOV      T10,TEMP2
4961 025444 004737 037056          JSR      PC,FATT1
4962 025450 104055          ERROR    55          ;SETUP TIMEOUT
4963                                     ;FIND ATTN
4964 025452 012765 100000 000000      MOV      #CCLR,RKCS1(R5)
4965 025460 013765 001222 000010      MOV      $UNIT,RKCS2(R5)          ;DRIVE#
4966 025466 012765 000005 000000      MOV      #CLEAR,RKCS1(R5)          ;DRIVE CLEAR CMD
4967 025474 013737 001412 007426      MOV      T10,TEMP1
4968 025502 004737 036542          JSR      PC,FRDY
4969 025506 104151          ERROR    151          ;SETUP TIMEOUT
4970 025510 004737 037024          JSR      PC,TSTATN
4971 025514 000401          BR       73$
4972 025516 104154          ERROR    154          ;FIND RDY
4973                                     ;NO RDY AFTER DRIVE CLEAR CMD
4974                                     ;TEST FOR ATTN
4975                                     ;ATTN NOT CLEARED AFTER DRIVE CLEAR CMD
4976 025520 005201          INC      R1          ;HEAD CTR
4977 025522 020127 000003      CMP      R1,#3          ;SEE IF ALL HEADS DONE
4978 025526 001402          BEQ     T$15          ;:BR IF YES
4979 025530 000137 024602          JMP     9$          ;ELSE REPEAT ALL FOR NEXT HEAD
4980 025534          10$:
4981          ;*****
4982          ;*TEST 15 WRITE WITH HEADS OFFSET
4983          ;*
4984          ;* THIS TEST VERIFIES THAT WHEN ATTEMPTING TO
4985          ;* WRITE WITH HEADS OFFSET THAT THE OFFSET WILL CLEAR
4986          ;* & THE DRIVE WILL WRITE
4987          ;* SINCE THE WRITE COMMAND HAS AN IMPLIED RTC.
4988          ;* THIS TEST IS PERFORMED FOR MAX POS & NEG OFFSETS ONLY
4989          ;*
4990          ;*****
4991 025534 000004          T$15: SCOPE
4992 025536 012737 000001 001174      MOV      #1,$TIMES          ;:DO 1 ITERATION
4993 025544 012706 001100          MOV      #STACK,SP          ;RESTORE STK PTR
4994
4995 025550 012700 000260          MOV      #260,R0          ;MAX NEG OFFSET
4996

```

H08

UNIBUS Rk06 DRIVE DIAGNOSTIC PART 2
DZR6IC.P11 07-OCT-76 13:50

MACY11 27(1006) 07-OCT-76 14:14 PAGE 98
T15 WRITE WITH HEADS OFFSET

SEG 0098

4997	025554	004737	040452	15:	JSR	PC, SUBCLR	
4998	025560	104024			ERROR	24	;CERR AFTER SCLR
4999							
5000	025562	010065	000016		MOV	RD, RKASOF(R5)	;SET OFFSET
5001							
5002	025566	012737	025662	001176	MOV	#64\$, \$ESCAPE	
5003	025574	012765	000015	000000	MOV	#OFFSET, RKCS1(R5)	;OFFSET CMD
5004	025602	013737	001420	007426	MOV	T100, TEMP1	;SETUP TIMEOUT
5005	025610	004737	036542		JSR	PC, FRDY	
5006	025614	104033			ERROR	33	;NO RDY AFTER OFFSET CMD
5007							
5008	025616	012737	032140	007460	MOV	#<D.PIP!D.SPIN!D.OFF!D.VV!D.DRA>, E.A0	;EXPECTED MSG A0
5009	025624	005037	007462		CLR	E.B0	
5010	025630	012737	001720	007464	MOV	#<D.SPOK!D.CART!D.DOOR!D.BRHM!D.SSP>, E.A1	
5011	025636	012737	000001	007466	MOV	#1, E.B1	
5012							
5013	025644	004737	037264		JSR	PC, CHKMSG	;CHECK MSGS A0, B0, A1, B1
5014	025650	000000			.WORD	0!0!0	; & MSGS SPECIFIED HERE
5015	025652	104035			ERROR	35	;MSG A0 ERROR DURING OFFSET CMD
5016	025654	104061			ERROR	61	;MSH B0 ERROR
5017	025656	104036			ERROR	36	;MSG A1 ERROR
5018	025660	104062			ERROR	62	;MSG B1 ERROR
5019							
5020	025662	005037	001176		CLR	\$ESCAPE	
5021	025666	013737	001422	007426	MOV	T5000, TEMP1	;SETUP TIMEOUT
5022	025674	004737	037152		JSR	PC, FATT2	;FIND ATTN
5023	025700	104034			ERROR	34	;NO ATTN AFTER OFFSET CMD
5024							
5025							
5026	025702	012737	052340	007460	MOV	#<D.DSC!D.OFF!D.SPIN!D.DRDY!D.VV!D.DRA>, E.A0	;EXPECTED MSG A0
5027	025710	005037	007462		CLR	E.B0	;EXPECTED MSG B0
5028	025714	012737	001720	007464	MOV	#<D.SPOK!D.CART!D.DOOR!D.BRHM!D.SSP>, E.A1	;EXPECTED A1
5029	025722	012737	000001	007466	MOV	#1, E.B1	;MSG ID FOR EXPECTED MSG B1
5030	025730	005037	007470		CLR	E.A2	;EXPECTED MSG A2
5031	025734	012737	000002	007472	MOV	#2, E.B2	;MSG ID FOR EXPECTED MSG B2
5032	025742	012737	000003	007476	MOV	#3, E.B3	;MSG ID FOR EXPECTED MSG B3
5033							
5034	025750	004737	037264		JSR	PC, CHKMSG	;CHECK MSGS A0, B0, A1, B1
5035	025754	000003			.WORD	T.A2!T.B2!0	; & MSGS SPECIFIED HERE
5036	025756	104260			ERROR	260	;MSG A0 ERROR AFTER OFFSET CMD
5037	025760	104261			ERROR	261	;MSH B0 ERROR
5038	025762	104037			ERROR	37	;MSG A1 ERROR
5039	025764	104040			ERROR	40	;MSG B1 ERROR
5040							
5041							
5042	025766	012765	100000	000000	MOV	#CLR, RKCS1(R5)	
5043	025774	013765	001222	000010	MOV	\$UNIT, RKCS2(R5)	;DRIVE#
5044	026002	012765	000005	000000	MOV	#CLEAR, RKCS1(R5)	;DRIVE CLEAR CMD
5045	026010	013737	001412	007426	MOV	T10, TEMP1	;SETUP TIMEOUT
5046	026016	004737	036542		JSR	PC, FRDY	;FIND RDY
5047	026022	104151			ERROR	151	;NO RDY AFTER DRIVE CLEAR CMD
5048	026024	004737	037024		JSR	PC, TSTATN	;TEST FOR ATTN
5049	026030	000401			BR	65\$	
5050	026032	104154			ERROR	154	;ATTN NOT CLEARED AFTER DRIVE CLEAR CMD
5051	026034						
5052							

65\$:

JOB

JNIBJS RK06 DRIVE DIAGNOSTIC PART 2
DZR6IC.P11 07-OCT-76 13:50

MACY11 27(1006) 07-OCT-76 14:14 PAGE 100
T15 WRITE WITH HEADS OFFSET

SEG 0100

5109	026346	004737	037264		JSR	PC,CHKMSG		;CHECK MSGS A0, B0, A1, B1
5110	026352	000003			.WORD	T.A2!T.B2!0		; & MSGS SPECIFIED HERE
5111	026354	104052			ERROR	52		;MSG A0 ERROR AFTER WRITE DATA CMD
5112	026356	104023			ERROR	23		;MSH B0 ERROR
5113	026360	104053			ERROR	53		;MSG A1 ERROR
5114	026362	104025			ERROR	25		;MSG B1 ERROR
5115								
5116	026364	005737	001362		TST	CYLDIF		;SEE IF MSG A2=0
5117	026370	001401			BEQ	70\$;BR IF YES
5118	026372	104104			ERROR	104		;MSG A2 NOT CLEARED AFTER WRITE CMD WITH OFFSET
5119	026374	005737	001364	70\$:	TST	CYLADD		;SEE IF MSG B2=0
5120	026400	001401			BEQ	71\$;BR IF YES
5121	026402	104105			ERROR	105		;MSG B2 NOT CLEARED AFTER WRITE CMD WITH OFFSET
5122	026404			71\$:				
5123								
5124	026404	104415			SCOP1			
5125	026406	012706	001100		MOV	#STACK,SP		;RESTORE STK PTR
5126								
5127	026412	004737	040452		JSR	PC,SUBCLR		
5128	026416	104024			ERROR	24		;CERR AFTER SCLR
5129								
5130								
5131	026420	012765	001510	000004	MOV	#DATA0,RKBA(R5)		

K08

UNIBUS RK06 DRIVE DIAGNOSTIC PART 2
DZR6IC.P11 07-OCT-76 13:50

MACY11 27(1006) 07-OCT-76 14:14 PAGE 101
T15 WRITE WITH HEADS OFFSET

SEG 0101

5132	026426	052765	000020	000010	BIS	#BAI,RKCS2(R5)	
5133	026434	012765	177400	000002	MOV	#-256,RKWC(R5)	
5134	026442	013765	001406	000006	MOV	SECTOR,RKDA(R5)	
5135							
5136	026450	012765	000031	000000	MOV	#<WRTCHK>,RKCS1(R5)	;WRITE CHECK CMD
5137	026456	013737	001424	007426	MOV	T5000,TEMP1	;SETUP TIMEOUT
5138	026464	004737	036542		JSR	PC,FRDY	;FIND RDY
5139	026470	104015			ERROR	15	;NO RDY AFTER WRITE CHECK CMD
5140	026472	004737	040100		JSR	PC,GSTAT	;GET FRESH STATUS
5141	026476	032737	100000	007370	BIT	#CERR,HCS1	
5142	026504	001453			BEQ	73\$	
5143	026506	032737	040000	007372	BIT	#WCE,HCS2	;SEE IF WRITE CHECK ERROR
5144	026514	001410			BEQ	72\$	
5145	026516	016537	000024	001466	MOV	RKDB(R5),WD1	;ACTUAL WORD FOR PRINTOUT
5146	026524	013737	001510	001470	MOV	DATA0,WD2	;EXPECTED WORD FOR TYPEOUT
5147	026532	104016			ERROR	16	;WCE AFTER WRITE CMD
5148	026534	000437			BR	73\$	
5149							
5150	026536	104022			72\$: ERROR	22	;CERR AFTER WRITE CHECK CMD
5151							
5152	026540	012737	010340	007460	MOV	#<O!D.SPIN!D.DRDY!D.VV!D.DRA>,E.A0	;EXPECTED MSG A0
5153	026546	005037	007462		CLR	E.B0	;EXPECTED MSG B0
5154	026552	012737	001720	007464	MOV	#<D.SPOK!D.CART!D.DOOR!D.BRHM!D.SSP>,E.A1	;EXPECTED A1
5155	026560	012737	000001	007466	MOV	#1,E.B1	;MSG ID FOR EXPECTED MSG B1
5156	026566	005037	007470		CLR	E.A2	;EXPECTED MSG A2
5157	026572	012737	000002	007472	MOV	#2,E.B2	;MSG ID FOR EXPECTED MSG B2
5158	026600	012737	000003	007476	MOV	#3,E.B3	;MSG ID FOR EXPECTED MSG B3
5159							
5160	026606	004737	037264		JSR	PC,CHKMSG	;CHECK MSGS A0, B0, A1, B1
5161	026612	000003			.WORD	T.A2!T.B2!0	; & MSGS SPECIFIED HERE
5162	026614	104057			ERROR	57	;MSG A0 ERROR AFTER WRITE CHECK CMD
5163	026616	104031			ERROR	31	;MSH B0 ERROR
5164	026620	104060			ERROR	60	;MSG A1 ERROR
5165	026622	104032			ERROR	32	;MSG B1 ERROR
5166	026624	104401	053155		TYPE	MSG26	;ABORTING DATA TESTS
5167	026630	000137	031706		JMP	TIMING	
5168							
5169	026634				73\$:		
5170							
5171	026634	012737	010340	007460	MOV	#<O!D.SPIN!D.DRDY!D.VV!D.DRA>,E.A0	;EXPECTED MSG A0
5172	026642	005037	007462		CLR	E.B0	;EXPECTED MSG B0
5173	026646	012737	001720	007464	MOV	#<D.SPOK!D.CART!D.DOOR!D.BRHM!D.SSP>,E.A1	;EXPECTED A1
5174	026654	012737	000001	007466	MOV	#1,E.B1	;MSG ID FOR EXPECTED MSG B1
5175	026662	005037	007470		CLR	E.A2	;EXPECTED MSG A2
5176	026666	012737	000002	007472	MOV	#2,E.B2	;MSG ID FOR EXPECTED MSG B2
5177	026674	012737	000003	007476	MOV	#3,E.B3	;MSG ID FOR EXPECTED MSG B3
5178							
5179	026702	004737	037264		JSR	PC,CHKMSG	;CHECK MSGS A0, B0, A1, B1
5180	026706	000003			.WORD	T.A2!T.B2!0	; & MSGS SPECIFIED HERE
5181	026710	104057			ERROR	57	;MSG A0 ERROR AFTER WRITE CHECK CMD
5182	026712	104031			ERROR	31	;MSH B0 ERROR
5183	026714	104060			ERROR	60	;MSG A1 ERROR
5184	026716	104032			ERROR	32	;MSG B1 ERROR
5185							

```

5186 026720 020027 000260      CMP      RO,#260
5187 026724 001004                BNE      2$      ;BR IF JUST DID POS OFFSET
5188 026726 012700 000060      MOV      #60,RO  ;ELSE SETUP FOR POS OFFSET
5189 026732 000137 025554      JMP
5190
5191 026736                2$:
5192
5193 026736 012765 100000 000000      MOV      #CCLR,RKCS1(R5)
5194 026744 013765 001222 000010      MOV      $UNIT,RKCS2(R5)
5195 026752 012765 000013 000000      MOV      #RECAL,RKCS1(R5)      ;RECAL CMD
5196                                ;RESET CYL DIFF/OFFSET & CYL ADDR REG
5197                                ;IN RKMR2 & RKMR3 RESP.
5198 026760 013737 001412 007426      MOV      T10,TEMP1      ;SETUP TIMEOUT
5199 026766 004737 036542                JSR      PC,FRDY      ;FIND RDY
5200 026772 104124                ERROR    124      ;RDY NOT SET AFTER RECAL CMD
5201
5202 026774 012765 000001 000026      MOV      #1,RKMR1(R5)      ;SELECT WORD 1
5203 027002 004737 040100                JSR      PC,GSTAT
5204 027006 032737 020000 007416      BIT      #D.RTZ,HMR2
5205 027014 001001                BNE      74$
5206 027016 104244                ERROR    244      ;RTZ NOT SET DURING RECAL CMD
5207 027020 013737 001412 007430 74$:      MOV      T10,TEMP2      ;SETUP TIMEOUT
5208 027026 004737 037056                JSR      PC,FATT1      ;FIND ATTN
5209 027032 104055                ERROR    55      ;NO ATTN AFTER RECAL CMD
5210
5211 027034 012765 100000 000000      MOV      #CCLR,RKCS1(R5)
5212 027042 013765 001222 000010      MOV      $UNIT,RKCS2(R5)      ;DRIVE#
5213 027050 012765 000005 000000      MOV      #CLEAR,RKCS1(R5)      ;DRIVE CLEAR CMD
5214 027056 013737 001412 007426      MOV      T10,TEMP1      ;SETUP TIMEOUT
5215 027064 004737 036542                JSR      PC,FRDY      ;FIND RDY
5216 027070 104151                ERROR    151      ;NO RDY AFTER DRIVE CLEAR CMD
5217 027072 004737 037024                JSR      PC,TSTATN      ;TEST FOR ATTN
5218 027076 000401                BR       75$
5219 027100 104154                ERROR    154      ;ATTN NOT CLEARED AFTER DRIVE CLEAR CMD
5220 027102                75$:
5221
5222
5223 027102                3$:
5224

```

5225
5226
5227
5228
5229
5230
5231
5232
5233
5234
5235
5236
5237
5238
5239
5240
5241 027102 000004
5242 027104 012737 000001 001174
5243 027112 012706 001100
5244
5245 027116 012700 001426
5246
5247 027122 004737 040452
5248 027126 104024
5249
5250 027130 005037 001406
5251 027134 013765 001406 000006
5252
5253 027142 011065 000020
5254 027146 012765 001516 000004
5255 027154 052765 000020 000010
5256 027162 012765 076000 000002
5257
5258 027170 012765 000023 000000
5259 027176 013737 001424 007426
5260 027204 004737 036542
5261 027210 104011
5262 027212 004737 040100
5263 027216 032737 100000 007370
5264 027224 001465
5265
5266 027226 032737 000200 007404
5267 027234 001421
5268 027236 004737 042122
5269 027242 000455
5270
5271 027244 005237 001406
5272 027250 023727 001406 000012
5273 027256 001003
5274 027260 104046
5275 027262 000137 030146
5276 027266 012765 100000 000000 64\$:
5277 027274 000137 027134
5278 027300 104012 65\$:
5279
5280 027302 012737 010340 007460

```

*****
*TEST 16      TEST CURRENT CROSS-OVER CYLINDERS
*
*   THIS TEST VERIFIES THAT THE DRIVE CAN WRITE & READ OFF
*   CURRENT CHANGE CYLINDERS X & Y IN THE FOLLOWING WAY:
*
*   SPIRAL WRITING IS PERFORMED FROM CYLINDER X TO CYLINDER Y
*   WITH A DATA PATTERN FILLING THE ENTIRE 2 CYLINDERS.
*
*   A WRITE CHECK IS THEN PREFORMED TO VERIFY DATA WAS PROPERLY WRITTEN.
*   THIS TEST IS PERFORMED FOR ALL 3 HEADS.
*
*   CYLINDER X:  63 127 191 255 319 383
*   CYLINDER Y:  64 128 192 256 320 384
*
*****
TST16: SCOPE
MOV     #1,STIMES      ;;DO 1 ITERATION
MOV     #STACK,SP
MOV     #CYL,RO        ;CYL ADDR TABLE
1$:     JSR     PC,SUBCLR ;CERR AFTER SCLR
        ERROR    24
5$:     CLR     SECTOR
        MOV     SECTOR,RKDA(R5)
        MOV     (RO),RKDC(R5) ;CYL #
        MOV     #DPAT1,RKBA(R5) ;DATA PATTERN
        BIS     #BAI,RKCS2(R5) ;BUSS ADDR INCREMENT INHIBIT
        MOV     #-6*22.*256.,RKWC(R5) ;WORD COUNT TO SPIRAL & FILL 2 CYLINDERS
        MOV     #<WRDATA>,RKCS1(R5) ;WRITE DATA CMD
        MOV     T50000,TEMP1 ;SETUP TIMEOUT
        JSR     PC,FRDY ;FIND RDY
        ERROR   11 ;NO RDY AFTER WRITE DATA CMD
        JSR     PC,GSTAT ;GET FRESH STATUS
        BIT     #CERR,HCS1
        BEQ     67$ ;BR IF NO ERRORS
        BIT     #BSE,HER ;SEE IF BAD SECTOR FLAG
        BEQ     65$ ;BR IF NO
        JSR     PC,TRUERR ;ELSE SEE IF SECTOR LISTED IN BSE TABLE
        BR     66$ ;RETURN HERE IF NO
        INC     SECTOR ;RETURN HERE IF YES
        CMP     SECTOR,#10. ;ARE 10 CONSEC. SECTORS BAD
        BNE     64$ ;BR IF NO
        ERROR   46 ;ABORTING TEST DETECTED 10 BAD SECTORS
        JMP     2$ ;BYPASS TEST
        MOV     #CCLR,RKCS1(R5) ;TRY ANOTHER SECTOR
        JMP     5$
        ERROR   12 ;CERR WITH WRITE DATA CMD
        MOV     #<D!D.SPIN!D.DRDY!D.VV!D.DRA>,E.AO ;EXPECTED MSG AO

```

N08

UNIBUS RK06 DRIVE DIAGNOSTIC PART 2
DZR6IC.P11 07-OCT-76 13:50

MACY11 27(1006) 07-OCT-76 14:14 PAGE 104
T16 TEST CURRENT CROSS-OVER CYLINDERS

SEQ 0104

5281	027310	005037	007462		CLR	E.B0	; EXPECTED MSG B0
5282	027314	012737	001720	007464	MOV	#<D.SPOK!D.CART!D.DOOR!D.BRHM!D.SSP>,E.A1	; EXPECTED A1
5283	027322	012737	000001	007466	MOV	#1,E.B1	; MSG ID FOR EXPECTED MSG B1
5284	027330	005037	007470		CLR	E.A2	; EXPECTED MSG A2
5285	027334	012737	000002	007472	MOV	#2,E.B2	; MSG ID FOR EXPECTED MSG B2
5286	027342	012737	000003	007476	MOV	#3,E.B3	; MSG ID FOR EXPECTED MSG B3
5287							
5288	027350	004737	037264		JSR	PC CHKMSG	; CHECK MSGS A0, B0, A1, B1
5289	027354	000003			.WORD	T.A2!T.B2!0	; & MSGS SPECIFIED HERE
5290	027356	104052			ERROR	52	; MSG A0 ERROR AFTER WRITE DATA CMD
5291	027360	104023			ERROR	23	; MSH B0 ERROR
5292	027362	104053			ERROR	53	; MSG A1 ERROR
5293	027364	104025			ERROR	25	; MSG B1 ERROR
5294	027366	104401	053155		TYPE	MSG26	; ABORTING DATA TESTS TO DO TIMING
5295	027372	000137	031706		JMP	TIMING	
5296	027376	104063			ERROR	63	; BAD SECTOR NOT LISTED IN TABLE
5297	027400						
5298							
5299	027400	012737	010340	007460	MOV	#<O!D.SPIN!D.DRDY!D.VV!D.DRA>,E.A0	; EXPECTED MSG A0
5300	027406	005037	007462		CLR	E.B0	; EXPECTED MSG B0
5301	027412	012737	001720	007464	MOV	#<D.SPOK!D.CART!D.DOOR!D.BRHM!D.SSP>,E.A1	; EXPECTED A1
5302	027420	012737	000001	007466	MOV	#1,E.B1	; MSG ID FOR EXPECTED MSG B1
5303	027426	005037	007470		CLR	E.A2	; EXPECTED MSG A2
5304	027432	012737	000002	007472	MOV	#2,E.B2	; MSG ID FOR EXPECTED MSG B2
5305	027440	012737	000003	007476	MOV	#3,E.B3	; MSG ID FOR EXPECTED MSG B3
5306							
5307	027446	004737	037264		JSR	PC CHKMSG	; CHECK MSGS A0, B0, A1, B1
5308	027452	000003			.WORD	T.A2!T.B2!0	; & MSGS SPECIFIED HERE
5309	027454	104052			ERROR	52	; MSG A0 ERROR AFTER WRITE DATA CMD
5310	027456	104023			ERROR	23	; MSH B0 ERROR
5311	027460	104053			ERROR	53	; MSG A1 ERROR
5312	027462	104025			ERROR	25	; MSG B1 ERROR
5313							
5314	027464	011065	000020		MOV	(R0),RKDC(R5)	
5315	027470	013765	001406	000006	MOV	SECTOR,RKDA(R5)	
5316	027476	012765	001516	000004	MOV	#DPAT1,RKBA(R5)	

66S:
67S:

```

5317 027504 052765 000020 000010      BIS      #BA1,RKCS2(R5)
5318 027512 012765 076000 000002      MOV      #-6*22,*256.,RKWC(R5)
5319
5320 027520 012765 000031 000000      MOV      #WATCH,RKCS1(R5) ;WRITE CHECK CMD
5321 027526 013737 001424 007426      MOV      T50000,TEMP1 ;SETUP TIMEOUT
5322 027534 004737 036542      JSR      PC,FRDY ;FIND RDY
5323 027540 104015      ERROR   15 ;NO RDY AFTER WRITE CHECK CMD
5324 027542 004737 040100      JSR      PC,GSTAT ;GET FRESH STATUS
5325 027546 032737 100000 007370      BIT      #CERR,HCS1
5326 027554 001453      BEQ     69$
5327 027556 032737 040000 007372      BIT      #WCE,HCS2 ;SEE IF WRITE CHECK ERROR
5328 027564 001410      BEQ     68$
5329 027566 016537 000024 001466      MOV      RKDB(R5),WD1 ;ACTUAL WORD FOR PRINTOUT
5330 027574 013737 001516 001470      MOV      DPAT1,WD2 ;EXPECTED WORD FOR TYPEOUT
5331 027602 104016      ERROR   16 ;WCE AFTER WRITE CMD
5332 027604 000437      BR      69$
5333
5334 027606 104022      69$:   ERROR   22 ;CERR AFTER WRITE CHECK CMD
5335
5336 027610 012737 010340 007460      MOV      #<0!D.SPIN!D.DRDY!D.VV!D.DRA>,E.A0 ;EXPECTED MSG A0
5337 027616 005037 007462      CLR     E.B0 ;EXPECTED MSG B0
5338 027622 012737 001720 007464      MOV      #<D.SPOK!D.CART!D.DOOR!D.BRHM!D.SSP>,E.A1 ;EXPECTED A1
5339 027630 012737 000001 007466      MOV      #1,E.B1 ;MSG ID FOR EXPECTED MSG B1
5340 027636 005037 007470      CLR     E.A2 ;EXPECTED MSG A2
5341 027642 012737 000002 007472      MOV      #2,E.B2 ;MSG ID FOR EXPECTED MSG B2
5342 027650 012737 000003 007476      MOV      #3,E.B3 ;MSG ID FOR EXPECTED MSG B3
5343
5344 027656 004737 037264      JSR      PC,CHKMSG ;CHECK MSGS A0, B0, A1, B1
5345 027662 000003      .WORD   T.A2!T.B2!0 ;& MSGS SPECIFIED HERE
5346 027664 104057      ERROR   57 ;MSG A0 ERROR AFTER WRITE CHECK CMD
5347 027666 104031      ERROR   31 ;MSG B0 ERROR
5348 027670 104060      ERROR   60 ;MSG A1 ERROR
5349 027672 104032      ERROR   32 ;MSG B1 ERROR
5350 027674 104401 053155      TYPE    MSG26 ;ABORTING DATA TESTS
5351 027700 000137 031706      JMP     TIMING
5352
5353 027704      69$:
5354
5355 027704 012737 010340 007460      MOV      #<0!D.SPIN!D.DRDY!D.VV!D.DRA>,E.A0 ;EXPECTED MSG A0
5356 027712 005037 007462      CLR     E.B0 ;EXPECTED MSG B0
5357 027716 012737 001720 007464      MOV      #<D.SPOK!D.CART!D.DOOR!D.BRHM!D.SSP>,E.A1 ;EXPECTED A1
5358 027724 012737 000001 007466      MOV      #1,E.B1 ;MSG ID FOR EXPECTED MSG B1
5359 027732 005037 007470      CLR     E.A2 ;EXPECTED MSG A2
5360 027736 012737 000002 007472      MOV      #2,E.B2 ;MSG ID FOR EXPECTED MSG B2
5361 027744 012737 000003 007476      MOV      #3,E.B3 ;MSG ID FOR EXPECTED MSG B3
5362
5363 027752 004737 037264      JSR      PC,CHKMSG ;CHECK MSGS A0, B0, A1, B1
5364 027756 000003      .WORD   T.A2!T.B2!0 ;& MSGS SPECIFIED HERE
5365 027760 104057      ERROR   57 ;MSG A0 ERROR AFTER WRITE CHECK CMD
5366 027762 104031      ERROR   31 ;MSG B0 ERROR
5367 027764 104060      ERROR   60 ;MSG A1 ERROR
5368 027766 104032      ERROR   32 ;MSG B1 ERROR
5369 027770 022027 000577      3$:   CMP     (R0)+,#383. ;ALL CYLINDERS DONE?
5370 027774 001402      BEQ     4$ ;BR IF YES
5371 027776 000137 027122      JMP     1$ ;ELSE REPEAT
5372

```

```

5373 030002 45:
5374
5375 030002 012765 100000 000000 MOV #CCLR,RKCS1(R5)
5376 030010 013765 001222 000010 MOV #UNIT,RKCS2(R5)
5377 030016 012765 000013 000000 MOV #RECCL,RKCS1(R5) ;RECCL CMD
5378 ;RESET CYL DIFF/OFFSET & CYL ADDR REG
5379 ;IN RKMR2 & RKMR3 RESP.
5380 030024 013737 001412 007426 MOV T10,TEMP1 ;SETUP TIMEOUT
5381 030032 004737 036542 JSR PC,FRDY ;FIND RDY
5382 030036 104124 ERROR 124 ;RDY NOT SET AFTER RECCL CMD
5383
5384 030040 012765 000001 000026 MOV #1,RKMR1(R5) ;SELECT WORD 1
5385 030046 004737 040100 JSR PC,GSTAT
5386 030052 032737 020000 007416 BIT #D,RTZ,HMR2
5387 030060 001001 BNE 705
5388 030062 104244 ERROR 244 ;RTZ NOT SET DURING RECCL CMD
5389 030064 013737 001412 007430 705: MOV T10,TEMP2 ;SETUP TIMEOUT
5390 030072 004737 037056 JSR PC,FATT1 ;FIND ATTN
5391 030076 104055 ERROR 55 ;NO ATTN AFTER RECCL CMD
5392
5393 030100 012765 100000 000000 MOV #CCLR,RKCS1(R5)
5394 030106 013765 001222 000010 MOV #UNIT,RKCS2(R5) ;DRIVE#
5395 030114 012765 000005 000000 MOV #CLEAR,RKCS1(R5) ;DRIVE CLEAR CMD
5396 030122 013737 001412 007426 MOV T10,TEMP1 ;SETUP TIMEOUT
5397 030130 004737 036542 JSR PC,FRDY ;FIND RDY
5398 030134 104151 ERROR 151 ;NO RDY AFTER DRIVE CLEAR CMD
5399 030136 004737 037024 JSR PC,TSTATN ;TEST FOR ATTN
5400 030142 000401 BR 715
5401 030144 104154 ERROR 154 ;ATTN NOT CLEARED AFTER DRIVE CLEAR CMD
5402
5403
5404
5405 030146 715:
5406
5407
5408
5409
5410
5411
5412
5413
5414
5415
5416
5417
5418
5419
5420
5421
5422
5423
5424
5425
5426
5427
5428
5429
5430
5431
5432
5433
5434
5435
5436
5437
5438
5439
5440
5441
5442
5443
5444
5445
5446
5447
5448
5449
5450
5451
5452
5453
5454
5455
5456
5457
5458
5459
5460
5461
5462
5463
5464
5465
5466
5467
5468
5469
5470
5471
5472
5473
5474
5475
5476
5477
5478
5479
5480
5481
5482
5483
5484
5485
5486
5487
5488
5489
5490
5491
5492
5493
5494
5495
5496
5497
5498
5499
5500

```

```

25:
*****
TEST 17 TEST HEAD SWITCHING TIME
*
* TESTS THE ABILITY TO SWITCH HEADS IN LESS THEN 10MS WHEN HEADS SPIRAL.
*
* 1. SECTOR 17 IS FIRST LOCATED AND A WRITE DATA COMMAND OF 512 WORDS
* TO SECTOR 21 IS ISSUED.
* 2. THE PROGRAM NOW KNOWS THAT THE DRIVE WILL NOT HAVE TO TRAVEL
* A FULL REVOLUTION BEFORE FINDING SECTOR 21.
* 3. SINCE EACH SECTOR TAKES APPROX. 1.2MS, THE TIME BETWEEN
* THE START OF THE WRITE COMMAND (FROM SECTOR 21, HEAD 0; TO
* SECTOR 0, HEAD 1) AND CONTROLLER READY SHOULD BE APPROX 6MS
*
* THE ABOVE IS REPEATED FOR HEAD SWITCHING BETWEEN 1 TO 2
*
* THIS TEST IS BYPASSED IF NEITHER L OR P CLOCK IS PRESENT
*
*****

```

```

5499 030146 000004 ST17: SCOPE
5500 030150 012737 000001 001174 MOV #1,STIMES ;:DO 1 ITERATION
5501 030156 012706 001100 MOV #STACK,SP

```

469	030162	005737	007536		TST	DO TIM	: BYPASS THIS TEST IF
470	030166	001001			BNE	18	: NEITHER L OR P CLOCK PRESENT
471	030170	000520			BR	*ST20	: GO TO NEXT TEST
472	030172	012737	000025	007436	18:	MOV	#25, TEMPS
473	030200	004737	040452		25:	JSR	PC, SUBCLR
474	030204	104024				ERROR	24
475							: CERR AFTER SOLR
476	030206	004737	041222			JSR	PC, FSEC23
477	030212	104106				ERROR	106
478	030214	000506				BR	TST20
479							: FIND SECTOR 23
480							: CANNOT FIND SECTOR 23
481							: GO TO NEXT TEST
482	030216	012765	001512	000004		MOV	#DATA01, RKBA(R5)
483	030224	052765	000020	000010		BIS	#BA1, RKCS2(R5)
484	030232	012765	177000	000002		MOV	#-512, RKWC(R5)
485	030240	013765	007436	000006		MOV	TEMPS, RKDA(R5)
486	030246	012765	000023	000000		MOV	#WRDATA, RKCS1(R5)
487							: DATA 0101
488							: BUSS ADDR INCREMENT INHIBIT
489							: WORD COUNT
490							: HEAD & SECTOR
491							: WRITE DATA CMD
492	030254	012727	002000	007426		MOV	#2000, #TEMP1
493	030262	004737	037232			JSR	PC, DLY
494							: DO DELAY
495	030266	032765	000200	000000	75:	BIT	#RDY, RKCS1(R5)
496	030274	001006				BNE	85
497	030276	004737	036542			JSR	PC, FRDY
498	030302	104011				ERROR	11
499	030304	004737	040100			JSR	PC, GSTAT
500	030310	104107				ERROR	107
501							: LOOK FOR CONTROLLER READY
502							: FIND RDY AND GET FRESH STATUS
503							: NO RDY AFTER SEL DRV CMD
504							: HEAD SWITCHING LONGER THAN DELAY
505	030312	032765	100000	000000	85:	BIT	#CERR, RKCS1(R5)
506	030320	001444				BEG	35
507	030322	104012				ERROR	12
508							: CERR AFTER WRITE DATA
509	030324	012737	010340	007460		MOV	#<0!D.SPIN!D.DRDY!D.VV!D.DRA>, E.A0
510	030332	005037	007462			CLR	E.B0
511	030336	012737	001720	007464		MOV	#<D.SPOK!D.CART!D.DOOR!D.BRHM!D.SSP>, E.A1
512	030344	012737	000001	007466		MOV	#1, E.B1
513	030352	005037	007470			CLR	E.A2
514	030356	012737	000002	007472		MOV	#2, E.B2
515	030364	012737	000003	007476		MOV	#3, E.B3
516							: EXPECTED MSG A0
517							: EXPECTED MSG B0
518							: EXPECTED A1
519							: MSG ID FOR EXPECTED MSG B1
520							: EXPECTED MSG A2
521							: MSG ID FOR EXPECTED MSG B2
522							: MSG ID FOR EXPECTED MSG B3
523	030372	004737	037264			JSR	PC, CHKMSG
524	030376	000000				.WORD	0!0!0
525	030400	104052				ERROR	52
526	030402	104023				ERROR	23
527	030404	104053				ERROR	53
528	030406	104025				ERROR	25
529							: CHECK MSGS A0, B0, A1, B1
530							: & MSGS SPECIFIED HERE
531							: MSG A0 ERROR AFTER WRITE DATA CMD
532							: MSG B0 ERROR
533							: MSG A1 ERROR
534							: MSG B1 ERROR
535	030410	023727	007436	000425		CMP	TEMPS, #425
536	030416	001405				BEG	TST20
537							: HEAD 1, SECTOR 21 DONE?
538							: GO TO NEXT TEST

030432 012737 000425 007436
030436 000437 030200
030432

MOV #425, TEMPS
JMP 25 ;ELSE REPEAT FOR HEAD 1. SECTOR 21

35:

:TEST 20 DRIVE OFF TRACK TEST

THIS TEST CHECKS FOR SERVO OSCILLATIONS DURING SETTling TIME BEYOND THE ALLOTTED 3MS.

- 1. INITIALLY, EVERY CYLINDER IS FORMATTED WITH IDENTICAL HEADERS (UNIQUE TO EACH CYLINDER)
- 2. A FULL SECTOR WRITE COMMAND IS ISSUED BY A SINGLE CYL SEEK FROM 0 TO 1. AS HEADERS ARE IDENTICAL, THE NEXT SECTOR TO COME UNDER THE HEADS WILL IMMEDIATELY BE WRITTEN.
- 3. IF THERE IS OSCILLATION SENSED BY READING THE TRIBITS, DRIVE OFF TRACK ERROR WILL SET.

IN THIS MANNER OSCILLATING SEEKS ARE PERFORMED BETWEEN ALL MAJOR CYLINDERS. 100 OSCILLATIONS ARE PERFORMED AT EACH MAJOR CYLINDER BEFORE DOING THE NEXT CYLINDER

5501 030432 000004
5502 030434 012737 000001 001174
5503 030442 012706 001100
5504 030446 005237 007354
5505 030452 005037 001352
5506 030456 012737 100000 007436
5507
5508 030464 004737 040452
5509 030470 104024
5510
5511 030472 012700 001536
5512
5513 030476 013720 001352
5514 030502 012720 140000
5515 030506 012710 140000
5516 030512 053720 001352
5517
5518 030516 020027 001742
5519 030522 001365
5520
5521 030524 012765 001536 000004
5522 030532 012765 177676 000002
5523 030540 013765 001352 000020
5524
5525 030546 012765 000027 000000
5526 030554 013737 001424 007426
5527 030562 004737 036542
5528 030566 104200
5529 030570 004737 040100
5530 030574 032737 100000 007370
5531 030602 001405
5532 030604 104201
5533 030606 104401 053155

5520: SCOPE
MOV #1, \$TIMES ;DO 1 ITERATION
MOV #STACK, SP ;RESTORE STK PTR
INC BADHDR ;USED FOR VALID HALT
CLR TOCYL
MOV #BIT15, TEMPS
15: JSR PC, SUBCLR ;CERR AFTER SCLR
ERROR 24
MOV #HDTAB, RD ;FORMAT HEADERS ON ALL MAJOR CYL.
25: MOV TOCYL, (RD)+ ;HEADER WORD 0: CYL #
MOV #140000, (RD)+ ;HEADER WORD 1: ALL SECTOR 0
MOV #140000, (RD) ;HEADER WORD 2: XOR OF 0 & 1
BIS TOCYL, (RD)+ ;ADD CYL # TO WORD 2
CMP RD, #HDTAB+132. ;ALL 22 SECTORS DONE? (22x6=132)
BNE 25 ;BR IF NO
MOV #HDTAB, RKBA(R5)
MOV #-66., RKWC(R5)
MOV TOCYL, RKDC(R5)
MOV #<WRHEAD>, RKCS1(R5) ;WRITE HEADER CMD
MOV T50000, TEMP1 ;SETUP TIMEOUT
JSR PC, FRDY ;FIND RDY
ERROR 200 ;NO RDY AFTER WRITE HEADER CMD
JSR PC, STAT ;GET FRESH STATUS
BIT #CERR, HCS1
BEQ 645
ERROR 201 ;CERR AFTER WRITE HEADER CMD
TYPE .MSG26 ;ABORTING DATA TESTS TO DO TIMING TESTS

5534	030612	000137	031706		JMP	TIMING	
5535	030616			64\$:			
5536							
5537							
5538	030616	006137	007436		ROL	TEMP5	;SET CARRY ONLY ONCE
5539	030622	006137	001352		ROL	TOCYL	;SELECT NEXT MAJOR CYL
5540	030626	023727	001352	001000	CMP	TOCYL,#1000	;ALL MAJOR CYL FORMATTED?
5541	030634	001313			BNE	1\$;BR IF NO
5542	030636	005065	000020		CLR	RKDC(R5)	;SETUP TO RETURN TO CYL 0
5543							
5544	030642	012765	000017	000000	MOV	#SEEK,RKCS1(R5)	;SEEK CMD
5545	030650	013737	001414	007426	MOV	T50,TEMP1	;SETUP TIMEOUT
5546	030656	004737	036542		JSR	PC,FRDY	;FIND RDY
5547	030662	104131			ERROR	131	;NO RDY AFTER SEEK CMD
5548							
5549	030664	013737	001424	007426	MOV	T50000,TEMP1	;SETUP TIMEOUT
5550	030672	004737	037152		JSR	PC,FATT2	;FIND ATTN
5551	030676	104132			ERROR	132	;NO ATTN AFTER SEEK CMD
5552							
5553	030700	032737	100000	007370	BIT	#CERR,HCS1	
5554	030706	001401			BEQ	65\$	
5555	030710	104210			ERROR	210	;CERR AFTER SEEK CMD
5556							
5557	030712			65\$:			
5558							
5559	030712	012737	031420	001176	MOV	#FORM,\$ESCAPE	
5560	030720	005000			CLR	RO	;ITERATION COUNTER
5561	030722	012737	000001	001352	MOV	#1,TOCYL	;SETUP TO CYL #
5562	030730	005037	001350		CLR	FRCYL	
5563							
5564	030734	104415			SCOPI		
5565	030736	012706	001100		MOV	#STACK,SP	;RESTORE STK PTR
5566							
5567	030742	013737	001352	001360	MOV	TOCYL,CALDIF	;SETUP FOR ERROR PRINTOUT
5568							
5569	030750	004737	040452		3\$:	JSR	PC,SUBCLR
5570	030754	104024			ERROR	24	;CERR AFTER SCLR,
5571							
5572	030756	013765	001352	000020	MOV	TOCYL,RKDC(R5)	;GO TO CYL #
5573	030764	012765	001514	000004	MOV	#DATA1,RKBA(R5)	;ALL 1'S
5574	030772	052765	000020	000010	BIS	#BAI,RKCS2(R5)	
5575	031000	012765	177400	000002	MOV	#-256.,RKWC(R5)	;SECTOR TO BE ALL 1'S
5576							
5577	031006	012765	000023	000000	MOV	#WRDATA,RKCS1(R5)	;WRITE DATA CMD
5578	031014	013737	001424	007426	MOV	T50000,TEMP1	
5579	031022	004737	036542		JSR	PC,FRDY	;FIND RDY
5580	031026	104011			ERROR	11	;NO RDY AFTER WRITE DATA CMD.
5581							
5582	031030	004737	040100		JSR	PC,GSTAT	;GET FRESH STATUS
5583	031034	032737	020000	007420	BIT	#D.DROT,HMR3	;SEE IF DRIVE OFF TRACK
5584	031042	001401			BEQ	5\$	
5585	031044	104112			ERROR	112	;DRIVE OFF TRACK AFTER WRITE DATA CMD
5586							
5587	031046	032737	100000	007370	5\$:	BIT	#CERR,HCS1
5588	031054	001401			BEQ	6\$	
5589	031056	104012			ERROR	12	;CERR SET AFTER WRITE DATA CMD

```

5590
5591 031060
5592
5593 031060 012737 010340 007460 MOV #<0!D.SPIN!D.DRDY!D.VV!D.DRA>,E.A0 ;EXPECTED MSG A0
5594 031066 005037 007462 CLR E.B0 ;EXPECTED MSG B0
5595 031072 012737 001720 007464 MOV #<D.SPOK!D.CART!D.DOOR!D.BRHM!D.SSP>,E.A1 ;EXPECTED A1
5596 031100 012737 000001 007466 MOV #1,E.B1 ;MSG ID FOR EXPECTED MSG B1
5597 031106 005037 007470 CLR E.A2 ;EXPECTED MSG A2
5598 031112 012737 000002 007472 MOV #2,E.B2 ;MSG ID FOR EXPECTED MSG B2
5599 031120 012737 000003 007476 MOV #3,E.B3 ;MSG ID FOR EXPECTED MSG B3
5600
5601 031126 004737 037264 JSR PC,CHKMSG ;CHECK MSGS A0, B0, A1, B1
5602 031132 000003 .WORD T.A2!T.B2!0 ;& MSGS SPECIFIED HERE
5603 031134 104052 ERROR 52 ;MSG A0 ERROR AFTER WRITE DATA CMD
5604 031136 104023 ERROR 23 ;MSH B0 ERROR
5605 031140 104053 ERROR 53 ;MSG A1 ERROR
5606 031142 104025 ERROR 25 ;MSG B1 ERROR
5607 031144 023737 001364 001352 CMP CYLADD,TOCYL
5608 031152 001401 BEQ 7$
5609 031154 104113 ERROR 113 ;CYL ADDR IN RKMR3 NOT = RKDC
5610
5611 031156
5612 031156 104415
5613 031160 012706 001100 SCOP1
5614 MOV #STACK,SP ;RESTORE STK PTR
5615 031164 004737 040452 JSR PC,SUBCLR
5616 031170 104024 ERROR 24 ;CERR AFTER SCLR
5617
5618
5619 031172 012765 001514 000004 MOV #DATA1,RKBA(R5)
5620 031200 052765 000020 000010 BIS #BA1,RKCS2(R5)
5621 031206 012765 177400 000002 MOV #-256.,RKWC(R5)
5622
5623 031214 012765 000023 000000 MOV #WRDATA,RKCS1(R5)
5624 031222 013737 001424 007426 MOV T50000,TEMP1
5625 031230 004737 036542 JSR PC,FRDY ;FIND RDY
5626 031234 104011 ERROR 11 ;NO RDY AFTER WRITE DATA CMD
5627
5628 031236 004737 040100 JSR PC,GSTAT ;GET FRESH STATUS
5629 031242 032737 020000 007420 BIT #D.DROT,HMR3
5630 031250 001401 BEQ 8$
5631 031252 104112 ERROR 112 ;DRIVE OFF TRACK AFTER WRITE DATA CMD
5632
5633 031254 032737 100000 007370 8$: BIT #CERR,HCS1
5634 031262 001401 BEQ 9$
5635 031264 104012 ERROR 12 ;CERR AFTER WRITE DATA CMD
5636
5637 031266
5638
5639 031266 012737 010340 007460 MOV #<0!D.SPIN!D.DRDY!D.VV!D.DRA>,E.A0 ;EXPECTED MSG A0
5640 031274 005037 007462 CLR E.B0 ;EXPECTED MSG B0
5641 031300 012737 001720 007464 MOV #<D.SPOK!D.CART!D.DOOR!D.BRHM!D.SSP>,E.A1 ;EXPECTED A1
5642 031306 012737 000001 007466 MOV #1,E.B1 ;MSG ID FOR EXPECTED MSG B1
5643 031314 005037 007470 CLR E.A2 ;EXPECTED MSG A2
5644 031320 012737 000002 007472 MOV #2,E.B2 ;MSG ID FOR EXPECTED MSG B2
5645 031326 012737 000003 007476 MOV #3,E.B3 ;MSG ID FOR EXPECTED MSG B3

```

```

5646
5647 031334 004737 037264 JSR PC,CHKMSG ;CHECK MSGS A0 B0 A1 B1
5648 031340 000003 .WORD T.A2!T.B2!0 ;& MSGS SPECIFIED HERE
5649 031342 104052 ERROR 52 ;MSG A0 ERROR AFTER WRITE DATA CMD
5650 031344 104023 ERROR 23 ;MSH B0 ERROR
5651 031346 104053 ERROR 53 ;MSG A1 ERROR
5652 031350 104025 ERROR 25 ;MSG B1 ERROR
5653 031352 005737 001364 TST CYLADD
5654 031356 001401 BEQ 10$
5655 031360 104042 ERROR 42 ;NOT BACK TO CYL 0
5656
5657 031362 005200 10$: INC R0
5658 031364 020027 000144 CMP R0,#100. ;ALL ITERATIONS DONE?
5659 031370 001402 BEQ 13$ ;BR IF YES
5660 031372 000137 030750 JMP 3$ ;ELSE DO AGAIN
5661
5662 031376 005000 13$: CLR R0 ;RESET ITERATION CTR
5663 031400 006337 001352 ASL TOCYL
5664 031404 023727 001352 001000 CMP TOCYL,#1000 ;ALL MAJOR CYL DONE?
5665 031412 001402 BEQ FORM ;BR IF YES
5666 031414 000137 030750 JMP 3$ ;ELSE DO NEXT CYL
5667
5668 031420 005037 001352 FORM: CLR TOCYL ;RESTORE TO ORIG 22 SECTOR FORMAT.
5669 031424 012737 100000 007436 MOV #BIT15,TEMPS
5670
5671 031432 004737 040452 12$: JSR PC,SUBCLR
5672 031436 104024 ERROR 24 ;CERR AFTER SCLR
5673
5674 031440 012765 001536 000004 MOV #HDTAB,RKBA(R5) ;REFORMAT ALL MAJOR CYLINDERS
5675 031446 012765 177676 000002 MOV #-66.,RKWC(R5)
5676 031454 013765 001352 000020 MOV TOCYL,RKDC(R5)
5677
5678 031462 013737 001352 001366 MOV TOCYL,CALADD ;SETUP
5679 031470 012737 000000 001474 MOV #0,HEAD ;TO FILL
5680 031476 012737 000000 001502 MOV #0,FORMAT ;HEADER
5681 031504 004737 041434 JSR PC,FHDTAB ;TABLE
5682
5683
5684 031510 012765 000027 000000 MOV #<WRHEAD>,RKCS1(R5) ;WRITE HEADER CMD
5685 031516 013737 001424 007426 MOV T5000,TEMP1 ;SETUP TIMEOUT
5686 031524 004737 036542 JSR PC,FRDY ;FIND RDY
5687 031530 104200 ERROR 200 ;NO RDY AFTER WRITE HEADER CMD
5688 031532 004737 040100 JSR PC,GSTAT ;GET FRESH STATUS
5689 031536 032737 100000 007370 BIT #CERR,HCS1
5690 031544 001405 BEQ 64$
5691 031546 104201 ERROR 201 ;CERR AFTER WRITE HEADER CMD
5692 031550 104401 053155 TYPE MSG26 ;ABORTING DATA TESTS TO DO TIMING TESTS
5693 031554 000137 031706 JMP TIMING
5694
5695
5696
5697 031560 006137 007436 64$: ROL TEMPS
5698 031564 006137 001352 ROL TOCYL
5699 031570 023727 001352 001000 CMP TOCYL,#1000 ;ALL MAJOR CYL REFORMATTED?
5700 031576 001315 BNE 12$ ;BR IF NO
5701

```

```

5702 031600 005065 000020 CLR RkDC(R5) ;SETUP TO RETURN TO CYL 0
5703 031604 005037 001176 CLR $ESCAPE
5704
5705 031610 012765 000017 000000 MOV #SEEK,RKCS1(R5) ;SEEK CMD
5706 031616 013737 001414 007426 MOV T50,TEMP1 ;SETUP TIMEOUT
5707 031624 004737 036542 JSR PC,FRDY ;FIND RDY
5708 031630 104131 ERROR 131 ;NO RDY AFTER SEEK CMD
5709
5710 031632 013737 001424 007426 MOV T50000,TEMP1 ;SETUP TIMEOUT
5711 031640 004737 037152 JSR PC,FAT2 ;FIND ATTN
5712 031644 104132 ERROR 132 ;NO ATTN AFTER SEEK CMD
5713
5714 031646 032737 100000 007370 BIT #CERR,HCS1
5715 031654 001401 BEQ 65$
5716 031656 104210 ERROR 210 ;CERR AFTER SEEK CMD

```

65\$:

```

5717
5718 031660
5719
5720
5721 031660 005737 007356 TST HPEND ;SEE IF HALT PENDING
5722 031664 001406 BEQ 4$ ;BR IF NO
5723 031666 005037 007356 CLR HPEND ;CLEAR FOR FUTURE FORMATTING
5724 031672 005037 007354 CLR BADHDR ;HEADERS NOW OK
5725 031676 000137 043460 JMP STOP ;GO BACK & HALT CPU
5726

```

4\$: CLR BADHDR ;HEADERS NOW OK

.SBTTL SERVO & SPINDLE TIMING TESTS

TIMING:

```

5727 031702 005037 007354
5728
5729
5730
5731 031706
5732
5733
5734
5735
5736
5737
5738
5739
5740
5741
5742
5743
5744
5745

```

```

*****
;TEST 21 TIME BETWEEN OUTER LIMIT TO HEADS HOME DURING UNLOAD
;
; TIME IS MEASURED FROM ATTN ASSERTING (APPROXIMATES REV & OUTER LIMIT)
; TO HEADS HOME ASSERTING. EXPECTED TIME APPROX 285MS
;
; ALL TIMING TESTS ARE BYPASSED IF NEITHER
; L OR P CLOCK IS PRESENT & WILL BE INDICATED BY A MESSAGE
;
*****

```

```

5746 031706 000004 TST21: SCOPE
5747 031710 012737 000001 001174 MOV #1,$TIMES ;;DO 1 ITERATION
5748 031716 012706 001100 MOV #STACK,SP
5749
5750 031722 005737 001342 TST BYPTIM ;SEE IF BYPASS TIMING TESTS
5751 031726 001404 BEQ TIME1 ;BR IF NO
5752 031730 104401 054456 TYPE MSG41 ;BYPASS TIMING TESTS
5753 031734 000137 036034 JMP $EOP
5754 031740
5755 031740 005737 007536 TIME1: TST DOTIM
5756 031744 001004 BNE 1$
5757 031746 104401 052212 TYPE ,MSG13 ;TIMING TESTS BYPASSED

```

```

5758 031752 000137 036034      JMP      $EOP
5759
5760 031756 004737 040452      1$:    JSR      PC,SUBCLR
5761 031762 104024          ERROR    24          ;CERR AFTER SCLR
5762
5763 031764 104401 054515      TYPE    ,MSG42      ;ALL TIMING TESTS BYPASSED
5764 031770 000137 036034      JMP      $EOP
5765
5766 031774 004737 042262      JSR      PC,CALCLK   ;CALIB TIME TO GO THRU 'LOOP'
5767 032000 005237 007352      INC     UNLD        ;USED FOR VALID HALT
5768
5769 032004 012737 032176 001176      MOV     #4$, $ESCAPE
5770 032012 012765 000007 000000      MOV     #UNLOAD,RKCS1(R5) ;UNLOAD CMD
5771 032020 013737 001412 007426      MOV     T10,TEMP1    ;SETUP TIMEOUT
5772 032026 004737 037152      JSR     PC,FATT2     ;FIND ATTN
5773 032032 104073          ERROR    73          ;NO ATTN AFTER UNLOAD CMD
5774
5775
5776 032034 012765 100000 000000      MOV     #CCLR,RKCS1(R5)
5777 032042 013765 001222 000010      MOV     $UNIT,RKCS2(R5) ;DRIVE#
5778 032050 012765 000005 000000      MOV     #CLEAR,RKCS1(R5) ;DRIVE CLEAR CMD
5779 032056 013737 001412 007426      MOV     T10,TEMP1    ;SETUP TIMEOUT
5780 032064 004737 036542      JSR     PC,FRDY      ;FIND RDY
5781 032070 104151          ERROR    151        ;NO RDY AFTER DRIVE CLEAR CMD
5782 032072 004737 037024      JSR     PC,TSTATN    ;TEST FOR ATTN
5783 032076 000401          BR      64$
5784 032100 104154          ERROR    154        ;ATTN NOT CLEARED AFTER DRIVE CLEAR CMD
5785 032102      64$:
5786
5787
5788 032102 005037 001452      CLR     LFCNT
5789 032106 012765 000001 000026      MOV     #1,RKMR1(R5) ;SELECT WORD 1
5790
5791 032114 004737 040100      2$:    JSR     PC,GSTAT
5792 032120 032737 000040 007416      BIT     #D.HDMM,HMR2
5793 032126 001006          BNE     3$          ;BR IF GOT HEAD HOME
5794 032130 004737 042726      JSR     PC,LOOP      ;ELSE GO THRU LOOP
5795 032134 005737 001452      TST    LFCNT        ;TEST FOR OVERFLOW
5796 032140 001365          BNE     2$          ;BR IF NO
5797 032142 104066          ERROR    66        ;NO HEAD HOME AFTER UNLOAD CMD
5798
5799 032144 104401 052411      3$:    TYPE    ,MSG18      ;TIME BEING MEASURED
5800 032150 013737 001452 001160      MOV     (LPCNT,$TMPD) ;SETUP FOR MULT
5801 032156 004737 043004      JSR     PC,TYPTIM    ;TYPE TIME IN USEC
5802 032162 104401 054567      TYPE    ,MSG43      ;LIMITS
5803 032166 104401 001205      TYPE    , $CRLF
5804 032172 104401 001205      TYPE    , $CRLF
5805
5806 032176 005037 001176      4$:    CLR     $ESCAPE
5807
5808
5809
5810
5811
5812
5813

```

```

*****
;TEST 22 TEST LOW VELOCITY TIMES DURING LOADING
;
; THIS TEST ISSUES A START SPINDLE COMMAND
; AFTER 'HEADS HOME' HAS BEEN DETECTED FROM THE PREVIOUS TEST.
; THE FOLLOWING "LOW VELOCITY" TIMES ARE CHECKED AGAINST

```

K09

JNIBUS Rk06 DRIVE DIAGNOSTIC PART 2
 DZR6IC.P11 07-OCT-76 13:50

MACY11 27(1006) 07-OCT-76 14:14 PAGE 114
 T22 TEST LOW VELOCITY TIMES DURING LOADING

SEQ 0114

5814
 5815
 5816
 5817
 5818
 5819
 5820
 5821
 5822
 5823
 5824
 5825
 5826
 5827
 5828
 5829
 5830
 5831
 5832 032202 000004
 5833 032204 012737 000001 001174
 5834 032212 012706 001100
 5835
 5836 032216 004737 040452
 5837 032222 104024
 5838
 5839 032224 004737 042262
 5840
 5841 032230 012765 000011 000000
 5842 032236 013737 001412 007426
 5843 032244 004737 036542
 5844 032250 104121
 5845
 5846 032252 012737 032656 001176
 5847
 5848 032260 013737 001370 001372
 5849 032266 012737 000074 001374
 5850 032274 012765 000001 000026
 5851 032302 004737 043236
 5852 032306 004737 040100
 5853 032312 032737 000040 007416
 5854 032320 001406
 5855 032322 005737 001376
 5856 032326 001767
 5857 032330 004737 043332
 5858 032334 104067
 5859
 5860 032336 004737 043332
 5861 032342 005037 001452
 5862 032346 012765 000001 000026
 5863
 5864 032354 004737 040100
 5865 032360 032737 000020 007416
 5866 032366 001006
 5867 032370 004737 042726
 5868 032374 005737 001452
 5869 032400 001365

```

: * LIMITS TO BE DEFINED:
: *
: * TIME 1: TIME BETWEEN HEADS HOME NEGATING & SERVO SIG PRES ASSERTING
: *           EXPECTED TIME APPROX 285 MS
: *
: * TIME 2: TIME BETWEEN OUTER LIMIT & INNER LIMIT
: *
: *           TIME IS MEASURED FROM SERVO SIG PRES (APPROX OUTER LIMIT)
: *           TO REV (APPROX INNER LIMIT)
: *           EXPECTED TIME APPROX 2.25 SEC
: *
: * TIME 3: TIME BETWEEN INNER LIMIT & OUTER LIMIT
: *
: *           TIME IS MEASURED FROM REV ASSERTING (FROM ABOVE)
: *           TO REV NEGATING (APPROX OUTER LIMIT)
: *           EXPECTED TIME APPROX 2.25 SEC
: *
: *****
T22: SCOPE
      MOV #1, $TIMES ; DO 1 ITERATION
      MOV #STACK, SP ; RESTORE STK PTR
      JSR PC, SUBCLR ; CERR AFTER SCLR
      ERROR 24
      JSR PC, CALCLK ; CALIB TIME TO GO THRU 'LOOP'
      MOV #SRTSPL RKCS1(R5) ; START SPINDLE CMD
      MOV T10, TEMP1 ; SETUP TIMEOUT
      JSR PC, FRDY ; FIND RDY
      ERROR 121 ; RDY NOT SET AFTER START SPIN CMD
      MOV #9$, $ESCAPE
      MOV HZ, COUNT
      MOV #60., SEC
      MOV #1, RKMR1(R5)
      JSR PC, CLKON ; TURN CLOCK ON FOR 60 SEC TIMEOUT
      JSR PC, GSTAT
      BIT #D.HDHM, HMR2
      BEQ 2$ ; BR IF HEAD HOME = 0
      TST TIMUP
      BEQ 1$
      JSR PC, CLKOF
      ERROR 67 ; HEAD HOME NOT CLEARED DURING LOAD CMD
      JSR PC, CLKOF
      CLR LPCNT ; SETUP FOR TIME 1
      MOV #1, RKMR1(R5) ; SELECT WORD 1
      JSR PC, GSTAT
      BIT #D.SSP, HMR2
      BNE 4$ ; BR IF GOT TFOK
      JSR PC, LOOP ; ELSE GO THRU LOOP
      TST LPCNT ; TEST FOR OVERFLOW
      BNE 3$ ; BR IF NO
  
```

```

5870 032402 104070          ERROR 70          ;SERVO SIG PRES NOT SET DURING LOAD CMD
5871 032404 013737 001452 001442 4$:  MOV    LPCNT,TIM1 ;STORE LOOP COUNT FOR TIME 1
5872
5873
5874 032412 005037 001452          CLR    LPCNT          ;SETUP FOR TIME 2
5875 032416 012765 000001 000026  MOV    #1,RKMR1(R5)
5876
5877 032424 004737 040100          JSR    PC,GSTAT
5878 032430 032737 004000 007416 5$:  BIT    #0,REV,HMR2
5879 032436 001006          BNE    6$           ;BR IF GOT REV
5880 032440 004737 042726          JSR    PC,LOOP
5881 032444 005737 001452          TST    LPCNT
5882 032450 001365          BNE    5$           ;TEST FOR OVERFLOW
5883 032452 104071          ERROR 5$           ;BR IF NO
5884 032454 013737 001452 001444 6$:  MOV    71          ;REV NOT SET DURING LOAD
5885          ERROR 71          ;STORE LOOP COUNT FOR TIME 2
5886
5887 032462 005037 001452          CLR    LPCNT          ;SETUP FOR TIME 3
5888 032466 012765 000001 000026  MOV    #1,RKMR1(R5)
5889
5890 032474 004737 040100          JSR    PC,GSTAT
5891 032500 032737 004000 007416 7$:  BIT    #0,REV,HMR2
5892 032506 001406          BEQ    8$
5893 032510 004737 042726          JSR    PC,LOOP
5894 032514 005737 001452          TST    LPCNT
5895 032520 001365          BNE    7$
5896 032522 104072          ERROR 7$           ;REV NOT CLEARED DURING LOAD
5897 032524 013737 001452 001446 8$:  MOV    72          ;STORE LOOP COUNT FOR TIME 3
5898
5899
5900 032532 104401 052542          TYPE   MSG20
5901 032536 013737 001442 001160  MOV    TIM1,$TMPO ;TIME 1 MEASUREMENT
5902 032544 004737 043004          JSR    PC,TYPTIM ;SETUP FOR MULT
5903 032550 104401 054567          TYPE   ,MSG43    ;TYPE TIME IN USEC
5904          ;LIMITS
5905 032554 104401 052636          TYPE   MSG21
5906 032560 013737 001444 001160  MOV    TIM2,$TMPO ;TIME 2 MEASUREMENT
5907 032566 004737 043004          JSR    PC,TYPTIM
5908 032572 104401 054635          TYPE   ,MSG44    ;LIMITS
5909
5910 032576 104401 052730          TYPE   MSG22
5911 032602 013737 001446 001160  MOV    TIM3,$TMPO ;TIME 3 MEASUREMENT
5912 032610 004737 043004          JSR    PC,TYPTIM
5913 032614 104401 054635          TYPE   ,MSG44    ;LIMITS
5914
5915 032620 104401 001205          TYPE   ,$CRLF
5916 032624 104401 001205          TYPE   ,$CRLF
5917 032630 005037 001176          CLR    $ESCAPE
5918
5919 032634 013737 001422 007426  MOV    T5000,TEMP1
5920 032642 004737 037152          JSR    PC,FATT2  ;FIND ATTN
5921 032646 104074          ERROR 74          ;NO ATTN AFTER START SPIN CMD
5922 032650 005037 007352          CLR    UNLD
5923 032654 000414          BR     TST23      ;USED FOR VALID HALT
5924          ;GO TO NEXT TEST
5925 032656 005037 001176          9$:  CLR    $ESCAPE

```

5926	032662	013737	001420	007430
5927	032670	004737	037056	
5928	032674	104074		
5929	032676	005037	007352	
5930	032702	000137	036034	

```

MOV T100,TEMP2
JSR PC,FATT1 ;FIND ATTN
ERROR 74 ;NO ATTN AFTER START SPIN CMD
CLR UNLD
JMP $EOP ;ABORT DRIVE

```

```

*****
*TEST 23 MEASURE ROTATIONAL SPEED

```

THIS TEST MEASURES INDEX TIMING BY:

- A. CHANGE FORMAT TO 20 SECTOR & READ HEADER.
CONTROLLER RDY STARTS THE TIMER
- B. CHANGE FORMAT TO 22 SECTOR & READ HEADER.
CONTROLLER RDY ENDS THE TIMER.

INDEX TIMING IS THE TIME BETWEEN THE 2 CONT. RDY TIMES.
THIS IS BECAUSE A CHANGE OF FORMAT INHIBITS SECTOR PULSES
UNTIL THE NEXT INDEX APPEARS-THUS KEEPING A GIVEN
FORMAT COMPLETE THROUGHOUT AN ENTIRE CYLINDER

THE TIME IS THE AVERAGE OF 100 READINGS.

```

*****
†ST23: SCOPE

```

5950	032706	000004		
5951	032710	012737	000001	001174
5952	032716	012706	001100	
5953				
5954	032722	004737	040452	
5955	032726	104024		
5956				
5957	032730	004737	042262	
5958				
5959	032734	005000		
5960	032736	005037	001456	
5961	032742	005037	001460	
5962				
5963	032746	012765	010025	000000 15:
5964	032754	013737	001424	007426
5965	032762	004737	036542	
5966	032766	104171		
5967				
5968	032770	005700		
5969	032772	001007		
5970				
5971	032774	004737	040532	
5972	033000	005737	001406	
5973	033004	001402		
5974	033006	104064		
5975				
5976	033010	000464		
5977				
5978	033012	005037	001452	45:
5979	033016	012765	000025	000000
5980	033024	013737	001424	007426
5981	033032	032765	000200	000000 25:

```

MOV #1,STIMES ;DO 1 ITERATION
MOV #STACK,SP ;RESTORE STK PTR
JSR PC,SUBCLR
ERROR 24 ;CERR AFTER SCLR
JSR PC,CALCLK ;CALIB TIME TO GO THRU 'LOOP'
CLR RD ;ITERATION COUNTER
CLR SUM ;LO WORD OF REV TIME SUM
CLR SUM+2 ;HI WORD OF REV TIME SUM
MOV #<CFMT!RDHEAD>,RKCS1(R5) ;CHANGE TO 20 SECTOR FMT & READ HEADER
MOV T5000,TEMP1 ;SETUP TIMEOUT
JSR PC,FRDY ;FIND RDY
ERROR 171 ;NO RDY AFTER READ HDR CMD
TST RD ;TEST FOR SECTOR 0 AFTER FORMAT CHANGE
BNE 45 ;DO ONLY ONCE
JSR PC,RDSEC
TST SECTOR
BEQ 45
ERROR 64 ;CANT FIND SECTOR 0 FROM INDEX
;AFTER FORMAT CHANGE & RDY REC'D
BR TST24 ;GOTO NEXT TEST
CLR LPCNT ;COUNT PASSES THRU LOOP
MOV #RDHEAD,RKCS1(R5) ;CHANGE TO 22 SECTOR FMT & READ HEADER
MOV T5000,TEMP1 ;SETUP TIMEOUT
BIT #RDY,RKCS1(R5)

```

```

5982 033040 001007      BNE      3$          ;EXIT IF GOT RDY
5983 033042 004737 042726 JSR      PC,LOOP    ;ELSE GO THRU LOOP
5984 033046 005337 007426 DEC      TEMP1
5985 033052 001367      BNE      2$
5986 033054 104171      ERROR    171        ;NO RDY AFTER READ HDR CMD
5987 033056 000441      BR       TST24      ;;GO TO NEXT TST
5988
5989 033060 004737 040532 3$:      JSR      PC,ROSEC
5990 033064 005737 001406      TST      SECTOR
5991 033070 001402      BEQ      5$
5992 033072 104064      ERROR    64        ;CANT FIND SECTOR 0 FROM INDEX
5993                                     ;AFTER FORMAT CHANGE & RDY REC'D
5994 033074 000432      BR       TST24      ;;GOTO NEXT TEST
5995
5996 033076 013702 001452 5$:      MOV      LPCNT,R2    ;FROM LOOP
5997 033102 013703 001454      MOV      LPTIM,R3   ;FROM CALCLK
5998 033106 010246      MOV      R2,-(SP)    ;;PUT THE MULTIPLIER ON THE STACK
5999 033110 010346      MOV      R3,-(SP)    ;;PUT THE MULTIPLICAND ON THE STACK
6000 033112 004737 050332      JSR      PC,2$MULT   ;;CALL THE MULTIPLY ROUTINE
6001 033116 012616      MOV      (SP)+,(SP)  ;;DISREGARD THE MSB'S
6002 033120 012603      MOV      (SP)+,R3   ;;GET THE LSB'S OF THE PRODUCT
6003
6004 033122 060337 001456      ADD      R3,SUM      ;R3 CONTAINS LOW ORDER PRODUCT
6005 033126 005537 001460      ADC
6006
6007 033132 005200      INC      R0
6008 033134 020027 000144      CMP      R0,#100.
6009 033140 001302      BNE      1$
6010
6011 033142 104401 053107      TYPE     ,MSG24     ;AVG ROTATIONAL TIME
6012
6013 033146 004737 043052      JSR      PC,AVGTIM  ;CALC & TYPEOUT AVG TIME
6014 033152 104401 001205      TYPE     ,SCLRF
6015 033156 104401 001205      TYPE     ,SCLRF
6016
6017
6018 *****
6019 *TEST 24      MEASURE MAX SEEK TIME
6020 *
6021 *      THIS TEST MEASURES THE MAX SEEK TIME BETWEEN CYLINDERS 0 & 410
6022 *      THE AVERAGE TIME OF 100 SEEKS IN BOTH DIRECTIONS ARE PRINTED
6023 *      IF NOT WITHIN LIMITS TO BE SUPPLIED.
6024 *      MAX SEEK TIME SHOULD BE LESS THAN 70MS.
6025 *****
6026 033162 000004      TST24: SCOPE
6027 033164 012737 000001 001174      MOV      #1,$TIMES  ;;DO 1 ITERATION
6028 033172 012706 001100      MOV      #STACK,SP
6029
6030
6031 033176 004737 042262      JSR      PC,CALCLK  ;CALIB TIME TO GO THRU 'LOOP'
6032
6033 033202 005000      CLR      R0          ;ITERATION COUNTER
6034 033204 005037 001456      CLR      SUM        ;LO WORD OF FOWARD SEEK TIME
6035 033210 005037 001460      CLR      SUM+2      ;HI WORD OF FOWARD SEEK TIME
6036 033214 005037 001462      CLR      SUM1       ;LO WORD OF REVERSE SEEK TIME
6037 033220 005037 001464      CLR      SUM1+2     ;HI WORD OF REVERSE SEEK TIME

```

B10

ATIBUS RK06 DRIVE DIAGNOSTIC PART 2
 07-00-76 13:50

MACY11 27(1006) 07-OCT-76 14:14 PAGE 118
 T24 MEASURE MAX SEEK TIME

SEG 0118

6038									
6039	033224	004737	040452		18:	JSR	PC, SUBCLR		
6040	033230	104024				ERROR	24		:CERR AFTER SCLR
6041									
6042	033232	012737	033524	001176		MOV	#75, \$ESCAPE		
6043									
6044	033240	012765	000632	000020		MOV	#410, RKDC(R5)		:SETUP FOR CYL 410.
6045	033246	012765	000017	000000	28:	MOV	#SEEK, RKCS1(R5)		:SEEK CMD
6046	033254	013737	001414	007426		MOV	T50, TEMP1		
6047	033262	004737	036542			JSR	PC, FRDY		:FIND RDY
6048	033266	104131				ERROR	131		:NO RDY AFTER SEEK CMD
6049									
6050	033270	005037	001452			CLR	LPCNT		:COUNT PASSES THRU LOOP
6051	033274	013704	001222			MOV	\$UNIT, R4		
6052	033300	004737	042752			JSR	PC, FATT3		:FIND ATTN AND STEP 'LPCNT'
6053	033304	104132				ERROR	132		:NO ATTN AFTER SEEK CMD
6054									
6055	033306	004737	040100			JSR	PC, GSTAT		
6056	033312	032737	100000	007370		BIT	#CERR, HCS1		
6057	033320	001401				BEQ	55		
6058	033322	104210				ERROR	210		:CERR AFTER SEEK CMD.
6059									
6060	033324	013737	001452	001160	55:	MOV	LPCNT, \$TMP0		:FROM LOOP
6061	033332	013737	001454	001162		MOV	LPTIM, \$TMP1		:FROM CALCLK
6062	033340	013746	001160			MOV	\$TMP0, -(SP)		::PUT THE MULTIPLIER ON THE STACK
6063	033344	013746	001162			MOV	\$TMP1, -(SP)		::PUT THE MULTIPLICAND ON THE STACK
6064	033350	004737	050332			JSR	PC, 2*\$MULT		::CALL THE MULTIPLY ROUTINE
6065	033354	012637	001162			MOV	(SP)+, \$TMP1		::GET THE LSB'S OF THE PRODUCT
6066	033360	012637	001164			MOV	(SP)+, \$TMP1+2		::GET THE MSB'S OF THE PRODUCT
6067									
6068	033364	005765	000020			TST	RKDC(R5)		
6069	033370	001414				BEQ	65		:BR IF THIS SEEK WAS REVERSE TO CYL 0
6070									
6071	033372	063737	001162	001456		ADD	\$TMP1, SUM		:SUM UP FOWARD SEEK *TIMES (LSB)
6072	033400	005537	001460			ADC	SUM+2		
6073	033404	063737	001164	001460		ADD	\$TMP2, SUM+2		:MSB
6074									
6075	033412	004737	040452			JSR	PC, SUBCLR		
6076	033416	104024				ERROR	24		:CERR AFTER SCLR
6077	033420	000712				BR	25		:SEEK TO CYL 0
6078									
6079	033422	063737	001162	001462	65:	ADD	\$TMP1, SUM1		:SUM UP REVERSE SEEK TIMES (LSB)
6080	033430	005537	001464			ADC	SUM1+2		
6081	033434	063737	001164	001464		ADD	\$TMP2, SUM1+2		:MSB
6082									
6083	033442	005200				INC	RO		
6084	033444	020027	000144			CMP	RO, #100.		:ALL SEEKS DONE?
6085	033450	001265				BNE	15		:BR & REPEAT IF NO TO CYL 410.
6086									
6087	033452	104401	053267			TYPE	MSG28		:FOWARD SEEK TIME
6088	033456	004737	043052			JSR	PC, AVGTIM		:CALC & TYPE AVG TIME (FOWARD)
6089									
6090	033462	104401	053346			TYPE	MSG29		:REVERSE SEEK TIME
6091	033466	013737	001462	001456		MOV	SUM1, SUM		:SETUP FOR
6092	033474	013737	001464	001460		MOV	SUM1+2, SUM+2		:AVGTIM ROUTINE
6093	033502	004737	043052			JSR	PC, AVGTIM		:CALC & TYPE AVG TIME (REVERSE)

```

6094
6095 033506 104401 001205 TYPE .SCLF
6096 033512 104401 001205 TYPE .SCLF
6097 033516 005037 001176 CLR $ESCAPE
6098 033522 000464 BR $T25 ;;GO TO NEXT TEST
6099
6100 033524 005037 001176 T5: CLR $ESCAPE
6101
6102 033530 012765 100000 000000 MOV #CLR,RKCS1(R5)
6103 033536 013765 001222 000010 MOV $UNIT,RKCS2(R5)
6104 033544 012765 000013 000000 MOV #RECAL,RKCS1(R5) ;RECAL CMD
6105 ;RESET CYL DIFF/OFFSET & CYL ADDR REG
6106 ;IN RKMR2 & RKMR3 RESP.
6107 033552 013737 001412 007426 MOV T10,TEMP1 ;SETUP TIMEOUT
6108 033560 004737 036542 JSR PC,FRDY ;FIND RDY
6109 033564 104124 ERROR 124 ;RDY NOT SET AFTER RECAL CMD
6110
6111 033566 012765 000001 000026 MOV #1,RKMR1(R5) ;SELECT WORD 1
6112 033574 004737 040100 JSR PC,GSTAT
6113 033600 032737 020000 007416 BIT #0,RTZ,HMR2
6114 033606 001001 BNE 64$
6115 033610 104244 ERROR 244 ;RTZ NOT SET DURING RECAL CMD
6116 033612 013737 001412 007430 64$: MOV T10,TEMP2 ;SETUP TIMEOUT
6117 033620 004737 037056 JSR PC,FATT1 ;FIND ATTN
6118 033624 104055 ERROR 55 ;NO ATTN AFTER RECAL CMD
6119
6120 033626 012765 100000 000000 MOV #CLR,RKCS1(R5)
6121 033634 013765 001222 000010 MOV $UNIT,RKCS2(R5) ;DRIVE#
6122 033642 012765 000005 000000 MOV #CLEAR,RKCS1(R5) ;DRIVE CLEAR CMD
6123 033650 013737 001412 007426 MOV T10,TEMP1 ;SETUP TIMEOUT
6124 033656 004737 036542 JSR PC,FRDY ;FIND RDY
6125 033662 104151 ERROR 151 ;NO RDY AFTER DRIVE CLEAR CMD
6126 033664 004737 037024 JSR PC,TSTATN ;TEST FOR ATTN
6127 033670 000401 BR 65$
6128 033672 104154 ERROR 154 ;ATTN NOT CLEARED AFTER DRIVE CLEAR CMD
6129
6130 65$:
6131
6132
6133
6134
6135 *****
6136 *TEST 25 MEASURE MIN SEEK TIME
6137 *
6138 * THIS TEST MEASURES THE MIN SEEK TIME BETWEEN CYLINDER 0 & 1
6139 * THE AVERAGE TIME OF 100 SEEKS IN BOTH DIRECTIONS ARE PRINTED
6140 * IF NOT WITHIN LIMITS TO BE SUPPLIED.
6141 * MIN SEEK TIME SHOULD BE LESS THAN 10MS.
6142 *
6143 *****
6144 †T25: SCOPE
6145 033674 000004 MOV #1,$TIMES ;;DO 1 ITERATION
6146 033676 012737 000001 001174 MOV #STACK,$P
6147 033704 012706 001100
6148 033710 004737 042262 JSR PC,CALCLK ;CALIB TIME TO GO THRU 'LOOP'
6149

```

6185	033714	005000			CLR	RC		: ITERATION COUNTER
6186	033716	005037	001456		CLR	SUM		: LO WORD OF FOWARD SEEK TIME
6187	033722	005037	001460		CLR	SUM+2		: HI WORD OF FOWARD SEEK TIME
6188	033726	005037	001462		CLR	SUM1		: LO WORD OF REVERSE SEEK TIME
6189	033732	005037	001464		CLR	SUM1+2		: HI WORD OF REVERSE SEEK TIME
6190	033736	004737	040452		JSR	PC, SUBCLR		
6191	033742	104024		18:	ERROR	24		: CERR AFTER SCLR
6192	033744	012737	034236	001176	MOV	#75, \$ESCAPE		
6193	033752	012765	000001	000020	MOV	#1, RKDC(R5)		: SETUP FOR CYL 1
6194	033760	012765	000017	000000	MOV	#SEEK, RKCS1(R5)		: SEEK CMD
6195	033766	013737	001414	007426	MOV	T50, TEMP1		
6196	033774	004737	036542		JSR	PC, FROY		: FIND R0Y
6197	034000	104131			ERROR	131		: NO R0Y AFTER SEEK CMD
6198	034002	005037	001452		CLR	_PCNT		: COUNT PASSES THRU LOOP
6199	034006	013704	001222		MOV	\$UNIT, R4		
6200	034012	004737	042752		JSR	PC, FATT3		: FIND ATTN AND STEP 'LPCNT'
6201	034016	104132			ERROR	132		: NO ATTN AFTER SEEK CMD
6202	034020	004737	040100		JSR	PC, GSTAT		
6203	034024	032737	100000	007370	BIT	#CERR, HCS1		
6204	034032	001401			BEQ	\$S		
6205	034034	104210			ERROR	210		: CERR AFTER SEEK CMD.
6206	034036	013737	001452	001160	MOV	LPCNT, \$TMP0		: FROM LOOP
6207	034044	013737	001454	001162	MOV	LPTIM, \$TMP1		: FROM CALCLK
6208	034052	013746	001160		MOV	\$TMP0, -(SP)		: PUT THE MULTIPLIER ON THE STACK
6209	034056	013746	001162		MOV	\$TMP1, - (SP)		: PUT THE MULTIPLICAND ON THE STACK
6210	034062	004737	050332		JSR	PC, #MULT		: CALL THE MULTIPLY ROUTINE
6211	034066	012637	001162		MOV	(SP)+, \$TMP1		: GET THE LSB'S OF THE PRODUCT
6212	034072	012637	001164		MOV	(SP)+, \$TMP1+2		: GET THE MSB'S OF THE PRODUCT
6213	034076	005765	000020		TST	RKDC(R5)		
6214	034102	001414			BEQ	\$S		: BR IF THIS SEEK WAS REVERSE TO CYL 0
6215	034104	063737	001162	001456	ADD	\$TMP1, SUM		: SUM UP FOWARD SEEK TIMES (LSB)
6216	034112	005537	001460		ADC	SUM+2		
6217	034116	063737	001164	001460	ADD	\$TMP2, SUM+2		: MSB
6218	034124	004737	040452		JSR	PC, SUBCLR		
6219	034130	104024			ERROR	24		: CERR AFTER SCLR
6220	034132	000712			BR	\$S		: SEEK TO CYL 0
6221	034134	063737	001162	001462	ADD	\$TMP1, SUM1		: SUM UP REVERSE SEEK TIMES (LSB)
6222	034142	005537	001464		ADC	SUM1+2		
6223	034146	063737	001164	001464	ADD	\$TMP2, SUM1+2		: MSB
6224	034154	005200			INC	RC		
6225	034156	020027	000144		CMP	RC, #100.		: ALL SEEKS DONE?
6226	034162	001265			BNE	\$S		: BR & REPEAT IF NO TO CYL 1
6227	034164	104401	053426		TYPE	MSG30		: FOWARD SEEK TIME
6228	034170	004737	043052		JSR	PC, AVGTIM		: CALC & TYPE AVG TIME (FOWARD)

```

6206
6207 034174 104401 053514          TYPE      MSG31          ;REVERSE SEEK TIME
6208 034200 013737 001462 001456      MOV      $SUM1,SUM      ;SETUP FOR
6209 034206 013737 001464 001460      MOV      SUM1+2,SUM+2    ;AVGTIM ROUTINE
6210 034214 004737 043052          JSR      PC,AVGTIM      ;CALC & TYPE AVG TIME (REVERSE)
6211
6212 034220 104401 001205          TYPE      $SCLF
6213 034224 104401 001205          TYPE      $SCLF
6214 034230 005037 001176          CLR      $ESCAPE
6215 034234 000464          BR      TST26          ;;GO TO NEXT TEST
6216
6217 034236 005037 001176          75:     CLR      $ESCAPE
6218
6219 034242 012765 100000 000000      MOV      #CLR,RKCS1(R5)
6220 034250 013765 001222 000010      MOV      $UNIT,RKCS2(R5)
6221 034256 012765 000013 000000      MOV      #RECAL,ACS1(R5) ;RECAL CMD
6222
6223
6224 034264 013737 001412 007426      MOV      T10,TEMP1      ;RESET CYL DIFF/OFFSET & CYL ADDR REG
6225 034272 004737 036542          JSR      PC,FRDY        ;IN RKMR2 & RKMR3 RESP.
6226 034276 104124          ERROR    124          ;SETUP TIMEOUT
6227
6228 034300 012765 000001 000026      MOV      #1,RKMR1(R5)   ;FIND RDY
6229 034306 004737 040100          JSR      PC,GSTAT      ;RDY NOT SET AFTER RECAL CMD
6230 034312 032737 020000 007416      BIT      #D.RTZ,HMR2
6231 034320 001001          BNE      64$
6232 034322 104244          ERROR    244          ;SELECT WORD 1
6233 034324 013737 001412 007430 64$:     MOV      T10,TEMP2      ;RTZ NOT SET DURING RECAL CMD
6234 034332 004737 037056          JSR      PC,FATT1      ;SETUP TIMEOUT
6235 034336 104055          ERROR    55          ;FIND ATTN
6236
6237 034340 012765 100000 000000      MOV      #CLR,RKCS1(R5) ;NO ATTN AFTER RECAL CMD
6238 034346 013765 001222 000010      MOV      $UNIT,RKCS2(R5) ;DRIVE#
6239 034354 012765 000005 000000      MOV      #CLEAR,RKCS1(R5) ;DRIVE CLEAR CMD
6240 034362 013737 001412 007426      MOV      T10,TEMP1      ;SETUP TIMEOUT
6241 034370 004737 036542          JSR      PC,FRDY        ;FIND RDY
6242 034374 104151          ERROR    151          ;NO RDY AFTER DRIVE CLEAR CMD
6243 034376 004737 037024          JSR      PC,TSTATN     ;TEST FOR ATTN
6244 034402 000401          BR      65$
6245 034404 104154          ERROR    154          ;ATTN NOT CLEARED AFTER DRIVE CLEAR CMD
6246 034406          65$:
6247
6248
6249
6250
6251
6252
6253
6254
6255
6256
6257
6258
6259
6260 034406 000004          TST26:  SCOPE
6261 034410 012737 000001 001174      MOV      #1,$TIMES      ;;DO 1 ITERATION

```

```

*****
*TEST 26          MEASURE 137 CYLINDER SEEK TIME
*
* THIS TEST MEASURES THE AVERAGE SEEK TIME BETWEEN CYLINDERS 0 & 137
* THE AVERAGE TIME OF 100 SEEKS IN BOTH DIRECTIONS ARE PRINTED
* IF NOT WITHIN LIMITS TO BE SUPPLIED.
* AVERAGE SEEK TIME SHOULD BE LESS THAN 40MS
*****

```

F10

UNIBUS RK06 DRIVE DIAGNOSTIC PART 2
 DZR6IC.P11 07-OCT-76 13:50

MACY11 27.1006) 07-OCT-76 14:14 PAGE 122
 T26 MEASURE 137 CYLINDER SEEK TIME

SEQ 0122

6262	034416	012706	001100		MOV	#STACK, SP	
6263							
6264							
6265	034422	004737	042262		JSR	PC, CALCLK	;CALIB TIME TO GO THRU 'LOOP'
6266							
6267	034426	005000			CLR	RO	; ITERATION COUNTER
6268	034430	005037	001456		CLR	SUM	;LO WORD OF FOWARD SEEK TIME
6269	034434	005037	001460		CLR	SUM+2	;HI WORD OF FOWARD SEEK TIME
6270	034440	005037	001462		CLR	SUM1	;LO WORD OF REVERSE SEEK TIME
6271	034444	005037	001464		CLR	SUM1+2	;HI WORD OF REVERSE SEEK TIME
6272							
6273	034450	004737	040452	15:	JSR	PC, SUBCLR	
6274	034454	104024			ERROR	24	;CERR AFTER SCLR
6275							
6276	034456	012737	034750	001176	MOV	#75, \$ESCAPE	
6277							
6278	034464	012765	000211	000020	MOV	#137, RKDC(R5)	;SETUP FOR CYL 137.
6279	034472	012765	000017	000000	MOV	#SEEK, RKCS1(R5)	;SEEK CMD
6280	034500	013737	001414	007426	MOV	T50, TEMPL	
6281	034506	004737	036542		JSR	PC, FRDY	;FIND RDY
6282	034512	104131			ERROR	131	;NO RDY AFTER SEEK CMD
6283							
6284	034514	005037	001452		CLR	LPCNT	;COUNT PASSES THRU LOOP
6285	034520	013704	001222		MOV	\$UNIT, R4	
6286	034524	004737	042752		JSR	PC, FATT3	;FIND ATTN AND STEP 'LPCNT'
6287	034530	104132			ERROR	132	;NO ATTN AFTER SEEK CMD
6288							
6289	034532	004737	040100		JSR	PC, GSTAT	
6290	034536	032737	100000	007370	BIT	#CERR, HCS1	
6291	034544	001401			BEQ	55	
6292	034546	104210			ERROR	210	;CERR AFTER SEEK CMD.
6293							
6294	034550	013737	001452	001160	MOV	LPCNT, \$TMP0	;FROM LOOP
6295	034556	013737	001452	001162	MOV	LPTIM, \$TMP1	;FROM CALCLK
6296	034564	013746	001160		MOV	\$TMP0, -(SP)	;:PUT THE MULTIPLIER ON THE STACK
6297	034570	013746	001162		MOV	\$TMP1, -(SP)	;:PUT THE MULTIPLICAND ON THE STACK
6298	034574	004737	050332		JSR	PC, \$MULT	;:CALL THE MULTIPLY ROUTINE
6299	034600	012637	001162		MOV	(SP)+, \$TMP1	;:GET THE LSB'S OF THE PRODUCT
6300	034604	012637	001164		MOV	(SP)+, \$TMP1+2	;:GET THE MSB'S OF THE PRODUCT
6301							
6302	034610	005765	000020		TST	RKDC(R5)	
6303	034614	001414			BEQ	65	;BR IF THIS SEEK WAS REVERSE TO CYL 0
6304							
6305	034616	063737	001162	001456	ADD	\$TMP1, SUM	;SUM UP FOWARD SEEK TIMES (LSB)
6306	034624	005537	001460		ADC	SUM+2	
6307	034630	063737	001164	001460	ADD	\$TMP2, SUM+2	;MSB
6308							
6309	034636	004737	040452		JSR	PC, SUBCLR	
6310	034642	104024			ERROR	24	;CERR AFTER SCLR
6311	034644	000712			BR	25	;SEEK TO CYL 0
6312							
6313	034646	063737	001162	001462	ADD	\$TMP1, SUM1	;SUM UP REVERSE SEEK TIMES (LSB)
6314	034654	005537	001464		ADC	SUM1+2	
6315	034660	063737	001164	001464	ADD	\$TMP2, SUM1+2	;MSB
6316							
6317	034666	005200			INC	RO	

G10

UNIBUS RK06 DRIVE DIAGNOSTIC PART 2
 DZR6IC.P11 07-OCT-76 13:50

MACY11 27(1006) 07-OCT-76 14:14 PAGE 123
 T26 MEASURE 137 CYLINDER SEEK TIME

SEQ 0123

6318	034670	020027	000144		CMP	RD, #100.		: ALL SEEKS DONE?
6319	034674	001265			BNE	1\$: BR & REPEAT IF NO TO CYL 137.
6320								
6321	034676	104401	053603		TYPE	MSG32		: FOWARD SEEK TIME
6322	034702	004737	043052		JSR	PC,AVGTIM		: CALC & TYPE AVG TIME (FOWARD)
6323								
6324	034706	104401	053662		TYPE	MSG33		: REVERSE SEEK TIME
6325	034712	013737	001462	001456	MOV	SUM1,SUM		: SETUP FOR
6326	034720	013737	001464	001460	MOV	SUM1+2,SUM+2		: AVGTIM ROUTINE
6327	034726	004737	043052		JSR	PC,AVGTIM		: CALC & TYPE AVG TIME (REVERSE)
6328								
6329	034732	104401	001205		TYPE	,SCLF		
6330	034736	104401	001205		TYPE	,SCLF		
6331	034742	005037	001176		CLR	\$ESCAPE		
6332	034746	000464			BR	TST27		: ;GO TO NEXT TEST
6333								
6334	034750	005037	001176		75:	CLR	\$ESCAPE	
6335								
6336	034754	012765	100000	000000	MOV	#CLR,RKCS1(R5)		
6337	034762	013765	001222	000010	MOV	\$UNIT,RKCS2(R5)		
6338	034770	012765	003013	000000	MOV	#RECAL,RKCS1(R5)		: RECAL CMD
6339								: RESET CYL DIFF/OFFSET & CYL ADDR REG
6340								: IN RKMR2 & RKMR3 RESP.
6341	034776	013737	001412	007426	MOV	T10,TEMP1		: SETUP TIMEOUT
6342	035004	004737	036542		JSR	PC,FRDY		: FIND RDY
6343	035010	104124			ERROR	124		: RDY NOT SET AFTER RECAL CMD
6344								
6345	035012	012765	000001	000026	MOV	#1,RKMR1(R5)		: SELECT WORD 1
6346	035020	004737	040100		JSR	PC,GSTAT		
6347	035024	032737	020000	007416	BIT	#D,RTZ,HMR2		
6348	035032	001001			BNE	64\$		
6349	035034	104244			ERROR	244		: RTZ NOT SET DURING RECAL CMD
6350	035036	013737	001412	007430	64\$:	MOV	T10,TEMP2	: SETUP TIMEOUT
6351	035044	004737	037056		JSR	PC,FATT1		: FIND ATTN
6352	035050	104055			ERROR	55		: NO ATTN AFTER RECAL CMD
6353								
6354	035052	012765	100000	000000	MOV	#CLR,RKCS1(R5)		
6355	035060	013765	001222	000010	MOV	\$UNIT,RKCS2(R5)		: DRIVE#
6356	035066	012765	000005	000000	MOV	#CLEAR,RKCS1(R5)		: DRIVE CLEAR CMD
6357	035074	013737	001412	007426	MOV	T10,TEMP1		: SETUP TIMEOUT
6358	035102	004737	036542		JSR	PC,FRDY		: FIND RDY
6359	035106	104151			ERROR	151		: NO RDY AFTER DRIVE CLEAR CMD
6360	035110	004737	037024		JSR	PC,TSTATN		: TEST FOR ATTN
6361	035114	000401			BR	65\$		
6362	035116	104154			65\$:	ERROR	154	: ATTN NOT CLEARED AFTER DRIVE CLEAR CMD
6363	035120							

6364
 6365
 6366
 6367
 6368
 6369
 6370
 6371
 6372
 6373

```

:*****
:TEST 27      MEASURE MAX VELOCITY OF HEADS
:
: THIS TESTS MAX VELOCITY BY DOING SEEKS BETWEEN
: CYL 0 & 383 AND MEASURING THE TIME BETWEEN CYLINDERS
: 128 & 256. SINCE THE DISTANCE BETWEEN CYL 128 & 256 IS KNOWN,
: THE AVERAGE VELOCITY OF 100 SEEKS IS CALCULATED & TYPED
:

```

H10

JNIBUS RKO6 DRIVE DIAGNOSTIC PART 2
DZR6IC.P11 07-OCT-76 13:50

MACY11 27(1006) 07-OCT-76 14:14 PAGE 124
T27 MEASURE MAX VELOCITY OF HEADS

SEQ 0124

```
6374                                     :*      IF NOT WITHIN THE SPECIFIED LIMITS TO BE SUPPLIED.
6375                                     :*
6376                                     :*
6377 035120 000004                       :*****
6378 035122 012737 000001 001174      †ST27: SCOPE
6379 035130 012706 001100              MOV      #1,STIMES      ;:DO 1 ITERATION
6380                                     MOV      #STACK,SP      ;RESTORE STK PTR
6381 035134 005000                       CLR      R0              ;ITERATION COUNTER
6382 035136 005037 001456              CLR      SUM            ;LO WORD OF FOWARD SEEK TIME
6383 035142 005037 001460              CLR      SUM+2          ;HI WORD OF FOWARD SEEK TIME
6384 035146 005037 001462              CLR      SUM1           ;LO WORD OF REVERSE SEEK TIME
6385 035152 005037 001464              CLR      SUM1+2        ;HI WORD OF REVERSE SEEK TIME
6386
6387 035156 004737 042502              JSR      PC,VELCAL      ;CLOCK CALIB SIMILAR TO CALCLK
6388
6389 035162 004737 040452              15:     JSR      PC,SUBCLR
6390 035166 104024                       ERROR    24              ;CERR AFTER SCLR
6391
6392 035170 012737 035664 001176      MOV      #145,$ESCAPE
6393
6394 035176 012765 000577 000020      MOV      #383.,RKDC(R5)
6395 035204 012765 000017 000000      25:     MOV      #SEEK,RKCS1(R5) ;SEEK CMD TO CYL 400
6396 035212 013737 001414 007426      MOV      T50,TEMP1
6397 035220 004737 036542              JSR      PC,FRDY        ;FIND RDY
6398 035224 104131                       ERROR    131            ;NO RDY AFTER SEEK CMD
6399
6400 035226 013737 001416 007426      MOV      T500,TEMP1
6401 035234 004737 040666              645:   JSR      PC,QKCYLD      ;READ CYL DIFF (NO SHIFTING)
6402 035240 032737 004000 001362      BIT      #BIT11,CYLDIF ;CHECK BIT 7 (NO SHIFTING)
6403 035246 001004                       BNE     65$             ;BR IF SET
6404 035250 005337 007426              DEC      TEMP1
6405 035254 001367                       BNE     64$
6406 035256 104110                       ERROR    110            ;CANNOT FIND CYL 128
6407
6408 035260 005037 001452              65$:   CLR      LPCNT
6409 035264 004737 040666              66$:   JSR      PC,QKCYLD
6410 035270 032737 004000 001362      BIT      #BIT11,CYLDIF ;CHECK BIT 7 (NO SHIFTING)
6411 035276 001406                       BEQ     67$
6412
6413 035300 004737 042726              JSR      PC,LOOP
6414 035304 005737 001452              TST     LPCNT
6415 035310 001365                       BNE     66$
6416 035312 104111                       ERROR    111            ;CANNOT FIND CYL 256
6417 035314
6418 035314 013737 001424 007426      67$:   MOV      T50000,TEMP1
6419 035322 004737 037152              JSR      PC,FAT2        ;FIND ATTN
6420 035326 104132                       ERROR    132            ;NO ATTN AFTER SEEK CMD
6421
6422 035330 032737 100000 007370      BIT      #CERR,HCS1
6423 035336 001401                       BEQ     7$
6424 035340 104210                       ERROR    210            ;CERR AFTER SEEK CMD
6425
6426 035342 013702 001452              7$:   MOV      LPCNT,R2      ;FROM LOOP
6427 035346 013703 001454              MOV      LPTIM,R3      ;FROM VELCAL
6428 035352 010246                       MOV      R2,-(SP)      ;;PUT THE MULTIPLIER ON THE STACK
6429 035354 010346                       MOV      R3,-(SP)      ;;PUT THE MULTIPLICAND ON THE STACK
```

6430	035356	004737	050332		JSR	PC, #SMULT	::CALL THE MULTIPLY ROUTINE
6431	035362	012616			MOV	(SP)+, (SP)	::DISREGARD THE MSB'S
6432	035364	012603			MOV	(SP)+, R3	::GET THE LSB'S OF THE PRODUCT
6433	035366	060337	001456		ADD	R3, SUM	;SUM UP FOWARD SEEK TIMES
6434	035372	005537	001460		ADC	SUM+2	
6435							
6436	035376	004737	040452		JSR	PC, SUBCLR	
6437	035402	104024			ERROR	24	;CERR AFTER SCLR
6438							
6439	035404	012765	000017	000000	MOV	#SEEK, RKCS1(R5)	;SEEK TO CYL 0
6440	035412	013737	001414	007426	MOV	T50, TEMP1	
6441	035420	004737	036542		JSR	PC, FRDY	;FIND RDY
6442	035424	104131			ERROR	131	;NO RDY AFTER SEEK CMD
6443							
6444	035426	013737	001416	007426	MOV	T500, TEMP1	
6445	035434	004737	040666	68\$:	JSR	PC, QKCYLD	;READ CYL DIFF (NO SHIFTING)
6446	035440	032737	004000	001362	BIT	#BIT11, CYLDIF	;CHECK BIT 7 (NO SHIFTING)
6447	035446	001004			BNE	69\$;BR IF SET
6448	035450	005337	007426		DEC	TEMP1	
6449	035454	001367			BNE	68\$	
6450	035456	104111			ERROR	111	;CANNOT FIND CYL 256
6451							
6452	035460	005037	001452	69\$:	CLR	LPCNT	
6453	035464	004737	040666	70\$:	JSR	PC, QKCYLD	
6454	035470	032737	004000	001362	BIT	#BIT11, CYLDIF	;CHECK BIT 7 (NO SHIFTING)
6455	035476	001406			BEQ	71\$	
6456							
6457	035500	004737	042726		JSR	PC, LOOP	
6458	035504	005737	001452		TST	LPCNT	
6459	035510	001365			BNE	70\$	
6460	035512	104110			ERROR	110	;CANNOT FIND CYL 128
6461	035514			71\$:			
6462	035514	013737	001424	007426	MOV	T50000, TEMP1	
6463	035522	004737	037152		JSR	PC, FAT2	;FIND ATTN
6464	035526	104132			ERROR	132	;NO ATTN AFTER SEEK CMD
6465							
6466	035530	032737	100000	007370	BIT	#CERR, HCS1	
6467	035536	001401			BEQ	12\$	
6468	035540	104210			ERROR	210	;CERR AFTER SEEK CMD
6469							
6470	035542	013702	001452	12\$:	MOV	LPCNT, R2	
6471	035546	013703	001454		MOV	LPTIM, R3	
6472	035552	010246			MOV	R2, -(SP)	::PUT THE MULTIPLIER ON THE STACK
6473	035554	010346			MOV	R3, -(SP)	::PUT THE MULTIPLICAND ON THE STACK
6474	035556	004737	050332		JSR	PC, #SMULT	::CALL THE MULTIPLY ROUTINE
6475	035562	012616			MOV	(SP)+, (SP)	::DISREGARD THE MSB'S
6476	035564	012603			MOV	(SP)+, R3	::GET THE LSB'S OF THE PRODUCT
6477	035566	060337	001462		ADD	R3, SUM1	;SUM UP REVERSE SEEK TIMES
6478	035572	005537	001464		ADC	SUM1+2	
6479							
6480	035576	005200			INC	RO	
6481	035600	020027	000144		CMP	RO, #100.	;DONE 100 SEEKS?
6482	035604	001402			BEQ	13\$;BR IF YES
6483	035606	000137	035162		JMP	1\$;ELSE DO AGAIN
6484							
6485	035612	104401	053742	13\$:	TYPE	,MSG34	;AVG MAX FWD SPEED


```

6532 .SBTTL END OF PASS ROUTINE
6533
6534 ;:*****
6535 ;*INCREMENT THE PASS NUMBER ($PASS)
6536 ;*TYPE "END PASS #XXXX" (WHERE XXXX IS A DECIMAL NUMBER)
6537 ;*IF THERES A MONITOR GO TO IT
6538 ;*IF THERE ISN'T JUMP TO STS
6539
6540 036034 $EOP:
6541
6542 036034 000004 SCOPE
6543 036036 012706 001100 MOV #STACK, SP
6544 036742 005237 001220 INC $DEVCT ; INCR COUNT FOR # DRIVES CHECKED
6545 036046 023737 007510 001220 CMP DRIVS, $DEVCT ; ARE ALL DRIVES PRESENT TESTED?
6546 036054 001403 BEQ $EOP1+2 ; BR IF YES
6547 036056 000137 015154 JMP NUDRV ; ELSE TEST NEXT DRIVE PRESENT
6548 036062 000004 $EOP1: SCOPE
6549 036064 005037 001102 CLR $TSTNM ; ZERO THE TEST NUMBER
6550 036070 005037 001174 CLR $TIMES ; ZERO THE NUMBER OF ITERATIONS
6551 036074 005237 001216 INC $PASS ; INCREMENT THE PASS NUMBER
6552 036100 042737 100000 001216 BIC #100000, $PASS ; DON'T ALLOW A NEG. NUMBER
6553 036106 005327 DEC (PC)+ ; LOOP?
6554 036110 000001 $EOPCT: .WORD 1
6555 036112 003022 BGT $DOAGN ; YES
6556 036114 012737 MOV (PC)+, a(PC)+ ; RESTORE COUNTER
6557 036116 000001 $ENDCT: .WORD 1
6558 036120 036110 $EOPCT
6559 036122 104401 036167 TYPE $ENDMG ; TYPE "END PASS #"
6560 036126 013746 001216 MOV $PASS, -(SP) ; SAVE $PASS FOR TYPEOUT
6561 036132 104405 TYPDS ; GO TYPE-... ICIMAL ASCII WITH SIGN
6562 036134 104401 036164 TYPE $ENULL ; TYPE NULL CHARACTER
6563 036140 013700 000042 $GET42: MOV a#42, R0 ; GET MONITOR ADDRESS
6564 036144 001405 BEQ $DOAGN ; BRANCH IF NO MONITOR
6565 036146 000005 RESET ; CLEAR THE WORLD
6566 036150 004710 $ENDAD: JSR PC, (R0) ; GO TO MONITOR
6567 036152 000240 NOP ; SAVE ROOM
6568 036154 000240 NOP ; FOR
6569 036156 000240 NOP ; ACT11
6570 036160 $DOAGN:
6571 036160 000137 JMP a(PC)+ ; RETURN
6572 036162 013534 $RTNAD: .WORD STS
6573 036164 377 377 000 $ENULL: .BYTE -1, -1, 0 ; NULL CHARACTER STRING
6574 036167 015 042412 042116 $ENDMG: .ASCIIZ <15><12>/END PASS #/
6575 036174 050040 051501 020123
6576 036202 000043

```

6577
6578
6579
6580
6581
6582
6583
6584
6585
6586
6587
6588
6589
6590
6591
6592
6593
6594
6595
6596
6597
6598
6599
6600
6601
6602
6603
6604
6605
6606
6607
6608
6609
6610
6611
6612
6613
6614
6615
6616
6617
6618
6619
6620
6621
6622
6623
6624
6625
6626
6627
6628
6629
6630
6631
6632

036204 012700 007500
036210 012701 177757
036214 005020
036216 005201
036220 001375
036222 000207

036224 005737 001344
036230 001024
036232 005237 001344
036236 104401 050630

036242 005737 000042
036246 001012
036250 123727 001230 000001
036256 001406
036260 023727 001140 000176
036266 001005
036270 104406
036272 000403
036274 112737 000001 001134
036302
036302 000207

036304 104411
036306 012600
036310 012701 177770
036314 112002
036316 042702 177400
036322 012703 007512
036326 012704 000060

036332 020402
036334 001415
036336 005723
036340 005204
036342 020427 000070
036346 001371
036350 005702
036352 001022
036354 020127 177770
036360 001426

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

;TYPE PROGRAM ID IF FTITLE=0
:
TITLE: TST FTITLE
BNE 1$
INC FTITLE
TYPE MSG1 ;PROGRAM ID
.SBTTL GET VALUE FOR SOFTWARE SWITCH REGISTER
TST 0#42 ;ARE WE RUNNING UNDER XXDP/ACT?
BNE 64$ ;BRANCH IF YES
CMPB $ENV, #1 ;ARE WE RUNNING UNDER APT?
BEQ 64$ ;BRANCH IF YES
CMP SWR, #SWREG ;SOFTWARE SWITCH REG SELECTED?
BNE 65$ ;BRANCH IF NO
GTSWR ;GET SOFT-SWR SETTINGS
BR 65$
64$: MOVB #1, $AUTOB ;SET AUTO-MODE INDICATOR
65$:
1$: RTS PC

;ROUTINE TO INPUT DRIVE NOS. TYPED IN & SET
;DRIVS, DRIVO-DRIV7 REGISTERS APPROPRIATELY
:
GDRVS: RDLIN
MOV (SP)+, R0 ;GET STARTING ADDR OF ASCII STRING
MOV #-8., R1 ;SET UP COUNT
1$: MOVB (R0)+, R2 ;GET ASCII CHAR
BIC #177400, R2 ;MASK HI BYTE
MOV #DRIVO, R3 ;DRIVE FLAG ADDR
MOV #60, R4

2$: CMP R4, R2 ;WAS TYPED CHAR 0 THRU 7?
BEQ 3$ ;BRANCH IF YES
TST (R3)+ ;NO, INCREMENT DR FLAG ADDR
INC R4
CMP R4, #70
BNE 2$ ;S/B 0-7 OR TERMINATOR
TST R2
BNE 4$
CMP R1, #-8.
BEQ 6$ ;DEFAULT ALL DRIVES
```

M10

JNIBUS RK06 DRIVE DIAGNOSTIC PART 2
DZR6IC.P11 07-OCT-76 13:50

MACY11 27(1006) 07-OCT-76 14:14 PAGE 129
GET VALUE FOR SOFTWARE SWITCH REGISTER

SEQ 0129

```

6633 036362 005037 007540 7$: CLR      SIZFLG      ;BYPASS TEST 1 (SIZING)
6634 036366 000207          RTS      PC          ;FOUND TERMINATOR, EXIT
6635
6636 036370 005213          3$: INC      @R3      ;SET UP FLAG FOR THE DRIVE
6637 036372 005237 007510   INC      DRVS      ;INCREMENT TOTAL # DRIVES TO BE TESTED
6638 036376 112002          MOV      (R0)+,R2   ;GET NEXT ASCII CHAR.
6639 036400 042702 177400   BIC      #177400,R2 ;MASK
6640 036404 022702 000054   CMP      #54,R2    ;IS IT A COMMA?
6641 036410 001407          BEQ      5$         ;YES, GO TO NEXT WORD.
6642 036412 005702          TST      R2        ;NO, IS IT A TERMINATOR?
6643 036414 001001          BNE      4$         ;IF NOT, SOMETHING WRONG.
6644 036416 000761          BR       7$         ;FOUND TERMINATOR, EXIT
6645
6646 036420 104401 055041   4$: TYPE     EMI      ;ONLY 0-7 ALLOWED.
6647 036424 000137 012742   JMP      PRGSRT    ;START ALL OVER
6648
6649 036430 005201          5$: INC      R1      ;S/B NO MORE THAN 8 DIFF
6650 036432 001330          BNE      1$        ;DRIVES TYPED IN.
6651 036434 000771          BR       4$        ;IF MORE, HAVE ERROR.
6652
6653 036436 005237 007540   6$: INC      SIZFLG  ;DO TEST 1 (SIZING)
6654 036442 000207          RTS      PC          ;EXIT.
6655
6656
6657 ;ROUTINE TO INPUT RKBAS OR DEFAULT.
6658 ;
6659
6660 036444 104412          GBA: RDOCT
6661 036446 012600          MOV      (SP)+,R0  ;GET LOW ORDER FROM STACK
6662 036450 005700          TST      R0
6663 036452 001403          BEQ      1$        ;BRANCH IF DEFAULT.
6664 036454 010037 001264   MOV      R0,$BASE
6665 036460 000207          RTS      PC
6666 036462 012737 177440 001264 1$: MOV      #177440,$BASE ;DEFAULT VALUE
6667 036470 000207          RTS      PC
6668
6669 ;ROUTINE TO INPUT RKVEC OR DEFAULT
6670 ;
6671 ;
6672 ;
6673 036472 104412          GINT: RDOCT
6674 036474 012600          MOV      (SP)+,R0 ;GET LOW ORDER FROM STACK
6675 036476 005700          TST      R0
6676 036500 001405          BEQ      1$        ;BRANCH IF DEFAULT
6677 036502 010037 001314   MOV      R0,RKVEC
6678 036506 004737 036524   2$: JSR      PC,SETINT
6679 036512 000207          RTS      PC
6680 036514 012737 000210 001314 1$: MOV      #210,RKVEC ;DEFAULT VALUE
6681 036522 000771          BR       2$
6682
6683 ;ROUTINE TO SETUP INTERRUPT VECTOR & PRIORITY
6684 ;
6685 ;
6686 ;
6687 036524 013700 001314   SETINT: MOV      RKVEC,R0
6688 036530 012720 044130   MOV      #INTER,(R0)+ ;INTER ADDR TO RKVEC

```

```

6689 036534 013710 001316      MOV      RKPRI,(R0)      ;PRS TO RKVEC+2
6690 036540 000207              RTS      PC
6691
6692
6693
6694
6695
6696
6697
6698
6699
6700 036542 032765 000200 000000  FRDY:   BIT      #RDY,RKCS1(R5)
6701 036550 001010              BNE     1$
6702 036552 005337 007426      DEC     TEMP1
6703 036556 001371              BNE     FRDY
6704 036560 004737 036676      JSR    PC,HOLD          ;STORE ALL RK611 REGS IN HOLDING REGS.
6705 036564 004737 040016      JSR    PC,CKCERR       ;CHECK FOR SPECIAL CERR
6706 036570 000207              RTS     PC              ;NO RDY, EXIT
6707 036572 062716 000002      1$:    ADD     #2,(SP)   ;SKIP OVER ERROR
6708 036576 004737 036676      JSR    PC,HOLD
6709 036602 004737 040016      JSR    PC,CKCERR       ;CHECK FOR SPECIAL CERR
6710 036606 000207              RTS     PC
6711
6712
6713
6714 036610 032765 000200 000000  FRDY1:  BIT      #RDY,RKCS1(R5)
6715 036616 001014              BNE     1$
6716 036620 005337 007426      DEC     TEMP1
6717 036624 001371              BNE     FRDY1
6718 036626 016537 000034 007416      MOV    RKMR2(R5),HMR2
6719 036634 016537 000036 007420      MOV    RKMR3(R5),HMR3
6720 036642 004737 040016      JSR    PC,CKCERR       ;CHECK FOR SPECIAL CERR CONDITIONS
6721 036646 000207              RTS     PC              ;NO RDY, EXIT
6722 036650 062716 000002      1$:    ADD     #2,(SP)   ;SKIP OVER ERROR
6723 036654 016537 000034 007416      MOV    RKMR2(R5),HMR2
6724 036662 016537 000036 007420      MOV    RKMR3(R5),HMR3
6725 036670 004737 040016      JSR    PC,CKCERR       ;CHECK FOR SPECIAL CERR CONDITIONS
6726 036674 000207              RTS     PC
6727
6728
6729
6730
6731
6732 036676 016537 000000 007370  HOLD:  MOV      RKCS1(R5),HCS1
6733 036704 016537 000010 007372      MOV    RKCS2(R5),HCS2
6734 036712 016537 000002 007374      MOV    RKWC(R5),HWC
6735 036720 016537 000004 007376      MOV    RKBA(R5),HBA
6736 036726 016537 000006 007400      MOV    RKDA(R5),HDA
6737 036734 016537 000012 007402      MOV    RKDS(R5),HDS
6738 036742 016537 000014 007404      MOV    RKER(R5),HER
6739 036750 016537 000016 007406      MOV    RKASOF(R5),HASOF
6740 036756 016537 000020 007410      MOV    RKDC(R5),HDC
6741 036764 016537 000026 007414      MOV    RKMR1(R5),HMR1
6742 036772 016537 000034 007416      MOV    RKMR2(R5),HMR2
6743 037000 016537 000036 007420      MOV    RKMR3(R5),HMR3
6744 037006 016537 000030 007422      MOV    RKECPS(R5),HPOS
    
```

```

6745 037014 016537 000032 007424      MOV    RRECPTR5),HPAT
6746 037022 000207                      RTS    PC
6747
6748
6749      ;ROUTINE TO CHECK FOR CORRECT ATTN
6750      ;RETURN IF ATTN NOT PRESENT (ERROR CONDITION)
6751      ;RETURN +2 IF ATTN PRESENT (SKIP OVER ERROR)
6752
6753 037024 010446      ;STATN: MOV    R4, -(SP)          ;SAV R4
6754 037026 013704 001222      MOV    $UNIT, R4
6755 037032 136437 007360 007407      BITB  ATTN(R4), HASOF+1
6756 037040 001404      BEQ    1$          ;BRANCH IF ATTN NOT PRESENT
6757 037042 012604      MOV    (SP)+, R4      ;RESTOR R4
6758 037044 062716 000002      ADD    #2, (SP)      ;INCR RET ADDR TO JUMP OVER ERROR.
6759 037050 000207                      RTS    PC
6760 037052 012604      1$:  MOV    (SP)+, R4      ;RESTOR R4
6761 037054 000207                      RTS    PC
6762
6763
6764      ;ROUTINE TO FIND ATTN WITHIN TIMES GREATER THAN 1 SEC
6765      ;ENTER WITH TIME IN SECONDS IN TEMP2
6766      ;RETURN IF NO ATTN (ERROR CONDITION)
6767      ;RETURN +2 IF ATTN FOUND
6768      ;STATUS IS OBTAINED BEFORE THE RETURN FOR EITHER CASE
6769
6770
6771 037056 010446      FATT1: MOV    R4, -(SP)          ;SAV R4
6772 037060 012737 177777 007426      3$:  MOV    #1, TEMP1
6773 037066 013704 001222      MOV    $UNIT, R4
6774 037072 136465 007360 000017      1$:  BITB  ATTN(R4), RKASOF+1(R5) ;FIND CORRECT ATTN
6775 037100 001014      BNE    2$
6776 037102 005337 007426      DEC    TEMP1
6777 037106 001371      BNE    1$
6778 037110 005337 007430      DEC    TEMP2
6779 037114 001361      BNE    3$
6780 037116 005065 000026      CLR    RKMRI(R5)      ;SELECT WORD 0
6781 037122 004737 040100      JSR    PC, GSTAT      ;GET LATEST STATUS
6782 037126 012604      MOV    (SP)+, R4      ;RESTOR R4
6783 037130 000207                      RTS    PC
6784 037132 005065 000026      2$:  CLR    RKMRI(R5)
6785 037136 004737 040100      JSR    PC, GSTAT      ;GET STATUS AFTER ATTN SEEN
6786 037142 012604      MOV    (SP)+, R4      ;RESTOR R4
6787 037144 062716 000002      ADD    #2, (SP)      ;SKIP OVER ERROR
6788 037150 000207                      RTS    PC
6789
6790
6791      ;ROUTINE TO FIND ATTN WITHIN 1 SEC
6792      ;ENTER WITH COUNT IN TEMP1
6793      ;RETURN IF NO ATTN (ERROR)
6794      ;RETURN +2 IF ATTN FOUND
6795      ;STATUS IS OBTAINED BEFORE THE RETURN FOR EITHER CASE
6796
6797
6798 037152 010446      FATT2: MOV    R4, -(SP)          ;SAV R4
6799 037154 013704 001222      2$:  MOV    $UNIT, R4
6800 037160 136465 007360 000017      BITB  ATTN(R4), RKASOF+1(R5) ;FIND CORRECT ATTN
  
```

6801	037166	001011		BNE	1\$		
6802	037170	005337	007426	DEC	TEMP1		
6803	037174	001367		BNE	2\$		
6804	037176	005365	000026	CLR	RKMR1(R5)	:SELECT WORD 0	
6805	037202	004737	040100	JSR	PC,GSTAT	:GET LATEST STATUS.	
6806	037206	012604		MOV	(SP)+,R4	:RESTOR R4	
6807	037210	000207		RTS	PC		
6808	037212	005365	000026	1\$: CLR	RKMR1(R5)		
6809	037216	004737	040100	JSR	PC,GSTAT		
6810	037222	012604		MOV	(SP)+,R4	:RESTOR R4	
6811	037224	062716	000002	ADD	#2,(SP)	:SKIP OVER ERROR	
6812	037230	000207		RTS	PC		
6813							
6814							
6815							
6816							
6817							
6818	037232	005737	007426	DLY:	TST TEMP1	:5.6 US	
6819	037236	001403			BEQ 1\$:1.9 US	
6820	037240	005337	007426		DEC TEMP1	:6.8 US	
6821	037244	000772			BR DLY	:2.5 US	
6822	037246	000207		1\$:	RTS PC	:3.8 US	
6823							
6824							
6825							
6826							
6827	037250	104401	052300	BYP:	TYPE MSG14	:BYPASS DRIVE	
6828	037254	010046			MOV RO,-(SP)	:SAVE RO FOR TYPEOUT	
6829						:TYPE DR#	
6830	037256	104403			TYPOS	:GO TYPE--OCTAL ASCII	
6831	037260	001			.BYTE 1	:TYPE 1 DIGIT(S)	
6832	037261	000			.BYTE 0	:SUPPRESS LEADING ZEROS	
6833	037262	000207			RTS PC		
6834							
6835							
6836							
6837	037264	017637	000000	001534	CHKMSG: MOV	2(SP),CHKFLG	:PASS MSGS TO BE TESTED
6838	037272	062716	000002		ADD	#2,(SP)	:BUMP RETURN ADDR TO 1ST ERROR
6839	037276	004737	040142		JSR	PC,GSTAT1	:GET ALL ACTUAL DRIVE & CONTR STATUS
6840							
6841	037302	053737	001222	007460	BIS	\$UNIT,E.A0	:SET JNIT #
6842	037310	053737	001222	007464	BIS	\$UNIT,E.A1	
6843	037316	053737	001222	007470	BIS	\$UNIT,E.A2	
6844	037324	053737	001222	007474	BIS	\$UNIT,E.A3	
6845							
6846	037332	013746	007426		MOV	TEMP1,-(SP)	:SAVE TEMP1
6847							
6848	037336	013737	007460	007426	MOV	E.A0,TEMP1	
6849	037344	004737	043354		JSR	PC,SBPARG	:GET PARITY FOR MSG A0
6850	037350	013737	007426	007460	MOV	TEMP1,E.A0	
6851							
6852	037356	013737	007464	007426	MOV	E.A1,TEMP1	
6853	037364	004737	043354		JSR	PC,SBPARG	:GET PARITY FOR MSG A1
6854	037370	013737	007426	007464	MOV	TEMP1,E.A1	
6855							
6856	037376	013737	007470	007426	MOV	E.A2,TEMP1	

:ENTER WITH A COUNT IN TEMP1
:THE DELAY IS APPROX 17 JS-ITERATION + 12 US TO EXIT
:WHEN COUNT IS 0. BASED ON AN 11/05

:THIS ROUTINE TYPES BYPASSED DRIVE#. ENTER WITH DRIVE# IN RO

:THIS ROUTINE READS ALL MSG A & B WORDS & CHECKS THEM AS REQ'D.

6857	037404	004737	043354			JSR	PC,SBPAR		:GET PARITY FOR MSG A2
6858	037410	013737	007426	00747C		MOV	TEMP1,E.A2		
6859									
6860	037416	013737	007462	007426		MOV	E.B0,TEMP1		
6861	037424	004737	043354			JSR	PC,SBPAR		:GET PARITY FOR MSG B0
6862	037430	013737	007426	007462		MOV	TEMP1,E.B0		
6863									
6864	037436	013737	007466	007426		MOV	E.B1,TEMP1		
6865	037444	004737	043354			JSR	PC,SBPAR		:GET PARITY FOR MSG B1
6866	037450	013737	007426	007466		MOV	TEMP1,E.B1		
6867									
6868	037456	013737	007472	007426		MOV	E.B2,TEMP1		
6869	037464	004737	043354			JSR	PC,SBPAR		:GET PARITY FOR MSG B2
6870	037470	013737	007426	007472		MOV	TEMP1,E.B2		
6871									
6872	037476	013737	007476	007426		MOV	E.B3,TEMP1		
6873	037504	004737	043354			JSR	PC,SBPAR		:GET PARITY FOR MSG B3
6874	037510	013737	007426	007476		MOV	TEMP1,E.B3		
6875									
6876	037516	012637	007426			MOV	(SP)+,TEMP1		:RESTORE TEMP1
6877	037522	013737	001176	001172		MOV	\$ESCAPE,\$TMP5		:SAVE ESCAPE
6878									
6879	037530	023737	007440	007460		CMP	H.A0,E.A0		:TEST MSG A0
6880	037536	001411				BEQ	2\$:BR IF OK
6881	037540	012737	037552	001176		MOV	#1\$, \$ESCAPE		:ELSE SETUP ESCAPE
6882	037546	011646				MOV	(SP),-(SP)		:COPY RET ADDR.
6883	037550	000207				RTS	PC		:& RETURN TO MAINLINE ERROR
6884									
6885	037552	032777	001000	141360	1\$:	BIT	#SW9,\$SWR		:RET HERE FROM MAINLINE ERROR
6886	037560	001107				BNE	20\$:& BR IF LOOP ON ERROR
6887	037562	062716	000002		2\$:	ADD	#2,(SP)		:BUMP RET ADDR TO NEXT ERROR
6888									
6889	037566	023737	007442	007462		CMP	H.B0,E.B0		:TEST MSG B0
6890	037574	001411				BEQ	5\$:BR IF OK
6891	037576	012737	037610	001176		MOV	#4\$, \$ESCAPE		:ELSE SETUP ESCAPE
6892	037604	011646				MOV	(SP),-(SP)		:COPY RET ADDR
6893	037606	000207				RTS	PC		:& RETURN TO MAINLINE ERROR
6894									
6895	037610	032777	001000	141322	4\$:	BIT	#SW9,\$SWR		:RETURN HERE FROM MAINLINE ERROR
6896	037616	001070				BNE	20\$:& BR IF LOOP ON ERROR
6897	037620	062716	000002		5\$:	ADD	#2,(SP)		:BUMP RET ADDR TO NEXT ERROR
6898									
6899	037624	023737	007444	007464		CMP	H.A1,E.A1		:TEST MSG A1
6900	037632	001411				BEQ	8\$:BR IF OK
6901	037634	012737	037646	001176		MOV	#7\$, \$ESCAPE		:ELSE SETUP ESCAPE
6902	037642	011646				MOV	(SP),-(SP)		:COPY RET ADDR
6903	037644	000207				RTS	PC		:& RETURN TO MAINLINE ERROR
6904									
6905	037646	032777	001000	141264	7\$:	BIT	#SW9,\$SWR		:RETURN HERE FROM MAINLINE ERROR
6906	037654	001051				BNE	20\$:& BR IF LOOP ON ERROR
6907	037656	062716	000002		8\$:	ADD	#2,(SP)		:BUMP RET ADDR TO NEXT ERROR
6908									
6909	037662	023737	007446	007466		CMP	H.B1,E.B1		:TEST MSG B1
6910	037670	001411				BEQ	11\$:BR IF OK
6911	037672	012737	037704	001176		MOV	#10\$, \$ESCAPE		:ELSE SETUP ESCAPE
6912	037700	011646				MOV	(SP),-(SP)		:COPY RET ADDR

E11

UNIBUS RKO6 DRIVE DIAGNOSTIC PART 2
DZR6TC.P11 07-OCT-76 13:50

MACY11 27(1006) 07-OCT-76 14:14 PAGE 134
GET VALUE FOR SOFTWARE SWITCH REGISTER

SEG 0134

```

6913 037702 000207          RTS      PC
6914
6915 037704 032777 001000 141226 10$: BIT      #SW9,2SWR
6916 037712 001032          BNE     20$
6917 037714 062716 000002          ADD     #2,(SP)
6918
6919 037720 032737 000001 001534 12$: BIT      #T.A2,CHKFLG ;TEST MSG A2?
6920 037726 001402          BEQ     13$ ;BR IF NO
6921 037730 004737 041026          JSR     PC,RCYLD ;PUT INFO CYLDIF, DO NOT CHECK
6922 037734 032737 000002 001534 13$: BIT      #T.B2,CHKFLG ;TEST MSG B2?
6923 037742 001402          BEQ     14$ ;BR IF NO
6924 037744 004737 041100          JSR     PC,RCYLA ;PUT INFO IN CYLADD, DO NOT CHECK
6925
6926 037750 032737 000004 001534 14$: BIT      #T.B3,CHKFLG ;TEST MSG B3?
6927 037756 001404          BEQ     15$
6928 037760 004737 041136          JSR     PC,RSEC ;PUT INFO IN SECTOR, DO NOT CHECK
6929 037764 004737 041174          JSR     PC,RHEAD ;PUT INFO IN HEADA, DO NOT CHECK
6930
6931 037770 013737 001172 001176 15$: MOV     $TMP5,$ESCAPE ;RESTORE ESCAPE
6932 037776 000207          RTS      PC
6933
6934 040000 012706 001100          MOV     #STACK,SP ;RESET STACK PTR
6935 040004 013737 001172 001176          MOV     $TMP5,$ESCAPE ;RESTORE ESCAPE
6936 040012 000177 141072          JMP     @SLPERR
6937
6938 ;THIS ROUTINE CHECKS FOR CERTAIN ERROR CONDITIONS ONLY
6939 ;I.E.: IF NED, CTO OR MDS SET MESSAGE A & B ARE INVALID
6940
6941 040016 005737 001532          CKCERR: TST     BYPCERR
6942 040022 001025          BNE     4$
6943 040024 032737 100000 007370          BIT      #CERR,HCS1
6944 040032 001001          BNE     1$ ;BR IF CERR
6945 040034 000207          RTS      PC
6946
6947 040036 032737 004000 007370 1$: BIT      #CTO,HCS1
6948 040044 001402          BEQ     2$ ;BR IF NOT CTO
6949 040046 104125          ERROR   125 ;CTO ERROR, MSG A & B INVALID
6950 040050 000207          RTS      PC
6951
6952 040052 032737 010000 007372 2$: BIT      #NED,HCS2
6953 040060 001401          BEQ     3$ ;BR IF NOT NED
6954 040062 104126          ERROR   126 ;NED ERROR, MSG A & B INVALID
6955
6956 040064 032737 001000 007372 3$: BIT      #MDS,HCS2
6957 040072 001401          BEQ     4$
6958 040074 104127          ERROR   127 ;MDS ERROR, MSG A & B INVALID
6959
6960 040076 000207          4$: RTS      PC
6961
6962 ;THIS ROUTINE DOES THE SELECT DRIVE COMMAND TO GET STATUS
6963 ;IT THEN WAITS FOR CONTROLLER READY
6964 ;IF RDY NOT RECEIVED BY THE TIMEOUT, AN ERROR IS FLAGGED
6965
6966
6967
6968 040100 013746 007426          GSTAT: MOV     TEMP1,-(SP) ;SAVE TEMP1

```

F11

JYIBUS RK06 DRIVE DIAGNOSTIC PART 2
 DZR6IC.P11 07-OCT-76 13:50

MACY11 27(1006) 07-OCT-76 14:14 PAGE 135
 GET VALUE FOR SOFTWARE SWITCH REGISTER

SEQ 0135

6969	040104	013765	001222	000010	MOV	\$UNIT,RKCS2(R5)	;CURRENT DRIVE #
6970	040112	012765	003001	000000	MOV	#SELDIV,RKCS1(R5)	;GET STATUS WITH SELECT DRIVE CMD
6971	040120	013737	001412	007426	MOV	T10,TEMP1	
6972	040126	004737	036542		JSR	PC,FRDY	;FIND RDY
6973	040132	104117			ERROR	117	;RDY NOT SET BY END OF SELECT DRIVE CMD
6974	040134	012637	007426		MOV	(SP)+,TEMP1	;RESTOR TEMP1.
6975	040140	000207			RTS	PC	
6976							
6977							
6978							
6979							
6980	040142	013746	007426				
6981	040146	004737	036676				
6982	040152	012765	100000	000000	GSTAT1: MOV	TEMP1,-(SP)	;SAVE TEMP1
6983	040160	013765	001222	000010	JSR	PC,HOLD	;GET ALL CONTR REG
6984	040166	012765	000003	000026	MOV	#CCLR,RKCS1(R5)	;CLEAR CONTR
6985	040174	012765	000001	000070	MOV	\$UNIT,RKCS2(R5)	;CURRENT DRIVE #
6986	040202	013737	001412	007426	MOV	#3,RKMR1(R5)	;SELECT WORD 3
6987	040210	004737	036610		MOV	#SELDIV,RKCS1(R5)	;GET STATUS
6988	040214	104117			MOV	T10,TEMP1	
6989	040216	013737	007416	007454	JSR	PC,FRDY1	;FIND RDY & STORE DRIVE REGS ONLY
6990	040224	013737	007420	007456	ERROR	117	;RDY NOT SET BY END OF SELECT DRV CMD
6991					MOV	HMR2,H.A3	;STORE MSG A3
6992	040232	012765	100000	000000	MOV	HMR3,H.B3	;STORE MSG B3
6993	040240	013765	001222	000010	MOV	#CCLR,RKCS1(R5)	
6994	040246	012765	000002	000026	MOV	\$UNIT,RKCS2(R5)	
6995	040254	012765	000001	000000	MOV	#2,RKMR1(R5)	;SELECT WORD 2
6996	040262	013737	001412	007426	MOV	#SELDIV,RKCS1(R5)	
6997	040270	004737	036610		MOV	T10,TEMP1	
6998	040274	104117			JSR	PC,FRDY1	;FIND RDY & STORE DRIVE REGS ONLY
6999	040276	013737	007416	007450	ERROR	117	;RDY NOT SET BY END OF SELECT DRV CMD
7000	040304	013737	007420	007452	MOV	HMR2,H.A2	;STORE MSG A2
7001					MOV	HMR3,H.B2	;STORE MSG B2
7002	040312	012765	100000	000000	MOV	#CCLR,RKCS1(R5)	
7003	040320	013765	001222	000010	MOV	\$UNIT,RKCS2(R5)	
7004	040326	012765	000001	000026	MOV	#1,RKMR1(R5)	;SELECT WORD 1
7005	040334	012765	000001	000000	MOV	#SELDIV,RKCS1(R5)	
7006	040342	013737	001412	007426	MOV	T10,TEMP1	
7007	040350	004737	036610		JSR	PC,FRDY1	;FIND RDY & STORE DRIVE REGS ONLY
7008	040354	104117			ERROR	117	;RDY NOT SET BY END OF SELECT DRV CMD
7009	040356	013737	007416	007444	MOV	HMR2,H.A1	;STORE MSG A1
7010	040364	013737	007420	007446	MOV	HMR3,H.B1	;STORE MSG B1
7011							
7012	040372	012765	100000	000000	MOV	#CCLR,RKCS1(R5)	
7013	040400	013765	001222	000010	MOV	\$UNIT,RKCS2(R5)	
7014	040406	012765	000001	000000	MOV	#SELDIV,RKCS1(R5)	;SELECT WORD 0
7015	040414	013737	001412	007426	MOV	T10,TEMP1	
7016	040422	004737	036610		JSR	PC,FRDY1	;FIND RDY & STORE DRIVE REGS ONLY
7017	040426	104117			ERROR	117	;RDY NOT SET BY END OF SEL DRV CMD
7018	040430	013737	007416	007440	MOV	HMR2,H.A0	;STORE MSG A0
7019	040436	013737	007420	007442	MOV	HMR3,H.B0	;STORE MSG B0
7020							
7021	040444	012637	007426		MOV	(SP)+,TEMP1	;RESTORE TEMP1
7022	040450	000207			RTS	PC	
7023							
7024							

G11

UNIBUS RK06 DRIVE DIAGNOSTIC PART 2
DZR6IC.P11 07-OCT-76 13:50

MACY11 27(1006) 07-OCT-76 14:14 PAGE 136
GET VALUE FOR SOFTWARE SWITCH REGISTER

SEQ 0136

7025				
7026				
7027				
7028				
7029				
7030				
7031	040452	012765	000040	000010
7032	040460	013737	001412	007426
7033	040466	004737	036542	
7034	040472	104120		
7035	040474	013765	001222	000010
7036	040502	005065	000026	

```

: THIS ROUTINE DOES A SUBSYSTEM CLEAR & WAITS FOR CONTROLLER READY
: IF RDY IS NOT RECEIVED BY THE END OF THE TIMEOUT, AN ERROR IS FLAGGED.
: THE ROUTINE THEN GETS CURRENT STATUS & CHECKS FOR CONTROLLER ERROR (CERR)
: RETURN IF CERR SET
: RETURN +2 IF CERR CLEAR

```

```

SUBCLR: MOV      #SCLR,RKCS2(R5)  :SUBSYS CLEAR
        MOV      T10,TEMP1
        JSR      PC,FRDY         :FIND RDY
        ERROR    120             :RDY NOT SET BY END OF SCLR
        MOV      $UNIT,RKCS2(R5) :CURRENT DRIVE #
        CLR      RKMRI(R5)      :SELECT WORD 0

```

H11

UNIBUS Rk06 DRIVE DIAGNOSTIC PART 2
DZR6IC.P11 07-OCT-76 13:50

MACY11 27(1006) 07-OCT-76 14:14 PAGE 137
GET VALUE FOR SOFTWARE SWITCH REGISTER

SEQ 0137

7037	040506	004737	040100		JSR	PC, GSTAT	:GET STATUS
7038	040512	032737	100000	007370	BIT	#CERR, HCS1	:CHECK FOR CONT ERROR
7039	040520	001401			BEQ	1\$	
7040	040522	000207			RTS	PC	
7041	040524	062716	000002	1\$:	ADD	#2, (SP)	:SKIP OVER ERROR
7042	040530	000207			RTS	PC	
7043							
7044							

UNIBUS RK06 DRIVE DIAGNOSTIC PART 2
DZR6IC.P11 07-OCT-76 13:50

I11

MACY11 27(1006) 07-OCT-76 14:14 PAGE 138
GET VALUE FOR SOFTWARE SWITCH REGISTER

SEG 0138

7045
7046

;READ THE SECTOR COUNT IN RKMR3, RIGHT JUSTIFY IT & STORE IT IN 'SECTOR'
;

```

7047 040532 012765 000003 000026 RDSEC: MOV #3,RKMR1(R5) ;WORD 3
7048 040540 004737 040100 JSR PC,GSTAT
7049 040544 013737 007420 001406 MOV HMR3,SECTOR
7050 040552 042737 177017 001406 BIC #1C<M.SECT>,SECTOR
7051 040560 006237 001406 ASR SECTOR ;RIGHT JUSTIFY
7052 040564 006237 001406 ASR SECTOR ;SECTOR
7053 040570 006237 001406 ASR SECTOR ;INFO
7054 040574 006237 001406 ASR SECTOR
7055 040600 000207 RTS PC
7056
7057 ;READ THE CYL DIFF/OFFSET IN RKMR2, RIGHT JUSTIFY IT & STORE IT IN 'CYLDIF'
7058
7059 040602 012765 000002 000026 RDCYLD: MOV #2,RKMR1(R5) ;WORD 2
7060 040610 004737 040100 JSR PC,GSTAT
7061 040614 013737 007416 001362 MOV HMR2,CYLDIF
7062 040622 042737 160017 001362 BIC #1C<M.CDIF>,CYLDIF
7063 040630 006237 001362 ASR CYLDIF ;RIGHT JUSTIFY
7064 040634 006237 001362 ASR CYLDIF ;CYL DIFF/OFFSET
7065 040640 006237 001362 ASR CYLDIF ;INFO
7066 040644 006237 001362 ASR CYLDIF
7067 040650 023727 001362 000777 CMP CYLDIF,#777 ;CHK TO SEE IF RET IN COMPL. FORM
7068 040656 001002 BNE IS ;BR IF NOT
7069 040660 005037 001362 CLR CYLDIF ;CLR IF YES
7070 040664 000207 1S: RTS PC
7071
7072
7073 ;QUICK SELECT DRIVE COMMAND TO OBTAIN CYL DIFF
7074
7075 040666 013746 007426 QKCYLD: MOV TEMP1,-(SP) ;SAVE TEMP1
7076 040672 012765 000002 000026 MOV #2,RKMR1(R5) ;SELECT WORD 2
7077 040700 012765 000001 000000 MOV #SELDRV,RKCS1(R5) ;SELECT DRIVE CMD
7078 040706 013737 001412 007426 MOV T10,TEMP1
7079 040714 032765 000200 000000 1S: BIT #RDY,RKCS1(R5) ;TEST FOR CONT RDY
7080 040722 001004 BNE 2S ;BR IF THERE
7081 040724 005337 007426 DEC TEMP1
7082 040730 001371 BNE 1S
7083 040732 104117 ERROR 117 ;NO RDY AFTER SEL DRV CMD
7084
7085 040734 016537 000034 001362 2S: MOV RKMR2(R5),CYLDIF
7086 040742 042737 160017 001362 BIC #1C<M.CDIF>,CYLDIF ;GET CYL DIFF ONLY (NO SHIFTING)
7087 040750 012637 007426 MOV (SP)+,TEMP1 ;RESTORE TEMP1
7088 040754 000207 RTS PC
7089
7090 ;READ THE CYL ADDR IN RKMR3, RIGHT JUSTIFY IT & STORE IT IN 'CYLADD'
7091
7092 040756 012765 000002 000026 RDCYLA: MOV #2,RKMR1(R5) ;WORD 2
7093 040764 004737 040100 JSR PC,GSTAT
7094 040770 013737 007420 001364 MOV HMR3,CYLADD
7095 040776 042737 160017 001364 BIC #1C<M.CADD>,CYLADD
7096 041004 006237 001364 ASR CYLADD ;RIGHT JUSTIFY
7097 041010 006237 001364 ASR CYLADD ;CYL ADDR
7098 041014 006237 001364 ASR CYLADD ;INFO
7099 041020 006237 001364 ASR CYLADD
7100 041024 000207 RTS PC
7101
7102 ;READ THE CYL DIFF/OFFSET IN H.A2, RIGHT JUSTIFY IT & STORE IT IN 'CYLDIF'

```

```

7103
7104 041026 013737 007450 001362 RCYLD: MOV H.A2,CYLDIF
7105 041034 042737 160017 001362 BIC #1C<M.CDIF>,CYLDIF ;CLEAR UNWANTED INFO
7106 041042 006237 001362 ASR CYLDIF ;RIGHT JUSTIFY
7107 041046 006237 001362 ASR CYLDIF
7108 041052 006237 001362 ASR CYLDIF
7109 041056 006237 001362 ASR CYLDIF
7110 041062 023727 001362 000777 CMP CYLDIF,#777 ;CHK TO SEE IF RET IN COMPL. FORM
7111 041070 001002 BNE 1$ ;BR IF NO
7112 041072 005037 001362 CLR CYLDIF ;ELSE CLEAR
7113 041076 000207 1$: RTS PC
7114
7115 ;READ THE CYL ADDR IN H.B2, RIGHT JUSTIFY IT & STORE IT IN 'CYLADD'
7116
7117 041100 013737 007452 001364 RCYLA: MOV H.B2,CYLADD
7118 041106 042737 160017 001364 BIC #1C<M.CADD>,CYLADD ;CLEAR UNWANTED INFO
7119 041114 006237 001364 ASR CYLADD ;RIGHT JUSTIFY
7120 041120 006237 001364 ASR CYLADD
7121 041124 006237 001364 ASR CYLADD
7122 041130 006237 001364 ASR CYLADD
7123 041134 000207 RTS PC
7124
7125 ;READ THE SECTOR COUNT IN H.B3, RIGHT JUSTIFY IT & STORE IT IN 'SECTOR'
7126
7127 041136 013737 007456 001406 RSEC: MOV H.B3,SECTOR
7128 041144 042737 177017 001406 BIC #1C<M.SECT>,SECTOR ;CLEAR UNWANTED INFO
7129 041152 006237 001406 ASR SECTOR ;RIGHT JUSTIFY
7130 041156 006237 001406 ASR SECTOR
7131 041162 006237 001406 ASR SECTOR
7132 041166 006237 001406 ASR SECTOR
7133 041172 000207 RTS PC
7134
7135 ;READ THE HEAD ADDR IN H.B3, RIGHT IT & STORE IT IN 'HEADA'
7136
7137 041174 013737 007456 001476 RHEAD: MOV H.B3,HEADA
7138 041202 042737 170777 001476 BIC #1C<M.HEAD>,HEADA ;CLEAR UNWANTED INFO
7139 041210 006237 001476 ASR HEADA ;RIGHT JUSTIFY IT
7140 041214 000337 001476 SWAB HEADA
7141 041220 000207 RTS PC
7142
7143 ;FIND SECTOR 23
7144 ;RETURN IF NOT FOUND
7145 ;RETURN +4 IF FOUND
7146
7147 041222 013737 001422 007426 FSEC23: MOV T5000,TEMP1 ;SETUP TIMEOUT
7148 041230 004737 040532 1$: JSR PC,RDSEC ;READ SECTOR
7149 041234 023727 001406 000023 CMP SECTOR,#23 ;TEST FOR SECTOR 23
7150 041242 001014 BNE 2$ ;BR IF NOT 23
7151
7152 041244 004737 040532 JSR PC,RDSEC
7153 041250 023727 001406 000023 CMP SECTOR,#23
7154 041256 001412 BEQ 3$ ;BR IF READ SAME TWICE
7155 041260 004737 040532 JSR PC,RDSEC ;ELSE TRY 1 MORE TIME
7156 041264 023727 001406 000023 CMP SECTOR,#23
7157 041272 001404 BEQ 3$ ;BR IF 17

```

```

7158
7159 041274 005337 007426
7160 041300 001353
7161 041302 000207
7162
7163 041304 062716 000004
7164 041310 000207
7165
7166 ;ROUTINE TO FIND HEADS HOME IN RKMR2 WORD 1 BEFORE SPECIFIED DELAY
7167 ;ENTER WITH TIME IN SECONDS IN TEMP2
7168 ;RETURN IF NOT FOUND
7169 ;RETURN+2 IF FOUND - SKIP OVER ERROR
7170
7171 041312 012737 177777 007426 FHDHM: MOV # -1, TEMP1 ;ALL 1'S
7172 041320 012765 000001 000026 MOV #1, RKMR1(R5) ;WORD 1
7173 041326 004737 040100 1S: JSR PC, GSTAT
7174 041332 032737 000040 007416 BIT #D, HDHM, HMR2
7175 041340 001007 BNE 2S
7176 041342 005337 007426 DEC TEMP1
7177 041346 001367 BNE 1S
7178 041350 005337 007430 DEC TEMP2
7179 041354 001356 BNE FHDHM
7180 041356 000207 RTS PC
7181 041360 062716 000002 2S: ADD #2, (SP) ;SKIP OVER ERROR
7182 041364 000207 RTS PC
7183
7184 ;ROUTINE TO FIND LOAD HEADS IN RKMR2 WORD 1 BEFORE THE TIMEOUT
7185 ;RETURN IF NOT FOUND
7186 ;RETURN+2 IF FOUND: SKIP OVER ERROR
7187
7188 041366 012737 000372 007426 FLOAD: MOV #25C, TEMP1
7189 041374 012765 000001 000026 MOV #1, RKMR1(R5) ;WORD 1
7190 041402 004737 040100 1S: JSR PC, GSTAT
7191 041406 032737 010000 007416 BIT #D, LOAD, HMR2
7192 041414 001004 BNE 2S
7193 041416 005337 007426 DEC TEMP1
7194 041422 001367 BNE 1S
7195 041424 000207 RTS PC
7196 041426 062716 000002 2S: ADD #2, (SP) ;SKIP OVER ERROR
7197 041432 000207 RTS PC
7198
7199 ;FILL HEADER TABLE WITH 66 WORDS OF VALID HEADERS
7200 ;ENTER WITH CYL # IN 'CALADD'
7201 ;ENTER WITH HEAD # IN 'HEAD'
7202 ;ENTER WITH FORMAT IN 'FORMAT'
7203
7204 041434 010046 FHDTAB: MOV R0, -(SP) ;SAV R0
7205 041436 010146 MOV R1, -(SP) ;SAV R1
7206 041440 012700 001536 MOV #HDTAB, R0 ;HEADER WORD TABLE ADDR
7207 041444 005001 CLR R1 ;SECTOR COUNTER
7208 041446 013737 001474 001500 MOV HEAD, HD1
7209 041454 006337 001500 ASL HD1
7210 041460 006337 001500 ASL HD1
7211 041464 006337 001500 ASL HD1
7212 041470 006337 001500 ASL HD1
7213 041474 006337 001500 ASL HD1 ;SETUP HEAD # FOR WORD 2 OF HEADER

```

M11

UNIBUS Rk06 DRIVE DIAGNOSTIC PART 2
DZR6IC.P11 07-OCT-76 13:50

MACY11 27(1006) 07-OCT-76 14:14 PAGE 142
GET VALUE FOR SOFTWARE SWITCH REGISTER

SEQ 0142

```

7214 041500 013737 001502 001504      MOV    FORMAT,FMT1
7215 041506 000337 001504              SWAB   FMT1
7216 041512 006337 001504              ASL    FMT1          ;SETUP FORMAT FOR WORD 2 OF HEADER
7217
7218 041516 013720 001366      1$:   MOV    CALADD,(R0)+  ;HEADER WORD 1-CYL ADDR
7219 041522 010110              MOV    R1,(R0)       ;HEADER WORD 2-SECTOR NO
7220 041524 053710 001500              BIS    HD1,(R0)      ;
7221 041530 053710 001504              BIS    FMT1,(R0)    ;
7222 041534 004737 041614              JSR    PC,SECFLG    ;
7223
7224 041540 013737 001366 007426      MOV    CALADD,TEMP1
7225 041546 011037 007430              MOV    (R0),TEMP2
7226 041552 043737 001366 007430      BIC    CALADD,TEMP2
7227 041560 042037 007426              BIC    (R0)+,TEMP1
7228 041564 053737 007426 007430      BIS    TEMP1,TEMP2
7229 041572 013720 007430              MOV    TEMP2,(R0)+  ;HEADER WORD 3-HEADER CHECK
7230
7231 041576 005201              INC    R1            ;SECTOR CTR
7232 041600 020127 000026      CMP    R1,#22       ;ALL 22 SECTORS DONE? (66 WORDS)
7233 041604 001344              BNE    1$           ;BR IF NO
7234
7235 041606 012601              MOV    (SP)+,R1     ;RESTOR R1
7236 041610 012600              MOV    (SP)+,R0     ;RESTOR R0
7237 041612 000207              RTS    PC
7238
7239      ; THIS ROUTINE GETS INFORMATION FROM THE BAD SECTOR TABLE FILLED BY A PREVIOUS
7240      ; TEST & SETS BITS 14 & 15 APPROPRIATLY.
7241
7242 041614 010246      SECFLG: MOV    R2,-(SP)   ;SAVE R2
7243 041615 005737 001502      TST    FORMAT
7244 041622 001016              BNE    1$           ;BR IF 20 SECTOR FORMAT
7245 041624 012702 003362      MOV    #BSE22H+8.,R2
7246 041630 004737 041714      JSR    PC,FLGTST   ;GET HARDWARE DETECTED FLAG
7247 041634 052710 100000      BIS    #BIT15,(R0) ;RETURN HERE IF GOOD SECTOR
7248
7249 041640 012702 005362      MOV    #BSE22S+8.,R2 ;ELSE RETURN HERE
7250 041644 004737 041714      JSR    PC,FLGTST   ;GET SOFTWARE DETECTED FLAG
7251 041650 052710 040000      BIS    #BIT14,(R0) ;RETURN HERE IF GOOD SECTOR
7252
7253 041654 012602              MOV    (SP)+,R2     ;ELSE RETURN HERE
7254 041656 000207              RTS    PC
7255
7256 041660 012702 002362      1$:   MOV    #BSE20H+8.,R2
7257 041664 004737 041714      JSR    PC,FLGTST   ;GET HARDWARE DETECTED FLAG
7258 041670 052710 100000      BIS    #BIT15,(R0) ;RETURN HERE IF GOOD SECTOR
7259
7260 041674 012702 004362      MOV    #BSE20S+8.,R2
7261 041700 004737 041714      JSR    PC,FLGTST   ;GET SOFTWARE DETECTED FLAG
7262 041704 052710 040000      BIS    #BIT14,(R0) ;RETURN HERE IF GOOD SECTOR
7263
7264 041710 012602              MOV    (SP)+,R2     ;RESTC
7265 041712 000207              RTS    PC
7266
7267
7268      ; THIS ROUTINE DOES THE ACTUAL SCANNING OF THE BAD SECTOR TABLES
7269      ; ENTER WITH THE ADDRESS OF TABLE (BSE22H, BSE22S, ETC.) IN TEMP1

```

```

7270 ;RETURN IF NO COMPARE
7271 ;RETURN+4 IF COMPARE
7272 ;
7273 041714 010346 FLGTST: MOV R3,-(SP) ;SAVE R3
7274
7275 041716 021227 177777 1S: CMP (R2),#-1 ;SEE IF ALL 1'S
7276 041722 001002 BNE 2S ;BR IF NO
7277 041724 012603 MOV (SP)+,R3 ;RESTORE R3
7278 041726 000207 RTS PC
7279
7280 041730 022237 001366 2S: CMP (R2)+,CALADD ;SEE IF=CYL # & ADR PTR TO TRK/SECTOR WORD
7281 041734 001403 BEQ 3S
7282 041736 062702 000002 ADD #2,R2 ;GO TO NEXT CYL WORD IN TABLE
7283 041742 000765 BR 1S
7284
7285 041744 013703 001474 3S: MOV HEAD,R3 ;GET HEAD # FROM FHDTAB ROUTINE
7286 041750 000303 SWAB R3
7287 041752 050103 BIS R1,R3 ;ADD SECTOR # FROM FHDTAB ROUTINE
7288 041754 022203 CMP (R2)+,R3 ;SEE IF SECTOR/HEAD COMPARE
7289 ;& INCR PTR TO NEXT CYL WORD
7290 041756 001401 BEQ 4S ;BR IF COMPARE
7291 041760 000756 BR 1S ;ELSE TRY NEXT CYL
7292
7293 041762 012603 4S: MOV (SP)+,R3 ;RESTORE R3
7294 041764 062716 000004 ADD #4,(SP) ;INCREMENT RET ADDR
7295 041770 000207 RTS PC
7296
7297 ;
7298 ;THIS ROUTINE SORTS THE RHTAB TABLE FROM WHATEVER SECTOR IT BEGINS
7299 ;WITH AND RE-WRITES THE INFO IN SRTTAB TABLE TO BEGIN WITH SECTOR 0
7300 ;
7301 041772 010046 SORT: MOV R0,-(SP) ;SAVE R0
7302 041774 010146 MOV R1,-(SP) ;SAVE R1
7303 041776 004737 040532 JSR PC,RDSEC
7304 042002 062737 000001 001406 ADD #1,SECTOR
7305 042010 004737 042100 JSR PC,MULT6 ;MULT SECTOR BY 6
7306
7307 042014 012700 000204 MOV #132,R0
7308 042020 163700 001406 SUB SECTOR,R0 ;R0-SECTOR TO R0 = INDEX
7309 042024 010037 001406 MOV R0,SECTOR
7310 042030 062737 001742 001406 ADD #RHTAB,SECTOR ;SAVE INDEX
7311
7312 042036 062700 001742 ADD #RHTAB,R0 ;INDEX TO BOT HALF OF RHTAB
7313 042042 012701 002146 MOV #SRTTAB,R1 ;INDEX TO TOP HALF OF SRTTAB
7314
7315 042046 012021 1S: MOV (R0)+,(R1)+ ;PUT BOTTOM OF RHTAB TO TOP OF SRTTAB
7316 042050 020027 002146 CMP R0,#RHTAB+132.
7317 042054 001374 BNE 1S
7318
7319 042056 012700 001742 2S: MOV #RHTAB,R0 ;PUT TOP OF RHTAB TO BOT OF SRTTAB
7320 042062 012021 MOV (R0)+,(R1)+
7321 042064 020037 001406 CMP R0,SECTOR
7322 042070 001374 BNE 2S
7323
7324 042072 012601 MOV (SP)+,R1 ;RESTOR R1
7325 042074 012600 MOV (SP)+,R0 ;RESTOR R0

```

```

7326 042076 000207          RTS      PC
7327
7328
7329
7330
7331 042100 006337 001406      MULT BY 6. ENTER WITH DESIRED # IN 'SECTOR'
7332 042104 013746 001406      MULT#6: ASL      SECTOR      ;2 X SECTOR
7333 042110 006337 001406      MOV      SECTOR, -(SP)
7334 042114 062637 001406      ASL      SECTOR      ;4 X SECTOR
7335 042120 000207          ADD      (SP)+, SECTOR ;(4 X 5)+(2 X 5) = 6 X SECTOR
7336
7337
7338
7339
7340
7341
7342
7343
7344
7345 042122 010446      TRUERR: MOV     R4, -(SP)      ;SAVE R4
7346
7347 042124 032737 010000 007370      BIT      #CFMT, HCS1      ;CHECK FORMAT
7348 042132 001014          BNE      2$              ;BR FOR 20 SECTOR FORMAT
7349
7350 042134 012704 003362      MOV     #BSE224+B., R4
7351 042140 004737 042222      JSR     PC, TERR1        ;SEE IF ON HARDWARE DETECTED TABLE
7352 042144 000422          BR      3$              ;RETURN HERE IF YES
7353
7354 042146 012704 005362      MOV     #BSE225+B., R4
7355 042152 004737 042222      JSR     PC, TERR1        ;SEE IF ON SOFTWARE DETECTED TABLE
7356 042156 000415          BR      3$              ;RETURN HERE IF YES
7357
7358 042160 012604          1$: MOV     (SP)+, R4      ;RESTORE R4
7359 042162 000207          RTS      PC              ;RETURN WITHOUT JUMPING OVER ERROR
7360
7361 042164 012704 002362      2$: MOV     #BSE204+B., R4
7362 042170 004737 042222      JSR     PC, TERR1        ;SEE IF ON HARDWARE DETECTED TABLE
7363 042174 000406          BR      3$              ;RETURN HERE IF YES
7364
7365 042176 012704 004362      MOV     #BSE205+B., R4
7366 042202 004737 042222      JSR     PC, TERR1        ;SEE IF ON SOFTWARE DETECTED TABLE
7367 042206 000401          BR      3$              ;RETURN HERE IF YES
7368 042210 000763          BR      1$              ;RETURN HERE IF NO
7369
7370 042212 012604          3$: MOV     (SP)+, R4      ;RESTORE R4
7371 042214 062716 000002      ADD     #2, (SP)         ;SKIP OVER ERROR ON RETURN
7372 042220 000207          RTS      PC
7373
7374
7375
7376
7377
7378
7379
7380 042222 021427 177777      THIS ROUTINE DOES THE ACTUAL COMPARING OF CYLINDER, HEAD & TRACK AGAINST
7381 042226 001405          THE BSE TABLE FOR THE ABOVE SUBROUTINE.
7382
7383
7384
7385
7386
7387
7388
7389
7390
7391
7392
7393
7394
7395
7396
7397
7398
7399
7400
7401
7402
7403
7404
7405
7406
7407
7408
7409
7410
7411
7412
7413
7414
7415
7416
7417
7418
7419
7420
7421
7422
7423
7424
7425
7426
7427
7428
7429
7430
7431
7432
7433
7434
7435
7436
7437
7438
7439
7440
7441
7442
7443
7444
7445
7446
7447
7448
7449
7450
7451
7452
7453
7454
7455
7456
7457
7458
7459
7460
7461
7462
7463
7464
7465
7466
7467
7468
7469
7470
7471
7472
7473
7474
7475
7476
7477
7478
7479
7480
7481
7482
7483
7484
7485
7486
7487
7488
7489
7490
7491
7492
7493
7494
7495
7496
7497
7498
7499
7500
    
```

7382	042230	022437	001354		CMP	(R4)+,CCYL	:SEE IF CYL MATCH
7383	042234	001405			BEQ	2\$:BR IF YES
7384	042236	005724			TSI	(R4)+	:ELSE ADV TO NEXT CYL WORD
7385	042240	000770			BR	TERR1	:& TRY AGAIN.
7386							
7387	042242	062716	000002	1\$:	ADD	#2.(SP)	
7388	042246	000207			RTS	PC	
7389							
7390	042250	022437	007400	2\$:	CMP	(R4)+,HDA	:SEE IF SECTOR & TRACK MATCH
7391	042254	001401			BEQ	3\$:BR IF YES
7392	042256	000761			BR	TERR1	:OR TRY AGAIN
7393							
7394	042260	000207		3\$:	RTS	PC	

7395
 7396
 7397
 7398
 7399
 7400
 7401
 7402
 7403
 7404
 7405
 7406
 7407
 7408
 7409
 7410
 7411
 7412
 7413
 7414
 7415
 7416
 7417
 7418
 7419
 7420
 7421
 7422
 7423
 7424
 7425
 7426
 7427
 7428
 7429
 7430
 7431
 7432
 7433
 7434
 7435
 7436
 7437
 7438
 7439
 7440
 7441
 7442
 7443
 7444
 7445
 7446
 7447
 7448
 7449
 7450

```

042262 104413
042264 005000
042266 005037 001456
042272 005037 001460
042276 012737 000001 001372
042304 012737 000001 001374
042312 012737 177777 007426
042320 004737 043236
042324 005737 001376
042330 001007
042332 005337 007426
042336 001372
042340 104401 053022
042344 000137 036034
042350 012737 000001 001372
042356 012737 000001 001374
042364 005037 001452
042370 005037 001376
042374 005737 001376
042400 001003
042402 004737 042726
042406 000772
042410 004737 043332
042414 063737 001452 001456
042422 005537 001460
042426 005200
042430 020027 000400
042434 001320
042436 000337 001456
042442 105037 001457
  
```

```

: THIS ROUTINE CALIBRATES ANY PDP-11 AGAINST 2 CONSECUTIVE 16MS TICKS FROM AN L OR 16SC
: THE 1ST TICK CLEARS A COUNTER 'LPCNT'. THE PROGRAM THEN GOES THRU A LOOP.
: EACH TIME IT GOES THRU THE LOOP, LPCNT IS INCREMENTED BY 1.
: THE 2ND TICK STOPS INCREMENTING LPCNT AND THE CLOCK INTERRUPT IS TURNED OFF.
: NOW, FOR THIS PARTICULAR PDP-11, THE TIME TO GO THRU THE ABOVE LOOP ONE IS
: 16.6MS DIVIDED BY THE CONTENTS OF LPCNT.
: AN APPROX VALUE FOR A PDP11-05 MAY BE 0.4MS TO GO THRU THE LOOP ONCE
: FROM THIS POINT ON, WHENEVER ACCURACIES GREATER THAN WHAT THE L CLOCK
: CAN PROVIDE ARE NECESSARY, THE EVENT TO BE TIMED IS COMPARED AGAINST
: THE NUMBER OF TIMES THE LOOP IS PASSED THRU AND MULT. BY THE ABOVE
: CALCULATED FIGURE
  
```

```

CALCLK: SAVREG
CLR RO ; ITERATION CTR
CLR SUM ; SUM OF 256 ITERATIONS
CLR SUM+2
5$: MOV #1,COUNT ; GO THRU CLOCK HANDLER
MOV #1,SEC ; ONLY ONCE
MOV #-1,TEMP1 ; ALL 1'S FOR TIMEOUT
JSR PC,CLKON
1$: TST TIMUP
BNE 2$ ; BR IF GOT 1'ST TICK
DEC TEMP1
BNE 1$ ; BR IF TIMEOUT NOT DONE
TYPE MSG23 ; NO CLOCK INTR PRESENT, ABORT TIMING TEST
JMP $EOP ; ABORT DRIVE
2$: MOV #1,COUNT ; GOT 1'ST TICK
MOV #1,SEC
CLR LPCNT ; LOOP COUNTER
CLR TIMUP ; CLEAR BEFORE 2'ND TICK
3$: TST TIMUP
BNE 4$ ; BR IF GOT 2'ND TICK
JSR PC,LOOP
BR 3$
4$: JSR PC,CLKOFF ; GOT 2'ND TICK, TURN OFF CLOCK
ADD LPCNT,SUM
ADC SUM+2 ; ADD POSSIBLE OVERFLOW
INC RO
CMP RO,#256. ; ALL ITERATIONS DONE?
BNE 5$ ; BR IF NO
SWAB SUM ; DIVIDE BY 256
CLRB SUM+1 ; TO GET AVERAGE
  
```

E12

UNIBUS RK06 DRIVE DIAGNOSTIC PART 2
DZR6IC.P11 07-OCT-76 13:50

MACY11 27(1006) 07-OCT-76 14:14 PAGE 147
GET VALUE FOR SOFTWARE SWITCH REGISTER

SEQ 0147

7451	042446	005000				CLR	R0	;CLEAR FOR DIV
7452	042450	005001				CLR	R1	
7453	042452	005002				CLR	R2	
7454	042454	005004				CLR	R4	
7455	042456	012703	040432			MOV	#16666.,R3	;LSB DIVIDEND (16666 US)
7456	042462	013705	001456			MOV	SUM,R5	;DIVISOR (AVERAGE OF 256)
7457	042466	004737	044252			JSR	PC,M.DPID	;DO DIVIDE
7458	042472	010337	001454			MOV	R3,LPTIM	;STORE QUOTIENT
7459								;THIS EQUALS THE TIME IN USEC
7460								;TO GO THRU THE LOOP ONCE
7461	042476	104414				RESREG		
7462	042500	000207				RTS	PC	
7463								
7464								
7465								
7466								
7467								
7468								
7469	042502	104413						
7470	042504	005000				CLR	R0	;ITERATION CTR
7471	042506	005037	001456			CLR	SUM	;SUM OF 256 ITERATIONS
7472	042512	005037	001460			CLR	SUM+2	
7473								
7474	042516	012737	000001	001372	5\$:	MOV	#1,COUNT	;SEE ABOVE CALCLK FOR COMMENTS
7475	042524	012737	000001	001374		MOV	#1,SEC	
7476	042532	012737	177777	007+26		MOV	#-1,TEMP1	
7477	042540	004737	043236			JSR	PC,CLKON	
7478								
7479	042544	005737	001376		1\$:	TST	TIMUP	
7480	042550	001007				BNE	2\$	
7481	042552	005337	007426			DEC	TEMP1	
7482	042556	001372				BNE	1\$	
7483	042560	104401	053022			TYPE	MSG23	;NO CLOCK
7484	042564	000137	036034			JMP	\$EOP	
7485								
7486	042570	012737	000001	001372	2\$:	MOV	#1,COUNT	
7487	042576	012737	000001	001374		MOV	#1,SEC	
7488	042604	005037	001452			CLR	LPCNT	
7489	042610	005037	001376			CLR	TIMUP	
7490								
7491	042614	005737	001376		3\$:	TST	TIMUP	
7492	042620	001005				BNE	4\$	
7493	042622	004737	040666			JSR	PC,QKCYLD	;THIS IS ONLY DIFFERENCE BETWEEN
7494								;PRECEDING SUBROUTINE
7495	042626	004737	042726			JSR	PC,LOOP	
7496	042632	000770				BR	3\$	
7497								
7498	042634	004737	043332		4\$:	JSR	PC,CLKOF	
7499	042640	063737	001452	001456		ADD	LPCNT,SUM	
7500	042646	005537	001460			ADC	SUM+2	;ADD POSSIBLE OVERFLOW
7501								
7502	042652	005200				INC	R0	
7503	042654	020027	000400			CMP	R0,#256.	
7504	042660	001316				BNE	5\$	
7505								
7506	042662	000337	001456			SWAB	SUM	

;; THIS ROUTINE IS EXACTLY THE SAME AS THE ABOVE EXCEPT THAT
 ;; A JSR PC QKCYLD HAS BEEN INSERTED TO TAKE UP SOME TIME
 ;; TO MATCH THE TIME TAKEN BY THE ROUTINE THAT USES THIS ROUTINE

VELCAL: SAVREG

7507 042666 105037 001457
7508
7509 042672 005000
7510 042674 005001
7511 042676 005002
7512 042700 005004
7513 042702 012703 040432
7514 042706 013705 001456
7515 042712 004737 044252
7516 042716 010337 001454
7517 042722 104414
7518 042724 000207
7519
7520
7521
7522
7523
7524
7525
7526
7527
7528
7529 042726 010046
7530 042730 012700 000400
7531 042734 005300
7532 042736 001401
7533 042740 000775
7534 042742 005237 001452
7535 042746 012600
7536 042750 000207
7537
7538
7539
7540 042752 136465 007360 000017
7541 042750 001006
7542 042762 004737 042726
7543 042766 005737 001452
7544 042772 001367
7545 042774 000207
7546 042776 062716 000002
7547 043002 000207
7548
7549
7550
7551
7552
7553 043004
7554 043004 013746 001454
7555 043010 013746 001160
7556 043014 004737 050332
7557 043020 012637 001160
7558 043024 012637 001162
7559 043030 012746 001160
7560
7561 043034 004737 050042
7562 043040 004737 050272

CLRB SUM+1
CLR R0
CLR R1
CLR R2
CLR R4
MOV #16666.,R3
MOV SUM,R5
JSR PC,M.DPID
MOV R3,LPTIM
RESREG
RTS PC

;; WITH 50 IN R0, IT TAKES APPROX 400 US FOR AN 11/05 TO GO THRU THIS
;; LOOP AND INCREMENT 'LPCNT' ONCE.
;; THIS 400 US IS APPROX 2.5% OF 16MS GIVING A RESOLUTION OF 0.4MS
;; PER COUNT IN 'LPCNT'.
;; WHEN USED BY THE 'CALCLK' ROUTINE, LPCLK SHOULD BE APPROX 40(10)=50(8)
;; WITH AN 11/05 TO 200(10)=264(8) FOR AN 11/70

LOOP: MOV R0,-(SP) ;SAVE R0
MOV #400,R0
1\$: DEC R0
BEQ 2\$
BR 1\$
2\$: INC LPCNT
MOV (SP)+,R0 ;RESTORE R0
RTS PC

;; THIS ROUTINE FINDS ATTN FROM A SEEK AND RETURNS

FATT3: BITB ATTN(R4),RKASOF+1(R5) ;TEST FOR ATTN
BNE 1\$;EXIT IF THERE
JSR PC,LOOP ;ELSE GO THRU LOOP
TST LPCNT ;TEST FOR OVERFLOW
BNE FATT3 ;BR IF NO AND TRY AGAIN
RTS PC
1\$: ADD #2,(SP) ;JUMP OVER ERROR
RTS PC

;; ROUTINE TO MULT & TYPE A 2 WORD PRODUCT
;; ENTER WITH LPCNT IN \$TMP0
;; USED FOR LOW VEL TIMES IN UNLD & LOADING

TYPTIM:
MOV LPTIM,-(SP) ;;PUT THE MULTIPLIER ON THE STACK
MOV \$TMP0,-(SP) ;;PUT THE MULTIPLICAND ON THE STACK
JSR PC,\$MULT ;;CALL THE MULTIPLY ROUTINE
MOV (SP)+,\$TMP0 ;;GET THE LSB'S OF THE PRODUCT
MOV (SP)+,\$TMP0+2 ;;GET THE MSB'S OF THE PRODUCT
MOV #TMP0,-(SP) ;;PUSH LSB ONTO STACK
JSR PC,\$OB2D ;;MSB IN \$TMP1 FROM MULT MACRO
JSR PC,\$SUPRS ;;CONVERT TO ASCII STRING
;TYPE TIMES

G12

UNIBUS RK06 DRIVE DIAGNOSTIC PART 2
DZR6IC.P11 07-OCT-76 13:50

MACY11 27(1006) 07-OCT-76 14:14 PAGE 149
GET VALUE FOR SOFTWARE SWITCH REGISTER

SEG 0149

7563	043044	104401	053151
7564	043050	000207	

TYPE	MSG25	: MICRO SEC
RTS	PC	

7565			
7566			
7567			
7568			

: ROUTINE USED BY TIMING TO OBTAIN THE AVERAGE OF 100 TIMES IN MICRO SECONDS

7569	043052	104413	
7570	043054	005000	
7571	043056	005001	
7572	043060	005004	
7573	043062	013703	001456
7574	043066	013702	001460
7575	043072	012705	000144
7576	043076	004737	043204
7577	043102	104401	053151
7578	043106	104414	
7579	043110	000207	
7580			
7581			
7582			
7583			

```
AVGTIM: SAVREG
        CLR R0 ;CLEAR FOR DIV
        CLR R1
        CLR R4
        MOV SUM,R3 ;LSB DIVIDEND
        MOV SUM+2,R2 ;MSB DIVIDEND
        MOV #100.,R5 ;DIVISOR
        JSR PC, DIVTYP
        TYPE ,MSG25 ;USEC
        RESREG
        RTS PC
```

7580			
7581			
7582			
7583			
7584	043112	104413	
7585	043114	005000	
7586	043116	005001	
7587	043120	013703	001456
7588	043124	013702	001460
7589	043130	012705	103240
7590	043134	012704	000001
7591			
7592			
7593	043140	004737	044252
7594	043144	010337	001162
7595			
7596	043150	005000	
7597	043152	005001	
7598	043154	005002	
7599	043156	005004	
7600	043160	012703	001231
7601	043164	013705	001162
7602	043170	004737	043204
7603	043174	104401	054117
7604	043200	104414	
7605	043202	000207	
7606			
7607			
7608			
7609	043204	004737	044252
7610	043210	010337	001160
7611	043214	010237	001162
7612	043220	012746	001160
7613	043224	004737	050042
7614	043230	004737	050272
7615	043234	000207	
7616			
7617			
7618			

: ROUTINE TO CALC & TYPE AVERAGE SPEED

```
AVGSP: SAVREG
        CLR R0 ;CLEAR FOR DIV
        CLR R1
        MOV SUM,R3 ;LSB DIVIDEND
        MOV SUM+2,R2 ;MSB DIVIDEND
        MOV #103240,R5 ;LSB DIVISOR
        MOV #1,R4 ;MSB DIVISOR
        ;TO DIVIDE BY 100000(10)
        ;100000(10)=303240(8)
        JSR PC,M.DPID ;GO DIVIDE
        MOV R3,STMP1 ;SAVE QUOTIENT

        CLR R0 ;CLEAR FOR DIV
        CLR R1
        CLR R2
        CLR R4
        MOV #665.,R3 ;LSB DIVIDEND
        MOV STMP1,R5 ;DIVISOR
        JSR PC, DIVTYP
        TYPE ,MSG36 ;INCHES/SEC
        RESREG
        RTS PC
```

: ROUTINE TO DIVIDE & TYPE WITH SUPPRESSED 0'S

```
DIVTYP: JSR PC,M.DPID ;GO DIVIDE
        MOV R3,STMP0 ;STORE LSB QUOTIENT
        MOV R2,STMP1 ;STORE MSB QUOTIENT
        MOV #STMP0,-(SP) ;PUSH LSB ON STACK
        JSR PC,$DB2D ;CONVERT TO ASCII
        JSR PC,$SUPRS ;TYPE IT
        RTS PC
```

: ROUTINE TO TURN L OR P CLOCK INTERRUPT ON

```

7619 043236 005037 001376      CLKON: CLR      TIMUP
7620 043242 005737 007534      TST      PCLKF
7621 043246 001004                BNE      1$          ;BRANCH IF P-CLOCK PRESENT
7622 043250 012777 000100 136050      MOV      #100,ALKS  ;L-CLOCK, ENABLE INT
7623 043256 000207                RTS      PC
7624 043260 012777 177777 136034 1$: MOV      #-1,APKSB ;P-CLOCK, ALL 1'S
7625 043266 012777 000135 136024      MOV      #135,APKS ;ENABLE INT, CT UP, REP INT
7626 043274 000207                RTS      PC          ;LINE FREQ & RUN
7627
7628      ;KW11-L & KW11-P INTERRUPT HANDLER
7629
7630 043276 005037 001376      CLOCK: CLR      TIMUP
7631 043302 005337 001372      DEC      COUNT
7632 043306 001010                BNE      1$
7633 043310 013737 001370 001372      MOV      HZ,COUNT
7634 043316 005337 001374      DEC      SEC
7635 043322 001002                BNE      1$
7636 043324 005237 001376      INC      TIMUP      ;SORRY, TIME IS UP
7637 043330 000002      1$: RTI
7638
7639      ;ROUTINE TO TURN L OR P CLOCK INTERRUPT OFF
7640
7641 043332 005737 007534      CLKOF: TST      PCLKF
7642 043336 001003                BNE      1$          ;BRACH IF P-CLOCK PRESENT
7643 043340 005077 135762      CLR      ALKS      ;L-CLOCK, CLEAR INTERRUPT
7644 043344 000207                RTS      PC
7645 043346 005077 135746      1$: CLR      APKSB ;P-CLOCK, CLEAR INTERRUPT
7646 043352 000207                RTS      PC
7647
7648      ;
7649      ;THIS ROUTINE GENERATES PARITY FOR THE EXPECTED MESSAGE
7650      ;ENTER WITH THE EXPECTED WORD IN TEMP1
7651      ;TEMP1 IS ROTATED LEFT 17 TIMES. EACH TIME THE CARRY BIT IS SET,
7652      ;R1 IS INCREMENTED. AT THE END OF 17 ROTATES (TEMP1 BACK TO ORIG).
7653      ;R1 BIT 0 IS EXAMINED. IF IT IS SET, INDICATING AN ODD # OF 1'S.
7654      ;THE PARITY BIT IS NOT SET IN B
7655      ;IF IT IS NOT SET, INDICATING AN EVEN # OF 1'S .THE PARITY BIT IS
7656      ;SET IN TEMP1
7657
7658 043354 010046      SBPAR: MOV      RO,-(SP) ;SAVE RO
7659 043356 010146      MOV      R1,-(SP) ;SAVE R1
7660 043360 012700 000021      MOV      #17,RO    ;SHIFT COUNTER
7661 043364 005001      CLR      R1        ;COUNT # OF 1'S IN TEMP1
7662 043366 000241      CLC                ;CLEAR CARRY
7663
7664 043370 006137 007426      1$:  ROL      TEMP1
7665 043374 103001      BCC      2$        ;BR IF CARRY CLEAR
7666 043376 005201      INC      R1        ;COUNT # OF 1'S
7667 043400 005300      2$:  DEC      RO
7668 043402 001372      BNE      1$        ;SHIFT COUNTER
7669
7670 043404 032701 000001      BIT      #BIT0,R1
7671 043410 001003      BNE      3$
7672 043412 052737 100000 007426      BIS      #M.PAR,TEMP1 ;SET PARITY BIT
7673 043420 012601      3$:  MOV      (SP)+,R1 ;RESTORE R1
7674 043422 012600      MOV      (SP)+,RO  ;RESTORE RO

```

```

7675 043424 000207          RTS      PC
7676
7677
7678          ; ROUTINE TO ENABLE LOOPING ON INTERMITTANT ERRORS
7679          ; WHEN $LPERR SET BY OTHER THAN SCOPE ROUTINE
7680          ; IE: MY LOOP MACRO
7681
7682 043426 032777 001000 135504 SCOPE1$ BIT      #SW9, $SWR      ; LOOP ON ERROR?
7683 043434 001406          BEQ      1$          ; BR IF NO
7684 043436 105737 001103          TSTB     $ERFLG      ; HAD ERROR?
7685 043442 001403          BEQ      1$          ; BR IF NO
7686 043444 013716 001110          MOV      $LPERR, (SP)
7687 043450 000002          RTI
7688
7689 043452 011637 001110          1$:      MOV      (SP), $LPERR      ; SET LOOP ADDR FOR TIGHT SCOPE LOOP
7690 043456 000002          RTI
7691
7692          ; THIS ROUTINE IS ENTERED BY TYPING A CONTROL-C.
7693          ; IT IS USED TO ALLOW THE OPERATOR TO HALT THE CPU WHILE INSURING
7694          ; THAT HEADS ARE LOADED & FORMATTING IS VALID BEFORE ACTUALLY HALTING
7695          ; THE CPU.
7696
7697 043460 022626          STOP:   CMP      (SP)+, (SP)+      ; RESTORE STACK FROM INTERRUPT
7698
7699 043462 004737 040452          JSR      PC, SUBCLR
7700 043466 104024          ERROR   24          ; CERR AFTER
7701
7702 043470 005737 007352          TST      UNLD          ; SEE IF HEADS UNLOADED
7703 043474 001432          BEQ      3$          ; BR IF NO
7704 043476 005737 000042          TST      42          ; SEE IF MANUAL OR AUTO MODE
7705 043502 001403          BEQ      1$          ; BR IF MANUAL MODE
7706 043504 104401 054677          TYPE    ,MSG74      ; PGM ABORT PENDING
7707 043510 000402          BR      2$
7708 043512 104401 054745          1$:      TYPE    ,MSG75      ; HALT PENDING
7709 043516          2$:
7710
7711 043516 004737 040452          JSR      PC, SUBCLR
7712 043522 104024          ERROR   24          ; CERR AFTER SCLR
7713
7714 043524 012765 000011 000000          MOV      #SRTSPL, RKCS1(R5) ; START SPINDLE CMD
7715 043532 013737 001412 007426          MOV      T10, TEMP1      ; SET TIMEOUT
7716 043540 004737 036542          JSR      PC, FRDY      ; FIND RDY
7717 043544 104121          ERROR   121         ; RDY NOT SET AFTER ST SPIN CMD.
7718
7719 043546 013737 001420 007430          MOV      T100, TEMP2     ; SETUP TIMEOUT
7720 043554 004737 037056          JSR      PC, FATT1      ; FIND ATTN
7721 043560 104074          ERROR   74          ; NO ATTN AFTER ST SPIN CMD.
7722
7723
7724 043562 005737 007354          3$:      TST      BADHDR      ; SEE IF HEADERS VALID
7725 043566 001466          BEQ      4$          ; BR IF YES
7726 043570 005237 007356          INC      HPEND
7727
7728 043574 012765 100000 000000          MOV      #CCLR, RKCS1(R5)
7729 043602 013765 001222 000010          MOV      $UNIT, RKCS2(R5)
7730 043610 012765 000013 000000          MOV      #RECAL, RKCS1(R5) ; RECAL CMD
  
```

```

7731                                     ;RESET CYL DIFF/OFFSET & CYL ADDR REG
7732                                     ;IN RKMR2 & RKMR3 RESP.
7733 043616 013737 001412 007426      MOV    T10,TEMP1      ;SETUP TIMEOUT
7734 043624 004737 036542              JSR    PC,FRDY       ;FIND RDY
7735 043630 104124                      ERROR  124           ;RDY NOT SET AFTER RECAL CMD
7736
7737 043632 012765 000001 000026      MOV    #1,RKMR1(R5)  ;SELECT WORD 1
7738 043640 004737 040100              JSR    PC,GSTAT
7739 043644 032737 020000 007416      BIT    #D,RTZ,HMR2
7740 043652 001001                      BNE    64$
7741 043654 104244                      ERROR  244           ;RTZ NOT SET DURING RECAL CMD
7742 043656 013737 001412 007430 64$: MOV    T10,TEMP2      ;SETUP TIMEOUT
7743 043664 004737 037056              JSR    PC,FATT1     ;FIND ATTN
7744 043670 104055                      ERROR  55           ;NO ATTN AFTER RECAL CMD
7745
7746 043672 012765 100000 000000      MOV    #CCLR,RKCS1(R5)
7747 043700 013765 001222 000010      MOV    $UNIT,RKCS2(R5) ;DRIVE#
7748 043706 012765 000005 000000      MOV    #CLEAR,RKCS1(R5) ;DRIVE CLEAR CMD
7749 043714 013737 001412 007426      MOV    T10,TEMP1      ;SETUP TIMEOUT
7750 043722 004737 036542              JSR    PC,FRDY       ;FIND RDY
7751 043726 104151                      ERROR  151           ;NO RDY AFTER DRIVE CLEAR CMD
7752 043730 004737 037024              JSR    PC,TSTATN    ;TEST FOR ATTN
7753 043734 000401                      BR     65$
7754 043736 104154                      ERROR  154           ;ATTN NOT CLEARED AFTER DRIVE CLEAR CMD
7755 043740                               65$:
7756
7757
7758 043740 000137 031420              JMP    FORM          ;WRITE VALID FORMATS
7759
7760 043744 005737 000042              4$: TST    42          ;SEE IF MANUAL OR AUTO MODE
7761 043750 001406                      BEQ    5$            ;BR IF MANUAL MODE
7762 043752 104401 055002              TYPE  ,MSG76         ;PGM ABORTED
7763 043756 005037 036110              CLR    $EOPCT        ;SET UP EOP TO EXIT TO MONITOR
7764 043762 000137 036034              JMP    $EOP
7765
7766 043766 104401 055024              5$: TYPE  ,MSG77         ;CPU HALTED
7767 043772 000000                      HALT
7768 043774 000137 013534              JMP    ST5           ;START OVER IF CONTINUE PRESSED
7769
7770
7771
7772 .SBTTL UNEXPECTED TIMEOUT HANDLER
7773
7774
7775 ;THIS ROUTINE IS ENTERED IF THERE IS
7776 ; A. NON EXISTANT MEMORY (NO SSYN)
7777 ; B. BOUNDARY ERROR
7778 ; C. STACK OVERFLOW
7779 ;
7780
7781 044000 011600      BADTMO: MOV    (SP),RO      ;SAVE PC WHERE TIMEOUT OCCURRED.
7782 044002 005740      TST    -(RO)         ;GET PC BEFORE UPDATE
7783 044004 032777 020000 135126      BIT    #SW13,$SWR    ;INHIBIT ERR TYP0UT?
7784 044012 001005      BNE    1$            ;YES, DON'T TYPE
7785 044014 104401 055205      TYPE  ,EM3           ;ABORT TESTS, UNEXP T.O. @ PC=
7786 044020 010046      MOV    RO,-(SP)      ;SAVE RO FOR TYP0UT

```

```

7787                                     ;; TYPE PC
7788 044022 104403 TYPOS                      ;; GO TYPE--OCTAL ASCII
7789 044024      006 .BYTE 6                ;; TYPE 6 DIGIT(S)
7790 044025      000 .BYTE 0                ;; SUPPRESS LEADING ZEROS
7791 044026 032777 001000 135104 1$: BIT #SW9,2SWR ;; LOOP ON ERROR?
7792 044034 001403 BEQ 2$                  ;; NO BRANCH
7793 044036 022626 CMP (SP)+,(SP)+        ;; YES, RESTORE STACK
7794 044040 000177 135042 JMP 2$LPADR     ;; GO TO STARTING ADDR OF TEST
7795                                     ;; THAT GAVE BAD TIMEOUT
7796 044044 032777 040000 135066 2$: BIT #SW14,2SWR ;; LOOP ON TEST?
7797 044052 001401 BEQ 3$                  ;; NO BRANCH
7798 044054 000002 RTI                    ;; YES
7799
7800 044056 000000 3$: HALT                ;; UNEXPECTED TIME OUT OCCURRED
7801                                     ;; AS INDICATED. YOU CAN LOOP ON
7802                                     ;; ERROR, LOOP ON TEST OR INHIBIT
7803                                     ;; ERROR TYPEOUT BY SETTING THOSE
7804                                     ;; SWITCHES.
7805
7806 044060 022626 CMP (SP)+,(SP)+        ;; RESTORE STACK
7807 044062 000137 036062 JMP $EOP1     ;; ABORT TESTS
7808
7809 .SBTTL MEMORY CHECK ENABLE TRAP
7810
7811 044066 012737 044102 001176 MEMERR: MOV #1$, $ESCAPE
7812 044074 011637 001334 MOV (SP), TRAPPC ;; STORE PC
7813 044100 104041 ERROR 41                ;; UNEXP MEM PARITY TRAP
7814 044102 005037 001176 1$: CLR $ESCAPE
7815 044106 032777 001000 135024 BIT #SW9,2SWR ;; CHECK IF LOOP ON ERROR
7816 044114 001001 BNE 2$                  ;; YES, FORCE STACK AND TRY AGAIN
7817 044116 000002 RTI                    ;; ELSE RETURN
7818
7819 044120 012706 001100 2$: MOV #STACK,SP ;; INIT STACK
7820 044124 000177 134760 JMP 2$LPERR ;; LOOP ON ERROR
7821
7822 .SBTTL RK06 INTERRUPT HANDLER
7823
7824 INTER: NOP
7825 044130 000240 NOP
7826 044132 000240 NOP
7827 044134 000240 MOV (SP),RO ;; SAVE PC WHERE INT OCCURRED.
7828 044136 011600 TST -(RO) ;; GET PC BEFORE UPDATE.
7829 044140 005740 TYPE MSG6 ;; INT AT PC=
7830 044142 104401 051762 MOV RO,-(SP) ;; SAVE RO FOR TYPEOUT
7831 044146 010046 ;; TYPE PC
7832                                     ;; GO TYPE--OCTAL ASCII
7833 044150 104403 TYPOS                      ;; TYPE 6 DIGIT(S)
7834 044152      006 .BYTE 6                ;; SUPPRESS LEADING ZEROS
7835 044153      000 .BYTE 0
7836 044154 000000 HALT
7837 044156 000240 NOP
7838 044160 000240 NOP
7839 044162 000002 RTI
7840
7841 .SBTTL POWER DOWN AND UP ROUTINES
7842

```

```

7843      :POWER DOWN ROUTINE
7844
7845 044164 012737 044176 000024 $PWRDN: MOV    #SPWRUP,PWRVEC ;SET UP VECTOR
7846 044172 000000                    HALT
7847 044174 000776                    BR      .-2          ;HANG UP.
7848
7849      ;POWER UP ROUTINE
7850
7851 044176 005037 044250 $PWRUP: CLR    $PWRCT          ;WAIT LOOP FOR TTY
7852 044202 005237 044250 1$:      INC    $PWRCT          ;WAIT FOR THE INCR
7853 044206 001375                    BNE    1$              ;OF WORD
7854 044210 012737 044164 000024  MOV    #SPWRDN,PWRVEC    ;SET POWER DOWN VECTOR
7855 044216 012737 000340 000026  MOV    #PR7,PWRVEC+2    ;PRIORITY 7
7856 044224 012737 000340 000036  MOV    #PR7,TRAPVEC+2  ;LOCKOUT ALL INTERRUPTS FOR TRAPS
7857 044232 012706 001100                    MOV    #STACK,SP      ;INITIALIZE STACK
7858 044236 104401 052152                    TYPE   ,MSG11         ;REPORT POWER FAIL
7859 044242 000005                    RESET
7860 044244 000137 015312                    JMP    PFSRT
7861
7862 044250 000000 $PWRCT: 0          ;WAIT COUNT FOR TTY
7863
7864      ;
7865      ;DIVISION UTILITY ROUTINE
7866      ;
7867      ;R0-R1-R2-R3=DIVIDEND
7868      ;R4-R5=DIVISOR
7869      ;R0-R1=REMAINDER AFTER DIVISION
7870      ;R2-R3=QUOTIENT AFTER DIVISION
7871      ;ENTER WITH JSR PC,M.DPID
7872      ;
7873 044252 012746 000040 M.DPID: MOV    #40,-(SP)    ;COUNTER FOR DIVISION CYCLES
7874 044256 010446                    MOV    R4,-(SP)        ;HI ORDER
7875 044260 010546                    MOV    R5,-(SP)        ;LO ORDER TO THE STACK
7876 044262 005466 000002                    NEG    2(SP)           ;FORM NEGATIVE
7877 044266 005416                    NEG    @SP              ;VERSION OF DIVISOR
7878 044270 005666 000002                    SBC    2(SP)
7879 044274 061601                    ADD    @SP,R1
7880 044276 005500                    ADC    R0                ;PERFORM INIT SUBT.
7881 044300 066600 000002                    ADD    2(SP),R0
7882 044304 103445                    BCS    M.DP50           ;IF CARRY THEN OVERFLOW HAS OCCURRED
7883 044306 005046                    CLR    -(SP)            ;THIS IS A LONGER LASTING CARRY BIT
7884 044310 006103 M.DP40: ROL    R3
7885 044312 006102                    ROL    R2
7886 044314 006101                    ROL    R1
7887 044316 006100                    ROL    R0
7888 044320 005716                    TST    @SP              ;TEST CARRY INDICATOR
7889 044322 001410                    BEQ    M.DP41           ;IF TO CARRY THEN ADD, ELSE SUBT.
7890 044324 005016                    CLR    @SP              ;CLEAR UP FOR NEXT TIME
7891 044326 066601 000002                    ADD    2(SP),R1
7892 044332 005500                    ADC    R0                ;ADD -(DIVISOR)
7893 044334 005516                    ADC    @SP              ;SET CARRY
7894 044336 066600 000004                    ADD    4(SP),R0
7895 044342 000404                    BR     M.DP42
7896
7897 044344 060501 M.DP41: ADD    R5,R1
7898 044346 005500                    ADC    R0                ;ADD +(DIVISOR)

```

7999	044350	005516		ADC	QSP		:SET CARRY
7900	044352	060400		ADD	R4, R0		
7901	044354	005516		M.DP42: ADC	QSP		:SET CARRY
7902	044356	005716		TST	QSP		:TEST THE UPDATE INDICATOR
7903	044360	001401		BEQ	.+4		:IF 0, FORGET IT
7904	044362	005203		INC	R3		:NO CARRY POSSIBLE HERE
7905	044364	005366	000006	DEC	6(SP)		:DECREMENT CTR
7906	044370	003347		BGT	M.DP40		:BR IF MORE TO DO
7907	044372	006003		ROR	R3		
7908	044374	103404		BCS	M.DP44		
7909	044376	060501		ADD	R5, R1		
7910	044400	005500		ADC	R0		
7911	044402	060400		ADD	R4, R0		
7912	044404	000241		CLC			
7913							
7914	044406	006103		M.DP44: ROL	R3		
7915	044410	062706	000010	ADD	#10, SP		:ADJUST STACK BY 4 WORDS
7916	044414	000242		CLV			
7917	044416	000207		RTS	PC		
7918							
7919	044420	062706	000006	M.DP50: ADD	#6, SP		
7920	044424	000262		SEV			
7921	044426	000237		RTS	PC		
7922							

```

7923
7924
7925
7926
7927
7928
7929
7930
7931
7932
7933
7934
7935
7936
7937 044430
7938 044430 104407
7939 044432 032777 040000 134500
7940 044440 001114
7941
7942 044442 000416
7943
7944 044444 013746 000004
7945 044450 012737 044470 000004
7946 044456 005737 177060
7947 044462 012637 000004
7948 044466 000463
7949 044470 022626
7950 044472 012637 000004
7951 044476 000423
7952 044500
7953 044500 032777 000400 134432
7954 044506 001404
7955 044510 127737 134424 001102
7956 044516 001465
7957 044520 105737 001103
7958 044524 001421
7959 044526 123737 001115 001103
7960 044534 101015
7961 044536 032777 001000 134374
7962 044544 001404
7963 044546 013737 001110 001106
7964 044554 000446
7965 044556 105037 001103
7966 044562 005037 001174
7967 044566 000415
7968 044570 032777 004000 134342
7969 044576 001011
7970 044600 005737 001216
7971 044604 001406
7972 044606 005237 001104
7973 044612 023737 001174 001104
7974 044620 002024
7975 044622 012737 000001 001104
7976 044630 013737 044706 001174
7977 044636 105237 001102
7978 044642 113737 001102 001114

```

```

.SBTTL SCOPE HANDLER ROUTINE
;*****
;THIS ROUTINE CONTROLS THE LOOPING OF SUBTESTS. IT WILL INCREMENT
;AND LOAD THE TEST NUMBER($TSTNM) INTO THE DISPLAY REG.(DISPLAY<7:0>)
;AND LOAD THE ERROR FLAG ($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

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

```

```

7989 044650 011637 001106
7990 044654 011637 001110
7991 044660 005037 001176
7992 044664 112737 000001 001115
7993 044672 013777 001102 134242
7994 044700 013716 001106
7995 044704 000002
7996 044706 003720

```

```

MOV (SP), SLPADR ;; SAVE SCOPE LOOP ADDRESS
MOV (SP), SLPERR ;; SAVE ERROR LOOP ADDRESS
CLR $ESCAPE ;; CLEAR THE ESCAPE FROM ERROR ADDRESS
MOV# B1, $ERMAX ;; ONLY ALLOW ONE(1) ERROR ON NEXT TEST
$OVER: MOV $STSTNM, $DISPLAY ;; DISPLAY TEST NUMBER
MOV SLPADR, (SP) ;; FUDGE RETURN ADDRESS
RTI ;; FIXES PS
$MXCNT: 2000. ;; MAX. NUMBER OF ITERATIONS
.SBTTL ERROR HANDLER ROUTINE

```

```

*****
* THIS ROUTINE WILL INCREMENT THE ERROR FLAG AND THE ERROR COUNT,
* SAVE THE ERROR ITEM NUMBER AND THE ADDRESS OF THE ERROR CALL
* AND GO TO TYPERR ON ERROR
* THE SWITCH OPTIONS PROVIDED BY THIS ROUTINE ARE:
* SW15=1 HALT ON ERROR
* SW13=1 INHIBIT ERROR TYPEOUTS
* SW10=1 BELL ON ERROR
* SW09=1 LOOP ON ERROR
* CALL ERROR N ;; ERROR=EMT AND N=ERROR ITEM NUMBER

```

```

8000 044710 104407
8001 044712 105237 001103
8002 044716 001775
8003 044720 013777 001102 134214
8004 044726 032777 002000 134204
8005 044734 001402
8006 044736 104401 001200
8007 044742 005237 001112
8008 044746 011637 001116
8009 044752 162737 000002 001116
8010 044760 117737 134132 001114
8011 044766 032777 020000 134144
8012 044774 001004
8013 044776 004737 064710
8014 045002 104401 001205
8015 045006 122737 000001 001230
8016 045014 001007
8017 045016 113737 001114 045030
8018 045024 004737 045634
8019 045030 000
8020 045031 000
8021 045032 000777
8022 045034 005777 134100
8023 045040 100002
8024 045042 000000
8025 045044 104407
8026 045046 032777 001000 134064
8027 045054 001402
8028 045056 013716 001110
8029 045062 005737 001176
8030 045066 001402
8031 045070 013716 001176

```

```

$ERROR:
75: CKSWR ;; TEST FOR CHANGE IN SOFT-SWR
INCB $ERFLG ;; SET THE ERROR FLAG
BEQ 75 ;; DON'T LET THE FLAG GO TO ZERO
MOV $STSTNM, $DISPLAY ;; DISPLAY TEST NUMBER AND ERROR FLAG
BIT #BIT10, $SWR ;; BELL ON ERROR?
BEQ 15 ;; NO - SKIP
TYPE $SBELL ;; RING BELL
15: INC $ERTL ;; COUNT THE NUMBER OF ERRORS
MOV (SP), $ERRPC ;; GET ADDRESS OF ERROR INSTRUCTION
SUB #2, $ERRPC
MOV# $ERRPC, $ITEMB ;; STRIP AND SAVE THE ERROR ITEM CODE
BIT #BIT13, $SWR ;; SKIP TYPEOUT IF SET
BNE 205 ;; SKIP TYPEOUTS
JSR PC, TYPERR ;; GO TO USER ERROR ROUTINE
TYPE , $CALF

205: CMPB #APTENV, $ENV ;; RUNNING IN APT MODE
BNE 25 ;; NO SKIP APT ERROR REPORT
MOV# $ITEMB, 215 ;; SET ITEM NUMBER AS ERROR NUMBER
JSR PC, SATY4 ;; REPORT FATAL ERROR TO APT

215: .BYTE 0
.BYTE 0

225: BR 225 ;; APT ERROR LOOP
25: TST $SWR ;; HALT ON ERROR
BPL 35 ;; SKIP IF CONTINUE
HALT ;; HALT ON ERROR!
35: BIT #BIT09, $SWR ;; TEST FOR CHANGE IN SOFT-SWR
BEQ 45 ;; LOOP ON ERROR SWITCH SET?
BR IF NO
MOV SLPERR, (SP) ;; FUDGE RETURN FOR LOOPING
45: TST $ESCAPE ;; CHECK FOR AN ESCAPE ADDRESS
BEQ 55 ;; BR IF NONE
MOV $ESCAPE, (SP) ;; FUDGE RETURN ADDRESS FOR ESCAPE

```

```

8035 045074
8036 045074 022737 036150 000042
8037 045074 001001
8038 045074 000000
8039 045074
8040 045074 000002
8041
8042
8043
8044
8045
8046
8047
8048
8049
8050
8051
8052
8053
8054
8055
8056
8057
8058 045110 105737 001157
8059 045114 100002
8060 045116 000000
8061 045120 000430
8062 045122 010046
8063 045124 017600 000002
8064 045130 122737 000001 001230
8065 045136 001011
8066 045140 132737 000100 001231
8067 045146 001405
8068 045150 010037 045160
8069 045154 004737 045624
8070 045160 000000
8071 045162 132737 000040 001231
8072 045170 001003
8073 045172 112046
8074 045174 001005
8075 045176 005726
8076 045200 012500
8077 045202 062716 000002
8078 045206 000002
8079 045210 122716 000011
8080 045214 001430
8081 045216 122716 000200
8082 045222 001006
8083 045224 005726
8084 045226 104401
8085 045230 001205
8086 045232 105037 045366
8087 045236 000755
8088 045240 004737 045322
8089 045244 123726 001156
8090 045250 001350

```

```

55:   CMP      #SENDAC,2#42      ;;ACT-11 AUTO-ACCEPT?
      BNE     65                ;;BRANCH IF NO
      HALT                    ;;YES
65:   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      $TPFLG          ;; IS THERE A TERMINAL?
      BPL      1$              ;; BR IF YES
      HALT     HERE IF NO TERMINAL
      BR      3$              ;; LEAVE
1$:   MOV      RO, -(SP)        ;; SAVE RO
      MOV      22(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
      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
      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$              ;;
      TST     (SP)+           ;; POP <CR><LF> EQUIV
      TYPE    A CR AND LF     ;; TYPE A CR AND LF
8$:   CLR     $CHARCNT         ;; CLEAR CHARACTER COUNT
      BR      2$              ;; GET NEXT CHARACTER
55:   JSR     PC, $TYPEC        ;; GO TYPE THIS CHARACTER
65:   CMPB    $FILLC, (SP)+   ;; IS IT TIME FOR FILLER CHARS.?
      BNE     2$              ;; IF NO GO GET NEXT CHAR.

```

```

8091 045252 013746 001154          MOV      $NULL,-(SP)      ;;GET # OF FILLER CHARS. NEEDED
8092                                     ;;AND THE NULL CHAR.
8093 045256 105366 000001      7S:    DECB      1(SP)      ;;DOES A NULL NEED TO BE TYPED?
8094 045262 002770                BLT      65              ;;BR IF NO--GO POP THE NULL OFF OF STACK
8095 045264 004737 045322          JSR      PC,$TYPEC      ;;GO TYPE A NULL
8096 045270 105337 045366          DECB      $CHARCNT      ;;DO NOT COUNT AS A COUNT
8097 045274 000770                BR       7S              ;;LOOP
8098
8099                                     ;HORIZONTAL TAB PROCESSOR
8100
8101 045276 112716 000040      8S:    MOV      #' (SP)      ;;REPLACE TAB WITH SPACE
8102 045302 004737 045322      9S:    JSR      PC,$TYPEC      ;;TYPE A SPACE
8103 045306 132737 000007 045366  BITB     #',$CHARCNT      ;;BRANCH IF NOT AT
8104 045314 001372                BNE     9S              ;;TAB STOP
8105 045316 005726                TST     (SP)+           ;;POP SPACE OFF STACK
8106 045320 000724                BR      2S              ;;GET NEXT CHARACTER
8107 045322 105777 133622      $TYPEC: *STB     2$TPS      ;;WAIT UNTIL PRINTER IS READY
8108 045326 100375                BPL     $TYPEC
8109 045330 116677 000002 133614  MOV     2(SP),2$TPB      ;;LOAD CHAR TO BE TYPED INTO DATA REG.
8110 045336 122766 000015 000002  CMPB    #CR,2(SP)        ;;IS CHARACTER A CARRIAGE RETURN?
8111 045344 001003                BNE     1S              ;;BRANCH IF NO
8112 045346 105037 045366          CLRB    $CHARCNT        ;;YES--CLEAR CHARACTER COUNT
8113 045352 000406                BR      $TYPEX          ;;EXIT
8114 045354 122766 000012 000002  1S:    CMPB    #LF,2(SP)      ;;IS CHARACTER A LINE FEED?
8115 045362 001402                BEQ     $TYPEX          ;;BRANCH IF YES
8116 045364 105227                INCB    (PC)+           ;;COUNT THE CHARACTER
8117 045366 000000          $CHARCNT: .WORD 0      ;;CHARACTER COUNT STORAGE
8118 045370 000207          $TYPEX: RTS      PC
8119
8120                                     .SBTTL CONVERT BINARY TO DECIMAL AND TYPE ROUTINE
8121
8122                                     ;*****
8123                                     ;*THIS ROUTINE IS USED TO CHANGE A 16-BIT BINARY NUMBER TO A 5-DIGIT
8124                                     ;*SIGNED DECIMAL (ASCII) NUMBER AND TYPE IT. DEPENDING ON WHETHER THE
8125                                     ;*NUMBER IS POSITIVE OR NEGATIVE A SPACE OR A MINUS SIGN WILL BE TYPED
8126                                     ;*BEFORE THE FIRST DIGIT OF THE NUMBER. LEADING ZEROS WILL ALWAYS BE
8127                                     ;*REPLACED WITH SPACES.
8128                                     ;*CALL:
8129                                     ;*      MOV      NUM,-(SP)      ;;PUT THE BINARY NUMBER ON THE STACK
8130                                     ;*      TYPDS      ;;GO TO THE ROUTINE
8131
8132      045372          $TYPDS:  MOV     R0,-(SP)      ;;PUSH R0 ON STACK
8133      045372 010046          MOV     R1,-(SP)      ;;PUSH R1 ON STACK
8134      045374 010146          MOV     R2,-(SP)      ;;PUSH R2 ON STACK
8135      045376 010246          MOV     R3,-(SP)      ;;PUSH R3 ON STACK
8136      045400 010346          MOV     R5,-(SP)      ;;PUSH R5 ON STACK
8137      045402 010546          MOV     #20200,-(SP)   ;;SET BLANK SWITCH AND SIGN
8138      045404 012746 020200  MOV     20(SP),R5      ;;GET THE INPUT NUMBER
8139      045410 016605 000020  BPL     1S              ;;BR IF INPUT IS POS.
8140      045414 100004          NEG     R5              ;;MAKE THE BINARY NUMBER POS.
8141      045416 005405          MOV     #'-,1(SP)      ;;MAKE THE ASCII NUMBER NEG.
8142      045420 112766 000055 000001  1S:    CLR     R0              ;;ZERO THE CONSTANTS INDEX
8143      045426 005000          MOV     #0BLK,R3      ;;SETUP THE OUTPUT POINTER
8144      045430 012703 045606          MOV     #' ,(R3)+      ;;SET THE FIRST CHARACTER TO A BLANK
8145      045434 112723 000040          MOV     R2              ;;CLEAR THE BCD NUMBER
8146      045440 005002          CLR     R2

```

E13

UNIBUS RK06 DRIVE DIAGNOSTIC PART 2
 DZRBIC.P11 07-OCT-76 13:50

MACY11 27(1006) 07-OCT-76 14:14 PAGE 160
 CONVERT BINARY TO DECIMAL AND TYPE ROUTINE

SEG 0160

0147	045442	016001	045576		MOV	\$DTBL(R0),R1	::GET THE CONSTANT
0148	045446	160105		3\$:	SUB	R1,R5	::FORM THIS BCD DIGIT
0149	045450	002402			BLT	4\$::BR IF DONE
0150	045452	005202			INC	R2	::INCREASE THE BCD DIGIT BY 1
0151	045454	000774			BR	3\$	
0152	045456	060105		4\$:	ADC	R1,R5	::ADD BACK THE CONSTANT
0153	045460	005702			TST	R2	::CHECK IF BCD DIGIT=0
0154	045462	001002			BNE	5\$::FALL THROUGH IF 0
0155	045464	105716			TSTB	(SP)	::STILL DOING LEADING 0'S?
0156	045466	100407			BMI	7\$::BR IF YES
0157	045470	106316		5\$:	ASLB	(SP)	::MSD?
0158	045472	103003			BCC	6\$::BR IF NO
0159	045474	116663	000001 177777		MOVB	1(SP),-1(R3)	::YES--SET THE SIGN
0160	045502	052702	000360	6\$:	BIS	#0,R2	::MAKE THE BCD DIGIT ASCII
0161	045506	052702	000040	7\$:	BIS	#7,R2	::MAKE IT A SPACE IF NOT ALREADY A DIGIT
0162	045512	110223			MOVB	R2,(R3)+	::PUT THIS CHARACTER IN THE OUTPUT BUFFER
0163	045514	005720			TST	(R0)+	::JUST INCREMENTING
0164	045516	020027	000010		CMP	R0,#10	::CHECK THE TABLE INDEX
0165	045522	002746			BLT	2\$::GO DO THE NEXT DIGIT
0166	045524	003002			BGT	8\$::GO TO EXIT
0167	045526	010502			MOV	R5,R2	::GET THE LSD
0168	045530	000764			BR	6\$::GO CHANGE TO ASCII
0169	045532	105726		8\$:	TSTB	(SP)+	::WAS THE LSD THE FIRST NON-ZERO?
0170	045534	100003			BPL	9\$::BR IF NO
0171	045536	116663	177777 177776		MOVB	-1(SP),-2(R3)	::YES--SET THE SIGN FOR TYPING
0172	045544	105013		9\$:	CLRP	(R3)	::SET THE TERMINATOR
0173	045546	012605			MOV	(SP)+,R5	::POP STACK INTO R5
0174	045550	012603			MOV	(SP)+,R3	::POP STACK INTO R3
0175	045552	012602			MOV	(SP)+,R2	::POP STACK INTO R2
0176	045554	012601			MOV	(SP)+,R1	::POP STACK INTO R1
0177	045556	012600			MOV	(SP)+,R0	::POP STACK INTO R0
0178	045560	104401	045606		TYPE	\$DBLK	::NOW TYPE THE NUMBER
0179	045564	016666	000002 000004		MOV	2(SP),4(SP)	::ADJUST THE STACK
0180	045572	012316			MOV	(SP)+,(SP)	
0181	045574	000002			RTI		::RETURN TO USER
0182	045576	023420		\$DTBL:	10000.		
0183	045600	001750			1000.		
0184	045602	000144			100.		
0185	045604	000012			10.		
0186	045606	000004		\$DBLK:	.BLKW 4		
0187				.SBTTL	APT COMMUNICATIONS ROUTINE		
0188							
0189							
0190	045616	112737	000001 046062		\$ATY1:	MOVB #1,\$FFLG	::TO REPORT FATAL ERROR
0191	045624	112737	000001 046060		\$ATY3:	MOVB #1,\$MFLG	::TO TYPE A MESSAGE
0192	045632	000403			BR	\$ATYC	
0193	045634	112737	000001 046062		\$ATY4:	MOVB #1,\$FFLG	::TO ONLY REPORT FATAL ERROR
0194	045642				\$ATYC:		
0195	045642	010046			MOV	R0,-(SP)	::PUSH R0 ON STACK
0196	045644	010146			MOV	R1,-(SP)	::PUSH R1 ON STACK
0197	045646	105737	046060		TSTB	\$MFLG	::SHOULD TYPE A MESSAGE?
0198	045652	001450			BEQ	5\$::IF NOT: BR
0199	045654	122737	000001 001230		CMPB	#APTENV,\$ENV	::OPERATING UNDER APT?
0200	045662	001031			BNE	3\$::IF NOT: BR
0201	045664	132737	000100 001231		BITB	#APTPOOL,\$ENVM	::SHOULD SPOOL MESSAGES?
0202	045672	001425			BEQ	3\$::IF NOT: BR

```

8203 045674 017600 000004      MOV      24(SP),R0      ;;GET MESSAGE ADDR.
8204 045700 062766 000002 000004  ADD      2,4(SP)      ;;BUMP RETURN ADDR.
8205 045706 005737 001210      TST      $MSGTYPE     ;;SEE IF DONE W/ LAST XMISSION?
8206 045712 001375      BNE      1$           ;;IF NOT: WAIT
8207 045714 010037 001224      MOV      R0,$MSGAD     ;;PUT ADDR IN MAILBOX
8208 045720 105720 25:      TSTB    (R0)+         ;;FIND END OF MESSAGE
8209 045722 001376      BNE      2$           ;;
8210 045724 163700 001224      SUB      $MSGAD,R0     ;;SUB START OF MESSAGE
8211 045730 006200      ASR      R0           ;;GET MESSAGE LNTH IN WORDS
8212 045732 010037 001226      MOV      R0,$MSGGLT   ;;PUT LENGTH IN MAILBOX
8213 045736 012737 000004 001210  MOV      4,$MSGTYPE   ;;TELL APT TO TAKE MSG.
8214 045744 000413      BR       5$           ;;
8215 045746 017637 000004 045772 3$:      MOV      24(SP),4$    ;;PUT MSG ADDR IN JSR LINKAGE
8216 045754 062766 000002 000004  ADD      2,4(SP)      ;;BUMP RETURN ADDRESS
8217 045762 013746 177776      MOV      177776,-(SP) ;;PUSH 177776 ON STACK
8218 045766 004737 045110      JSR     PC,$TYPE     ;;CALL TYPE MACRO
8219 045772 000000      4$:      .WORD   0
8220 045774      5$:
8221 045774 105737 046062 10$:      TSTB    $FFLG        ;;SHOULD REPORT FATAL ERROR?
8222 046000 001416      BEQ     12$          ;;IF NOT: BR
8223 046002 005737 001230      TST     $ENV         ;;RUNNING UNDER APT?
8224 046006 001413      BEQ     12$          ;;IF NOT: BR
8225 046010 005737 001210 11$:      TST     $MSGTYPE     ;;FINISHED LAST MESSAGE?
8226 046014 001375      BNE     11$         ;;IF NOT: WAIT
8227 046016 017637 000004 001212  MOV      24(SP),$FATAL ;;GET ERROR #
8228 046024 062766 000002 000004  ADD      2,4(SP)      ;;BUMP RETURN ADDR.
8229 046032 005237 001210      INC     $MSGTYPE     ;;TELL APT TO TAKE ERROR
8230 046036 105037 046062 12$:      CLRB   $FFLG        ;;CLEAR FATAL FLAG
8231 046042 105037 046061      CLRB   $LFLG        ;;CLEAR LOG FLAG
8232 046046 105037 046060      CLRB   $MFLG        ;;CLEAR MESSAGE FLAG
8233 046052 012601      MOV     (SP)+,R1     ;;POP STACK INTO R1
8234 046054 012600      MOV     (SP)+,R0     ;;POP STACK INTO R0
8235 046056 000207      RTS    PC           ;;RETURN
8236 046060      $MFLG: .BYTE 0      ;;MESSG. FLAG
8237 046061      $LFLG: .BYTE 0      ;;LOG FLAG
8238 046062      $FFLG: .BYTE 0      ;;FATAL FLAG

```

```

8239      046064      .EVEN
8240      000200  APTSIZE=200
8241      000001  APTENV=001
8242      000100  APTSPool=100
8243      000040  APTCSUP=040
8244      .SBTTL  BINARY TO OCTAL (ASCII) AND TYPE

```

```

8245
8246 *****
8247 *THIS ROUTINE IS USED TO CHANGE A 16-BIT BINARY NUMBER TO A 6-DIGIT
8248 *OCTAL (ASCII) NUMBER AND TYPE IT.
8249 *$TYPOS---ENTER HERE TO SETUP SUPPRESS ZEROS AND NUMBER OF DIGITS TO TYPE
8250 *CALL:
8251 *      MOV      NUM,-(SP)      ;;NUMBER TO BE TYPED
8252 *      TYPOS    ;;CALL FOR TYPEOUT
8253 *      .BYTE   N              ;;N=1 TO 6 FOR NUMBER OF DIGITS TO TYPE
8254 *      .BYTE   M              ;;M=1 OR 0
8255 *
8256 *      ;;1=TYPE LEADING ZEROS
8257 *      ;;0=SUPPRESS LEADING ZEROS
8258 *
8259 *$STYPON----ENTER HERE TO TYPE OUT WITH THE SAME PARAMETERS AS THE LAST

```

```

8259 ;*STYPOS OR STYPOC
8260 ;*CALL:
8261 ;*   MOV   NUM,-(SP)   ;;NUMBER TO BE TYPED
8262 ;*   TYPON                ;;CALL FOR TYPEOUT
8263 ;*
8264 ;*STYPOC---ENTER HERE FOR TYPEOUT OF A 16 BIT NUMBER
8265 ;*CALL:
8266 ;*   MOV   NUM,-(SP)   ;;NUMBER TO BE TYPED
8267 ;*   TYPOC                ;;CALL FOR TYPEOUT
8268 ;*
8269 046064 017646 000000 046307 STYPOS: MOV   2(SP),-(SP)   ;;PICKUP THE MODE
8270 046070 116637 000001 MOVVB 1(SP),%SOFILL ;;LOAD ZERO FILL SWITCH
8271 046076 112637 046311 MOVVB (SP)+,%SOMODE+1 ;;NUMBER OF DIGITS TO TYPE
8272 046102 062716 000002 ADD   #2,(SP)   ;;ADJUST RETURN ADDRESS
8273 046106 000406 BR    STYPON
8274 046110 112737 000001 046307 STYPOC: MOVVB #1,%SOFILL   ;;SET THE ZERO FILL SWITCH
8275 046116 112737 000006 046311 MOVVB #6,%SOMODE+1 ;;SET FOR SIX(6) DIGITS
8276 046124 112737 000005 046306 STYPON: MOVVB #5,%SOCNT ;;SET THE ITERATION COUNT
8277 046132 010346 MOV   R3,-(SP)   ;;SAVE R3
8278 046134 010446 MOV   R4,-(SP)   ;;SAVE R4
8279 046136 010546 MOV   R5,-(SP)   ;;SAVE R5
8280 046140 113704 046311 MOVVB %SOMODE+1,R4 ;;GET THE NUMBER OF DIGITS TO TYPE
8281 046144 005404 NEG   R4
8282 046146 062704 000006 ADD   #6,R4   ;;SUBTRACT IT FOR MAX. ALLOWED
8283 046152 110437 046310 MOVVB R4,%SOMODE ;;SAVE IT FOR USE
8284 046156 113704 046307 MOVVB %SOFILL,R4 ;;GET THE ZERO FILL SWITCH
8285 046162 016605 000012 MOV   12(SP),R5 ;;PICKUP THE INPUT NUMBER
8286 046166 005003 CLR   R3   ;;CLEAR THE OUTPUT WORD
8287 046170 006105 1$: ROL  R5   ;;ROTATE MSB INTO "C"
8288 046172 000404 BR    3$   ;;GO DO MSB
8289 046174 006105 2$: ROL  R5   ;;FORM THIS DIGIT
8290 046176 006105 ROL  R5
8291 046200 006105 ROL  R5
8292 046202 010503 MOV   R5,R3
8293 046204 006103 3$: ROL  R3   ;;GET LSB OF THIS DIGIT
8294 046206 105337 046310 DECB  %SOMODE ;;TYPE THIS DIGIT?
8295 046212 100016 BPL  7$   ;;BR IF NO
8296 046214 042703 177770 BIC  #177770,R3 ;;GET RID OF JUNK
8297 046220 001002 BNE  4$   ;;TEST FOR 0
8298 046222 005704 TST  R4   ;;SUPPRESS THIS 0?
8299 046224 001403 BEQ  5$   ;;BR IF YES
8300 046226 005204 4$: INC  R4   ;;DON'T SUPPRESS ANYMORE 0'S
8301 046230 052703 000060 BIS  #'0,R3 ;;MAKE THIS DIGIT ASCII
8302 046234 052703 000040 5$: BIS  #' ,R3 ;;MAKE ASCII IF NOT ALREADY
8303 046240 110337 046304 MOVVB R3,%S   ;;SAVE FOR TYPING
8304 046244 104401 046304 TYPE  #8$   ;;GO TYPE THIS DIGIT
8305 046250 105337 046306 7$: DECB  %SOCNT ;;COUNT BY 1
8306 046254 003347 BGT  2$   ;;BR IF MORE TO DO
8307 046256 002402 BLT  6$   ;;BR IF DONE
8308 046260 005204 INC  R4   ;;INSURE LAST DIGIT ISN'T A BLANK
8309 046262 000744 BR    2$   ;;GO DO THE LAST DIGIT
8310 046264 012605 6$: MOV  (SP)+,R5 ;;RESTORE R5
8311 046266 012604 MOV  (SP)+,R4 ;;RESTORE R4
8312 046270 012603 MOV  (SP)+,R3 ;;RESTORE R3
8313 046272 016666 000002 000004 MOV  2(SP),4(SP) ;;SET THE STACK FOR RETURNING
8314 046300 012616 MOV  (SP)+,(SP)
  
```

```

8315 046302 000002
8316 046304 000
8317 046305 000
8318 046306 000
8319 046307 000
8320 046310 000000
8321
8322
8323
8324
8325 046312 000000
8326 046314 000000
8327 046316 000000
8328 046320 000001
8329 046321
8330 046322
8331
8332
8333
8334
8335
8336
8337
8338
8339
8340 046322 005037 046312
8341 046326 012737 046320 046314
8342 046334 013737 046314 046316
8343 046342 012737 046372 000060
8344 046350 012737 000200 000062
8345 046356 005777 132564
8346 046362 012777 000100 132554
8347 046370 000207
8348
8349
8350
8351
8352
8353
8354
8355
8356 046372 117746 132550
8357 046376 042716 177600
8358 046402 021627 000003
8359 046406 001007
8360 046410 104401 047520
8361 046414 004737 046322
8362 046420 005726
8363 046422 000137 043460
8364 046426 021627 000007
8365 046432 001004
8366 046434 022737 000176 001140
8367 046442 001500
8368
8369 046444
8370 046444 022737 000001 046312

```

```

RTI ;; RETURN
8$: .BYTE 0 ;; STORAGE FOR ASCII DIGIT
      .BYTE 00 ;; TERMINATOR FOR TYPE ROUTINE
$OCNT: .BYTE 00 ;; OCTAL DIGIT COUNTER
$OFILL: .BYTE 00 ;; ZERO FILL SWITCH
$OMODE: .WORD 00 ;; 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
$TKQSRV: .BLKB 1 ;; TTY KEYBOARD QUEUE
$TKQEND=.
.EVEN

; *TK INITIALIZE ROUTINE
; *THIS ROUTINE WILL INITIALIZE THE TTY KEYBOARD INPUT QUEUE
; *SETUP THE INTERRUPT VECTOR AND TURN ON THE KEYBOARD INTERRUPT
; *CALL:
; * JSR PC,$TKINT
; * RETURN
$TKINT: CLR $TKCNT ;; CLEAR COUNT OF ITEMS IN QUEUE
        MOV $TKQSRV,$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: MOV $TKB,-(SP) ;; PICKUP THE CHARACTER
        BIC #↑C177,(SP) ;; STRIP THE JUNK
        CMP (SP),#3 ;; IS IT A CONTROL C?
        BNE IS ;; 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
IS: 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?

```

8371	046452	001004			BNE	3\$::BRANCH IF NO
8372	046454	104401	001200		TYPE	, \$BELL	::RING THE TTY BELL
8373	046460	005726			TST	(SP)+	::CLEAN CHARACTER OFF OF STACK
8374	046462	000451			BR	5\$::EXIT
8375	046464	021627	000023	3\$:	CMP	(SP), #23	::IS IT A CONTROL-S?
8376	046470	001021			BNE	32\$::BRANCH IF NO
8377	046472	005077	132446		CLR	\$STKS	::DISABLE TTY KEYBOARD INTERRUPTS
8378	046476	005726			TST	(SP)+	::CLEAN CHAR OFF STACK
8379	046500	105777	132440	31\$:	TSTB	\$STKS	::WAIT FOR A CHAR
8380	046504	100375			BPL	31\$::LOOP UNTIL ITS THERE
8381	046506	117746	132434		MOVB	\$STKB, -(SP)	::GET THE CHARACTER
8382	046512	042716	177600		BIC	#1C177, (SP)	::MAKE IT 7-BIT ASCII
8383	046516	022627	000021		CMP	(SP)+, #21	::IS IT A CONTROL-Q?
8384	046522	001366			BNE	31\$::BRANCH IF NO
8385	046524	012777	000100	132412	MOV	#100, \$STKS	::REENABLE TTY KEYBOARD INTERRUPTS
8386	046532	000002			RTI		::RETURN
8387	046534	005237	046312	32\$:	INC	\$STKCNT	::COUNT THIS CHARACTER
8388	046540	021627	000140		CMP	(SP), #140	::IS IT UPPER CASE?
8389	046544	002405			BLT	4\$::BRANCH IF YES
8390	046546	021627	000175		CMP	(SP), #175	::IS IT A SPECIAL CHAR?
8391	046552	003002			BGT	4\$::BRANCH IF YES
8392	046554	042716	000040		BIC	#40, (SP)	::MAKE IT UPPER CASE
8393	046560	112677	177530	4\$:	MOVB	(SP)+, \$STKQIN	::AND PUT IT IN QUEUE
8394	046564	005237	046314		INC	\$STKQIN	::UPDATE THE POINTER
8395	046570	023727	046314	046321	CMP	\$STKQIN, #STKQEND	::GO OFF THE END?
8396	046576	001003			BNE	5\$::BRANCH IF NO
8397	046600	012737	046320	046314	MOV	#STKQSR, \$STKQIN	::RESET THE POINTER
8398	046606	000002		5\$:	RTI		::RETURN

8399 *****

8400 ::*SOFTWARE SWITCH REGISTER CHANGE ROUTINE.
8401 ::*ROUTINE IS ENTERED FROM THE TRAP HANDLER, AND WILL
8402 ::*SERVICE THE TEST FOR CHANGE IN SOFTWARE SWITCH REGISTER TRAP
8403 ::*CALL WHEN OPERATING IN TTY INTERRUPT MODE.

8404					\$CKSWR:	CMP	#SWREG, SWR	::IS THE SOFT-SWR SELECTED
8405	046610	022737	000176	001140		BNE	15\$::EXIT IF NOT
8406	046616	001124			TSTB	\$STKS	::IS A CHAR WAITING?	
8407	046620	105777	132320		BPL	15\$::IF NOT, EXIT	
8408	046624	100121			MOVB	\$STKB, -(SP)	::YES	
8409	046626	117746	132314		BIC	#1C177, (SP)	::MAKE IT 7-BIT ASCII	
8410	046632	042716	177600		CMP	(SP), #7	::IS IT A CONTROL-G?	
8411	046636	021627	000007		BNE	25	::IF NOT, PUT IT IN THE TTY QUEUE	
8412	046642	001300					::AND EXIT	

8413 *****
8414 ::*CONTROL IS PASSED TO THIS POINT FROM EITHER THE TTY INTERRUPT SERVICE
8415 ::*ROUTINE OR FROM THE SOFTWARE SWITCH REGISTER TRAP CALL, AS A RESULT OF A
8416 ::*CONTROL-G BEING TYPED, AND THE SOFTWARE SWITCH REGISTER BEING SELECTED.

8417					6\$:	CMPB	\$AUTOB, #1	::ARE WE RUNNING IN AUTO-MODE?
8418						BEQ	25	::BRANCH IF YES
8419	046644	123727	001134	000001		TST	(SP)+	::CLEAR CONTROL-G OFF STACK
8420	046652	001674			JSR	PC, \$TKINT	::FLUSH THE TTY INPUT QUEUE	
8421	046654	005726			CLR	\$STKS	::DISABLE TTY KEYBOARD INTERRUPTS	
8422	046656	004737	046322		MOV	#1, \$INTAG	::SET INTERRUPT MODE INDICATOR	
8423	046662	005077	132256					
8424	046666	112737	000001	001135				
8425								
8426	046674	104401	047532		TYPE	, \$CNTLG	::ECHO THE CONTROL-G (1G)	

8427	046700	104401	047537		SGTSWR: TYPE	\$MSWR	:: TYPE CURRENT CONTENTS
8428	046704	013746	000176		MOV	\$WREG, -(SP)	:: SAVE SWREG FOR TYPEOUT
8429	046710	104402			TYPOC		:: GO TYPE--OCTAL ASCII(ALL DIGITS.
8430	046712	104401	047550		TYPE	, \$MNEW	:: PROMPT FOR NEW SWR
8431	046716	005046		19\$:	CLR	-(SP)	:: CLEAR COUNTER
8432	046720	005046			CLR	-(SP)	:: THE NEW SWR
8433	046722	105777	132216	7\$:	TSTB	\$STKS	:: CHAR THERE?
8434	046726	100375			BPL	7\$:: IF NOT TRY AGAIN
8435							
8436	046730	117746	132212		MOVB	\$STKB, -(SP)	:: PICK UP CHAR
8437	046734	042716	177600		BIC	#1C177, (SP)	:: MAKE IT 7-BIT ASCII
8438							
8439	046740	021627	000003		CMP	(SP), #3	:: IS IT A CONTROL-C?
8440	046744	001015			BNE	9\$:: BRANCH IF NOT
8441	046746	104401	047520		TYPE	\$CNTLC	:: YES, ECHO CONTROL-C (↑C)
8442	046752	062706	000006		ADD	#6, SP	:: CLEAN UP STACK
8443	046756	123727	001135	000001	CMPB	\$INTAG, #1	:: REENABLE TTY KEYBOARD INTERRUPTS?
8444	046764	001003			BNE	8\$:: BRANCH IF NO
8445	046766	012777	000100	132150	MOV	#100, \$STKS	:: ALLOW TTY KEYBOARD INTERRUPTS
8446	046774	000137	043460	8\$:	JMP	STOP	:: CONTROL-C RESTART
8447							
8448							
8449	047000	021627	000025	9\$:	CMP	(SP), #25	:: IS IT A CONTROL-U?
8450	047004	001005			BNE	10\$:: BRANCH IF NOT
8451	047006	104401	047525		TYPE	\$CNTLU	:: YES, ECHO CONTROL-U (↑U)
8452	047012	062706	000006	20\$:	ADD	#6, SP	:: IGNORE PREVIOUS INPUT
8453	047016	000737			BR	19\$:: LET'S TRY IT AGAIN
8454							
8455							
8456	047020	021627	000015	10\$:	CMP	(SP), #15	:: IS IT A <CR>?
8457	047024	001022			BNE	16\$:: BRANCH IF NO
8458	047026	005766	000004		TST	4(SP)	:: YES, IS IT THE FIRST CHAR?
8459	047032	001403			BEQ	11\$:: BRANCH IF YES
8460	047034	016677	000002	132076	MOV	2(SP), \$SWR	:: SAVE NEW SWR
8461	047042	062706	000006	11\$:	ADD	#6, SP	:: CLEAN UP STACK
8462	047046	104401	001205	14\$:	TYPE	\$CRLF	:: ECHO <CR> AND <LF>
8463	047052	123727	001135	000001	CMPB	\$INTAG, #1	:: RE-ENABLE TTY KBD INTERRUPTS?
8464	047060	001003			BNE	15\$:: BRANCH IF NOT
8465	047062	012777	000100	132054	MOV	#100, \$STKS	:: RE-ENABLE TTY KBD INTERRUPTS
8466	047070	000002		15\$:	RTI		:: RETURN
8467	047072	004737	045322	16\$:	JSR	PC, \$TYPEC	:: ECHO CHAR
8468	047076	021627	000060		CMP	(SP), #60	:: CHAR < 0?
8469	047102	002420			BLT	18\$:: BRANCH IF YES
8470	047104	021627	000067		CMP	(SP), #67	:: CHAR > 7?
8471	047110	003015			BGT	18\$:: BRANCH IF YES
8472	047112	042726	000060		BIC	#60, (SP)+	:: STRIP-OFF ASCII
8473	047116	005766	000002		TST	2(SP)	:: IS THIS THE FIRST CHAR
8474	047122	001403			BEQ	17\$:: BRANCH IF YES
8475	047124	006316			ASL	(SP)	:: NO, SHIFT PRESENT
8476	047126	006316			ASL	(SP)	:: CHAR OVER TO MAKE
8477	047130	006316			ASL	(SP)	:: ROOM FOR NEW ONE.
8478	047132	005266	000002	17\$:	INC	2(SP)	:: KEEP COUNT OF CHAR
8479	047136	056616	177776		BIS	-2(SP), (SP)	:: SET IN NEW CHAR
8480	047142	000667			BR	7\$:: GET THE NEXT ONE
8481	047144	104401	001204	18\$:	TYPE	\$QUES	:: TYPE ?<CR><LF>
8482	047150	000720			BR	20\$:: SIMULATE CONTROL-U

```

0483 .DSABL LSB
0484
0485
0486
0487
0488
0489
0490
0491
0492
0493
0494 047152 011646 000004 000002 $RDCHR: MOV (SP), -(SP) ;; PUSH DOWN THE PC AND
0495 047154 016666 000004 000002 MOV 4(SP), 2(SP) ;; THE PS
0496 047162 005066 000004 CLR 4(SP) ;; GET READY FOR A CHARACTER
0497 047166 005046 CLR -(SP) ;; PUT NEW PS ON STACK
0498 047170 012746 047176 MOV #64$, -(SP) ;; PUT NEW PC ON STACK
0499 047174 000002 RTI ;; POP NEW PC AND PS
0500 047176 64$:
0501 047176 005737 046312 1$: TST $STKCNT ;; WAIT ON A CHARACTER
0502 047202 001775 BEQ 1$
0503 047204 005337 046312 DEC $STKCNT ;; DECREMENT THE COUNTER
0504 047210 117766 177102 000004 MOV $STKQOUT, 4(SP) ;; GET ONE CHARACTER
0505 047216 005237 046316 INC $STKQOUT ;; UPDATE THE POINTER
0506 047222 023727 046316 046321 CMP $STKQOUT, # $STKQEND ;; DID IT GO OFF OF THE END?
0507 047230 001003 BNE 2$ ;; BRANCH IF NO
0508 047232 012737 046320 046316 MOV # $STKQSR, $STKQOUT ;; RESET THE POINTER
0509 047240 000002 2$: RTI ;; RETURN
0510
0511
0512
0513
0514
0515
0516
0517 047242 010346 $RDLIN: MOV R3, -(SP) ;; SAVE R3
0518 047244 005046 CLR -(SP) ;; CLEAR THE RUBOUT KEY
0519 047246 012703 047476 1$: MOV # $TTYIN, R3 ;; GET ADDRESS
0520 047252 022703 047520 2$: CMP # $TTYIN+22, R3 ;; BUFFER FULL?
0521 047256 101456 BLOS 4$ ;; BR IF YES
0522 047260 104410 RDCHR ;; GO READ ONE CHARACTER FROM THE TTY
0523 047262 112613 MOV (SP)+, (R3) ;; GET CHARACTER
0524 047264 122713 000177 10$: CMPB #177, (R3) ;; IS IT A RUBOUT
0525 047270 001022 BNE 5$ ;; BR IF NO
0526 047272 005716 TST (SP) ;; IS THIS THE FIRST RUBOUT?
0527 047274 001007 BNE 6$ ;; BR IF NO
0528 047276 112737 000134 047474 MOV #'\, 9$ ;; TYPE A BACK SLASH
0529 047304 104401 047474 TYPE 9$
0530 047310 012716 177777 MOV # -1, (SP) ;; SET THE RUBOUT KEY
0531 047314 005303 6$: DEC R3 ;; BACKUP BY ONE
0532 047316 020327 047476 CMP R3, # $TTYIN ;; STACK EMPTY?
0533 047322 103434 BLO 4$ ;; BR IF YES
0534 047324 111337 047474 MOV (R3), 9$ ;; SETUP TO TYPEOUT THE DELETED CHAR.
0535 047330 104401 047474 TYPE 9$ ;; GO TYPE
0536 047334 000746 BR 2$ ;; GO READ ANOTHER CHAR.
0537 047336 005716 5$: TST (SP) ;; RUBOUT KEY SET?
0538 047340 001406 BEQ 7$ ;; BR IF NO

```

```

8539 047342 112737 000134 047474      MOVB    #' \, 9$      ;; TYPE A BACK SLASH
8540 047350 104401 047474      TYPE    9$
8541 047354 005016                CLR     (SP)          ;; CLEAR THE RUBOUT KEY
8542 047356 122713 000025      7$:    CMPB    #25, (R3) ;; IS CHARACTER A CTRL U?
8543 047362 001003                BNE     8$           ;; BR IF NO
8544 047364 104401 047525      TYPE    $CNTLU       ;; TYPE A CONTROL "U"
8545 047370 000726                BR      1$           ;; GO START OVER
8546 047372 122713 000022      8$:    CMPB    #22, (R3) ;; IS CHARACTER A "↑R"?
8547 047376 001011                BNE     3$           ;; BRANCH IF NO
8548 047400 105013                CLRB   (R3)          ;; CLEAR THE CHARACTER
8549 047402 104401 001205      TYPE    $CRLF        ;; TYPE A "CR" & "LF"
8550 047406 104401 047476      TYPE    $TTYIN       ;; TYPE THE INPUT STRING
8551 047412 000717                BR      2$           ;; GO PICKUP ANOTHER CHACTER
8552 047414 104401 001204      4$:    TYPE    $QUES   ;; TYPE A '?'
8553 047420 000712                BR      1$           ;; CLEAR THE BUFFER AND LOOP
8554 047422 111337 047474      3$:    MOVB    (R3), 9$  ;; ECHO THE CHARACTER
8555 047426 104401 047474      TYPE    9$
8556 047432 122723 000015      CMPB    #15, (R3)+   ;; CHECK FOR RETURN
8557 047436 001305                BNE     2$           ;; LOOP IF NOT RETURN
8558 047440 105063 177777      CLRB   -1(R3)        ;; CLEAR RETURN (THE 15)
8559 047444 104401 001206      TYPE    $LF          ;; TYPE A LINE FEED
8560 047450 005726                TST    (SP)+         ;; CLEAN RUBOUT KEY FROM THE STACK
8561 047452 012603                MOV    (SP)+, R3     ;; RESTORE R3
8562 047454 011646                MOV    (SP), -(SP)  ;; ADJUST THE STACK AND PUT ADDRESS OF THE
8563 047456 016666 000004 000002      MOV    4(SP), 2(SP) ;; FIRST ASCII CHARACTER ON IT
8564 047464 012766 047476 000004      MOV    #TTYIN, 4(SP)
8565 047472 000002                RTI
8566 047474      000      9$:    .BYTE   0          ;; STORAGE FOR ASCII CHAR. TO TYPE
8567 047475      000                .BYTE   0          ;; TERMINATOR
8568 047476 000022                $TTYIN: .BLKB  22    ;; RESERVE 22 BYTES FOR TTY INPUT
8569 047520 041536 005015      000      $CNTLC: .ASCIZ  /↑C/<15><12> ;; CONTROL "C"
8570 047525      136 006525 000012      $CNTLU: .ASCIZ  /↑U/<15><12> ;; CONTROL "U"
8571 047532 043536 005015      000      $CNTLG: .ASCIZ  /↑G/<15><12> ;; CONTROL "G"
8572 047537      015 051412 051127      $MSWR:  .ASCIZ  <15><12>/SWR = /
8573 047544 036440 000040                $MNEW:  .ASCIZ  / NEW = /
8574 047550 020040 042516 020127
8575 047556 020075      000
8576      047562
8577
8578
8579
8580
8581
8582
8583
8584
8585
8586
8587
8588
8589
8590
8591 047562 011646      000004 000002      $RDOCT: MOV    (SP), -(SP) ;; PROVIDE SPACE FOR THE
8592 047564 016666                MOV    4(SP), 2(SP) ;; INPUT NUMBER
8593 047572 010046                MOV    R0, -(SP)    ;; PUSH R0 ON STACK
8594 047574 010146                MOV    R1, -(SP)    ;; PUSH R1 ON STACK

```

.EVEN
 .SBTTL READ AN OCTAL NUMBER FROM THE TTY

 *THIS ROUTINE WILL READ AN OCTAL (ASCII) NUMBER FROM THE TTY AND
 *CHANGE IT TO BINARY.
 *THE INPUT CHARACTERS WILL BE CHECKED TO INSURED THEY ARE LEGAL
 *OCTAL DIGITS. IF AN ILLEGAL CHARACTER IS READ A "?" WILL BE TYPED
 *FOLLOWED BY A CARRIAGE RETURN-LINE FEED. THE COMPLETE NUMBER MUST
 *THEN BE RETYPED. THE INPUT IS TERMINATED BY TYPING A CARRIAGE RETURN.
 *CALL:

* RDOCT ;; READ AN OCTAL NUMBER
 * RETURN HERE ;; LOW ORDER BITS ARE ON TOP OF THE STACK
 * ;; HIGH ORDER BITS ARE IN \$HIOCT

```

8595 047576 010246
8596 047600 104411
8597 047602 012600
8598 047604 010037 047710
8599 047610 005001
8600 047612 005002
8601 047614 112046
8602 047616 001420
8603 047620 122716 000060
8604 047624 003026
8605 047626 122716 000067
8606 047632 002423
8607 047634 006301
8608 047636 006102
8609 047640 006301
8610 047642 006102
8611 047644 006301
8612 047646 006102
8613 047650 042716 177770
8614 047654 062601
8615 047656 000756
8616 047660 005726
8617 047662 010166 000012
8618 047666 010237 047720
8619 047672 012602
8620 047674 012601
8621 047676 012600
8622 047700 000002
8623 047702 005726
8624 047704 105010
8625 047706 104401
8626 047710 000000
8627 047712 104401 001204
8628 047716 000730
8629 047720 000000
8630
8631
8632
8633
8634
8635
8636
8637
8638
8639
8640
8641 047722 104413
8642 047724 016601 000002
8643 047730 012705 050041
8644 047734 012704 000014
8645 047740 012703 177770
8646 047744 012100
8647 047746 012101
8648 047750 005002
8649 047752 110245
8650 047754 010002

MOV R2,-(SP) ;;PUSH R2 ON STACK
1$: RDLIN ;;READ AN ASCIZ LINE
MOV (SP)+,R0 ;;GET ADDRESS OF 1ST CHARACTER
MOV R0,5$ ;;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)
R1T 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$: TST (SP)+ ;;CLEAN TERMINATOR FROM STACK
MOV R1,12(SP) ;;SAVE THE RESULT
MOV R2,$HI OCT
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
TYPE $QUES ;;?" "CR" & "LF"
BR 1$ ;;TRY AGAIN
$HI OCT: .WORD 0 ;;HIGH ORDER BITS GO HERE
.SBTL 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#$SDB20 ;; CALL THE ROUTINE
;* RETURN ;; THE ADDRESS OF THE FIRST ASCIZ CHAR. IS ON THE STACK

$SDB20: SAVREG ;;SAVE ALL REGISTERS
MOV 2(SP),R1 ;;PICKUP THE POINTER TO LOW WORD
MOV #$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

```

```

8651 047756 005304          DEC      R4          ;; COUNT THIS CHARACTER
8652 047760 003007          BGT     3$          ;; BR IF NOT THE LAST DIGIT
8653 047762 001405          BEQ     2$          ;; BR IF IT IS THE LAST DIGIT
8654 047764 005205          INC     R5          ;; ALL DIGITS DONE-ADJUST POINTER FOR FIRST
8655 047766 010566 000002  MOV     R5,2(SP)    ;; ASCIZ CHAR. & PUT IT ON THE STACK
8656 047772 104414          RESREG          ;; RESTORE ALL REGISTERS
8657 047774 000207          RTS     PC          ;; RETURN TO USER
8658 047776 006203          2$: ASR     R3          ;; POSITION THE MASK FOR THE LAST DIGIT
8659 050000 006001          3$: ROR     R1          ;; POSITION THE BINARY NUMBER FOR
8660 050002 006000          ROR     R0          ;; THE NEXT OCTAL DIGIT
8661 050004 006001          ROR     R1
8662 050006 006000          ROR     R0
8663 050010 006001          ROR     R1
8664 050012 006000          ROR     R0
8665 050014 040302          BIC     R3,R2      ;; MASK OUT ALL JUNK
8666 050016 062702 000060  ADD     #'0,R2     ;; MAKE THIS CHAR. ASCII
8667 050022 000753          BR      1$          ;; GO PUT IT IN THE DATA TABLE
8668 050024 000016          $OCTVL: .BLKB 14.  ;; RESERVE DATA TABLE
8669                                     .SBTTL DOUBLE LENGTH BINARY TO DECIMAL ASCII CONVERT ROUTINE
8670
8671                                     ;;*****
8672                                     ;;*THIS ROUTINE WILL CONVERT A 32-BIT BINARY NUMBER TO AN UNSIGNED
8673                                     ;;*DECIMAL (ASCII) NUMBER. THE SIGN OF THE BINARY NUMBER MUST BE
8674                                     ;;*POSITIVE.
8675                                     ;;*CALL
8676                                     ;;*
8677                                     ;;*   MOV     #PNTR, -(SP)  ;; POINTER TO LOW WORD OF BINARY NUMBER
8678                                     ;;*   JSR     PC, @#$DB2D
8679                                     ;;*   RETURN
8680                                     ;;*
8681                                     ;;*
8682 050042 104413          $DB2D: SAVREG      ;; SAVE REGISTERS
8683 050044 016602 000002  MOV     2(SP), R2  ;; PICKUP THE DATA POINTER
8684 050050 012700 050222  MOV     #$DECVL, R0 ;; GET ADDRESS OF "$DECVL" STRING
8685 050054 010066 000002  MOV     R0, 2(SP)  ;; PUT ADDRESS OF ASCIZ STRING ON STACK
8686 050060 012201          MOV     (R2)+, R1  ;; PICKUP THE BINARY NUMBER
8687 050062 012202          MOV     (R2)+, R2
8688 050064 012737 000012 050140  MOV     #10, 4$    ;; SET UP TO DO 10 CONVERSIONS
8689 050072 012704 050152  MOV     #$TNPWR, R4 ;; ADDRESS OF TEN POWER
8690 050076 012705 050154  MOV     #$TNPWR+2, R5
8691 050102 005003          1$: CLR     R3          ;; CLEAR PARTIAL
8692 050104 161401          2$: SUB     (R4), R1  ;; SUBTRACT TEN POWER
8693 050106 005602          SBC     R2
8694 050110 161502          SUB     (R5), R2
8695 050112 002402          BLT     3$          ;; BR IF TEN POWER TO LARGE
8696 050114 005203          INC     R3          ;; ADD 1 TO PARTIAL
8697 050116 000772          BR      2$          ;; LOOP
8698 050120 062401          3$: ADD     (R4)+, R1  ;; RESTORE SUBTRACTED VALUE
8699 050122 005502          ADC     R2
8700 050124 062402          ADD     (R4)+, R2
8701 050126 022525          CMP     (R5)+, (R5)+ ;; MOVE TO NEXT TEN POWER
8702 050130 052703 000060  BIS     #'0, R3     ;; CHANGE PARTIAL TO ASCII
8703 050134 110320          MOVB   R3, (R0)+   ;; SAVE IT
8704 050136 005327          DEC     (PC)+      ;; DONE?
8705 050140 000000          4$: .WORD 0
8706 050142 001357          BNE    1$          ;; BR IF NO

```

8707	050144	105020	
8708	050146	104414	
8709	050150	000207	
8710	050152	145000	
8711	050154	035632	
8712	050156	160400	
8713	050160	002765	
8714	050162	113200	
8715	050164	000230	
8716	050166	041100	
8717	050170	000017	
8718	050172	103240	
8719	050174	000001	
8720	050176	023420	
8721	050200	000000	
8722	050202	001750	
8723	050204	000000	
8724	050206	000144	
8725	050210	000000	
8726	050212	000012	
8727	050214	000000	
8728	050216	000001	
8729	050220	000000	
8730	050222	000014	
8731			
8732			
8733			
8734			
8735			
8736			
8737			
8738			
8739			
8740			
8741			
8742	050236	016637	000002
8743	050244	012746	050266
8744	050250	004737	050042
8745	050254	062716	000005
8746	050260	012666	000002
8747	050264	000207	
8748	050266	000000	000000
8749			
8750			
8751			
8752			
8753			
8754			
8755			
8756			
8757			
8758			
8759	050272	010046	
8760	050274	016600	000004
8761	050300	105710	
8762	050302	001403	

```

CLRB (R0) :: TERMINATOR
RESREG :: RESTORE REGISTERS
RTS PC :: RETURN
STMPWR: 145000 :: 1.0E09
        35632
        160400 :: 1.0E08
        2765
        113200 :: 1.0E07
        230
        041100 :: 1.0E06
        17
        103240 :: 1.0E05
        1
        23420 :: 1.0E04
        0
        1750 :: 1.0E03
        0
        144 :: 1.0E02
        0
        12 :: 1.0E01
        0
        1 :: 1.0E00
        0
SDECVL: .BLKB 12. :: RESERVE STORAGE FOR ASCIZ STRING
.SBTTL SINGLE LENGTH BINARY TO DECIMAL ASCII ROUTINE

:*****
: THIS ROUTINE WILL CONVERT A 16-BIT UNSIGNED BINARY NUMBER TO AN
: UNSIGNED DECIMAL ASCII NUMBER.
: CALL
: * MOV NUMBER, -(SP) :: PUT BINARY NUMBER ON THE STACK
: * JSR PC, 2*SSB2D :: CALL
: * RETURN :: ADDRESS OF THE 1ST ASCII CHAR. IS ON THE STACK

SSB2D: MOV 2(SP), 1S :: SAVE BINARY NUMBER
      MOV #1S, -(SP) :: SET POINTER
      JSR PC, 2*SSB2D :: CALL DOUBLE LENGTH CONVERT
      ADD #5, (SP) :: ONLY ALLOW FIVE CHARACTERS
      MOV (SP)+, 2(SP) :: PICKUP POINTER
      RTS PC :: RETURN
1S: .WORD 0,0
.SBTTL TYPE NUMERICAL ASCII STRING SUPPRESS LEADING ZEROS

:*****
: THIS ROUTINE IS USED TO TYPE AN ASCII NUMBER SUPPRESSING THE
: LEADING NUMBERS.
: CALL
: * MOV #NUMADR, -(SP) :: FIRST ADDRESS OF ASCII STRING
: * JSR PC, 2*SSUPRS

SSUPRS: MOV R0, -(SP) :: SAVE R0
      MOV 4(SP), R0 :: PICKUP THE POINTER
1S: TSTB (R0) :: TERMINATOR?
     BEQ 2S :: BR IF YES

```

```

8763 050304 122720 000060
8764 050310 001773
8765 050312 005300
8766 050314 010037 050322
8767 050320 104401
8768 050322 000000
8769 050324 012600
8770 050326 012616
8771 050330 000207
8772
8773
8774
8775
8776
8777
8778
8779
8780
8781
8782
8783
8784
8785
8786 050332
8787 050332 010046
8788 050334 010146
8789 050336 010246
8790 050340 005046
8791 050342 016601 000012
8792 050346 100002
8793 050350 005216
8794 050352 005401
8795 050354 016602 000014
8796 050360 100002
8797 050362 005316
8798 050364 005402
8799 050366 012746 000021
8800 050372 005007
8801 050374 103001
8802 050376 060200
8803 050400 006000
8804 050402 006001
8805 050404 005016
8806 050406 001772
8807 050410 022516
8808 050412 001703
8809 050414 005400
8810 050416 005401
8811 050420 005600
8812 050422 005726
8813 050424 010066 000012
8814 050430 010166 000010
8815 050434 012602
8816 050436 012601
8817 050440 012600
8818 050442 000207

```

```

CMPB 0,(RO)+      ;;S THIS AN ASCII "0" ?
BEQ 1$           ;;R IF YES
DEC  RC          ;;BACKJP BY "1"
MOV  RC,3$      ;;SAVE FOR TYPING
TYPE          ;;GO TYPE
WORD 0          ;;ASCIZ POINTER GOES HERE
MOV  (SP)+,RO   ;;RESTORE RO
MOV  (SP)+,(SP) ;;RESTORE THE STACK
RTS  PC         ;;RETURN

.SBTTL INTEGER MULTIPLY ROUTINE

*****
*CALL
*   MOV  MULTIPLIER, -(SP)
*   MOV  MULTIPLICAND, -(SP)
*   JSR  PC, @SMULT
*   RETURN ;;PRODUCT IS ON THE STACK
*
*   STACK  PRODUCT
*   -----
*   TOP    LSB'S
*   +2     MSB'S
*
SMULT:
MOV  R0, -(SP)      ;;PUSH R0 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
MOV  14(SP), R2     ;;GET THE MULTIPLIER
BPL  2$            ;;BR IF PLUS
DEC  (SP)           ;;UPDATE THE SIGN KEY
NEG  R2             ;;MAKE THE MULTIPLIER POSTIVE
MOV  17, -(SP)      ;;SET THE LOOP COUNT
CLR  R0            ;;SETUP FOR THE MULTIPLY LOOP
BCC  4$            ;;DON'T ADD IF MULTIPLICAND = 0
ADD  R2, R0
ROR  R0             ;;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  R0             ;;YES--SO MAKE IT SO
SBC  R0
TST  (SP)+         ;;CLEAR SIGN INFO. OFF OF STACK
MOV  R0, 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)+, R0     ;;POP STACK INTO R0
RTS  PC

```

```

8819
8820
8821
8822
8823
8824
8825
8826
8827
8828
8829
8830
8831
8832
8833
8834
8835
8836 050444
8837 050444 010046
8838 050446 010146
8839 050450 010246
8840 050452 010346
8841 050454 010446
8842 050456 010546
8843 050460 016646 000022
8844 050464 016646 000022
8845 050470 016646 000022
8846 050474 016646 000022
8847 050500 000002
8848
8849
8850
8851
8852 050502
8853 050502 012666 000022
8854 050506 012666 000022
8855 050512 012666 000022
8856 050516 012666 000022
8857 050522 012605
8858 050524 012604
8859 050526 012603
8860 050530 012602
8861 050532 012601
8862 050534 012600
8863 050536 000002
8864
8865
8866
8867
8868
8869
8870
8871
8872 050540 010046 000002
8873 050542 016600
8874 050546 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(SR), -(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

```

```

8875 050550 111000          MOVB  (RO),RO          ;;GET RIGHT BYTE OF TRAP
8876 050552 006300          ASL  RO              ;;POSITION FOR INDEXING
8877 050554 016000 050574  MOV  $TRPAD(RO),RC    ;;INDEX TO TABLE
8878 050560 000200          RTS   RO              ;;GO TO ROUTINE
8879
8880
8881
8882

```

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

```

8883 050562 011646 000004 000002 $TRAP2: MOV  (SP),-(SP)  ;;MOVE THE PC DOWN
8884 050564 016666          MOV  4(SP),2(SP)      ;;MOVE THE PSW DOWN
8885 050572 000002          RTI                    ;;RESTORE THE PSW
8886
8887
8888
8889
8890
8891
8892
8893

```

.SBTTL TRAP TABLE

;*THIS TABLE CONTAINS THE STARTING ADDRESSES OF THE ROUTINES CALLED
;*BY THE "TRAP" INSTRUCTION.

```

:          ROUTINE
:          -----
8894 050574 050562 $TRPAD: .WORD  $TRAP2
8895 050576 045110 $TYPE  ;;CALL=TYPE   TRAP+1(104401) TTY TYPEOUT ROUTINE
8896 050600 046110 $TYPOC ;;CALL=TYPOC  TRAP+2(104402) TYPE OCTAL NUMBER (WITH LEADING ZEROS)
8897 050602 046064 $TYPOS ;;CALL=TYPOS  TRAP+3(104403) TYPE OCTAL NUMBER (NO LEADING ZEROS)
8898 050604 046124 $TYPON ;;CALL=TYPON   TRAP+4(104404) TYPE OCTAL NUMBER (AS PER LAST CALL)
8899 050606 045372 $TYPDS ;;CALL=TYPDS  TRAP+5(104405) TYPE DECIMAL NUMBER (WITH SIGN)
8900
8901 050610 046700 $GTSWR ;;CALL=GTSWR  TRAP+6(104406) GET SOFT-SWR SETTING
8902
8903 050612 046610 $CKSWR ;;CALL=CKSWR  TRAP+7(104407) TEST FOR CHANGE IN SOFT-SWR
8904 050614 047152 $RDCHR ;;CALL=RDCHR  TRAP+10(104410) TTY TYPEIN CHARACTER ROUTINE
8905 050616 047242 $RDLIN ;;CALL=RDLIN  TRAP+11(104411) TTY TYPEIN STRING ROUTINE
8906 050620 047562 $RDOCT ;;CALL=RDOCT  TRAP+12(104412) READ AN OCTAL NUMBER FROM TTY
8907 050622 050444 $$SAVREG ;;CALL=SAVREG TRAP+13(104413) SAVE RO-R5 ROUTINE
8908 050624 050502 $RESREG ;;CALL=RESREG TRAP+14(104414) RESTORE RO-R5 ROUTINE
8909 050626 043426 $SCOPI$ ;;CALL=SCOPI TRAP+15(104415) INTERNAL LOOP ON ERROR
8910

```

8911
8912
8913
8914
8915 050630 005015 047125 041111
8916 050636 051525 051040 030113
8917 050644 020066 051104 053111
8918 050652 020105 044504 043501
8919 050660 047516 052123 041511
8920 050666 005015 050011 051101
8921 050674 020124 062
8922 050677 015 046412 044501
8923 050704 042116 041505 030455
8924 050712 026461 055104 033122
8925 050720 026511 026503 041120
8926 050726 005015
8927 050730 005015 025011 025052
8928 050736 025052 041440 052501
8929 050744 044524 047117 025040
8930 050752 025052 025052 005015
8931 050760 005015 044124 051511
8932 050766 050040 07522 051107
8933 050774 046501 051440 047510
8934 051002 046125 020104 041040
8935 051010 020105 040510 052114
8936 051016 042105 047440 046116
8937 051024 020131 054502 052040
8938 051032 050131 047111 020107
8939 051040 047503 052116 047522
8940 051046 026514 103
8941 051051 015 047412 044124
8942 051056 051105 044527 042523
8943 051064 020054 040503 052122
8944 051072 044522 043504 020105
8945 051100 047506 046522 052101
8946 051106 044524 043516 040440
8947 051114 042116 020054 051117
8948 051122 052040 042510 042040
8949 051130 044522 042526
8950 051134 005015 040515 020131
8951 051142 042502 046040 043105
8952 051150 020124 047111 040440
8953 051156 020116 047125 042504
8954 051164 042524 046522 047111
8955 051172 042105 051440 040524
8956 051200 042524
8957 051202 005015 047111 052111
8958 051210 040511 046114 026131
8959 051216 042040 044522 042526
8960 051224 020123 047524 041040
8961 051232 020105 042524 052123
8962 051240 042105 051440 047510
8963 051246 046125 020104 040510
8964 051254 042526 006472 012
8965 051261 015 040412 020056
8966 051266 042510 042101 020123

.SBTTL SERVICE MESSAGES

MSG1: .ASCII <CR><LF>/UNIBUS RK06 DRIVE DIAGNOSTIC/

.ASCII <CR><LF>/ PART 2/

.ASCII <CR><LF>/MAINDEC-11-DZR6I-C-PB/<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 DRIVE/

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

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

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

8967	051274	040515	052516	046101	
8968	051302	054514	046040	040517	
8969	051310	042504	104		
8970	051313	015	041012	020056	.ASCII <CR><LF>/B. CORRECT PORT SELECTED/
8971	051320	047503	051122	041505	
8972	051326	020124	047520	052122	
8973	051334	051440	046105	041505	
8974	051342	042524	104		
8975	051345	015	041412	020056	.ASCII <CR><LF>/C. WRITE LOCK DISABLED/
8976	051352	051127	052111	020105	
8977	051360	047514	045503	042040	
8978	051366	051511	041101	042514	
8979	051374	104			
8980	051375	015	042012	020056	.ASCII <CR><LF>/D. DRIVE READY INDICATOR ON/<CR><LF>
8981	051402	051104	053111	020105	
8982	051410	042522	042101	020131	
8983	051416	047111	044504	040503	
8984	051424	047524	020122	047117	
8985	051432	005015			
8986	051434	005015	051104	053111	.ASCII <CR><LF>/DRIVES NOT TO BE TESTED MUST HAVE/
8987	051442	051505	047040	052117	
8988	051450	052040	020117	042502	
8989	051456	052040	051505	042524	
8990	051464	020104	052515	052123	
8991	051472	044040	053101	105	
8992	051477	015	041012	052117	.ASCIZ <CR><LF>/BOTH PORTS DESELECTED/<CR><LF>
8993	051504	020110	047520	052122	
8994	051512	020123	042504	042523	
8995	051520	042514	052103	042105	
8996	051526	005015	000		
8997					
8998					
8999	051531	015	041012	020105	MSG2: .ASCIZ <CR><LF>/BE SURE TO PUT SCRATCH PACK IN DRIVE 0/
9000	051536	052523	042522	052040	
9001	051544	020117	052520	020124	
9002	051552	041523	040522	041524	
9003	051560	020110	040520	045503	
9004	051566	044440	020116	051104	
9005	051574	053111	020105	000060	
9006	051602	005015	051104	053111	MSG3: .ASCIZ <CR><LF>/DRIVE(S) TO BE TESTED: /
9007	051610	024105	024523	052040	
9008	051616	020117	042502	052040	
9009	051624	051505	042524	035104	
9010	051632	000040			
9011	051634	005015	054524	042520	MSG4: .ASCIZ <CR><LF>/TYPE BUSS ADDRESS IF NOT 177440: /
9012	051642	041012	051525	020123	
9013	051650	042101	051104	051505	
9014	051656	020123	043111	047040	
9015	051664	052117	030440	033467	
9016	051672	032064	035060	020040	
9017	051700	000			
9018	051701	015	052012	050131	MSG5: .ASCIZ <CR><LF>/TYPE CONTROLLER INTERRUPT VECTOR IF NOT 210: /
9019	051706	020105	047503	052116	
9020	051714	047522	046114	051105	
9021	051722	044440	052116	051105	
9022	051730	052522	052120	053040	

9023	051736	041505	047524	020122	
9024	051744	043111	047040	052117	
9025	051752	031040	030061	020072	
9026	051760	000040			
9027	051762	005015	047111	042524	MSG6: .ASCIZ <CR><LF>/INTERRUPT OCCURRED AT PC=/ /
9028	051770	051122	050125	020124	
9029	051776	041517	052503	051122	
9030	052004	042105	040440	020124	
9031	052012	041520	000075		
9032	052016	005015	051104	053111	MSG7: .ASCIZ <CR><LF>/DRIVE 0 WILL NOT BE TESTED/ /
9033	052024	020105	020060	044527	
9034	052032	046114	047040	052117	
9035	052040	041040	020105	042524	
9036	052046	052123	042105	000	
9037	052053	015	051012	040505	MSG8: .ASCIZ <CR><LF>/READ DATA WITH OFFSET TEST/<CR><LF> /
9038	052060	020104	040504	040524	
9039	052066	053440	052111	020110	
9040	052074	043117	051506	052105	
9041	052102	052040	051505	006524	
9042	052110	000012			
9043	052112	005015	042510	042101	MSG9: .ASCIZ <CR><LF>/HEAD NO./ /
9044	052120	047040	027117	000	
9045	052125	015	005012	044527	MSG10: .ASCIZ <CR><LF><LF>/WILL TEST DRIVES:/ /
9046	052132	046114	052040	051505	
9047	052140	020124	051104	053111	
9048	052146	051505	000072		
9049	052152	005015	050012	053517	MSG11: .ASCIZ <CR><LF><LF>/POWER UP RESTART TO TEST 1/<CR><LF> /
9050	052160	051105	052440	020120	
9051	052166	042522	052123	051101	
9052	052174	020124	047524	052040	
9053	052202	051505	020124	006461	
9054	052210	000012			
9055	052212	005015	047516	046040	MSG13: .ASCII <CR><LF>/NO L OR P CLOCKS PRESENT/ /
9056	052220	047440	020122	020120	
9057	052226	046103	041517	051513	
9058	052234	050040	042522	042523	
9059	052242	052116			
9060	052244	005015	046101	020114	.ASCIZ <CR><LF>/ALL TIMING TESTS BYPASSED/ /
9061	052252	044524	044515	043516	
9062	052260	052040	051505	051524	
9063	052266	041040	050131	051501	
9064	052274	042523	000104		
9065	052300	005015	054502	040520	MSG14: .ASCIZ <CR><LF>/BYPASSING DRIVE / /
9066	052306	051523	047111	020107	
9067	052314	051104	053111	020105	
9068	052322	000			
9069	052323	015	005012	051104	MSG15: .ASCIZ <CR><LF><LF>/DRIVE / /
9070	052330	053111	020105	000	
9071	052335	015	042012	044522	MSG16: .ASCIZ <CR><LF>/DRIVE SERIAL NO. / /
9072	052342	042526	051440	051105	
9073	052350	040511	020114	047516	
9074	052356	020056	000		
9075	052361	015	041412	051101	MSG17: .ASCIZ <CR><LF>/CARTRIDGE SERIAL NO. / /
9076	052366	051124	042111	042507	
9077	052374	051440	051105	040511	
9078	052402	020114	047516	020056	

UNIBUS RK06 DRIVE DIAGNOSTIC PART 2
DZR6IC.P11 07-OCT-76 13:50

MACY11 27(1006) 07-OCT-76 14:14 PAGE 177
SERVICE MESSAGES

SEG 0177

9079	052410	000			
9080	052411	015	052012	046511	MSG18: .ASCIZ <CR><LF>/TIME BETWEEN OUTER LIMIT & HEADS HOME DURING UNLOADING: /
9081	052416	020105	042502	053524	
9082	052424	042505	020116	052517	
9083	052432	042524	020122	044514	
9084	052440	044515	020124	020046	
9085	052446	042510	042101	020123	
9086	052454	047510	042515	042040	
9087	052462	051125	047111	020107	
9088	052470	047125	047514	042101	
9089	052476	047111	035107	000040	
9090	052504	005015	054502	040520	MSG19: .ASCIZ <CR><LF>/BYPASSING ALL WRITE TESTS/<CR><LF>
9091	052512	051523	047111	020107	
9092	052520	046101	020114	051127	
9093	052526	052111	020105	042524	
9094	052534	052123	006523	000012	
9095	052542	005015	044524	042515	MSG20: .ASCIZ <CR><LF>/TIME BETWEEN HEADS HOME & SERVO SIG PRES DURING LOADING: /
9096	052550	041040	052105	042527	
9097	052556	047105	044040	040505	
9098	052564	051504	044040	046517	
9099	052572	020105	020046	042523	
9100	052600	053122	020117	044523	
9101	052606	020107	051120	051505	
9102	052614	042040	051125	047111	
9103	052622	020107	047514	042101	
9104	052630	047111	035107	000040	
9105	052636	005015	044524	042515	MSG21: .ASCIZ <CR><LF>/TIME BETWEEN OUTER LIMIT & INNER LIMIT DURING LOADING: /
9106	052644	041040	052105	042527	
9107	052652	047105	047440	052125	
9108	052660	051105	046040	046511	
9109	052666	052111	023040	044440	
9110	052674	047116	051105	046040	
9111	052702	046511	052111	042040	
9112	052710	051125	047111	020107	
9113	052716	047514	042101	047111	
9114	052724	035107	000040		
9115	052730	005015	044524	042515	MSG22: .ASCIZ <CR><LF>/TIME BETWEEN INNER LIMIT & OUTER LIMIT DURING LOADING: /
9116	052736	041040	052105	042527	
9117	052744	047105	044440	047116	
9118	052752	051105	046040	046511	
9119	052760	052111	023040	047440	
9120	052766	052125	051105	046040	
9121	052774	046511	052111	042040	
9122	053002	051125	047111	020107	
9123	053010	047514	042101	047111	
9124	053016	035107	000040		
9125	053022	005015	047516	041440	MSG23: .ASCIZ <CR><LF>/NO CLOCK INTERRUPTS PRESENT, ABORTING TIMING TESTS/
9126	053030	047514	045503	044440	
9127	053036	052116	051105	052522	
9128	053044	052120	020123	051120	
9129	053052	051505	047105	026124	
9130	053060	040440	047502	052122	
9131	053066	047111	020107	044524	
9132	053074	044515	043516	052040	
9133	053102	051505	051524	000	
9134	053107	015	040412	042526	MSG24: .ASCIZ <CR><LF>/AVERAGE TIME FOR 1 REVOLUTION: /

9135	053114	040522	042507	052040
9136	053122	046511	020105	047506
9137	053130	020122	020061	042522
9138	053136	047526	052514	044524
9139	053144	047117	020072	000
9140	053151	040	051525	000
9141	053155	015	005012	041101
9142	053162	051117	044524	043516
9143	053170	042040	052101	020101
9144	053176	042524	052123	020123
9145	053204	020046	047507	047111
9146	053212	020107	047524	052040
9147	053220	046511	047111	020107
9148	053226	042524	052123	006523
9149	053234	005012	000	
9150	053237	015	005012	046101
9151	053244	020114	051104	053111
9152	053252	051505	052040	051505
9153	053260	042524	006504	005012
9154	053266	000		
9155	053267	015	040412	042526
9156	053274	040522	042507	052040
9157	053302	046511	020105	047506
9158	053310	020122	030464	020060
9159	053316	054503	044514	042116
9160	053324	051105	043040	053517
9161	053332	051101	020104	042523
9162	053340	045505	020072	000040
9163	053346	005015	053101	051105
9164	053354	043501	020105	044524
9165	053362	042515	043040	051117
9166	053370	032040	030061	041440
9167	053376	046131	047111	042504
9168	053404	020122	042522	042526
9169	053412	051522	020105	042523
9170	053420	045505	020072	000040
9171	053426	005015	053101	051105
9172	053434	043501	020105	044524
9173	053442	042515	043040	051117
9174	053450	030440	041440	046131
9175	053456	047111	042504	020122
9176	053464	030050	052040	020117
9177	053472	024461	043040	053517
9178	053500	051101	020104	042523
9179	053506	045505	020072	000040
9180	053514	005015	053101	051105
9181	053522	043501	020105	044524
9182	053530	042515	043040	051117
9183	053536	030440	041440	046131
9184	053544	047111	042504	020122
9185	053552	030050	052040	020117
9186	053560	024461	051040	053105
9187	053566	051105	042523	051440
9188	053574	042505	035113	020040
9189	053602	000		
9190	053603	015	040412	042526

MSG25: .ASCIZ / US/
MSG26: .ASCIZ <CR><LF><LF>/ABORTING DATA TESTS & GOING TO TIMING TESTS/<CR><LF><LF>

MSG27: .ASCIZ <CR><LF><LF>/ALL DRIVES TESTED/<CR><LF><LF>

MSG28: .ASCIZ <CR><LF>/AVERAGE TIME FOR 410 CYLINDER FOWARD SEEK: /

MSG29: .ASCIZ <CR><LF>/AVERAGE TIME FOR 410 CYLINDER REVERSE SEEK: /

MSG30: .ASCIZ <CR><LF>/AVERAGE TIME FOR 1 CYLINDER (0 TO 1) FOWARD SEEK: /

MSG31: .ASCIZ <CR><LF>/AVERAGE TIME FOR 1 CYLINDER (0 TO 1) REVERSE SEEK: /

MSG32: .ASCIZ <CR><LF>/AVERAGE TIME FOR 137 CYLINDER FOWARD SEEK: /

9191	053610	040522	042507	052040
9192	053616	046511	020105	047506
9193	053624	020122	031461	020067
9194	053632	054503	044514	042116
9195	053640	051105	043040	053517
9196	053646	051101	020104	042523
9197	053654	045505	020072	000040
9198	053662	005015	053101	051105
9199	053670	043501	020105	044524
9200	053676	042515	043040	051117
9201	053704	030440	033463	041440
9202	053712	046131	047111	042504
9203	053720	020122	042522	042526
9204	053726	051522	020105	042523
9205	053734	045505	020072	000040
9206	053742	005015	053101	051105
9207	053750	043501	020105	047506
9208	053756	040527	042122	051440
9209	053764	042520	042105	041040
9210	053772	052105	042527	047105
9211	054000	041440	046131	047111
9212	054006	042504	051522	030440
9213	054014	034062	023040	031040
9214	054022	033065	020072	000040
9215	054030	005015	053101	051105
9216	054036	043501	020105	042522
9217	054044	042526	051522	020105
9218	054052	050123	042505	020104
9219	054060	042502	053524	042505
9220	054066	020116	054503	044514
9221	054074	042116	051105	020123
9222	054102	031061	020070	020046
9223	054110	032462	035066	020040
9224	054116	000		
9225	054117	040	044440	041516
9226	054124	042510	020123	042520
9227	054132	020122	042523	047503
9228	054140	042116	000	
9229	054143	015	047012	020117
9230	054150	051127	052111	020105
9231	054156	044103	041505	020113
9232	054164	051105	047522	020122
9233	054172	052101	046440	054101
9234	054200	050040	051517	052111
9235	054206	053111	020105	043117
9236	054214	051506	052105	000
9237	054221	015	047012	020117
9238	054226	051127	052111	020105
9239	054234	044103	041505	020113
9240	054242	051105	047522	020122
9241	054250	052101	046440	054101
9242	054256	047040	043505	052101
9243	054264	053111	020105	043117
9244	054272	051506	052105	005015
9245	054300	000		
9246	054301	015	053412	044522

MSG33: .ASCIZ <CR><LF>/AVERAGE TIME FOR 137 CYLINDER REVERSE SEEK: /

MSG34: .ASCIZ <CR><LF>/AVERAGE FOWARD SPEED BETWEEN CYLINDERS 128 & 256: /

MSG35: .ASCIZ <CR><LF>/AVERAGE REVERSE SPEED BETWEEN CYLINDERS 128 & 256: /

MSG36: .ASCIZ / INCHES PER SECOND/

MSG37: .ASCIZ <CR><LF>/NO WRITE CHECK ERROR AT MAX POSITIVE OFFSET/

MSG38: .ASCIZ <CR><LF>/NO WRITE CHECK ERROR AT MAX NEGATIVE OFFSET/<CR><LF>

MSG39: .ASCIZ <CR><LF>/WRITE CHECK FAILURE AT OFFSET =/

9247	054306	042524	041440	042510	
9248	054314	045503	043040	044501	
9249	054322	052514	042522	040440	
9250	054330	020124	043117	051506	
9251	054336	052105	036440	000	
9252	054343	015	041412	052517	MSG40: .ASCII <CR><LF>/COULD NOT READ BAD SECTOR INFO ON CYL 410/
9253	054350	042114	047040	052117	
9254	054356	051040	040505	020104	
9255	054364	040502	020104	042523	
9256	054372	052103	051117	044440	
9257	054400	043116	020117	047117	
9258	054406	041440	046131	032040	
9259	054414	030061			
9260	054416	005015	051117	040440	.ASCIZ <CR><LF>/OR ALIGNMENT CARTRIDGE USED/<CR><LF>
9261	054424	044514	047107	042515	
9262	054432	052116	041440	051101	
9263	054440	051124	042111	042507	
9264	054446	052440	042523	006504	
9265	054454	000012			
9266	054456	005015	054502	040520	MSG41: .ASCIZ <CR><LF>/BYPASSING ALL TIMING TESTS/<CR><LF>
9267	054464	051523	047111	020107	
9268	054472	046101	020114	044524	
9269	054500	044515	043516	052040	
9270	054506	051505	051524	005015	
9271	054514	000			
9272	054515	015	040412	046114	MSG42: .ASCIZ <CR><LF>/ALL TIMING TESTS TEMPORARILY BYPASSED/<CR><LF>
9273	054522	052040	046511	047111	
9274	054530	020107	042524	052123	
9275	054536	020123	042524	050115	
9276	054544	051117	051101	046111	
9277	054552	020131	054502	040520	
9278	054560	051523	042105	005015	
9279	054566	000			
9280	054567	015	051412	047510	MSG43: .ASCIZ <CR><LF>/SHOULD BE BETWEEN 0.140 & 0.429 SEC/
9281	054574	046125	020104	042502	
9282	054602	041040	052105	042527	
9283	054610	047105	030040	030456	
9284	054616	030064	023040	030040	
9285	054624	032056	034462	051440	
9286	054632	041505	000		
9287	054635	015	051412	047510	MSG44: .ASCIZ <CR><LF>/SHOULD BE BETWEEN 1.9 & 2.6 SEC/
9288	054642	046125	020104	042502	
9289	054650	041040	052105	042527	
9290	054656	047105	030440	034456	
9291	054664	023040	031040	033056	
9292	054672	051440	041505	000	
9293	054677	015	050012	047522	MSG74: .ASCIZ <CR><LF>/PROGRAM ABORT PENDING...PLEASE WAIT/
9294	054704	051107	046501	040440	
9295	054712	047502	052122	050040	
9296	054720	047105	044504	043516	
9297	054726	027056	050056	042514	
9298	054734	051501	020105	040527	
9299	054742	052111	000		
9300	054745	015	044012	046101	MSG75: .ASCIZ <CR><LF>/HALT PENDING...PLEASE WAIT/
9301	054752	020124	042520	042116	
9302	054760	047111	027107	027056	

9303	054766	046120	040505	042523	
9304	054774	053440	044501	000124	
9305	055002	005015	051120	043517	MSG76: .ASCIZ <CR><LF>/PROGRAM ABORTED/
9306	055010	040522	020115	041101	
9307	055016	051117	042524	000104	
9308	055024	005015	050103	020125	MSG77: .ASCIZ <CR><LF>/CPU HALTED/
9309	055032	040510	052114	042105	
9310	055040	000			
9311					
9312					.SBTTL ERROR MESSAGES
9313					
9314	055041	015	042412	051122	EM1: .ASCIZ <CR><LF>/ERROR, ONLY 0 THRU 7 ALLOWED, TRY AGAIN/<CR><LF>
9315	055046	051117	020054	047117	
9316	055054	054514	030040	052040	
9317	055062	051110	020125	020067	
9318	055070	046101	047514	042527	
9319	055076	026104	052040	054522	
9320	055104	040440	040507	047111	
9321	055112	005015	000		
9322	055115	104	044522	042526	EM2: .ASCIZ /DRIVE # IN RKCS2 CANNOT BE READ BACK CORRECTLY IN RKMR2/
9323	055122	021440	044440	020116	
9324	055130	045522	051503	020062	
9325	055136	040503	047116	052117	
9326	055144	041040	020105	042522	
9327	055152	042101	041040	041501	
9328	055160	020113	047503	051122	
9329	055166	041505	046124	020131	
9330	055174	047111	051040	046513	
9331	055202	031122	000		
9332	055205	015	040412	047502	EM3: .ASCIZ <CR><LF>/ABORT TESTS...UNEXPECTED TIME OUT AT PC=
9333	055212	052122	052040	051505	
9334	055220	051524	027056	052456	
9335	055226	042516	050130	041505	
9336	055234	042524	020104	044524	
9337	055242	042515	047440	052125	
9338	055250	040440	020124	041520	
9339	055256	000075			
9340	055260	005015	041101	051117	EM4: .ASCIZ <CR><LF>/ABORT TESTS...UNEXPECTED INTERRUPT AT PC=
9341	055266	020124	042524	052123	
9342	055274	027123	027056	047125	
9343	055302	054105	042520	052103	
9344	055310	042105	044440	052116	
9345	055316	051105	052522	052120	
9346	055324	040440	020124	041520	
9347	055332	000075			
9348	055334	042115	020123	042523	EM5: .ASCIZ /MDS SET IN RKCS2/
9349	055342	020124	047111	051040	
9350	055350	041513	031123	000	
9351	055355	125	042506	051440	EM6: .ASCIZ /UFE SET IN RKCS2/
9352	055362	052105	044440	020116	
9353	055370	045522	051503	000062	
9354	055376	051104	020101	047111	EM7: .ASCIZ /DRA IN RKDS & NED IN RKCS2 RESET; WRONG PORT SELECTED?/
9355	055404	051040	042113	020123	
9356	055412	020046	042516	020104	
9357	055420	047111	051040	041513	
9358	055426	031123	051040	051505	

9359	055434	052105	020073	051127	
9360	055442	047117	020107	047520	
9361	055450	052122	051440	046105	
9362	055456	041505	042524	037504	
9363	055464	000			
9364	055465	104	044522	042526	EM8: .ASCIZ /DRIVE PRESENT BUT NOT SPECIFIED BY OPERATOR/
9365	055472	050040	042522	042523	
9366	055500	052116	041040	052125	
9367	055506	047040	052117	051440	
9368	055514	042520	044503	044506	
9369	055522	042105	041040	020131	
9370	055530	050117	051105	052101	
9371	055536	051117	000		
9372	055541	104	044522	042526	EM9: .ASCIZ /DRIVE NOT PRESENT BUT SPECIFIED BY OPERATOR/
9373	055546	047040	052117	050040	
9374	055554	042522	042523	052116	
9375	055562	041040	052125	051440	
9376	055570	042520	044503	044506	
9377	055576	042105	041040	020131	
9378	055604	050117	051105	052101	
9379	055612	051117	000		
9380	055615	101	047502	052122	EM10: .ASCIZ /ABORT TESTS...CANNOT REFERENCE CONTROLLER REGISTER/
9381	055622	052040	051505	051524	
9382	055630	027056	041456	047101	
9383	055636	047516	020124	042522	
9384	055644	042506	042522	041516	
9385	055652	020105	047503	052116	
9386	055660	047522	046114	051105	
9387	055666	051040	043505	051511	
9388	055674	042524	000122		
9389	055700	051104	020101	047111	EM11: .ASCIZ /DRA IN RKDS & NED IN RKCS2 BOTH SET/
9390	055706	051040	042113	020123	
9391	055714	020046	042516	020104	
9392	055722	047111	051040	041513	
9393	055730	031123	041040	052117	
9394	055736	020110	042523	000124	
9395	055744	047503	052116	047522	EM12: .ASCIZ /CONTROLLER NOT READY IN RKCS1/
9396	055752	046114	051105	047040	
9397	055760	052117	051040	040505	
9398	055766	054504	044440	020116	
9399	055774	045522	051503	000061	
9400	056002	047516	040440	052124	EM13: .ASCIZ /NO ATTN IN RKASOF/
9401	056010	020116	047111	051040	
9402	056016	040513	047523	000106	
9403	056024	047125	054105	042520	EM14: .ASCIZ /UNEXPECTED MEMORY PARITY TRAP/
9404	056032	052103	042105	046440	
9405	056040	046505	051117	020131	
9406	056046	040520	044522	054524	
9407	056054	052040	040522	000120	
9408	056062	045522	041504	023040	EM15: .ASCII /RKDC & RKDA INDICATE THAT WCE OCCURRED AT/
9409	056070	051040	042113	020101	
9410	056076	047111	044504	040503	
9411	056104	042524	052040	040510	
9412	056112	020124	041527	020105	
9413	056120	041517	052503	051122	
9414	056126	042105	040440	124	

9415	056133	015	041412	046131		.ASCIZ	<CR><LF>/CYL 411, TRACK 2, SECTOR 21/
9416	056140	032040	030461	020054			
9417	056146	051124	041501	020113			
9418	056154	026062	051440	041505			
9419	056162	047524	020122	030462			
9420	056170	000					
9421	056171	103	047101	047516	EM16:	.ASCIZ	CANNOT READ BAD SECTOR INFORMATION/
9422	056176	020124	042522	042101			
9423	056204	041040	042101	051440			
9424	056212	041505	047524	020122			
9425	056220	047111	047506	046522			
9426	056226	052101	047511	000116			
9427	056234	042515	051523	043501	EM17:	.ASCIZ	/MESSAGE A0 ERROR/
9428	056242	020105	030101	042440			
9429	056250	051122	051117	000			
9430	056255	115	051505	040523	EM18:	.ASCIZ	/MESSAGE B0 ERROR/
9431	056262	042507	041040	020060			
9432	056270	051105	047522	000122			
9433	056276	042515	051523	043501	EM19:	.ASCIZ	/MESSAGE A1 ERROR/
9434	056304	020105	030501	042440			
9435	056312	051122	051117	000			
9436	056317	115	051505	040523	EM20:	.ASCIZ	/MESSAGE B1 ERROR/
9437	056324	042507	041040	020061			
9438	056332	051105	047522	000122			
9439	056340	042503	051122	051440	EM21:	.ASCIZ	/CERR SET IN RKCS1/
9440	056346	052105	044447	020116			
9441	056354	045522	051503	000061			
9442	056362	047516	042040	044522	EM22:	.ASCII	/NO DRIVES FOUND IN DEVICE MAP (SDEVN)/<CR><LF>
9443	056370	042526	020123	047506			
9444	056376	047125	020104	047111			
9445	056404	042040	053105	041511			
9446	056412	020105	040515	020120			
9447	056417	022050	042504	046526			
9448	056424	006451	012				
9449	056430	123	052105	050125		.ASCIZ	/SETUP CORRECTLY AND RESTART/<CR><LF>
9450	056436	041440	051117	042522			
9451	056444	052103	054514	040440			
9452	056452	042116	051040	051505			
9453	056460	040524	052122	005015			
9454	056466	000					
9455	056467	116	020117	051104	EM23:	.ASCII	/NO DRIVES FOUND ON BUSS/<CR><LF>
9456	056474	053111	051535	043040			
9457	056502	052517	042116	047440			
9458	056510	020116	052502	051523			
9459	056516	005015					
9460	056520	042523	052524	020120		.ASCIZ	/SETUP CORRECTLY AND PRESS 'CONTINUE'/<CR><LF>
9461	056526	047503	051122	041505			
9462	056534	046124	020131	047101			
9463	056542	020104	051120	051505			
9464	056550	020123	041447	047117			
9465	056556	044524	052516	023505			
9466	056564	005015	000				
9467	056567	126	046117	053040	EM24:	.ASCIZ	/VOL VALID NOT SET IN RKMR2.
9468	056574	046101	042111	047040			
9469	056602	052117	051440	052105			
9470	056610	044440	020116	045522			

9471	056616	051115	000062		
9472	056622	005015	042504	042524	EM25: .ASCIZ <CR><LF>/DETECTED 10 BAD SECTORS...ABORTING TEST/
9473	056630	052103	042105	030440	
9474	056636	020360	040502	020104	
9475	056644	042523	052103	051117	
9476	056652	027123	027056	041101	
9477	056660	051117	044524	043516	
9478	056666	052040	051505	000124	
9479	056674	042504	042524	052103	EM26: .ASCIZ /DETECTED BSE BUT NOT LISTED IN BAD SECTOR FILE/
9480	056702	042105	041040	042523	
9481	056710	041040	052123	047040	
9482	056716	052117	046040	051511	
9483	056724	042524	020104	047111	
9484	056732	041040	042101	051440	
9485	056740	041505	047524	020122	
9486	056746	044506	042514	000	
9487	056753	104	052105	041505	EM27: .ASCII /DETECTED BSE IN READ COMMAND/
9488	056760	042524	020104	051502	
9489	056766	020105	047111	051040	
9490	056774	040505	020104	047503	
9491	057002	046515	047101	104	
9492	057007	015	041012	052125	.ASCIZ <CR><LF>/BUT NOT IN PREVIOUS WRITE COMMAND TO SAME SECTOR/
9493	057014	047040	052117	044440	
9494	057022	020116	051120	053105	
9495	057030	047511	051525	053440	
9496	057036	044522	042524	041440	
9497	057044	046517	040515	042116	
9498	057052	052040	020117	040523	
9499	057060	042515	051440	041505	
9500	057066	047524	000122		
9501	057072	054503	020114	042101	EM36: .ASCIZ /CYL ADDR IN RKMR3 NOT SAME AS RKDC/
9502	057100	051104	044440	020116	
9503	057106	045522	051115	020063	
9504	057114	047516	020124	040523	
9505	057122	042515	040440	020123	
9506	057130	045522	041504	000	
9507	057135	103	046131	042040	EM39: .ASCIZ /CYL DIFF & OFFSET IN RKMR2 NOT CLEARED/
9508	057142	043111	020106	020046	
9509	057150	043117	051506	052105	
9510	057156	044440	020116	045522	
9511	057164	051115	020062	047516	
9512	057172	020124	046103	040505	
9513	057200	042522	000104		
9514	057204	054503	020114	042101	EM40: .ASCIZ /CYL ADDR IN RKMR3 NOT CLEARED/
9515	057212	051104	044440	020116	
9516	057220	045522	051115	020063	
9517	057226	047516	020124	046103	
9518	057234	040505	042522	000104	
9519	057242	054503	020114	042101	EM41: .ASCIZ /CYL ADDR IN B2 DID NOT REMAIN CLEARED/
9520	057250	051104	044440	020116	
9521	057256	031102	042040	042111	
9522	057264	047040	052117	051040	
9523	057272	046505	044501	020116	
9524	057300	046103	040505	042522	
9525	057306	000104			
9526	057310	052101	047124	047040	EM55: .ASCIZ /ATTN NOT CLEARED IN RKASOF/

9527	057316	052117	041440	042514		
9528	057324	051101	042105	044440		
9529	057332	020116	045522	051501		
9530	057340	043117	000			
9531	057343	110	040505	051504	EM60:	.ASCIZ HEADS HOME NOT FOUND IN RKMR2/
9532	057350	044040	046517	020105		
9533	057356	047516	020124	047506		
9534	057364	047125	020104	047111		
9535	057372	051040	046513	031122		
9536	057400	000				
9537	057401	104	042524	051440	EM62:	.ASCIZ /DTE SET IN RKER/
9538	057406	052105	044440	020116		
9539	057414	045522	051105	000		
9540	057421	104	052114	051440	EM63:	.ASCIZ /DLT SET IN RKCS2/
9541	057426	052105	044440	020116		
9542	057434	045522	051503	000062		
9543	057442	042522	042101	044040	EM65:	.ASCIZ /READ HEADER ERROR/
9544	057450	040505	042504	020122		
9545	057456	051105	047522	000122		
9546	057464	054503	020114	042101	EM66:	.ASCIZ /CYL ADDR IN RKMR3 INCORRECT/
9547	057472	051104	044440	020116		
9548	057500	045522	051115	020063		
9549	057506	047111	047503	051122		
9550	057514	041505	000124			
9551	057520	046101	04351	046516	EM69:	.ASCIZ /ALIGNMENT CARTRIDGE USED/
9552	057526	047105	020124	040503		
9553	057534	052122	044522	043504		
9554	057542	020105	051525	042105		
9555	057550	000				
9556	057551	103	047524	051440	EM73:	.ASCIZ /CTO SET IN RKCS1/
9557	057556	052105	044440	020116		
9558	057564	045522	051503	000061		
9559	057572	052122	020132	047516	EM74:	.ASCIZ /RTZ NOT SET IN RKMR2/
9560	057600	020124	042523	020124		
9561	057606	047111	051040	046513		
9562	057614	031122	000			
9563	057617	116	042105	051440	EM79:	.ASCIZ /NED SET IN RKCS2/
9564	057624	052105	044440	020116		
9565	057632	045522	051503	000062		
9566	057640	051127	052111	020105	EM80:	.ASCIZ /WRITE CHECK ERROR SET IN RKCS2/
9567	057646	044103	041505	020113		
9568	057654	051105	047522	020122		
9569	057662	042523	020124	047111		
9570	057670	051040	041513	031123		
9571	057676	000				
9572	057677	127	044522	042524	EM81:	.ASCIZ /WRITE CHECK COMMAND NOT FUNCTIONING/
9573	057704	041440	042510	045503		
9574	057712	041440	046517	040515		
9575	057720	042116	047040	052117		
9576	057726	043040	047125	052103		
9577	057734	047511	044516	043516		
9578	057742	000				
9579	057743	122	040505	020104	EM82:	.ASCIZ /READ DATA DID NOT COMPARE WITH WRITE DATA
9580	057750	040504	040524	042040		
9581	057756	042111	047040	052117		
9582	057764	041440	046517	040520		

9583	057772	042522	053440	052111	
9584	060000	020110	051127	052111	
9585	060006	020105	040504	040524	
9586	060014	000			
9587	060015	104	052101	020101	EM83: .ASCIZ /DATA CHECK ERROR SET IN RKER/
9588	060022	044103	041505	020113	
9589	060030	051105	047522	020122	
9590	060036	042523	020124	047111	
9591	060044	051040	042513	000122	
9592	060052	044127	046111	020105	EM84: .ASCIZ /WHILE WAITING FOR CONTR READY OR AFTER CONTR READY REC'D/
9593	060060	040527	052111	047111	
9594	060066	020107	047506	020122	
9595	060074	047503	052116	020122	
9596	060102	042522	042101	020131	
9597	060110	051117	040440	052106	
9598	060116	051105	041440	047117	
9599	060124	051124	051040	040505	
9600	060132	054504	051040	041505	
9601	060140	042047	000		
9602	060143	117	043106	042523	EM85: .ASCIZ /OFFSET STATUS BIT IN RKMR2 CLEARED/
9603	060150	020124	052123	052101	
9604	060156	051525	041040	052111	
9605	060164	044440	020116	045522	
9606	060172	051115	020062	046103	
9607	060200	040505	042522	000104	
9608	060206	043117	051506	052105	EM86: .ASCIZ /OFFSET REG IN A2 NOT = RKASOF/
9609	060214	051040	043505	044440	
9610	060222	020116	031101	047040	
9611	060230	052117	036440	051040	
9612	060236	040513	047523	000106	
9613	060244	044504	020104	047516	EM88: .ASCIZ /DID NOT FIND SECTOR 0 FROM INDEX/
9614	060252	020124	044506	042116	
9615	060260	051440	041505	047524	
9616	060266	020122	020060	051106	
9617	060274	046517	044440	042116	
9618	060302	054105	000		
9619	060305	110	040505	051504	EM89: .ASCIZ /HEADS HOME NOT CLEARED IN RKMR2/
9620	060312	044040	046517	020105	
9621	060320	047516	020124	046103	
9622	060326	040505	042522	020104	
9623	060334	047111	051040	046513	
9624	060342	031122	000		
9625	060345	123	051105	047526	EM90: .ASCIZ /SERVO SIG PRES NOT SET IN RKMR2/
9626	060352	051440	043511	050040	
9627	060360	042522	020123	047516	
9628	060366	020124	042523	020124	
9629	060374	047111	051040	046513	
9630	060402	031122	000		
9631	060405	122	053105	047040	EM91: .ASCIZ /REV NOT SET IN RKMR2/
9632	060412	052117	051440	052105	
9633	060420	044440	020116	045522	
9634	060426	051115	000062		
9635	060432	042522	020126	047516	EM92: .ASCIZ /REV NOT CLEARED IN RKMR2/
9636	060440	020124	046103	040505	
9637	060446	042522	020104	047111	
9638	060454	051040	046513	031122	

9639	060462	000				
9640	060463	122	040505	044504	EM93:	.ASCIZ /READING WRONG CYLINDER # IN HEADER...MISPOSITION/
9641	060470	043516	053440	047522		
9642	060476	043516	041440	046131		
9643	060504	047111	042504	020122		
9644	060512	020043	047111	044040		
9645	060520	040505	042504	027122		
9646	060526	027056	044515	050123		
9647	060534	051517	052111	047511		
9648	060542	000116				
9649	060544	043117	051506	052105	EM94:	.ASCIZ /OFFSET IT IN A2 NOT CLEARED/
9650	060552	044440	020124	047111		
9651	060560	040440	020062	047516		
9652	060566	020124	046103	040505		
9653	060574	042522	000104			
9654	060600	047506	046522	052101	EM95:	.ASCIZ /FORMAT BIT NOT SET IN RKMR2/
9655	060606	041040	052111	047040		
9656	060614	052117	051440	052105		
9657	060622	044440	020116	045522		
9658	060630	051115	000062			
9659	060634	040503	047116	052117	EM96:	.ASCIZ /CANNOT FIND SECTOR 23(8)/
9660	060642	043040	047111	020104		
9661	060650	042523	052103	051117		
9662	060656	031040	024063	024470		
9663	060664	000				
9664	060665	110	040505	020104	EM97:	.ASCIZ /HEAD SWITCHING REQ'D ANOTHER FULL REVOLUTION OF DISK/
9665	060672	053523	052111	044103		
9666	060700	047111	020107	042522		
9667	060706	023521	020104	047101		
9668	060714	052117	042510	020122		
9669	060722	052506	046114	051040		
9670	060730	053105	046117	052125		
9671	060736	047511	020116	043117		
9672	060744	042040	051511	000113		
9673	060752	040503	047116	052117	EM98:	.ASCIZ /CANNOT FIND CYLINDER 128/
9674	060760	043040	047111	020104		
9675	060766	054503	044514	042116		
9676	060774	051105	030440	034062		
9677	061002	000				
9678	061003	103	047101	047516	EM99:	.ASCIZ /CANNOT FIND CYLINDER 256/
9679	061010	020124	044506	042116		
9680	061016	041440	046131	047111		
9681	061024	042504	020122	032462		
9682	061032	000066				
9683	061034	051104	053111	020105	EM100:	.ASCIZ /DRIVE OFF TRACK SET IN RKMR3/
9684	061042	043117	020106	051124		
9685	061050	041501	020113	042523		
9686	061056	020124	047111	051040		
9687	061064	046513	031522	000		
9688	061071	104	042111	047040	EM101:	.ASCIZ /DID NOT GO TO CYLINDER 10/
9689	061076	052117	043440	020117		
9690	061104	047524	041440	046131		
9691	061112	047111	042504	020122		
9692	061120	030061	000			
9693						
9694						.SBTTL DATA HEADERS

H15

JNIBUS RK06 DRIVE DIAGNOSTIC PART 2
DZR6IC.P11 07-OCT-76 13:50

MACY11 27(1006) 07-OCT-76 14:14 PAGE 189
DATA HEADERS

SEQ 0199

9751	061601	122	046513	031122	DH12:	.ASCIZ	/RKMR2 RKMR3 RKCS1 RKCS2 RKDC RKDA/
9752	061606	051011	046513	031522			
9753	061614	051011	041513	030523			
9754	061622	051011	041513	031123			
9755	061630	051011	042113	004503			
9756	061636	045522	040504	000			
9757	061643	117	020116	042523	DH13:	.ASCIZ	/ON SECTORS 10, 12, 14, 16, 18 OR 20 CYL 410 TRACK 2/
9758	061650	052103	051117	020123			
9759	061656	030061	020054	031061			
9760	061664	020054	032061	020054			
9761	061672	033061	020054	034061			
9762	061700	047440	020122	030062			
9763	061706	041440	046131	032040			
9764	061714	030061	052040	040522			
9765	061722	045503	031040	000			
9766	061727	117	020116	042523	DH14:	.ASCIZ	/ON SECTORS 11, 13, 15, 17, 19 OR 21 CYL 410 TRACK 2/
9767	061734	052103	051117	020123			
9768	061742	030461	020054	031461			
9769	061750	020054	032461	020054			
9770	061756	033461	020054	034461			
9771	061764	047440	020122	030462			
9772	061772	041440	046131	032040			
9773	062000	030061	052040	040522			
9774	062006	045503	031040	000			
9775	062013	101	052106	051105	DH17:	.ASCIZ	/AFTER RECAL COMMAND/
9776	062020	051040	041505	046101			
9777	062026	041440	046517	040515			
9778	062034	042116	000				
9779	062037	101	052106	051105	DH18:	.ASCIZ	/AFTER UNLOAD COMMAND/
9780	062044	052440	046116	040517			
9781	062052	020104	047503	046515			
9782	062060	047101	000104				
9783	062064	043101	042524	020122	DH19:	.ASCIZ	/AFTER PACK COMMAND/
9784	062072	040520	045503	041440			
9785	062100	046517	040515	042116			
9786	062106	000					
9787	062107	101	052106	051105	DH20:	.ASCIZ	/AFTER SELECT DRIVE COMMAND/
9788	062114	051440	046105	041505			
9789	062122	020124	051104	053111			
9790	062130	020105	047503	046515			
9791	062136	047101	000104				
9792	062142	043101	042524	020122	DH21:	.ASCIZ	/AFTER SUBSYSTEM CLEAR/
9793	062150	052523	051502	051531			
9794	062156	042524	020115	046103			
9795	062164	040505	000122				
9796	062170	043101	042524	020122	DH22:	.ASCIZ	/AFTER DRIVE CLEAR COMMAND/
9797	062176	051104	053111	020105			
9798	062204	046103	040505	020122			
9799	062212	047503	046515	047101			
9800	062220	000104					
9801	062222	043101	042524	020122	DH24:	.ASCIZ	/AFTER OFFSET COMMAND/
9802	062230	043117	051506	052105			
9803	062236	041440	046517	040515			
9804	062244	042116	000				
9805	062247	101	052106	051105	DH25:	.ASCIZ	/AFTER SEEK COMMAND/
9806	062254	051440	042505	020113			

9863	062736	051503	020061	020040					
9864	062744	045522	051503	020062					
9865	062752	043040	047522	020115					
9866	062760	042523	052103	020040					
9867	062766	047524	051440	041505					
9868	062774	000124							
9869	062776	043101	042524	020122	DH39:	.ASCIZ	/AFTER WRITE HEADER COMMAND/		
9870	063004	051127	052111	020105					
9871	063012	042510	042101	051105					
9872	063020	041440	046517	040515					
9873	063026	042116	000						
9874	063031	122	046513	031122	DH40:	.ASCIZ	/RKMR2 RKMR3 RKDA WORD# HEADER WAS SHOULD BE/		
9875	063036	051011	046513	031522					
9876	063044	051011	042113	004501					
9877	063052	047527	042122	004443					
9878	063060	042510	042101	051105					
9879	063066	053440	051501	020040					
9880	063074	044123	052517	042114					
9881	063102	041040	000105						
9882	063106	052504	044522	043516	DH41:	.ASCIZ	/DURING RECAL COMMAND/		
9883	063114	051040	041505	046101					
9884	063122	041440	046517	040515					
9885	063130	042116	000						
9886	063133	117	020116	042523	DH42:	.ASCIZ	/ON SECTORS 0,2,4,6 OR 8 CYL 410 TRACK 2/		
9887	063140	052103	051117	020123					
9888	063146	026060	026062	026064					
9889	063154	020066	051117	034040					
9890	063162	020040	054503	020114					
9891	063170	030464	020060	051124					
9892	063176	041501	020113	000062					
9893	063204	047117	051440	041505	DH43:	.ASCIZ	/ON SECTORS 1,3,5,7 OR 9 CYL 410 TRACK 2/		
9894	063212	047524	051522	030440					
9895	063220	031454	032454	033454					
9896	063226	047440	020122	020071					
9897	063234	041440	046131	032040					
9898	063242	030061	052040	040522					
9899	063250	045503	031040	000					
9900	063255	106	051117	040515	DH44:	.ASCIZ	/FORMAT & ALL READ-WRITE TESTS WILL BE BYPASSED/		
9901	063262	020124	020046	046101					
9902	063270	020114	042522	042101					
9903	063276	053455	044522	042524					
9904	063304	052040	051505	051524					
9905	063312	053440	046111	020114					
9906	063320	042502	041040	050131					
9907	063326	051501	042523	000104					
9908	063334	043101	042524	020122	DH47:	.ASCIZ	/AFTER READ HEADER COMMAND WITH MOVEMENT/		
9909	063342	042522	042101	044040					
9910	063350	040505	042504	020122					
9911	063356	047503	046515	047101					
9912	063364	020104	044527	044124					
9913	063372	046440	053117	046505					
9914	063400	047105	000124						
9915	063404	051515	020107	020101	DH49:	.ASCIZ	/MSG A & B IN RKMR2 & RKMR3 RESP. ARE INVALID/		
9916	063412	020046	020102	047111					
9917	063420	051040	046513	031122					
9918	063426	023040	051040	046513					

9919	063434	031522	051040	051505	
9920	063442	027120	040440	042522	
9921	063450	044440	053116	046101	
9922	063456	042111	000		
9923	063461	101	052106	051105	DH51: .ASCIZ /AFTER SEEK TO SELF COMMAND/
9924	063466	051440	042505	020113	
9925	063474	047524	051440	046105	
9926	063502	020106	047503	046515	
9927	063510	047101	000104		
9928	063514	044527	044124	044440	DH52: .ASCIZ /WITH INTENTIONAL MISCOMPARE/
9929	063522	052116	047105	044524	
9930	063530	047117	046101	046440	
9931	063536	051511	047503	050115	
9932	063544	051101	000105		
9933	063550	052504	044522	043516	DH53: .ASCIZ /DURING OFFSET COMMAND/
9934	063556	047440	043106	042523	
9935	063564	020124	047503	046515	
9936	063572	047101	000104		
9937	063576	043101	042524	020122	DH54: .ASCIZ /AFTER FORMAT CHANGE AND CONTR READY REC'D/
9938	063604	047506	046522	052101	
9939	063612	041440	040510	043516	
9940	063620	020105	047101	020104	
9941	063626	047503	052116	020122	
9942	063634	042522	042101	020131	
9943	063642	042522	023503	000104	
9944	063650	045522	051115	004462	DH56: .ASCIZ /RKMR2 RKMR3 RKDC CYL # HEADER WORD 0/
9945	063656	045522	051115	004463	
9946	063664	045522	041504	041411	
9947	063672	046131	021440	044011	
9948	063700	040505	042504	020122	
9949	063706	047527	042122	030040	
9950	063714	000			
9951	063715	101	052106	051105	DH57: .ASCIZ /AFTER WRITE COMMAND WITH OFFSET/
9952	063722	053440	044522	042524	
9953	063730	041440	046517	040515	
9954	063736	042116	053440	052111	
9955	063744	020110	043117	051506	
9956	063752	052105	000		
9957	063755	104	052101	020101	DH58: .ASCIZ /DATA WAS SHOULD BE/
9958	063762	040527	020123	051411	
9959	063770	047510	046125	020104	
9960	063776	042502	000		
9961					
9962					.SBTTL ERROR OUTPUT DATA
9963					
9964		064002			.EVEN
9965	064002	001214	001116	007416	DT1: \$TESTN,\$ERRPC,HMR2,HMR3,HER,HDS,HCS1,HCS2,HASOF
9966	064010	007420	007404	007402	
9967	064016	007370	007372	007406	
9968	064024	001214	001334		DT3: \$TESTN,TRAPP
9969	064030	001214	001116	007416	DT4: \$TESTN,\$ERRPC,HMR2,HMR3,HDC,FRCYL,TOCYL,CALDIF
9970	064036	007420	007410	001350	
9971	064044	001352	001360		
9972	064050	001214	001116	007416	DT5: \$TESTN,\$ERRPC,HMR2,HMR3,HER,HDS,HDA,HCS1,HCS2
9973	064056	007420	007404	007402	
9974	064064	007400	007370	007372	

9975	064072	001214	001116	007416	DT6:	STESTN, SERRPC, HMR2, HMR3, HCS1, HCS2, WD2, WD1
9976	064100	007420	007370	007372		
9977	064106	001470	001466			
9978	064112	001214	001116	007416	DT7:	STESTN, SERRPC, HMR2, HMR3, HDA, WDCNT, HDWD, TEMP1
9979	064120	007420	007400	01506		
9980	064126	001524	007426			
9981	064132	001214	001116	007416	DT8:	STESTN, SERRPC, HMR2, HMR3, HDC, TOCYL, FRCYL, CALDIF
9982	064140	007420	007410	001352		
9983	064146	001350	001360			
9984	064152	001214	001116	007416	DT9:	STESTN, SERRPC, HMR2, HMR3, HDC, TOCYL, RHTAB
9985	064160	007420	007410	001352		
9986	064166	001742				
9987	064170	001214	001116	007416	DT10:	STESTN, SERRPC, HMR2, HMR3, HCS1, HCS2, HDC, HDA
9988	064176	007420	007370	007372		
9989	064204	007410	007400			
9990	064210	001214	001116	007460	DT13:	STESTN, SERRPC, E.A0, E.B0, E.A1, E.B1, H.A0, H.B0, H.A1, H.B1
9991	064216	007462	007464	007466		
9992	064224	007440	007442	007444		
9993	064232	007446				
9994	064234	007370	007372	007406		HCS1, HCS2, HASOF, HER, HDS, HDC
9995	064242	007404	007402	007410		
9996	064250	001214	001116	007460	DT14:	STESTN, SERRPC, E.A0, E.B0, E.A1, E.B1, E.A2, E.B2
9997	064256	007462	007464	007466		
9998	064264	007470	007472			
9999	064270	007440	007442	007444		H.A0, H.B0, H.A1, H.B1, H.A2, H.B2
10000	064276	007446	007450	007452		
10001	064304	007370	007372	007406		HCS1, HCS2, HASOF, HER, HDS, HDC
10002	064312	007404	007402	007410		
10003	064320	001214	001116	007460	DT15:	STESTN, SERRPC, E.A0, E.B0, E.A1, E.B1, E.A2, E.B2, E.B3
10004	064326	007462	007464	007466		
10005	064334	007470	007472	007476		
10006	064342	007440	007442	007444		H.A0, H.B0, H.A1, H.B1, H.A2, H.B2, H.B3
10007	064350	007446	007450	007452		
10008	064356	007456				
10009	064360	007370	007372	007406		HCS1, HCS2, HASOF, HER, HDS, HDC
10010	064366	007404	007402	007410		
10011						
10012					.SBTTL	ERROR DATA FORMATS
10013						
10014	064374	000002			DF1:	2
10015	064376	002	000			.BYTE 2,0
10016	064400	061140				DH2
10017	064402	007	000			.BYTE 7,0
10018						
10019	064404	000001			DF2:	1
10020	064406	002	000			.BYTE 2,0
10021						
10022	064410	000003			DF3:	3
10023	064412	000	000			.BYTE 0,0
10024	064414	061123				DH1
10025	064416	002	000			.BYTE 2,0
10026	064420	061534				DH11
10027	064422	006	000			.BYTE 6,0
10028						
10029	064424	000002			DF4:	2
10030	064426	002	000			.BYTE 2,0

10031	064430	061601		DH12	
10032	064432	006	000	.BYTE	6,0
10033					
10034	064434	000004		DF5:	4
10035	064436	000	000	.BYTE	0,0
10036	064440	063404		DH49	
10037	064442	000	000	.BYTE	0,0
10038	064444	061123		DH1	
10039	064446	002	000	.BYTE	2,0
10040	064450	061140		DH2	
10041	064452	007	000	.BYTE	7,0
10042					
10043	064454	000003		DF6:	3
10044	064456	000	000	.BYTE	0,0
10045	064460	061123		DH1	
10046	064462	002	000	.BYTE	2,0
10047	064464	061260		DH6	
10048	064466	006	000	.BYTE	6,0
10049					
10050					
10051	064470	000003		DF10:	3
10052	064472	000	000	.BYTE	0,0
10053	064474	061123		DH1	
10054	064476	002	000	.BYTE	2,0
10055	064500	061140		DH2	
10056	064502	007	000	.BYTE	7,0
10057					
10058	064504	000002		DF14:	2
10059	064506	002	000	.BYTE	2,0
10060	064510	063031		DH40	
10061	064512	006	000	.BYTE	6,0
10062					
10063					
10064	064514	000003		DF15:	3
10065	064516	000	000	.BYTE	0,0
10066	064520	061123		DH1	
10067	064522	002	000	.BYTE	2,0
10068	064524	061342		DH7	
10069	064526	007	000	.BYTE	7,0
10070					
10071	064530	000004		DF17:	4
10072	064532	000	000	.BYTE	0,0
10073	064534	063255		DH44	
10074	064536	000	000	.BYTE	0,0
10075	064540	061123		DH1	
10076	064542	002	000	.BYTE	2,0
10077	064544	061140		DH2	
10078	064546	007	000	.BYTE	7,0
10079	064550	000003		DF20:	3
10080	064552	000	000	.BYTE	0,0
10081	064554	061123		DH1	
10082	064556	002	000	.BYTE	2,0
10083	064560	063650		DH56	
10084	064562	005	000	.BYTE	5,0
10085					
10086	064564	000007		DF21:	7

10087	064566	000	000
10088	064570	061123	
10089	064572	002	000
10090	064574	062353	
10091	064576	000	000
10092	064600	062547	
10093	064602	004	000
10094	064604	062434	
10095	064606	000	000
10096	064610	062547	
10097	064612	004	000
10098	064614	062652	
10099	064616	006	000
10100			
10101	064620	000007	
10102	064622	000	000
10103	064624	061123	
10104	064626	002	000
10105	064630	062353	
10106	064632	000	000
10107	064634	062547	
10108	064636	006	000
10109	064640	062434	
10110	064642	000	000
10111	064644	062547	
10112	064646	006	000
10113	064650	062652	
10114	064652	006	000
10115			
10116	064654	000007	
10117	064656	000	000
10118	064660	061123	
10119	064662	002	000
10120	064664	062353	
10121	064666	000	000
10122	064670	062547	
10123	064672	007	000
10124	064674	062434	
10125	064676	000	000
10126	064700	062547	
10127	064702	007	000
10128	064704	062652	
10129	064706	006	000
10130			
10131			
10132			
10133			
10134			
10135			
10136			
10137			
10138			
10139			
10140			
10141	064710	104413	
10142	064712	113700	001114

.BYTE	0,0
DH1	
.BYTE	2,0
DH28	
.BYTE	0,0
DH31	
.BYTE	4,0
DH29	
.BYTE	0,0
DH31	
.BYTE	4,0
DH34	
.BYTE	6,0
DF22:	7
.BYTE	0,0
DH1	
.BYTE	2,0
DH28	
.BYTE	0,0
DH31	
.BYTE	6,0
DH29	
.BYTE	0,0
DH31	
.BYTE	6,0
DH34	
.BYTE	6,0
DF23:	7
.BYTE	0,0
DH1	
.BYTE	2,0
DH28	
.BYTE	0,0
DH31	
.BYTE	7,0
DH29	
.BYTE	0,0
DH31	
.BYTE	7,0
DH34	
.BYTE	6,0

```

.EVEN
;*****
;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
MOV B $ITEMB,RO ;ENTER ERROR NUMBER

```

10143	064716	042700	177400	BIC	#177400,R0	:CLEAR SIGN EXTENSION	
10144	064722	005300		DEC	R0	:FORM INDEX FOR ERROR TABLE	
10145	064724	006300		ASL	R0		
10146	064726	006300		ASL	R0		
10147	064730	006300		ASL	R0		
10148	064732	062700	007542	18:	ROD	#ERRTB,R0	:FORM ADDRESS OF ERROR ENTRY
10149	064736	012037	064752	MOV	(R0)+,2\$:GET EM POINTER	
10150	064742	001404		BEG	3\$:BRANCH IF THERE ISN'T ONE	
10151	064744	104401	001205	TYPE	.\$CRLF	:TYPE CARRIAGE RETURN LINE FEED	
10152	064750	104401		TYPE		:TYPE ERROR MESSAGE (EM)	
10153	064752	000000		28:	.WORD	0	:EM POINTER GOES HERE
10154	064754	012037	064770	38:	MOV	R0)+,4\$:GET DH POINTER
10155	064760	001404		BEG	5\$:BRANCH IF THERE ISN'T ONE	
10156	064762	104401	001205	TYPE	.\$CRLF	:TYPE CR-LF	
10157	064766	104401		TYPE		:TYPE DATA HEADER	
10158	064770	000000		48:	.WORD	0	:DH POINTER GOES HERE
10159	064772	012001		58:	MOV	(R0)+,R1	:GET DT POINTER
10160	064774	001455		BEG	20\$:BRANCH IF THERE ARE NONE	
10161	064776	005304		CLR	R4	:SET INDENT SWITCH	
10162	065000	012000		MOV	(R0)+,R0	:GET DF POINTER	
10163	065002	012032		MOV	(R0)+,R2	:STORE NUMBER OF DH'S	
10164	065004	001446		BEG	17\$:DH NUM IS 0-BRANCH	
10165	065006	005104		COM	R4	:NO INDENT	
10166	065010	104401	001205	TYPE	.\$CRLF		
10167	065014	112003		108:	MOV	(R0)+,R3	:GET & STORE NUMBER OF DATA WORDS
10168	065016	105720		TSTB	(R0)+	:BUMP PAST FORMAT WORD	
10169	065020	005703		TST	R3	:TEST IF ANY DATA FOR THIS HEADER	
10170	065022	001407		BEG	14\$:NO - SKIP DATA PRINT	
10171	065024	013146		118:	MOV	2(R1)+,-(SP)	:PUT FIRST DATA WORD ON STACK
10172	065026	104402		TYPOC		:TYPE IT	
10173	065030	005303		DEC	R3	:MORE DATA WORDS	
10174	065032	001403		BEG	14\$:NO-BRANCH	
10175	065034	104401	065164	TYPE	.\$SPACE2	:TYPE SEPARATORS	
10176	065040	000771		BR	11\$:LOOP	
10177	065042	005302		148:	DEC	R2	:MORE DH'S?
10178	065044	003431		BLE	20\$:NO-BRANCH	
10179	065046	104401	001205	TYPE	.\$CRLF		
10180	065052	005760	000002	TST	2(R0)	:ONLY A DH IN THIS REQUEST?	
10181	065056	001404		BEG	15\$:YES-BRANCH BYPASS INDENT	
10182	065060	005104		COM	R4	:INDENT?	
10183	065062	001002		BNE	15\$:NO-BRANCH	
10184	065064	104401	065164	TYPE	.\$SPACE2	:YES-TYPE SPACES	
10185	065070	012037	065076	158:	MOV	(R0)+,16\$:GET NEXT DH POINTER
10186	065074	104401		TYPE		:TYPE DH	
10187	065076	000000		168:	.WORD	0	:DH POINTER GOES HERE
10188	065100	105710		TSTB	(R0)	:TYPE A DT?	
10189	065102	001003		BNE	21\$:YES-BRANCH	
10190	065104	062700	000002	ADD	#2,R0	:INCREMENT DF POINTER	
10191	065110	000754		BR	14\$:SEE IF END OF DF BLOCK	
10192	065112	104401	001205	218:	TYPE	.\$CRLF	
10193	065116	005704		TST	R4	:INDENT?	
10194	065120	001335		BNE	10\$:NO-BRANCH	
10195	065122	104401	065164	178:	TYPE	.\$SPACE2	:YES-TYPE SPACES
10196	065126	000732		BR	10\$:LOOP	
10197	065130	104414		208:	RESREG		

```

10199 065132 032777 010000 114000
10200 065140 001410
10201 065142 023727 001103 000024
10202 065150 001004
10203 065152 012706 001100
10204 065156 000137 036034
10205
10206 065162 000207
10207 065164 020040 000
10208
10209
10210
10211
10212
10213
10214
10215
10216 065170
10217 065250
10218 000000
10219 000001
10220 000002
10221 000003
10222 000004
10223 000005
10224 000006
10225 000007
10226 177776
10227
10228 000014
10229 000340
10230 000020
10231 000003
10232 000006
10233
10234
10235
10236
10237
10238

```

```

BIT #SW12,DSWR :SEE IF ABORT DRV AFTER 20 ERRORS
BEQ 255 :BR IF NO
CMP SERFLG,#20. :ELSE SEE IF HAVE 20 ERRORS
BNE 255 :BR IF NC
MOV #STACK,SP :ELSE RESTORE STACK PTR
JMP SEOP :AND GO TO NEXT DRV

```

```

255: RTS PC
SPACE2: .ASCIZ/ / :2 SPACES
: OCT-11 -- VOOSA

```

: 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

```

```

: R5 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).

```

```

10239      177562
10240      177560
10241      177566
10242      177564
10243
10244
10245
10246
10247
10248
10249
10250      065250 000413
10251      065252 000417
10252      065254 013737 177776 065230
10253      065262 013737 000016 177776
Z 10254      065270 010737 065226
10255      065274 000137 066426
10256
10257      065300 012706 065210
10258      065304 010637 065224
10259      065310 000414
10260      065312 004037 066634
10261      065316 013777 065246 177716
10262      065324 113704 065232
10263      065330 106004
10264      065332 106004
10265      065334 106004
10266      065336 110437 177776
Z 10267      065342 000127
10268      065344 000403
10269      065346 012737 000002 066336
10270      065354 105037 067255
10271      065360 012737 000340 000016
10272      065366 012737 066416 000014
10273      065374 000417
10274
10275
10276
10277
10278      065376 004537 067056
10279      065402 012704 067301
10280      065406 120024
10281      065410 001413
10282      065412 022704 067307
10283      065416 101373
10284      065420 042700 177770
10285      065424 010004
10286      065426 006304
10287      065430 062704 065210
10288      065434 005202
10289      065436 006444
10290      065440 162704 067272
10291      065444 000770
10292
10293
10294

```

```

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.STAT :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.STAT: MOV #O.URD,SP :SET UP STACK
: MOV SP,O.USP :FAKE THE SAVED STACK
: BR O.RST1 :CLEAR BREAKPOINT TABLES
O.RST: JSR O.O.SVR :SAVE REGISTERS
: MOV O.UIN,O.ADR1 :REMOVE THE BREAKPOINT
: MOV O.PRI,R4 :GET ODT PRIORITY
: RORB R4 :SHIFT
: RORB R4 :INTO
: RORB R4 :POSITION
: MOV R4,ST :STORE IN STATUS
O.RST1: JMP (PC)+
: BR O.45
: MOV #RTI,O.RTI1 :SET TO RTI IF 11/20 OR /05
O.45: CLRB O.P :DISALLOW PROCEED
: MOV #O.STM,O.TVEC+2 :STATUS WORD TO TRT VECTOR + 2
: MOV #O.BRK,O.TVEC :PC TO TRT VECTOR
: BR O.RALL :CLEAR BREAKPOINT TABLES
:
: SPECIAL NAME HANDLER
: DEPENDS UPON THE EXPLICIT ORDER OF THE TWO TABLES O.TL AND O.URD
:
O.REGT: JSR S,O.GET :SPECIAL NAME, GET ONE MORE CHARACTER
: MOV #O.TL,R4 :TABLE START ADDRESS
O.RSP: CMPB RD,(R4)+ :IS THIS THE CORRECT CHARACTER?
: BEQ O.SP :JUMP IF YES
: CMP #O.TL+O.LG,R4 :IS THE SEARCH DONE?
: BHI O.RSP :BRANCH IF NOT
: BIC #177770,RD :MASK OFF OCTAL
O.SP1: MOV RD,R4
: ASL R4
: ADD #O.URD,R4 :GENERATE ADDRESS
: INC R2 :SET FOUND FLAG
O.SP: BR O.SCAN :GO FIND NEXT CHARACTER
: SUB #O.TL-7,R4 :CORRECT CONSTANT
: BR O.SP1
:
: * HANDLER - OPEN INDEXED ON THE PC
:

```

10295	065446	004737	067202	0.0RPC:	JSR	PC,0.TCLS		
10296	065452	010502			MOV	R5,R2	:CURRENT ADDRESS IN R2	
10297	065454	061202			ADD	2R2,R2	:COMPUTE	
10298	065456	006202			ASR	R2	:MOVE ONE BIT TO CARRY	
10299	065460	103421			BCS	0.ERR	:ERROR IF ODD NUMBER	
10300	065462	006302			ASL	R2	:RESTORE WORD	
10301	065464	005722			TST	(R2)+	:AND INCREMENT BY TWO	
10302	065466	010205			MOV	R2,R5	:UPDATE CAD	
10303	065470	000137	065742		JMP	0.0P2	:GO FINISH UP	
10304				:				
10305				:				
10306				:				
10307	065474	005702		0.BKPT:	TST	R2	:IF NO NUMBER TYPED	
10308	065476	001406			BEQ	0.RALL	:REMOVE BREAKPOINT	
10309	065500	006204			ASR	R4	:CHECK IF ODD	
10310	065502	103410			BCS	0.ERR	:JUMP IF ODD	
10311	065504	006304			ASL	R4	:RESTORE ONE BIT	
10312	065506	010437	065242		MOV	R4,0.ADR1	:SET A BREAKPOINT	
10313	065512	000412			BR	0.DCD		
10314	065514	012737	067316	065242	0.RALL:	MOV	#0,TRTC,0.ADR1	:CLEAR BREAKPOINT
10315	065522	000406			BR	0.DCD		
10316				:				
10317				:				
10318				:				
10319				:				
10320				:				
10321				:				
10322	065524	052705	000001	0.ERR:	BIS	#1,R5	:CLOSE EVERYTHING	
10323	065530	012700	000077		MOV	#',R0	:? TO BE TYPED	
10324	065534	004537	067134		JSR	5,0.FTYP	:OUTPUT ?	
10325	065540	004537	067234	0.DCD:	JSR	5,0.CRLS	:TYPE <CR><LF>*	
10326	065544	005004		0.DCD1:	CLR	R4	:R4 CONTAINS THE CONVERTED OCTAL	
10327	065546	005002			CLR	R2	:R2 IS THE NUMBER FOUND FLAG	
10328	065550	004537	067056	0.SCAN:	JSR	5,0.GET	:GET A CHAR, RETURN IN R0	
10329	065554	022700	070060		CMP	#0,R0	:COMPARE WITH ASCII 0	
10330	065560	101013			BHI	0.CLGL	:CHECK LEGALITY IF NON-NUMERIC	
10331	065562	022700	000067		CMP	#7,R0	:COMPARE WITH ASCII 7	
10332	065566	103410			BLO	0.CLGL	:CHECK LEGALITY IF NOT OCTAL	
10333	065570	042700	177770		BIC	#177770,R0	:CONVERT TO BCD	
10334	065574	006304			ASL	R4	:MAKE ROOM	
10335	065576	006304			ASL	R4	:IN	
10336	065600	006304			ASL	R4	:R4	
10337	065602	060004			ADD	R0,R4	:PACK THREE BITS IN R4	
10338	065604	005202			INC	R2	:R2 HAS NUMERIC FLAG	
10339	065606	000760			BR	0.SCAN	:AND TRY AGAIN	
10340	065610	005001		0.CLGL:	CLR	R1	:CLEAR INDEX	
10341	065612	120061	067265	0.LGL1:	CMPB	R0,0.LGCH(R1)	:DO THE CODES MATCH?	
10342	065616	001405			BEQ	0.LGL2	:JUMP IF YES	
10343	065620	005201			INC	R1	:SET INDEX FOR NEXT SEARCH	
10344	065622	020127	000014		CMP	R1,#0.CLGT	:IS THE SEARCH DONE?	
10345	065626	103336			BHIS	0.ERR	:OOPS!	
10346	065630	000770			BR	0.LGL1	:RE-LOOP	
10347	065632	006301		0.LGL2:	ASL	R1	:MULTIPLY BY TWO	
10348	065634	000171	065640		JMP	20.LGDR(R1)	:GO TO PROPER ROUTINE	
10349				:				
10350	065640	065670		0.LGDR:	0.WRD	;	OPEN WORD	

F16

UNIBUS RK06 DRIVE DIAGNOSTIC PART 2
DZR6IC.P11 07-OCT-76 13:50

MACY11 27(1006) 07-OCT-76 14:14 PAGE 200
TYPE ERROR ROUTINE

SEQ 0200

10351 065642 065722
10352 065644 065376
10353 065646 066232
10354 065650 065734
10355 065652 065446
10356 065654 065766

O.CRET : CARRIAGE RETURN CLOSE
O.REGT : S REGISTER OPS
O.GO : G GO TO ADDRESS K
O.OP1 : (LF) MODIFY CLOSE OPEN NEXT
O.ORPC : + OPEN RELATED INDEX - PC
O.BACK : † OPEN PREVIOUS

10357	065656	065776		0.OFST	:	0	OFFSET	
10358	065660	066054		0.WSCH	:	W	SEARCH WORD	
10359	065662	066050		0.EFF	:	E	SEARCH EFFECTIVE ADDRESS	
10360	065664	065474		0.BKPT	:	B	BREAKPOINTS	
10361	065666	066340		0.PROC	:	P	PROCEED	
10362		000030		0.LGL	=	-0.LGDR		;LGL MUST EQUAL 2X CHLGT ALWAYS
10363								
10364								
10365								
10366	065670	005702		0.WRD:	TST	R2		;GET VALUE IF R2 IS NON-ZERO
10367	065672	00141C			BEQ	0.WRDA		;SKIP OTHERWISE
10368	065674	010405			MOV	R4,R5		;PUT VALUE IN CAD
10369	065676	006205		0.WRD1:	ASR	R5		;MOVE ONE BIT TO CARRY
10370	065700	103711		0.ERR2:	BCS	0.ERR		;JUMP IF ODD ADDRESS
10371	065702	006305			ASL	R5		;RESTORE THE CARRY BIT
10372	065704	011500			MOV	R5,R0		;GET CONTENTS OF WORD
10373	065706	004537	066772		JSR	S,0.CAL /		;GO GET AND TYPE OUT @CAD
10374	065712	000714			BR	0.DCD1		;GO BACK TO DECODER
10375	065714	042705	000001	0.WRDA:	BIC	#1,R5		;CLEAR CLOSED BIT
10376	065720	000766			BR	0.WRD1		;GO BACK TO MAIN-LINE
10377								
10378								
10379								
10380	065722	004737	067202	0.CRET:	JSR	PC,0.TCLS		;CLOSE LOCATION
10381	065726	052705	000001		BIS	#1,R5		;CLOSE EVERYTHING
10382	065732	000702			BR	0.DCD		;RETURN TO DECODER
10383								
10384								
10385								
10386	065734	004737	067202	0.OP1:	JSR	PC,0.TCLS		;CLOSE PRESENT CELL
10387	065740	005725			TST	(R5)+		;GENERATE NEW ADDRESS
10388	065742	004537	067226	0.OP2:	JSR	S,0.CRLF		; <CR><LF>
10389	065746	010500			MOV	R5,R0		;NUMBER TO TYPE
10390	065750	004537	066772		JSR	S,0.CADV		;TYPE OUT ADDRESS
10391	065754	012700	000057		MOV	# /,R0		;TYPE A /
10392	065760	004537	067134		JSR	S,0.FTYP		
10393	065764	000744			BR	0.WRD1		;GO PROCESS IT
10394								
10395								
10396								
10397	065766	004737	067202	0.BACK:	JSR	PC,0.TCLS		;GENERATE NEW ADDRESS
10398	065772	005745			TST	-(R5)		;GO DO THE REST
10399	065774	000762			BR	0.OP2		
10400								
10401								
10402								
10403	065776	006205		0.OFST:	ASR	R5		;GET LOW ORDER BIT
10404	066000	103737			BCS	0.ERR2		;ERROR IF CLOSED
10405	066002	006305			ASL	R5		;RESTORE WORD
10406	066004	012700	000040		MOV	#,R0		;TYPE ONE BLANK
10407	066010	004537	067134		JSR	S,0.FTYP		;AS A SEPARATOR
10408	066014	160504			SUB	R5,R4		;COMPUTE
10409	066016	005304			DEC	R4		
10410	066020	005304			DEC	R4		;16 BIT OFFSET
10411	066022	010400			MOV	R4,R0		;TYPE A
10412	066024	010402			MOV	R4,R2		;SAVE R4

H16

JNIBUS RK06 DRIVE DIAGNOSTIC PART 2
DZR6IC.P11 07-OCT-76 13:50

MACY11 27(1006) 07-OCT-76 14:14 PAGE 202
TYPE ERROR ROUTINE

SEQ 0202

10413 066026 004537 066772
10414 066032 010200
10415 066034 006200
10416 066036 103402
10417 066040 004537 066772
10418 066044 000137 065544
10419
10420
10421
10422

JSR S.O.CADV ;NUMBER IN RO - WORD MODE
MOV R2,R0
ASR R0 ;DIVIDE BY TWO
BCS 0,OF1 ;BRANCH IF ODD
JSR S.O.CADV ;NUMBER IN RO - BYTE MODE
0,OF1: JMP 0,DCD1 ;ALL DONE
:
: SEARCHES - SMSK HAS THE MASK
: SMSK+2 HAS THE FWA
: SMSK+4 HAS THE LWA
:

10423				0.EFF:	INC R1	;SET EFFECTIVE SEARCH	
10424	066050	005201			BR 0.WDS		
10425	066052	000401		0.WSCH:	CLR R1	;SET WORD SEARCH	
10426	066054	005001		0.WDS:	TST R2	;CHECK FOR OBJECT FOUND	
10427	066056	005702		0.ERR1:	BEQ 0.ERR	;ERROR IF NO OBJECT	
10428	066060	001621			MOV 0.MSK+2,R2	;SET ORIGIN	
10429	066062	013702	065236		MOV 0.MSK,R5	;SET MASK	
10430	066066	013705	065234		COM R5	;AND COMPLEMENT IT	
10431	066072	005105		0.WDS2:	CMP R2,0.MSK+4	; IS THE SEARCH ALL DONE?	
10432	066074	020237	065240		BHI 0.DCD	; YES	
10433	066100	101217			MOV 0R2,R0	; GET OBJECT	
10434	066102	011200			TST R1	;NO	
10435	066104	005701			BNE 0.EFF1	;BRANCH IF EFFECTIVE SEARCH	
10436	066106	001027			MOV R0,-(SP)		
10437	066110	010046			MOV R4,R3	;EXCLUSIVE OR	
10438	066112	010403			BIC R4,R0	; IS DONE	
10439	066114	040400			BIC (SP)+,R3	; IN A VERY	
10440	066116	042603			BIS R0,R3	; FANCY MANNER HERE	
10441	066120	050003			BIC R5,R3	; AND RESULT WITH MASK	
10442	066122	040503		0.WDS3:	BNE 0.WDS4	;RE-LOOP IF NO MATCH	
10443	066124	001016			MOV R4,-(SP)	;REGISTERS R2,R4, AND R5 ARE SAFE	
10444	066126	010446			JSR 5,0.CRLF	;TYPE <CR,LF>	
10445	066130	004537	067226		MOV R2,R0	;GET READY TO TYPE	
10446	066134	010200			JSR 5,0.CADV	;TYPE ADDRESS	
10447	066136	004537	066772		MOV #/,R0	;SLASH TO R0	
10448	066142	012700	000057		JSR 5,0.FTYP	;TYPE IT	
10449	066146	004537	067134		MOV 0R2,R0	;GET CONTENTS	
10450	066152	011200			JSR 5,0.CADV	;TYPE CONTENTS	
10451	066154	004537	066772		MOV (SP)+,R4	;RESTORE R4	
10452	066160	012604		0.WDS4:	TST (R2)+	;INCREMENT TO NEXT CELL AND	
10453	066162	005722			BR 0.WDS2	;RETURN	
10454	066164	000743		0.EFF1:	CMP R0,R4	; IS (X)=K?	
10455	066166	020004			BEQ 0.WDS3	;TYPE IF EQUAL	
10456	066170	001755			MOV R0,R3	; (X) TO R3	
10457	066172	010003			ADD R2,R3	; (X)+X	
10458	066174	060203			INC R3		
10459	066176	005203			INC R3	; (X)+X+2	
10460	066200	005203			CMP R3,R4	; IS (X)+X+2=K?	
10461	066202	020304			BEQ 0.WDS3	;BRANCH IF EQUAL	
10462	066204	001747			BIC #177400,R0	;WIPE OUT EXTRANEIOUS BITS	
10463	066206	042700	177400		MOV R0,R0	;EXTEND SIGN	
10464	066212	110000			CCC		
10465	066214	000257			ASL R0	;MULTIPLY BY TWO	
10466	066216	006300			INC R0	;ADD TWO	
10467	066220	005200			INC R0		
10468	066222	005200			ADD R2,R0	;ADD PC	
10469	066224	060200			CMP R0,R4	; IS THE RESULT A PROPER REL. BRANCH?	
10470	066226	020004			BR 0.WDS3		
10471	066230	000735					
10472							
10473							
10474							
10475	066232	105037	067255		0.GO:	CLRB 0.P	;DISALLOW PROCEED
10476	066236	006204			ASR R4	;CHECK LOW ORDER BIT	
10477	066240	103617			BCS 0.ERR2	;ERROR IF ODD NUMBER	
10478	066242	006304			ASL R4	;RESTORE WORD	

```

10479 066244 010437 065226          MOV      R4,0.UPC          ;SET UP NEW PC
10480 066250 112737 000340 177776  MOVB     #0.STM,ST        ;SET HIGH PRIORITY
10481 066256 004537 066724          JSR      5,0.RSTT        ;RESTORE TELETYPE
10482 066262 105037 067254          CLR      0.T             ;CLEAR BOTH
10483 066266 042737 000020 065230  O.TBIT: BIC      #0.TBT,0.UST ; T-BIT FLAGS
10484 066274 017737 176742 065246  MOV      @0.ADR1,0.UIN    ;SAVE INSTRUCTION
10485 066302 013777 067316 176732  MOV      0.TRTC,@0.ADR1  ;REPLACE WITH TRAP
10486 066310 012600          MOV      (SP)+,R0        ;RESTORE
10487 066312 012601          MOV      (SP)+,R1        ; R0
10488 066314 012602          MOV      (SP)+,R2        ; THRU
10489 066316 012603          MOV      (SP)+,R3
10490 066320 012604          MOV      (SP)+,R4
10491 066322 012605          MOV      (SP)+,R5
10492 066324 012606          MOV      (SP)+,SP        ; R5
10493 066326 013746 065230          MOV      0.UST,-(SP)     ; AND SP
10494 066332 013746 065226          MOV      0.UPC,-(SP)    ; AND STATUS
10495 066336 000006          O.RTIT: RTT             ; AND PC
10496                                     ;CHANGED TO RTI FOR 11/20 AND /05
10497                                     ;
10498                                     ; PROCESS P - PROCEED
10499                                     ; ONLY ALLOWED AFTER A BREAKPOINT
10500 066340 105737 067255          O.PROC: TSTB     0.P      ;CHECK LEGALITY OF PROCEED
10501 066344 001645          BEQ      0.ERR1         ;NOT LEGAL
10502 066346 105037 067255          CLR      0.P           ;CLEAR PROCEED FLAG
10503 066352 005702          TST      R2            ;WAS COUNT SPECIFIED?
10504 066354 001402          BEQ      0.PR1         ;NO
10505 066356 010437 065244          MOV      R4,0.CT       ;YES, PUT AWAY COUNT
10506 066362 112737 000340 177776  O.PR1: MOVB     #0.STM,ST  ;FORCE HIGH PRIORITY
10507 066370 004537 066724          JSR      5,0.RSTT        ;RESTORE TTY
10508 066374 112737 000340 177776  O.C1:  MOVB     #0.STM,ST  ;SET HIGH PRIORITY
10509 066402 105237 067254          INCB     0.T           ;SET T-BIT FLAG
10510 066406 052737 000020 065230  BIS      #0.TBT,0.UST   ;SET T-BIT
10511 066414 000735          BR       0.G02
10512                                     ;
10513                                     ; BREAKPOINT HANDLER
10514                                     ; A TRT BREAKPOINT CAUSES 0.BRK TO BE ENTERED, WHICH SAVES
10515                                     ; VARIOUS ODDS AND ENDS, FINDS OUT IF THE BREAKPOINT WAS LEGAL,
10516                                     ; AND GIVES CONTROL TO THE COMMAND DECODER
10517                                     ;
10518 066416 012637 065226          O.BRK: MOV      (SP)+,0.UPC ;PRIORITY IS 7 UPON ENTRY
10519 066422 012637 065230          MOV      (SP)+,0.UST    ;SAVE STATUS AND PC
10520 066426 004037 066634          O.BK1: JSR      0,0.SVR   ;SAVE VARIOUS REGISTERS
10521 066432 105737 067254          TSTB     0.T           ;CHECK FOR T-BIT SET
10522 066436 001311          BNE      0.TBIT        ;JUMP IF SET
10523 066440 013777 065246 176574  MOV      0.UIN,@0.ADR1  ;REMOVE BREAKPOINTS
10524 066446 105737 065232          TSTB     0.PRI         ;CHECK IF PRIORITY
10525 066452 100003          BPL      0.BK2        ; IS AS SAME AS USER PGM
10526 066454 113705 065230          MOVB     0.UST,R5      ;PICK UP USER UST IF SO
10527 066460 000407          BR       0.BK3        ;AND DON'T COMPUTE THE PRIORITY
10528 066462 113705 065232          O.BK2: MOVB     0.PRI,R5 ;OTHERWISE PICK UP ACTUAL PRIORITY
10529 066466 000257          CCC
10530 066470 106005          RORB     R5           ;CLEAR CARRY
10531 066472 106005          RORB     R5           ;SHIFT LOW ORDER BITS
10532 066474 106005          RORB     R5           ; INTO
10533 066476 106005          RORB     R5           ; HIGH ORDER
10534 066500 110537 177776          O.BK3: MOVB     R5,ST    ; POSITION
                                     ; PUT THE STATUS AWAY WHERE IT BELONGS

```

```

10535 066504 013705 065226      MOV      0.UPC,R5      ;GET PC, IT POINTS TO THE TRT
10536 066510 005745              TST      -(R5)        ;SUBTRACT TWO
10537 066512 010537 065226      MOV      R5,0.UPC     ;FROM THE USER'S PC
10538 066516 020537 065242      CMP      R5,0.ADR1    ;COMPARE WITH LIST
10539 066522 001417              BEQ      0.B2         ;JUMP IF FOUND
10540 066524 004537 066672      JSR      5,0.SVTT     ;SAVE TELETYPE STATUS
10541 066530 004537 067226      JSR      5,0.CRLF     ;
10542 066534 012704 067260      MOV      #0,BD,R4     ;ERROR, NOTHING FOUND
10543 066540 012703 067261      MOV      #0,BD+1,R3   ;
10544 066544 004537 067120      JSR      5,0.TYPE     ;OUTPUT "BE" FOR BAD ENTRY
10545 066550 010500              MOV      R5,R0        ;
10546 066552 042737 000020 065230  BIC      #0,TBT,0.UST ;CLEAR OUT ANY POSSIBLE FAKE T-BIT
10547 066560 000420              BR       0.B3         ; AND CONTINUE
10548 066562 005337 065244      0.B2:   DEC      0.CT   ;
10549 066566 003302              BGT      0.C1         ; JUMP IF REPEAT
10550 066570 012737 000001 065244  MOV      #1,0.CT     ;RESET COUNT TO 1
10551 066576 105237 067255      INCB     0.P          ;ALLOW PROCEED
10552 066602 004537 066672      JSR      5,0.SVTT     ;SAVE TELETYPE STATUS, R4 IS SAFE
10553 066606 012700 000102      MOV      #1,B,R0     ;
10554 066612 004537 067134      JSR      5,0.FTYP     ;TYPE "B"
10555 066616 013700 065242      MOV      0.ADR1,R0   ;GET ADDRESS OF BREAK
10556 066622 004537 066772      0.B3:   JSR      5,0.CADV   ;TYPE ADDRESS
10557 066626 005005              CLR      R5          ;CLEAR CAD
10558 066630 000137 065540      JMP      0.DCD        ;GO TO DECODER
10559
10560      ; SAVE REGISTERS R0-R6 IN INTERNAL STACK
10561
10562 066634 012637 067252      0.SVR:  MOV      (SP)+,0.XXX ;PICK REGISTER FROM STACK AND SAVE
10563 066640 010637 065224      MOV      SP,0.USP    ;SAVE USER STACK ADDRESS
10564 066644 012706 065224      MOV      #0,USP,SP   ;SET TO INTERNAL STACK
10565 066650 010546              MOV      R5,-(SP)    ;SAVE
10566 066652 010446              MOV      R4,-(SP)    ;REGISTERS
10567 066654 010346              MOV      R3,-(SP)    ;1
10568 066656 010246              MOV      R2,-(SP)    ;THRU
10569 066660 010146              MOV      R1,-(SP)    ;5
10570 066662 013746 067252      MOV      0.XXX,-(SP) ;PUT SAVED REGISTER ON STACK
10571 066666 005746              TST      -(SP)
10572 066670 000200              RTS      R0
10573
10574      ; SAVE TELETYPE STATUS
10575
10576 066672 113737 177560 067256  0.SVTT: MOVB     0.RCSR,0.CSR1 ;SAVE R C/SR
10577 066700 113737 177564 067257  MOVB     0.TCSR,0.CSR2 ;SAVE T C/SR
10578 066706 105037 177560      CLRB     0.RCSR      ;CLEAR ENABLE AND MAINTENANCE
10579 066712 105037 177564      CLRB     0.TCSR      ;BITS IN BOTH C/SR
10580 066716 004537 067226      JSR      5,0.CRLF    ;TYPE <CR,LF>
10581 066722 000205      RTS      R5
10582
10583      ; RESTORE TELETYPE STATUS
10584
10585 066724 004537 067226      0.RSTT: JSR      5,0.CRLF ;<CR,LF> BEFORE RESTORING
10586 066730 105737 177564      TSTB     0.TCSR      ;WAIT READY ON PRINTER
10587 066734 100375              BPL      -4
10588 066736 032737 004000 177560  BIT      #4000,0.RCSR ;CHECK BUSY FLAG ON READER
10589 066744 001403              BEQ      0.RSE1      ;SKIP READY LOOP IF NOT BUSY
10590 066746 105737 177560      TSTB     0.RCSR      ;WAIT READY

```

```

10591 066752 100375          BPL      -4          ; ON READER
10592 066754 113737 067256 177560 0.RSE1: MOVB  0.CSR1,0.RCSR ;RESTORE
10593 066762 113737 067257 177564  MOVB  0.CSR2,0.TCSR ; THE STATUS REGISTERS
10594 066770 000205          RTS      R5
10595
10596          ; TYPE OUT CONTENTS OF WORD OR BYTE WITH ONE TRAILING SPACE
10597          ; WORD IS IN R0
10598
10599 066772 010246          0.CADV: MOV   R2,-(SP)      ;SAVE R2
10600 066774 012704 067315  MOV   #0.BUF+6,R4      ;BUFFER START ADDRESS
10601 067000 012746 000060  MOV   #'0,-(SP)      ;CONSTANT ASCII 0
10602 067004 010002          0.SPC: MOV   R0,R2      ; GET
10603 067006 042702 177770  BIC   #177770,R2      ; OCTAL CHARACTER
10604 067012 061602          ADD   @SP,R2          ; CONVERT TO ASCII
10605 067014 110244          MOVB  R2,-(R4)        ; STORE IN BUFFER
10606 067016 006200          ASR   R0              ; SHIFT THIS MESS
10607 067020 006200          ASR   R0              ; RIGHT
10608 067022 006200          ASR   R0              ; THREE WHOLE PLACES
10609 067024 020427 067310  CMP   R4,#0.BUF+1    ; DONE?
10610 067030 101365          BHI   0.SPC          ; NO
10611 067032 042700 177776  BIC   #177776,R0      ; GET LAST BIT
10612 067036 062600          ADD   (SP)+,R0      ; CONVERT TO ASCII
10613 067040 110044          MOVB  R0,-(R4)        ; AND PUT IT AWAY
10614 067042 012703 067315  MOV   #0.BUF+6,R3      ;LWA
10615 067046 004537 067120  JSR   5,0.FTYP        ;TYPE WHOLE STRING OF CHARACTERS
10616 067052 012602          MOV   (SP)+,R2      ;RESTORE R2
10617 067054 000205          RTS      R5
10618
10619          ; GENERAL CHARACTER INPUT ROUTINE
10620          ; CHARACTER INPUT GOES TO R0
10621
10622 067056 105737 177560 0.GET: TSTB  0.RCSR      ;WAIT FOR
10623 067062 100375          BPL   -4              ; INPUT FROM KEYBOARD
10624 067064 113700 177562  MOVB  0.RDB,FO        ;GET A CHARACTER
10625 067070 004537 067134  JSR   5,0.FTYP        ;ECHO CHARACTER
10626 067074 042700 177600  BIC   #177600,R0      ;STRIP OFF PARITY FROM CHARACTER
10627 067100 001766          BEQ   0.GET          ;IGNORE NULLS
10628 067102 122700 000040  CMPB  #40,R0          ;CHECK FOR SPACES
10629 067106 001763          BEQ   0.GET          ;IGNORE NULLS
10630 067110 122700 000073  CMPB  #';,R0          ;CHECK FOR SEMI-COLON
10631 067114 001760          BEQ   0.GET          ;IGNORE THEM IF FOUND
10632 067116 000205          RTS      R5
10633
10634          ; GENERAL CHARACTER OUTPUT ROUTINE
10635          ; ADDRESS OF FIRST BYTE IN R4,
10636          ; ADDRESS OF LAST BYTE IN R3, (R3)>(R4)
10637
10638 067120 020304          0.TYPE: CMP   R3,R4      ;CHECK FOR COMPLETION
10639 067122 103426          BLO   0.TYP1        ; EXIT WHEN DONE
10640 067124 112400          MOVB  (R4)+,R0      ;GET A CHARACTER
10641 067126 004537 067134  JSR   5,0.FTYP        ;TYPE ONE CHARACTER
10642 067132 000772          BR    0.TYPE        ;LOOP UNTIL DONE
10643
10644          ; TYPE ONLY ONE CHARACTER (CONTAINED IN R0)
10645
10646 067134 105737 177564 0.FTYP: TSTB  0.TCSR      ;CHECK STATUS

```

10647	067140	100375		BPL	.-4		;WAIT UNTIL READY
10648	067142	110037	177566	MOVB	R0,0.TDB		;TYPE ONE CHARACTER
10649	067146	120037	000045	CMPB	R0,0#45		;IS CHAR TO BE FILLED?
10650	067152	001012		BNE	0.TYP1		;NO
10651	067154	113746	000044	MOVB	0#44, -(SP)		;YES, INIT THE COUNT
10652	067160	105737	177564	O.TYP2: TSTB	0.TCSR		
10653	067164	100375		BPL	0.TYP2		
10654	067166	105037	177566	CLRB	0.TDB		;GENERATE NULL FILLER
10655	067172	105316		DECB	0SP		
10656	067174	003371		BGT	0.TYP2		
10657	067176	005726		TST	(SP)+		;POP STACK
10658	067200	000205		O.TYP1: RTS	R5		
10659							
10660							
10661							
10662							
10663	067202	006205		O.TCLS: ASR	R5		;GET LOW ORDER BIT
10664	067204	103405		BCS	0.TC		;JUMP IF ALREADY CLOSED
10665	067206	006305		ASL	R5		
10666	067210	005702		TST	R2		;IF NO NUMBER WAS TYPED THERE IS
10667	067212	001401		BEQ	0.CLS1		;NO CHANGE TO THE OPEN CELL
10668	067214	010415		MOV	R4,0R5		;STORE WORD
10669	067215	000207		O.CLS1: RTS	PC		
10670	067220	005746		O.TC: TST	-(SP)		;POP EXTRA CELL FROM STACK
10671	067222	000137	065524	JMP	0.ERR		;AND SCREAM BLOODY MURDER
10672							
10673							
10674							
10675							
10676	067226	012703	067263	O.CRLF: MOV	#0.CR+1,R3		;LWA <CR,LF>
10677	067232	000402		BR	0.CRS		
10678	067234	012703	067264	O.CRLS: MOV	#0.CR+2,R3		;LWA <CR,LF>*
10679	067240	012704	067262	O.CRS: MOV	#0.CR,R4		;FWA
10680	067244	004537	067120	JSR	5,0.TYPE		;TYPE SOMETHING
10681	067250	000205		RTS	R5		
10682							
10683	067252	000000		O.XXX: .WORD	0		;TEMPORARY STORAGE
10684	067254	000		O.T: .BYTE	0		; T-BIT FLAG
10685	067255	000		O.P: .BYTE	0		;PROCEED FLAG = 0 IF PROCEED NOT ALLOWED
10686							= 1 IF PROCEED ALLOWED
10687	067256	000		O.CSR1: .BYTE	0		;SAVE CELL - R C/SR
10688	067257	000		O.CSR2: .BYTE	0		;SAVE CELL - T C/SR
10689							
10690							
10691	067260	042502		O.BD: .WORD	"BE		
10692							
10693	067262	015		O.CR: .BYTE	015		; <CR>
10694	067263	012		.BYTE	012		; <LF>
10695	067264	052		.BYTE	'*		; *
10696							
10697	067265	057		O.LGCH: .BYTE	'/'		; /
10698	067266	015		.BYTE	015		; CARRIAGE RETURN
10699	067267	044		.BYTE	'\$; \$
10700	067270	107		.BYTE	'G		; G
10701	067271	012		.BYTE	012		; <LF>
10702	067272	137		.BYTE	'↑		; ↑

RBASE	=	177440	1658	1699	17130			
RCOM1	=	000000	1658	1701				
RCOM2	=	000000	1658	1702				
RCLO	=	000010	1658					
RCPUOP	=	000000	1658	1673				
ACT11	=	007504	1658	3108				
ROOM0	=	000000	1658	1703				
ROOM1	=	000000	1658	1704				
ROOM10	=	000000	1658					
ROOM11	=	000000	1658					
ROOM12	=	000000	1658					
ROOM13	=	000000	1658					
ROOM14	=	000000	1658					
ROOM15	=	000000	1658					
ROOM2	=	000000	1658	1705				
ROOM3	=	000000	1658	1706				
ROOM4	=	000000	1658	1707				
ROOM5	=	000000	1658	1708				
ROOM6	=	000000	1658	1709				
ROOM7	=	000000	1658	1710				
ROOM8	=	000000	1658					
ROOM9	=	000000	1658					
ROEVT	=	000000	1658	1664				
ROEVM	=	000000	1658	1700				
ROENV	=	000000	1658	1669				
ROEVM	=	000000	1658	1670				
ROEVM	=	000000	1658	1661				
ROEVM	=	000000	1658	1686				
ROEVM	=	000000	1658	1690				
ROEVM	=	000000	1658	1693				
ROEVM	=	000000	1658	1696				
ROEVM	=	000000	1658	1680				
ROEVM	=	000000	1658	1688				
ROEVM	=	000000	1658	1691				
ROEVM	=	000000	1658	1694				
ROEVM	=	000000	1658	1666				
ROEVM	=	000000	1658	1667				
ROEVM	=	000000	1658	1660				
ROEVM	=	000000	1658	1681				
ROEVM	=	000000	1658	1689				
ROEVM	=	000000	1658	1692				
ROEVM	=	000000	1658	1695				
ROEVM	=	000000	1658	1663				
ROEVM	=	000000	1658					
RPTCSU	=	000040	8071	8243				
RPTENV	=	000001	8018	8064	8199	8241		
RPTGIZ	=	000200	3037	8240				
RPTSPC	=	000100	8066	8201	8242			
REAFEC	=	000000	1658	1671				
REESTN	=	000000	1658	1662				
RTIN	=	007360	1832	6755	6774	6800	7540	
RUNIT	=	000000	1658	1665				
RUSWR	=	000000	1658	1672				
RVECT1	=	000000	1658	1697				
RVECT2	=	000000	1658	1698				
R:GSP	=	043112	6486	6491	7584			

DO1

UNIBUS Rk06 DRIVE DIAGNOSTIC PART 2
DZ61C.P11 07-OCT-76 13:50

MACY11 27(1006) 07-OCT-76 14:14 PAGE 211
CROSS REFERENCE TABLE -- USER SYMBOLS

SEQ 0210

AVGTIME	043052	6073	6088	6093	6205	6210	6322	6327	7569*						
BACMDR	007354	1822*	3116*	3790*	3924*	4194*	4384*	5504*	5724*	5727*	7724				
BACTMO	044000	2188	7781*												
BA:	= 000020	1326*	3935	4086	4224	4294	4697	4760	4888	5055	5132	5255	5317	5442	
		5574*	5620												
BA16	= 000400	1292*													
BA17	= 001000	1293*													
BA170	= 000001	1293*	1289	1321	1340	1889	3227	7670							
BA1700	= 000001	1293*	1235												
BA1701	= 000002	1293*	1234												
BA1702	= 000004	1293*	1233												
BA1703	= 000010	1293*	1232												
BA1704	= 000020	1293*	1231												
BA1705	= 000040	1293*	1230												
BA1706	= 000100	1293*	1229												
BA1707	= 000200	1293*	1228												
BA1708	= 000400	1293*	1227	7953											
BA1709	= 001000	1293*	1226	7961	8029										
BA171	= 000002	1293*	1222	1890											
BA1710	= 002000	1293*	1294	1312	1331	1363	1377	1391	1404	1418	8006				
BA1711	= 004000	1293*	1295	1313	1332	1349	1364	1378	1392	1405	1419	6402	6410	6446	
		6454*	7968												
BA1712	= 010000	1293*	1296	1314	1333	1365	1379	1393	1406	1420					
BA1713	= 020000	1293*	1297	1315	1334	1350	1366	1380	1394	1407	1421	8013			
BA1714	= 040000	1293*	1298	1316	1335	1351	1367	1381	1395	1408	1422	1435	7251	7262	
		7939													
BA1715	= 100000	1293*	1299	1300	1317	1336	1352	1368	1438	5536	5669	7247	7258		
BA172	= 000004	1293*	1323	1342	1891										
BA173	= 000010	1293*	1305	1324	1343										
BA174	= 000020	1293*	1306	1325	1344	1357	1385	1412							
BA175	= 000040	1293*	1307	1326	1345	1358	1372	1386	1399	1413					
BA176	= 000100	1293*	1290	1308	1327	1346	1359	1373	1387	1400	1414				
BA177	= 000200	1293*	1291	1309	1328	1347	1360	1374	1388	1401	1415	3217	4456	4500	
		4952*	4651	4913											
BA178	= 000400	1293*	1292	1310	1329	1348	1361	1375	1389	1402	1416				
BA179	= 001000	1293*	1293	1311	1330	1362	1376	1390	1403	1417					
BPFVFC	= 000014	1293*													
BSE	= 000200	1293*	3837	3946	4011	4236	4712	5068	5266						
BSEFR	= 001526	1805*	3692*	3697*	3781										
BSE201	= 002252	1814*	3713	7256	7361										
BSE205	= 004352	1815*	3722	7260	7365										
BSE221	= 002352	1815*	3632	3694	3731	7245	7350								
BSE225	= 005352	1817*	3704	7249	7354										
BYP	= 037250	3297	3320	3324	3339	3391	3399	3417	3421	3425	3429	5827*			
BYP CER	= 001532	1807*	3214*	3443*	6941										
BYP TIM	= 001342	1730*	2950*	2955*	2960*	2965*	2970*	2975*	5750						
BYP MAT	= 001340	1729*	2949*	2954*	2959*	2964*	2969*	2974*	3775						
BYP TIM	= 012642	1454	2958*												
BYP TIM	= 012704	1460	2968*												
BYP MAT	= 012622	1452	2953*												
BYP MAT	= 012662	1458	2963*												
CALADD	= 001366	1742*	3906*	4197*	4366*	5678*	7218	7224	7226	7280					
CALCLK	= 042262	5766	5839	5957	6031	6148	6265	7412*							
CALDIF	= 001360	1739*	5567*	9969	9981										
CCLR	= 100000	1300*	3593	3611	3847	3956	4246	4470	4481	4722	4876	4946	4964	5042	
		5078	5193	5211	5276	5375	5393	5776	6102	6120	6219	6237	6336	6354	

F01

07-0CT-76 13:50 DRIVE DIAGNOSTIC PART 2

MACY11 27(1006) 07-0CT-76 14:14 PAGE 213
CROSS REFERENCE TABLE -- USER SYMBOLS

SEQ 0212

0621	064564	2035	2045	2050	2061	2066	2071	2086	2091	2096	2101	2151	2156	2161
		2176	2181	2186	2191	2421	2426	2431	2436	2687	2692	2843	2848	
		2868	2873	2878	2883	2888	2893	2898	2903	2908	2913	2918	2923	2928
		10096#	2136	2141	2266	2281	2286	2321	2326	2351	2356	10101#		
		10111#	2020	10022#										
		10116#	10029#											
		2010	2376	10034#										
		2121	2316	2637	2778	10043#								
		3366	1943	1949	1955	1960	1966	1972	1978	2119	2204	2244	2249	2289
		2311	9696#	10024	10028	10045	10053	10066	10075	10081	10088	10103	10118	
		1927	2585	2590	9740#									
		2239	10026											
		9744#	10031											
		2233	9757#											
		061505	9766#											
		061534	2134	2139	2164	2359	2685	2690	2906	2911	9775#			
		061601	2209	2234	9779#									
		061643	2209	2234	9783#									
		061727	2209	2234	9783#									
		062013	9699#	10016	10040	10055	10077							
		062037	2134	2139	2164	2359	2685	2690	2906	2911	9775#			
		062064	2209	2234	9779#									
		061140	2209	2234	9783#									
		062107	9699#	10016	10040	10055	10077							
		062142	2134	2139	2164	2359	2685	2690	2906	2911	9775#			
		062170	2038	2239	9792#									
		062222	2114	2224	2464	2479	2866	2871	2896	2901	9796#			
		062247	2074	2279	2094	2099	2109	2819	2841	2846	9801#			
		062272	2144	2299	2304	2384	2389	2414	2635	2640	2776			
		062322	1993	1998	2018	2023	2048	2059	2159	2169	2710	9805#		
		062353	1983	1988	2033	2043	2129	2149	2154	2194	2269	2294	9809#	2314
		062434	9818#	10090	10105	10120								9813#
		062515	9827#	10094	10109	10124								
		062547	2349	2565	2575	2580	2886	2891	2926	2931	9836#			
		062574	9841#	10092	10096	10107	10111	10122	10126					
		062626	2003	2008	2028	2064	2069	2174	2179	2274	9845#			
		062692	9850#											
		062714	9854#	10098	10113	10128								
		062776	9860#											
		063031	2600	2605	2876	2881	2916	2921	9869#					
		063106	9874#	10060										
		063133	2781	9882#										
		063204	2735	9886#										
		063255	2740	9893#										
		063334	2745	9900#	10073									
		063404	2264	9908#										
		061211	9915#	10036										
		063461	9706#											
		063514	2124	2354	2419	2424	2429	2434	9923#					
		063550	2013	9928#										
		063576	2084	2089	2184	2189	9933#							
		063650	2199	9937#										
		063715	9944#	10083										
		063755	2279	2284	9951#									
		061260	9957#											
		061342	9713#	10047										
		061411	9722#	10068										
		061432	2104	9729#										
			2214	2219	2224	2229	2344	9732#						

G01

UNIBUS RK06 DRIVE DIAGNOSTIC PART 2
 DZR6IC.P11 07-OCT-76 13:50

MACY11: 27(1006) 07-OCT-76 14:14 PAGE 214
 CROSS REFERENCE TABLE -- USER SYMBOLS

SEQ 0213

DI = 040000	1298#													
DISPLA 001142	1633#	3025*	3033*	7983*	8005*									
DISPRE 000174	1447#	3033												
DIYTYP 043204	7576	7602	7609#											
DI = 100000	1317#	4591												
CLY = 037232	5448	6818#	6821											
DMD = 000040	1358#													
DCTIM 007536	1917#	3136*	3145*	5429	5755									
DOWRT 016650	3776	3779#												
DPAT1 001516	1799#	5254	5316	5330										
DPAT2 001520	1800#													
DRA = 000001	1340#	3314	3407											
DRCY = 000200	1347#													
DRIVS 007510	1901#	3222*	3229*	3246	3261*	3287*	3304	3395*	3461	6545	6637*			
DRIV0 007512	1906#	3118	3224	3263	3363	6620								
DRIV1 007514	1907#													
DRIV2 007516	1908#													
DRIV3 007520	1909#													
DRIV4 007522	1910#													
DRIV5 007524	1911#													
DRIV6 007526	1912#													
DRIV7 007530	1913#													
DRPT = 000040	1345#													
DRPAR = 000010	1324#													
DRVMSK = 000007	1304#	3280	3380											
DRVPTR 001346	1733#	3118*	3466	3476*										
DSC = 040000	1351#													
DSWR = 177570	1156#	1632	3024											
DYE = 010000	1333#													
DYE = 000040	1326#													
DT1 064002	1938	1944	1950	1956	1961	1967	1973	1979	1984	1989	1994	1999	2004	
	2014	2024	2029	2039	2055	2075	2080	2115	2125	2130	2165	2195	2200	
	2205	2210	2215	2220	2225	2230	2235	2240	2245	2250	2270	2275	2290	
	2300	2305	2330	2335	2340	2345	2360	2365	2370	2375	2395	2390	2415	
	2465	2480	2566	2586	2591	2641	2711	2721	2726	2736	2741	2746	2782	
	9965#													
DT10 064170	2120	9987#												
DT13 064210	2034	2044	2049	2060	2065	2070	2085	2090	2095	2100	2150	2155	2160	
	2170	2175	2180	2185	2190	2420	2425	2430	2435	2686	2691	2842	2847	
	2867	2872	2877	2882	2887	2892	2897	2902	2907	2912	2917	2922	2927	
	2932	9990#												
	2110	2135	2140	2265	2280	2285	2320	2325	2350	2355	9996#			
DT14 064250	10003#													
DT15 064320	2105	9968#												
DT3 064024	2310	2315	2636	9969#										
DT4 064030	2295	2371	2576	2581	2601	2606	2716	9972#						
DT5 064050	2009	2019	9975#											
DT6 064070	2611	9978#												
DT7 064110	2777	9981#												
DT8 064130	2145	9984#												
DT9 064150	1400#													
D.ACLO= 000100	1387#	3650	3853	3872	3962	3981	4029	4046	4109	4128	4163	4252	4280	
D.BRHM= 000100	4315	4343	4418	4436	4489	4538	4601	4729	4747	4782	4801	4844	4862	
	5010	5028	5084	5103	5154	5173	5282	5301	5338	5357	5463	5595	5641	
	1389#	3650	3853	3872	3962	3981	4029	4046	4109	4128	4163	4252	4280	
D.CART= 000400	4315	4343	4418	4436	4489	4532	4601	4729	4747	4782	4801	4844	4862	

NO1

UNIBUS RK06 DRIVE DIAGNOSTIC PART 2
DZR6IC.P11 07-OCT-76 13:50

MACY11 27(1006) 07-OCT-76 14:14 PAGE 221
CROSS REFERENCE TABLE -- USER SYMBOLS

SEG 0220

O. BO	067260	10542	10543	10691*					
O. BKPT	065474	10307*	10360						
O. BK1	066426	10255	10520*						
O. BK2	066462	10525	10529*						
O. BK3	066500	10527	10534*						
O. BRK	066416	10272	10518*						
O. BUF	067307	10600	10609	10614	10719*				
O. B2	066562	10539	10548*						
O. B3	066622	10547	10556*						
O. CADV	066772	10373	10390	10413	10417	10447	10451	10556	10599*
O. CLGL	065610	10330	10332	10340*					
O. CLGT =	000014	10344	10709*						
O. CLS1	067216	10667	10669*						
O. CR	067262	10676	10678	10679	10693*				
O. CRET	065722	10351	10380*						
O. CRLF	067226	10388	10445	10541	10580	10585	10676*		
O. CRLS	067234	10325	10678*						
O. CRS	067240	10677	10679*						
O. CSR1	067256	10576*	10592	10687*					
O. CSR2	067257	10577*	10593	10688*					
O. CT	065244	10505*	10548*	10550*	10747*				
O. C1	066374	10508*	10549						
O. DCD	065540	10313	10315	10325*	10382	10433	10558		
O. DCD1	065544	10326*	10374	10418					
O. EFF	066050	10359	10424*						
O. EFF1	066166	10436	10455*						
O. ENTR	065254	10252*							
O. ERR	065524	10299	10310	10322*	10345	10370	10428	10671	
O. ERR1	066060	10428*	10501						
O. ERR2	065700	10370*	10404	10477					
O. FTYP	067134	10324	10392	10407	10449	10554	10625	10641	10646*
O. GET	067056	10278	10328	10622*	10627	10629	10631		
O. GO	066232	10353	10475*						
O. GO2	066310	10486*	10511						
O. LG =	000006	10282	10717*						
O. LGCH	067265	10341	10697*	10709					
O. LGDR	065640	10348	10350*	10362					
O. LGL =	000030	10362*							
O. LGL1	065612	10341*	10346						
O. LGL2	065632	10342	10347*						
O. MSK	065234	10429	10430	10432	10739*				
O. ODT	065250	1463	10250*	10728					
O. OFST	065776	10357	10403*						
O. OF1	066044	10416	10418*						
O. OP1	065734	10354	10386*						
O. OP2	065742	10303	10388*	10399					
O. ORPC	065446	10295*	10355						
O. P	067255	10270*	10475*	10500	10502*	10551*	10685*		
O. PR1	065232	10262	10524	10528	10738*				
O. PROC	066340	10361	10500*						
O. PR1	066362	10504	10506*						
O. RALL	065514	10273	10308	10314*					
O. RCSP =	177560	10240*	10576	10578*	10588	10590	10592*	10622	
O. RDB =	177562	10239*	10624						
O. REGT	065376	10278*	10352						
O. PSE:	066754	10589	10592*						

PK5	001320	1716	3126	3133	7625*	7645*								
PK5B	001322	1717	7624*											
PPTP	007506	1900	3075*											
PRGSRT	012742	2951	2956	2961	2966	2971	2976	6647						
PR0	000000	1172	2978	3165										
PR1	000040	1173												
PR2	000100	1174												
PR3	000140	1175												
PR4	000200	1176												
PR5	000240	1177	1715											
PR6	000300	1178	3138											
PR7	000340	1179	3042	3048	7855	7856								
PS	177776	1152	1153											
PSEC	001402	1751												
PSA	177776	1153												
PWAVEC	000024	1244	3012*	3013*	7845*	7854*	7855*							
RKCYLD	040666	6401	6409	6445	6453	7075	7493							
RCYLA	041100	6924	7117											
RCYLD	041026	6921	7104											
ROCHR	104410	8522	8904											
ROCYLA	040756	7092												
ROCYLD	040602	7059												
RODATA	000021	1283	3639	4004										
ROGATE	100000	1358												
ROHEAD	000025	1285	4576	5963	5979									
ROLIN	104411	6615	8596	8905										
ROOCT	104412	6660	6673	8906										
ROSEC	040532	5971	5989	7047	7148	7152	7155	7303						
ROTAB	006352	1818	4000	4059	4067									
ROY	000200	1291	5450	5981	6700	6714	7079							
RECAL	000013	1280	3595	4948	5195	5377	6104	6221	6338	6502	7730			
RESREG	104414	7461	7517	7578	7604	8656	8708	8908	10197					
RESVEC	000010	1239												
RHEAD	041174	6929	7137											
RTAB	001742	1811	4575	4614	7310	7312	7316	7319	9984					
RKASOF	000016	1262	3179	4408*	4454*	4498*	4550*	4831*	5000*	6739	6774	6800	7540	
RKBA	000034	1257	3175	3633*	3687*	3705*	3714*	3723*	3801*	3825*	3902*	3933*	4000*	4074*
		4087*	4147*	4191*	4223*	4293*	4361*	4696*	4759*	4887*	5054*	5131*	5254*	5316*
		5441*	5521*	5573*	5619*	5674*	6735							
RKCS1	000000	1255	3172	3273*	3373*	3551*	3593*	3595*	3611*	3613*	3639*	3741*	3804*	3829*
		3847*	3912*	3938*	3956*	4004*	4091*	4150*	4203*	4228*	4246*	4268*	4297*	4331*
		4372*	4411*	4470*	4472*	4481*	4514*	4576*	4635*	4704*	4722*	4764*	4837*	4876*
		4878*	4891*	4946*	4948*	4964*	4966*	5003*	5042*	5044*	5060*	5078*	5136*	5193*
		5195*	5211*	5213*	5258*	5276*	5320*	5375*	5377*	5393*	5395*	5445*	5450	5457
		5525*	5544*	5577*	5623*	5684*	5705*	5770*	5776*	5778*	5841*	5963*	5979*	5981
		6045*	6102*	6104*	6120*	6122*	6162*	6219*	6221*	6237*	6239*	6279*	6336*	6338*
		6354*	6356*	6395*	6439*	6500*	6502*	6518*	6520*	6700	6714	6732	6970*	6982*
		6985*	6992*	6995*	7002*	7005*	7012*	7014*	7077*	7079	7714*	7728*	7730*	7746*
		7748*												
RKCS2	000010	1259	3173	3254*	3268*	3272*	3368*	3372*	3594*	3612*	3935*	4086*	4224*	4294*
		4471*	4591	4697*	4760*	4877*	4888*	4947*	4965*	5043*	5055*	5132*	5194*	5212*
		5255*	5317*	5376*	5394*	5442*	5574*	5620*	5777*	6103*	6121*	6220*	6238*	6337*
		6355*	6501*	6519*	6733	6969*	6983*	6993*	7003*	7013*	7031*	7035*	7729*	7747*
RKDA	000006	1258	3176	3635*	3689*	3707*	3716*	3725*	3824*	3932*	3999*	4076*	4089*	4222*
		4295*	4702*	4762*	4833*	4890*	5058*	5134*	5251*	5315*	5444*	6736		
RKDB	000024	1264	3181	4100	4306	4586	4587	4588	4773	4899	5145	5329		

F02

TEMP3	007432	6202*	6202*	6514*	6778*	7178*	7225*	7226*	7228*	7229	7719*	7742*		
TEMP4	007434	1862*	3632*	3669*	3674*	3679*	3700*	3701	3710	3719				
TEMPS	007436	1863*	3633*	3633*	3687*	3704*	3705	3713*	3714	3722*	3723			
TERR1	042222	1863*	3634*	3635*	3688*	3689	3706*	3707	3715*	3716	3724*	3725	5433*	5444
TIME1	031740	5476*	5478*	5506*	5538*	5669*	5697*							
TIMING	031706	7351	7355	7362	7366	7380*	7385	7392						
TIMUP	001376	3778	3785	3813	3866	3921	3975	4016	4122	4212	4265	4328	4381	4584
TIM1	001442	4596	4741	4795	5097	5167	5295	5351	5534	5693	5731*			
TIM2	001444	1749*	5855	7423	7433*	7435	7479	7489*	7491	7619*	7630*	7636*		
TIM3	001446	1770*	5871*	5901										
TIM4	001450	1771*	5884*	5906										
TITLE	036224	1772*	5897*	5911										
TKVEC =	000060	1773*												
TOCYL	001352	3068	3105	6593*										
		1247*	8343*	8344*										
		1736*	3904*	3906	4193*	4197	4363*	4366	4569*	4614	5505*	5513	5516	5523
		5539*	5540	5561*	5567	5572	5607	5663*	5664	5668*	5576	5678	5698*	5699
		9969	9981	9984										
TPVEC =	000064	1248*												
TRAPPC	001334	1726*	7812*	9968										
TRAPVE =	000034	1246*	3010*	3011*	7856*									
TRT =	000003	10231*	10724											
TRTVEC =	000014	1241*												
TRUERR	042122	3839	3948	4238	4714	5070	5268	7345*						
TSTATN	037024	3617	4476	4882	4970	5048	5217	5399	5782	6126	6243	6360	6524	6753*
		7752												
TST1	013714	3139	3161*											
TST10	016616	3693	3729	3771*										
TST11	017352	3895*												
TST12	021152	4184*												
TST13	022410	4399*												
TST14	023722	4666	4689*											
TST15	025534	4936	4978	4991*										
TST16	027102	5241*												
TST17	030146	5425*												
TST18	014070	3189	3210*											
TST20	030432	5431	5439	5477	5501*									
TST21	031706	5746*												
TST22	032202	5832*												
TST23	032706	5923	5950*											
TST24	033162	5976	5987	5994	6026*									
TST25	033674	6098	6143*											
TST26	034406	6215	6260*											
TST27	035120	6332	6377*											
TST3	014630	3259	3359*											
TST4	015160	3389	3455*											
TST5	015212	3495*												
TST6	015450	3500	3535*											
TST7	015552	3543	3557	3585*										
TYPE =	104405	6561	8899*											
TYPE =	104401	3072	3087	3089	3091	3106	3146	3221	3231	3260	3289	3330	3463	3477
		3485	3504	3526	3527	3730	3734	3735	3777	3783	3784	3812	3865	3920
		3974	4015	4121	4211	4264	4327	4380	4583	4595	4740	4794	4814	4818
		4824	4904	4910	4911	4928	4941	5096	5166	5294	5350	5533	5692	5752
		5757	5763	5799	5802	5803	5804	5900	5903	5905	5908	5910	5913	5915

AVGVEL	1602#	6399	6443												
CALIB	1534#	3592	4944	5191	5373	6101	6218	6335	6499	7727					
CHECK	1522#	3655	3858	3877	3967	3986	4034	4051	4114	4133	4168	4257	4285	4320	4348
	4420#	4441	4491	4543	4606	4733	4752	4787	4806	4846	4867	5012	5033	5089	5108
	5159	5178	5287	5306	5343	5362	5468	5600	5646						
COMMEN	1250#														
CMO2	1528#	5115													
DACLR	1529#	3610	4469	4875	4963	5041	5210	5392	5775	6119	6236	6353	6517	7745	
ENDCOM	1250#														
EOPGM	1592#	6541													
ERROR	1144#														
	3400	3192	3250	3257	3271	3276	3298	3306	3321	3325	3336	3340	3371	3376	3392
	3619	3418	3422	3426	3430	3502	3540	3549	3554	3558	3590	3600	3606	3609	3616
	3744	3625	3642	3646	3658	3659	3660	3661	3664	3671	3676	3681	3684	3695	3738
	3880	3748	3752	3788	3807	3811	3821	3832	3845	3849	3861	3862	3863	3864	3867
	3976	3881	3882	3883	3900	3915	3919	3929	3941	3954	3958	3970	3971	3972	3973
	4055	3989	3990	3991	3992	3997	4007	4013	4020	4023	4037	4038	4039	4040	4054
	4139	4056	4057	4065	4084	4094	4102	4105	4117	4118	4119	4120	4136	4137	4138
	4260	4145	4153	4157	4171	4172	4173	4174	4189	4206	4210	4219	4221	4244	4248
	4324	4261	4262	4263	4266	4271	4274	4288	4289	4290	4291	4300	4308	4311	4323
	4424	4325	4326	4334	4337	4351	4352	4353	4354	4359	4375	4379	4406	4414	4423
	4495	4425	4426	4431	4444	4445	4446	4447	4451	4461	4466	4475	4478	4485	4494
	4567	4425	4437	4505	4510	4517	4521	4525	4532	4546	4547	4548	4549	4557	4562
	4646	4579	4582	4594	4609	4610	4611	4612	4616	4622	4626	4630	4632	4638	4642
	4775	4694	4707	4720	4724	4736	4737	4738	4739	4742	4755	4756	4757	4758	4767
	4852	4778	4790	4791	4792	4793	4809	4810	4811	4812	4829	4840	4849	4850	4851
	5006	4857	4870	4871	4872	4873	4881	4884	4894	4953	4959	4962	4969	4972	4998
	5092	5015	5016	5017	5018	5023	5036	5037	5038	5039	5047	5050	5063	5076	5080
	5162	5093	5094	5095	5098	5111	5112	5113	5114	5118	5121	5128	5139	5147	5150
	5274	5163	5164	5165	5181	5182	5183	5184	5200	5206	5209	5216	5219	5248	5261
	5347	5278	5290	5291	5292	5293	5296	5309	5310	5311	5312	5323	5331	5324	5346
	5455	5348	5349	5365	5366	5367	5368	5382	5388	5391	5398	5401	5435	5428	5453
	5589	5459	5471	5472	5473	5474	5509	5528	5532	5547	5551	5555	5570	5580	5585
	5672	5603	5604	5605	5606	5609	5616	5626	5631	5635	5649	5650	5651	5652	5655
	5883	5687	5691	5708	5712	5716	5761	5773	5781	5784	5797	5837	5844	5858	5870
	6115	5896	5921	5928	5955	5966	5974	5986	5992	6040	6048	6053	6058	6076	6109
	6282	6118	6125	6128	6157	6165	6170	6175	6193	6226	6232	6235	6242	6245	6274
	6437	6287	6292	6310	6343	6349	6352	6359	6362	6390	6398	6406	6416	6420	6424
	6988	6442	6450	6460	6464	6468	6507	6513	6516	6523	6526	6949	6954	6958	6973
	7813	6998	7008	7017	7034	7083	7700	7712	7717	7721	7735	7741	7744	7751	7754
ESCAPE	1250#														
F.ERB	1512#	3647	3850	3869	3959	3978	4025	4043	4106	4125	4159	4249	4276	4312	4339
	4433	4534	4598	4725	4744	4779	4798	4859	5025	5081	5100	5151	5170	5279	5298
	5335	5354	5460	5591	5637										
GETPRI	1250#														
GETSWR	1250#	6597													
HCHK3	1550#	4573													
HDTBL	1554#	3905	4196	4365	5677										
LOOP	1506#	3264	3364	3545	3817	3925	3993	4079	4141	4215	4355	5124	5564	5611	
MSG	3151#	3153	3196#	3198	3344#	3346	3446#	3448	3488#	3490	3529#	3531	3562#	3564	3761#
	3763	3885#	3887	4177#	4179	4385#	4388	4669#	4671	4981#	4983	5225#	5227	5407#	5409
	5482#	5484	5736#	5738	5808#	5810	5932#	5934	6017#	6019	6134#	6136	6251#	6253	6367#
	6369														
MULT	1250#	5998	6062	6179	6296	6428	6472	7553							
NEWTST	1250#	3151	3196	3344	3446	3488	3529	3562	3761	3885	4177	4386	4669	4981	5225
	5407	5482	5736	5808	5932	6017	6134	6251	6367						

.SCATC	1099#	1440
.SCMTA	1099#	1605
.SOB20	1099#	8669
.SOB20	1099#	8630
.SEOP	1099#	6532
.SERRO	1099#	7987
.SMULT	1099#	8772
.SRDOC	1099#	8577
.SREAD	1099#	8321
.SSAVE	1099#	8819
.SSB20	1099#	8731
.SSCOP	1099#	7923
.SSUPR	1099#	8749
.STRAP	1099#	8864
.STYPO	1099#	8120
.STYPE	1099#	8041
.STYPO	1099#	8244

. ABS. 067320 000

% ERRORS DETECTED: 0 HARD 2 SOFT
 DEFAULT GLOBALS GENERATED: 0

DZR6IC,DZR6IC.SEG/SOL/NL:MD/EQ:SDC/CRF/NL:TOC/DOC=DZR6IC.P11
 RUN-TIME: 80 80 9 SECONDS
 RUN-TIME RATIO: 248/172=1.4
 CORE USED: 32K (63 PAGES)

DOCUMENT PAGES: 233

B03

Spooler runtime 35 Seconds, 147 KCS, 925 disk reads, 0 disk writes, 233 pages
Date 01-Sep-76 17:00:09 Monitor IPC-0 602 (680) #END#

00000000111111112222222233333333444444445555555566666666777777778888888899999999
.....