

RK611
RK06, RK07

RK6L CTG FMTR
CZR6LDO

AH-9133D-MC
FICHE 1 OF 1

MAR 1982
COPYRIGHT © 76-81
MADE IN USA



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

.REM %

IDENTIFICATION

PRODUCT CODE: AC-9132D-MC
PRODUCT NAME: CZR6LDO RK06L CTG FMTR
DATE: AUGUST 10 1981
MAINTAINER: DIAGNOSTIC GROUP
AUTHOR: BRIAN LEBLANC

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

THE SOFTWARE DISCRIBED 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) 1976,1981 BY DIGITAL EQUIPMENT CORPORATION

38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60
61
62
63
64
65
66
67
68
69
70
71
72
73
74
75
76
77
78
79
80
81
82
83
84
85

TABLE OF CONTENTS

1.0 ABSTRACT

2.0 REQUIREMENTS

- 2.1 EQUIPMENT
- 2.2 PRELIMINARY PROGRAMS
- 2.3 MEDIA

3.0 OPERATING PROCEDURES

- 3.1 LOADING PROCEDURE
- 3.2 STARTING PROCEDURES
 - 3.2.1 STARTING ADDRESS
 - 3.2.2 RESTART ADDRESS
- 3.3 RK611 ADDRESS
- 3.4 OPTIONAL SWITCH SETTINGS
- 3.5 'SOFTWARE' SWITCH REGISTER
- 3.6 RUN TIME

4.0 OPERATING PROCEDURE

- 4.1 SUMMARY OF PARAMETERS
- 4.2 PARAMETER DESCRIPTION
 - 4.2.1 DRIVE TYPE
 - 4.2.2 DRIVE NUMBER
 - 4.2.3 SECTOR/TRACK
 - 4.2.4 MODE
 - 4.2.5 EVEN CYLINDER PATTERN AND ODD CYLINDER PATTERN
 - 4.2.6 TRACK LIMITS AND CYLINDER LIMITS
 - 4.2.7 OFFSET
- 4.3 BAD SECTOR OPTION
 - 4.3.1 SPECIFYING SECTORS THAT ARE TO BE FLAGGED BAD
 - 4.3.2 PRESERVING SOFTWARE BAD SECTOR FILES

5.0 PROGRAM DESCRIPTION

6.0 ERROR REPORTING

S

86
87
88
89
90
91
92
93
94
95
96
97
98
99
100
101
102
103
104
105
106
107
108
109
110
111
112
113
114
115
116
117
118
119
120
121
122
123
124
125
126
127
128
129
130
131
132
133
134
135
136
137
138
139
140
141

1.0 ABSTRACT

THE RK06-RK07 CARTRIDGE FORMATTER PROGRAM IS DESIGNED TO PROVIDE A MEANS TO WRITE AND VERIFY HEADER AND DATA INFORMATION ON AN RK06K-RK07K DISK PACK AT ALL POSSIBLE DISK PACK ADDRESSES. THE PROGRAM IS DESIGNED TO PROVIDE FLEXIBILITY IN THE TYPES OF OPERATION AND THE DATA USED. (SEE OPERATING PROCEDURE MODES OF OPERATION.) FORMATTING IS DONE ON THE BASIS OF 411 CYLINDERS, 3 TRACKS, AND 20 OR 22 SECTORS PER TRACK (SELECTED WITH A PARAMETER ENTRY).

THE PROGRAM UTILIZES THE BAD SECTOR FILE DESCRIBED IN THE RK06K-RK07K DISK CARTRIDGE SPECIFICATION TO:

- A. REPORT THE SERIAL NUMBER OF THE CARTRIDGE BEING FORMATTED,
- B. CHECK IF THE CARTRIDGE IS AN ALIGNMENT CARTRIDGE AND ABORT THE PROGRAM IF IT IS,
- C. AND IDENTIFY THE SECTORS THAT ARE TO BE MARKED BAD.

AS FORMATTING IS BEING DONE, THE SECTORS IDENTIFIED IN THE "FACTORY DETECTED BAD SECTOR FILE" (FDBSF) ARE UNCONDITIONALLY FLAGGED BAD BY RESETTING BIT 15 IN THE SECOND HEADER WORD. NORMALLY THE "SOFTWARE DETECTED BAD SECTOR FILE" (SDBSF) IS DESTROYED AND ONLY THOSE SECTORS THAT THIS PROGRAM FINDS ARE MARKED BAD AND ENTERED IN THE SDBSF. OPTIONALLY THE PROGRAM WILL PRESERVE THE SDBSF AND MARK THE SECTORS FOUND IN THE SDBSF BAD BY RESETTING BIT 14 OF THE SECOND HEADER WORD.

SECTORS FOUND BAD BY THIS PROGRAM ARE ADDED TO THE SDBSF AND MARKED BAD BY RESETTING BIT 14 OF THE SECOND HEADER WORD.

IF THE SDBSF IS TO BE PRESERVED OR IF THE PROGRAM IS RUN IN A VERIFICATION MODE, THE BAD SECTOR FILES FOR SOFTWARE DETECTED BAD SECTORS MUST EMPLOY THE SAME FORMAT AS DESCRIBED IN THE RK06K-RK07K CARTRIDGE SPECIFICATION FOR FACTORY DETECTED BAD SECTORS. (SEE THE REQUIREMENT IN PARAGRAPH 2.3.) THEREFORE, THE PROGRAM WILL EXPECT ALL EVEN SECTORS 10 THROUGH 20 TO CONTAIN THE ADDRESS OF THE SECTORS FOUND BAD IN 22 SECTOR/TRACK MODE AND ALL ODD SECTORS 11 THROUGH 21 TO CONTAIN THE ADDRESS OF THE SECTORS FOUND BAD IN 20 SECTORS/TRACK MODE. IF FORMATTING IS DONE WITHOUT PRESERVING THE SDBSF THE SDBSF FOR THE SECTOR/TRACK FORMAT IS WRITTEN AS DESCRIBED.

IF THE CARTRIDGE IS FORMATTED (HEADERS WRITTEN) THE DATA USED IS THAT SPECIFIED FOR THE ODD AND EVEN CYLINDERS. THE DATA USED CAN BE SPECIFIED BY THE OPERATOR. WRITING IS DONE UTILIZING THE BUS ADDRESS INCREMENT INHIBIT FUNCTIONS AND A WORD COUNT EQUAL TO ONE TRACK. VERIFICATION AFTER WRITING IS ALSO DONE WITH BUS ADDRESS INCREMENT INHIBIT SET IN A WRITE CHECK OPERATION. AGAIN THE WORD COUNT IS LARGE ENOUGH FOR ONE TRACK.

IF THE PACK IS NOT FORMATTED (VERIFICATION ONLY) THE DATA IS READ ONE SECTOR AT A TIME. HARDWARE ECC IS RELIED UPON TO DETECT ANY ERRORS.

THE PROGRAM HAS THE ABILITY TO DO THE VERIFICATION OPERATION (EITHER AFTER FORMATTING OR AS VERIFICATION ONLY) WITH THE HEADS OFFSET. A

142
143
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

SINGLE OFFSET VALUE IS SPECIFIED IN PROGRAM STARTUP AND THAT OFFSET IS APPLIED TO EVERY READ OPERATION.

C A U T I O N

IF THE PROGRAM IS HALTED WHILE FORMATTING IS IN PROGRESS, THE STATE OF THE CARTRIDGE CANNOT BE PREDICTED.

2.0 REQUIREMENTS

2.1 EQUIPMENT

PDP-11 SYSTEM (16K CORE MEMORY)
CONSOLE TERMINAL
DECTAPE, PAPER TAPE READER, OR DECDISK
RK611 CONTROLLER
RK06/RK07 DISK DRIVE
RK06/RK07 DISK CARTRIDGE

2.2 PRELIMINARY PROGRAMS

ALL RK06 DIAGNOSTICS SHOULD HAVE PREVIOUSLY BEEN RUN SUCCESSFULLY.

2.3 MEDIA

THE CARTRIDGE TO BE FORMATTED MUST HAVE CYLINDER 411(10), TRACK 2 FORMATTED IN 22(10) SECTOR/TRACK. EACH SECTOR OF THE FCBSF MUST HAVE THE CARTRIDGE SERIAL NUMBER IN THE FIRST TWO WORDS, FOLLOWED BY TWO WORDS OF ZEROS, FOLLOWED BY THE ADDRESS OF THE BAD SECTOR (IF ANY), FOLLOWED BY A WORD OF ALL ONES. IF THE SDBSF IS TO BE PRESERVED OR IF THE OPERATION IS VERIFICATION ONLY, THE SDBSF MUST ALSO CONFORM TO THE ABOVE.

NOTE: FOR THE RK07K, CYLINDER 815(10), TRACK 2 MUST BE FORMATTED IN 22(10) SECTOR/TRACK FIRST.

3.0 OPERATING PROCEDURE

3.1 LOADING PROCEDURE

198
199
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

THE PROGRAM CAN BE LOADED FROM PAPER TAPE USING THE ABSOLUTE LOADER OR FROM ANY MEDIA SUPPORTED BY XXDP BUT IT IS NOT CHAINABLE. IT CAN ALSO BE LOADED BY APT OR ACT IN DUMP MODE ONLY.

THE PROGRAM DOES NOT DESTROY THE LOADER.

3.2 STARTING PROCEDURE

3.2.1 STARTING ADDRESS

PROGRAM STARTING LOCATION IS 200.

3.2.2 RESTART ADDRESS

LOCATION 200 - RESTART THE PROGRAM TO REQUEST PARAMETERS.

LOCATION 204 - RESTART THE PROGRAM USING THE PARAMETERS AS LAST ENTERED.

3.3 RK611 ADDRESSES

THE PROGRAM IS ASSEMBLED TO USE 177440 AS THE RK611 ADDRESS. THE VECTOR IS ASSUMED TO BE 210. TO ALTER THESE ADDRESS ASSIGNMENTS, THE FOLLOWING MEMORY LOCATIONS MUST BE CHANGED BEFORE STARTING:

RK611 ADDRESS - RKBAS - LOCATION 2570
RK611 VECTOR - RKVEC - LOCATION 2572

3.4 OPTIONAL SWITCH SETTINGS

SW15 - HALT PROGRAM ON ERROR.

SW14 - LOOP ON THE CURRENT CYLINDER AND TRACK OPERATION.

SW13 - INHIBIT ERROR TYPEOUT.

SW10 - BELL ON ERROR

SW07 - REPORT SUMMARY OF ALL BAD SECTORS

SW01 - REPORT ALL DATA IN ERROR

SW00 - LOOP PROGRAM WITH PARAMETERS AS LAST SPECIFIED.

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

3.5 '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 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 SETTINGS 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.

3.6 RUN TIME

THE PROGRAM RUN TIME FOR A ENTIRE FORMAT OPERATION IS APPROXIMATELY 2.5 MINUTES. THE RUN TIME FOR VERIFICATION ONLY IS APPROXIMATELY 1.5 MINUTES.

NOTE: TIMES ARE APPROX. DOUBLED FOR RK07 FORMATTING.

4.0 OPERATING PROCEDURE

4.1 SUMMARY OF PARAMETERS

WHEN STARTING AT LOCATION 200, THE PROGRAM IDENTIFIES ITSELF AND ASKS IF THE OPERATOR WANTS HELP. IF 'HELP' IS TYPED A SUMMARY OF THE PARAMETERS AND THEIR USES IS TYPED. IF A CARRIAGE RETURN IS TYPED THE PROGRAM CONTINUES IN THE PARAMETERIZATION SEQUENCE.

THE PARAMETER REQUEST IS PRINTED ON THE CONSOLE FOLLOWED BY AN EQUAL SIGN. THE PARAMETER IS THEN ENTERED FOLLOWED BY A CARRIAGE RETURN. AN INVALID OR ILLEGAL PARAMETER REQUEST IS ECHOED BACK TO THE CONSOLE PRECEDED BY A QUESTION MARK. THAT PARAMETER MAY THEN BE ENTERED WITHOUT RETURNING TO THE BEGINNING OF THE PARAMETER INSERTION SEQUENCE. ALL VALUES ENTERED MUST BE OCTAL.

310
311
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

TYPING A CONTROL Z (Z) (CR) AT ANY TIME DURING PARAMETER ENTRY WILL CAUSE THE REMAINING PARAMETERS TO DEFAULT.

NOTE: DO NOT USE CONTROL Z (CR) AS A RESPONSE FOR THE DRIVE TYPE (FIRST QUESTION). ALL OTHER ENTRIES CAN BE A CONTROL Z (CR).

TYPING A CONTROL C (C) AT ANY TIME WILL RETURN TO FIRST PARAMETER ENTRY REQUEST.

TYPING A CARRIAGE RETURN AFTER THE PARAMETER REQUEST CAUSES THAT PARAMETER TO DEFAULT.

TYPING A RUBOUT CANCELS THE LAST CHARACTER TYPED.

TYPING A CONTROL U (U) CANCELS THE LINE BEING TYPED.

TYPING A CONTROL S (S) WILL STOP THE PROGRAM (LOOPS WAITING FOR Q).

TYPING A CONTROL Q (Q) CONTINUES PROGRAM EXECUTION AFTER A S.

ALL ENTRIES MUST BE TERMINATED BY A CARRIAGE RETURN

THE FOLLOWING TABLE SHOWS THE PARAMETERS THAT WILL BE REQUESTED, THE OPTIONS AVAILABLE, AND THE DEFAULT. EACH OF THE PARAMETERS IS DESCRIBED IN DETAIL IN THE FOLLOWING PARAGRAPHS.

PARAMETER	OPTION	DEFAULT
-----	-----	-----
DRIVE TYPE	6-7	6
DRIVE NUM	0-7	0
SECTORS/TRACK	20,22	22
MODE	0-4	2
EVEN CYLINDER PAT	6 CHAR	135143
ODD CYLINDER PAT	6 CHAR	072307
TRACK LIMITS	0-2,0-2	0,2
CYLINDER LIMITS	0-632,0-632	0,632 FOR RK06
	0-1456,0-1456	0,1456 FOR RK07
OFFSET	060-160	000 (0 OFFSET)

4.2 PARAMETER DESCRIPTION

4.2.1 DRIVE TYPE

THIS PARAMETER DEFINES THE TYPE OF DRIVE AND MEDIA TO BE FORMATTED. A 6 INDICATES AN RK06, 7 INDICATES AN RK07.

4.2.2 DRIVE NUMBER

366
367
368
369
370
371
372
373
374
375
376
377
378
379
380
381
382
383
384
385
386
387
388
389
390
391
392
393
394
395
396
397
398
399
400
401
402
403
404
405
406
407
408
409
410
411
412
413
414
415
416
417
418
419
420
421

THIS PARAMETER DEFINES THE DRIVE ADDRESS TO BE FORMATTED.

4.2.3 SECTOR/TRACK

THIS PARAMETER DEFINES THE TRACK FORMAT. A 24 INDICATES 20(10) SECTOR/TRACK AND A 26 INDICATES 22(10) SECTOR/TRACK.

4.2.4 MODE

THIS PARAMETER DEFINES THE MODE OF OPERATION FOR THE PROGRAM. THE MODES ARE:

- 0 = WRITE HEADERS AND DATA
- 1 = WRITE HEADERS AND DATA: VERIFY HEADERS
- 2 = WRITE HEADERS & DATA: VERIFY HEADERS & DATA
- 3 = VERIFY HEADERS
- 4 = VERIFY HEADERS & VERIFY DATA.

MODE 0 CONSISTS OF A SINGLE PASS OPERATION, WRITING THE HEADERS AND DATA.

MODE 1 CONSISTS OF TWO PASSES. FIRST THE OPERATION OF MODE 2 IS DONE, THEN A SECOND PASS IS DONE WHERE ALL HEADERS ARE READ AND COMPARED.

MODE 2 PERFORMS THE SAME OPERATION AS MODE 0 AND MODE 1 WITH THE ADDITION OF WRITE CHECKING ALL DATA IN THE SECOND PASS.

MODE 3 AND MODE 4 ARE READ ONLY OPERATIONS. IN MODE 3 ONLY THE HEADERS ARE READ AND COMPARED. IN MODE 4 THE DATA IS ALSO READ.

4.2.5 EVEN CYLINDER PATTERN AND ODD CYLINDER PATTERN

THESE PARAMETERS DEFINE THE DATA WRITTEN IN ALL TRACKS AND SECTORS OF THE EVEN AND ODD NUMBERED CYLINDERS, RESPECTIVELY.

4.2.6 TRACK LIMITS & CYLINDER LIMITS

THESE PARAMETERS DEFINE THE BOUNDARIES WHERE THE DESIRED OPERATIONS ARE PERFORMED. THE TRACK LIMITS DEFINE INCLUSIVELY WHICH TRACKS ARE TO BE USED AND CYLINDER LIMITS DEFINE INCLUSIVELY WHICH CYLINDERS ARE TO BE USED. FOR EXAMPLE, TRACK LIMITS OF 1.1 AND CYLINDER LIMITS OF 0,632 IN MODE 0 WOULD WRITE ALL HEADERS ON TRACK 1 OF ALL CYLINDERS.

422
423
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

4.2.7 OFFSET

THE OFFSET VALUE SUPPLIED MUST BE IN THE RANGE OF 060-160. 060 IS THE MAXIMUM POSITIVE OFFSET AND 160 IS THE MAXIMUM NEGATIVE OFFSET. 000 AND 100 ARE BOTH OFFSET OF ZERO. REFER TO THE RK06 DISK DRIVE SPECIFICATION FOR THE SPECIFIC VALUE-OFFSET RELATIONSHIP. THE OFFSET WILL AFFECT ALL THE VERIFICATION OPERATIONS (VERIFY HEADERS AND VERIFY DATA).

4.3 BAD SECTOR OPTIONS

TWO OPTIONS ARE PROVIDED IN RESPECT TO BAD SECTOR HANDLING. THESE ARE:

- A. SPECIFYING SECTORS THAT ARE TO BE FLAGGED AS BAD AND
- B. PRESERVING THE ODD SOFTWARE DETECTED BAD SECTOR FILES.

4.3.1 SPECIFYING SECTORS THAT ARE TO BE FLAGGED BAD

THE FOLLOWING LINES IS TYPED ON THE CONSOLE:

"ANY SECTOR TO BE FLAGGED BAD?(TYPE Y OR N)(CR)"

IF "N" IS TYPED THE PROGRAM PROCEEDS TO THE QUESTION. IF "Y" IS TYPED THE FOLLOWING LINES ARE TYPED ON THE CONSOLE:

"ENTER OCTAL ADDRESS TO BE FLAGGED BAD IN FORMAT CCC,T,SS"
"ENTER ADDRESS OF (CR) TO TERMINATE"

THE ADDRESS TO BE FLAGGED BAD IS THEN ENTERED. CCC IS THE CYLINDER ADDRESS, T IS THE TRACK ADDRESS, AND SS IS THE SECTOR NUMBER. SEPARATING COMMAS ARE REQUIRED.

IF AN UNACCEPTABLE ADDRESS IS GIVEN (ILLEGAL IN ANY RESPECT) AN ERROR MESSAGE IS TYPED AND THE LINE "ENTER ADDRESS OR (CR) TO TERMINATE" IS REPEATED. IF THE ADDRESS IS ACCEPTABLE THE "ENTER ADDRESS OR (CR) TO TERMINATE" MESSAGE IS TYPED WITHOUT THE ERROR MESSAGE TO INDICATE ANOTHER ADDRESS MAY BE ENTERED.

TYPING THE (CR) TERMINATES THE BAD SECTOR ENTRY OPERATION.

4.3.2 PRESERVING SOFTWARE BAD SECTOR FILES

THE FOLLOWING LINE IS TYPED ON THE CONSOLE:

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

"PRESERVE SOFTWARE BAD SECTOR FILES?(TYPE Y OR N)(CR)"

IF 'N' IS TYPED THE PROGRAM PROCEEDS TO FORMAT THE CARTRIDGE AND ANY ENTRIES IN THE SDBSF ARE DISREGARDED.

IF 'Y' IS TYPED THE SDBSF ENTRIES ARE USED IN DETERMINING WHICH SECTORS ARE TO BE FLAGGED BAD. ANY SECTOR FOUND BAD IN THE FORMAT OPERATION OR SPECIFIED BAD AS IN 5.3.1 ARE ADDED TO THE FILE.

5.0 PROGRAM DESCRIPTION

AFTER THE PARAMETERS HAVE BEEN ENTERED AS DESCRIBED IN PARAGRAPH 4 AND 5, THE PROGRAM PERFORMS VARIOUS CHECKS ON THE PARAMETERS. INCLUDED IN THE CHECKS ARE CYLINDER AND TRACK VALUE CHECKS TO INSURE THE STARTING VALUES ARE LESS THAN OR EQUAL TO THE ENDING VALUES AND THAT ALL VALUES ARE LEGAL ADDRESSES.

THE PROGRAM THEN SETS UP TO FORMAT THE CARTRIDGE IN 20 OR 22 SECTORS/TRACK FORMAT. THIS REMAINS IN EFFECT FOR THE DURATION OF ALL OPERATIONS REQUIRED BY THE MODE SELECTED.

THE TRACK OF BAD SECTORS (CYLINDER 410, TRACK 2) IS READ. SECTOR 0, 2, 4, 6, OR 8 IS READ IF THE REQUESTED FORMAT IS 22 SECTORS/TRACK OR SECTOR 1, 3, 5, 7, OR 9 IS READ IF THE REQUESTED FORMAT IS 20 SECTORS/TRACK. ONE OF THE SECTORS MUST BE READ WITHOUT ERROR OR A MESSAGE IS TYPED OUT AND THE OPERATION STOPS.

ALL PACK WRITING IS DONE NEXT. THE HEADERS AND DATA ARE WRITTEN IN THE FIRST PASS OF HEADS ACROSS THE DISK SURFACE. CYLINDER 410, TRACK 2 IS NOT WRITTEN, PROTECTING THE BLOCKS OF BAD SECTOR INFORMATION. THE HEADER OF ANY SECTOR THAT IS KNOWN TO BE BAD (ENTERED IN THE FDBSF) IS WRITTEN WITH BIT 15 OF THE SECOND HEADER WORD RESET. IF THE SDBSF IS TO BE PRESERVED AND/OR IF ANY SECTORS WERE SPECIFIED TO BE MARKED BAD, THESE ARE WRITTEN WITH BIT 14 OF THE SECOND HEADER WORD RESET.

WHEN DATA IS BEING WRITTEN IT IS POSSIBLE TO GET OPERATION INCOMPLETE (OPI) OR HEADER CHECK (HVRC) ERRORS. IF THIS OCCURS ON A SECTOR THAT HAS NOT BEEN MARKED BAD AND THE SECTOR WHERE THE ERROR OCCURRED CANNOT BE SUCCESSFULLY WRITTEN IN 8 RETRIES, THE SECTOR IS MARKED BAD. ITS ADDRESS IS ENTERED IN THE SOFTWARE TABLE OF BAD SECTORS, BIT 14 IS RESET IN THE HVRC WORD OF THAT SECTOR HEADER, THE HEADERS ARE REWRITTEN, AND DATA WRITING PROCEEDS. HOWEVER, THE HVRC OR OPI ERROR MAY PERSIST. IF IT DOES, THE PROGRAM PRINTS A MESSAGE INDICATING THE PHYSICAL LOCATION OF A HEADER IS DAMAGED AND THE PROGRAM HALTS. THIS CARTRIDGE MUST BE CONSIDERED UNFORMATTABLE.

AFTER ALL WRITING IS COMPLETED, THE VERIFICATION PASS IS STARTED. THIS INVOLVES A SECOND PASS OF THE HEADS ACROSS THE DISK SURFACE. ALL HEADERS ON A TRACK ARE READ IN AND COMPARED WORD FOR WORD AGAINST SOFTWARE GENERATED HEADERS. THE GENERATED HEADERS TAKE ENTRIES IN THE BAD SECTOR BLOCKS, BOTH FACTORY WRITTEN AND SOFTWARE DETECTED, INTO

534
535
536
537
538
539
540
541
542
543
544
545
546
547
548
549
550
551
552
553
554
555
556
557
558
559
560
561
562
563
564
565
566
567
568
569
570
571
572
573
574
575
576
577
578
579
580
581
582
583
584
585
586
587
588
589

ACCOUNT. ANY MISCOMPARES ARE REPORTED.

IF THE DATA ON THIS PACK HAS JUST BEEN WRITTEN (MODE 2), A WRITE CHECK OPERATION IS STARTED. ANY OPI, HVRC, DCK (DATA CHECK), OR WCE (WRITE CHECK ERROR) OCCURRING ON A SECTOR THAT HAS NOT BEEN MARKED BAD IS RETRIED 8 TIMES. IF UNSUCCESSFUL AND IF IT IS AN OPI, HVRC, OR DCK WITH HARD ECC ERROR, THE SECTOR IS MARKED BAD. IF IT IS A WCE ERROR THE SECTOR IN QUESTION IS READ WHICH WILL CAUSE A DCK ERROR. IF THE DCK ERROR IS A HARD ECC ERROR, THE SECTOR IS ALSO MARKED BAD. IF THE ORIGINAL ERROR WAS DCK WITHOUT THE HARD ECC ERROR, THE ERROR IS REPORTED BUT THE SECTOR IS NOT MARKED BAD. IF A SECTOR HAS BEEN MARKED BAD (INSERTED IN THE SOFTWARE SELECTED BAD SECTOR TABLE) BIT 14 OF THE HVRC WORD OF THE HEADER IS RESET, THE HEADERS ARE REWRITTEN FOR THIS TRACK, THE DATA IS REWRITTEN, THE HEADERS ARE VERIFIED, AND THIS WRITE CHECK OPERATION IS PERFORMED AGAIN.

IF THE MODE SELECTED WAS 4, THE DATA VERIFY OPERATION IS DONE AS A READ, RELYING ON THE HARDWARE TO DETECT ERRORS. ANY OPI, HVRC, OR DCK ERROR THAT OCCURS AN A SECTOR THAT IS NOT MARKED BAD IS REPORTED. IF THE SECTOR IS MARKED BAD, NO ERROR IS REPORTED.

AFTER THE WRITE AND VERIFY OPERATIONS IT IS THEN DETERMINED IF THE SOFTWARE DETECTED BAD SECTOR BLOCKS ARE TO BE WRITTEN. IF PACK WRITING WAS DONE, THESE BLOCKS ARE WRITTEN. SECTOR 10, 12, 14, 16, 18, AND 20 ARE WRITTEN IF THE FORMAT IS 22 SECTORS/TRACK AND 11, 13, 15, 17, 19, AND 21 ARE WRITTEN IF THE FORMAT IS 20 SECTORS/TRACK. THIS OPERATION IS BYPASSED IF THE PACK HAS NOT BEEN WRITTEN.

THE PROGRAM WILL THEN PRINT A COMPLETE LIST OF ALL BAD SECTORS IF SWITCH 7 IS SET. THIS LIST INCLUDES BOTH THE BAD SECTORS AS DETECTED IN THE FACTORY AND THE BAD SECTORS DETECTED BY SOFTWARE.

IF SWITCH 2 IS SET, THE PROGRAM WILL REPEAT THE ENTIRE OPERATION WITH THE CURRENT PARAMETER SET.

6.0 ERROR REPORTING

THE GENERAL FORMAT OF ERROR REPORT FOR ERRORS DETECTED BY THE RK06 SUBSYSTEM IS:

ERROR MESSAGE

(COMMAND START VALUES)
(DRIVE) (CMND) (CYLNDR) (TRACK) (SECTOR) (WD CNT) (BUFFER)

(RK611 REGISTERS AND DRIVE STATUS)
(RKCS1) (RKCS2) (RKER) (RKDS) (RKWC) (RKDC) (RKDA) (RKBA)

(A 0) (B 0) (A 1) (B 1) (A 2) (B 2) (A 3) (B 3)

THE FORMAT FOR HEADER VERIFICATION ERRORS IS:

590
591
592
593
594
595
596
597
598
599
600
601
602
603
604
605
606
607
608
609
610
611
612
613
614
615
616
617
618
619
620
621
622
623
624
625
626

HEADER COMPARE ERROR
HEADER SHOULD BE:
WORD 1 WORD 2 WORD 3
BAD HEADER IS:
WORD 1 WORD 2 WORD 3

THE FORMAT FOR REPORTING THE DATA IN ERROR DURING A DATA VERIFICATION WITH A WRITE CHECK IS:

DATA MISCOMPARE
PACK ADDRESS OF ERROR IS:
CYLNDR TRACK SECTOR
GOOD BAD WORD NUM

THE FORMAT FOR REPORTING DATA CHECK DURING A DATA VERIFICATION WITH A READ DATA IS:

BAD DATA VERIFICATION WITH READ
PACK ADDRESS OF ERROR IS:
CYLNDR TRACK SECTOR

ECC DATA IS:
PAT POS CORRECTABLE?

IF CORRECTABLE? = 1, ERROR IS ECC CORRECTABLE.
IF CORRECTABLE? = 0, ERROR IS NOT ECC CORRECTABLE.

%

627
628
629
630
631
632
633
634
635
636
637
638
639
640
641
642
643
644
645
646
647
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

163000

000001

001100

000011

000012

000015

000200

177776

177774

177772

177570

177570

000000

000001

000002

000003

000004

000005

000006

```

:*** REV 004 ***
      .NLIST MC,MD,CND
      .LIST ME
      .ENABL ABS,AMA
      $SWR= 163000
.TITLE CZR6LDO RK06L CTG FMTR
:*COPYRIGHT (C) 1976,1981
:*DIGITAL EQUIPMENT CORP.
:*MAYNARD, MASS. 01754
:*
:*PROGRAM BY MARV TEGROTHUIS
:*
:*THIS PROGRAM WAS ASSEMBLED USING THE PDP-11 MAINDEC SYSMAC
:*PACKAGE (MAINDEC-11-DZQAC-C5), JAN, 1981.
:*
$TN=1
.SBTTL OPERATIONAL SWITCH SETTINGS
:*
:*      SWITCH          USE
:*      -----
:*      15             HALT ON ERROR
:*      14             LOOP ON TEST
:*      13             INHIBIT ERROR TYPEOUTS
:*      10             BELL ON ERROR
:*      9              LOOP ON ERROR
:*      7              PRINT LIST OF BAD SECTORS
:*      1              LIMIT DATA COMPARE ERROR REPORTS TO 10
:*      0              REPEAT ENTIRE FORMAT OPERATION
.SBTTL BASIC DEFINITIONS
:*INITIAL ADDRESS OF THE STACK POINTER *** 1100 ***
STACK= 1100
.EQUIV EMT,ERROR      ;;BASIC DEFINITION OF ERROR CALL
.EQUIV IOT,SCOPE     ;;BASIC DEFINITION OF SCOPE CALL
:*MISCELLANEOUS DEFINITIONS
HT= 11              ;;CODE FOR HORIZONTAL TAB
LF= 12              ;;CODE FOR LINE FEED
CR= 15              ;;CODE FOR CARRIAGE RETURN
CRLF= 200           ;;CODE FOR CARRIAGE RETURN-LINE FEED
PS= 177776         ;;PROCESSOR STATUS WORD
.EQUIV PS,PSW
STKLMT= 177774     ;;STACK LIMIT REGISTER
PIRQ= 177772       ;;PROGRAM INTERRUPT REQUEST REGISTER
DSWR= 177570       ;;HARDWARE SWITCH REGISTER
DDISP= 177570      ;;HARDWARE DISPLAY REGISTER
:*GENERAL PURPOSE REGISTER DEFINITIONS
R0= %0             ;;GENERAL REGISTER
R1= %1             ;;GENERAL REGISTER
R2= %2             ;;GENERAL REGISTER
R3= %3             ;;GENERAL REGISTER
R4= %4             ;;GENERAL REGISTER
R5= %5             ;;GENERAL REGISTER
R6= %6             ;;GENERAL REGISTER

```

683	000007	R7=	%7	::GENERAL REGISTER
684	000006	SP=	%6	::STACK POINTER
685	000007	PC=	%7	::PROGRAM COUNTER
686				
687		:*PRIORITY LEVEL DEFINITIONS		
688	000000	PR0=	0	::PRIORITY LEVEL 0
689	000040	PR1=	40	::PRIORITY LEVEL 1
690	000100	PR2=	100	::PRIORITY LEVEL 2
691	000140	PR3=	140	::PRIORITY LEVEL 3
692	000200	PR4=	200	::PRIORITY LEVEL 4
693	000240	PR5=	240	::PRIORITY LEVEL 5
694	000300	PR6=	300	::PRIORITY LEVEL 6
695	000340	PR7=	340	::PRIORITY LEVEL 7
696				
697		:*"SWITCH REGISTER" SWITCH DEFINITIONS		
698	100000	SW15=	100000	
699	040000	SW14=	40000	
700	020000	SW13=	20000	
701	010000	SW12=	10000	
702	004000	SW11=	4000	
703	002000	SW10=	2000	
704	001000	SW09=	1000	
705	000400	SW08=	400	
706	000200	SW07=	200	
707	000100	SW06=	100	
708	000040	SW05=	40	
709	000020	SW04=	20	
710	000010	SW03=	10	
711	000004	SW02=	4	
712	000002	SW01=	2	
713	000001	SW00=	1	
714		.EQUIV	SW09,SW9	
715		.EQUIV	SW08,SW8	
716		.EQUIV	SW07,SW7	
717		.EQUIV	SW06,SW6	
718		.EQUIV	SW05,SW5	
719		.EQUIV	SW04,SW4	
720		.EQUIV	SW03,SW3	
721		.EQUIV	SW02,SW2	
722		.EQUIV	SW01,SW1	
723		.EQUIV	SW00,SW0	
724				
725		:*DATA BIT DEFINITIONS (BIT00 TO BIT15)		
726	100000	BIT15=	100000	
727	040000	BIT14=	40000	
728	020000	BIT13=	20000	
729	010000	BIT12=	10000	
730	004000	BIT11=	4000	
731	002000	BIT10=	2000	
732	001000	BIT09=	1000	
733	000400	BIT08=	400	
734	000200	BIT07=	200	
735	000100	BIT06=	100	
736	000040	BIT05=	40	
737	000020	BIT04=	20	
738	000010	BIT03=	10	

```

739          000004          BIT02= 4
740          000002          BIT01= 2
741          000001          BIT00= 1
742          .EQUIV BIT09,BIT9
743          .EQUIV BIT08,BIT8
744          .EQUIV BIT07,BIT7
745          .EQUIV BIT06,BIT6
746          .EQUIV BIT05,BIT5
747          .EQUIV BIT04,BIT4
748          .EQUIV BIT03,BIT3
749          .EQUIV BIT02,BIT2
750          .EQUIV BIT01,BIT1
751          .EQUIV BIT00,BIT0
752
753          ;*BASIC "CPU" TRAP VECTOR ADDRESSES
754          000004          ERRVEC= 4          ;; TIME OUT AND OTHER ERRORS
755          000010          RESVEC= 10         ;; RESERVED AND ILLEGAL INSTRUCTIONS
756          000014          TBITVEC=14        ;; "T" BIT
757          000014          TRTVEC= 14         ;; TRACE TRAP
758          000014          BPTVEC= 14         ;; BREAKPOINT TRAP (BPT)
759          000020          IOTVEC= 20         ;; INPUT/OUTPUT TRAP (IOT) **SCOPE**
760          000024          PWRVEC= 24         ;; POWER FAIL
761          000030          EMTVEC= 30         ;; EMULATOR TRAP (EMT) **ERROR**
762          000034          TRAPVEC=34        ;; "TRAP" TRAP
763          000060          TKVEC= 60          ;; TTY KEYBOARD VECTOR
764          000064          TPVEC= 64          ;; TTY PRINTER VECTOR
765          000240          PIRQVEC=240       ;; PROGRAM INTERRUPT REQUEST VECTOR
766          .SBTTL TRAP CATCHER
767
768          000000          .=0
769          ;*ALL UNUSED LOCATIONS FROM 4 - 776 CONTAIN A ".+2,HALT"
770          ;*SEQUENCE TO CATCH ILLEGAL TRAPS AND INTERRUPTS
771          ;*LOCATION 0 CONTAINS 0 TO CATCH IMPROPERLY LOADED VECTORS
772          000174          .=174
773          000174          000000          DISPREG: .WORD 0          ;; SOFTWARE DISPLAY REGISTER
774          000176          000000          SWREG:   .WORD 0          ;; SOFTWARE SWITCH REGISTER
775          .SBTTL STARTING ADDRESS(ES)
776          000200          000137          016202          JMP @#START ;; JUMP TO STARTING ADDRESS OF PROGRAM
777          000204          000204          .=204
778          000204          000137          020154          JMP @#DRINIT
  
```



```
779 .SBTTL COMMON TAGS
780
781 ::*****
782 :*THIS TABLE CONTAINS VARIOUS COMMON STORAGE LOCATIONS
783 :*USED IN THE PROGRAM.
784
785 001100 .=1100
786 001100 $CMTAG: .WORD 0 ::START OF COMMON TAGS
787 001100 000000 $PASS: .WORD 0 ::CONTAINS PASS COUNT
788 001102 000 $STSTM: .BYTE 0 ::CONTAINS THE TEST NUMBER
789 001103 000 $ERFLG: .BYTE 0 ::CONTAINS ERROR FLAG
790 001104 000000 $ICNT: .WORD 0 ::CONTAINS SUBTEST ITERATION COUNT
791 001106 000000 $LPADR: .WORD 0 ::CONTAINS SCOPE LOOP ADDRESS
792 001110 000000 $LPERR: .WORD 0 ::CONTAINS SCOPE RETURN FOR ERRORS
793 001112 000000 $ERTTL: .WORD 0 ::CONTAINS TOTAL ERRORS DETECTED
794 001114 000 $ITEMB: .BYTE 0 ::CONTAINS ITEM CONTROL BYTE
795 001115 001 $ERMAX: .BYTE 1 ::CONTAINS MAX. ERRORS PER TEST
796 001116 000000 $ERRPC: .WORD 0 ::CONTAINS PC OF LAST ERROR INSTRUCTION
797 001120 000000 $GDADR: .WORD 0 ::CONTAINS ADDRESS OF 'GOOD' DATA
798 001122 000000 $BDADR: .WORD 0 ::CONTAINS ADDRESS OF 'BAD' DATA
799 001124 000000 $GDDAT: .WORD 0 ::CONTAINS 'GOOD' DATA
800 001126 000000 $BDDAT: .WORD 0 ::CONTAINS 'BAD' DATA
801 001130 000000 .WORD 0 ::RESERVED--NOT TO BE USED
802 001132 000000 .WORD 0
803 001134 000 $AUTOB: .BYTE 0 ::AUTOMATIC MODE INDICATOR
804 001135 000 $INTAG: .BYTE 0 ::INTERRUPT MODE INDICATOR
805 001136 000000 .WORD 0
806 001140 177570 SWR: .WORD DSWR ::ADDRESS OF SWITCH REGISTER
807 001142 177570 DISPLAY: .WORD DDISP ::ADDRESS OF DISPLAY REGISTER
808 001144 177560 $TKS: 177560 ::TTY KBD STATUS
809 001146 177562 $TKB: 177562 ::TTY KBD BUFFER
810 001150 177564 $TPS: 177564 ::TTY PRINTER STATUS REG. ADDRESS
811 001152 177566 $TPB: 177566 ::TTY PRINTER BUFFER REG. ADDRESS
812 001154 000 $NULL: .BYTE 0 ::CONTAINS NULL CHARACTER FOR FILLS
813 001155 002 $FILLS: .BYTE 2 ::CONTAINS # OF FILLER CHARACTERS REQUIRED
814 001156 012 $FILLC: .BYTE 12 ::INSERT FILL CHARS. AFTER A 'LINE FEED'
815 001157 000 $TPFLG: .BYTE 0 ::'TERMINAL AVAILABLE' FLAG (BIT<07>=0=YES)
816 001160 000000 $REGAD: .WORD 0 ::CONTAINS THE ADDRESS FROM
817 WHICH ($REGO) WAS OBTAINED
818 001162 000000 $REG0: .WORD 0 ::CONTAINS (($REGAD)+0)
819 001164 000000 $REG1: .WORD 0 ::CONTAINS (($REGAD)+2)
820 001166 000000 $REG2: .WORD 0 ::CONTAINS (($REGAD)+4)
821 001170 000000 $REG3: .WORD 0 ::CONTAINS (($REGAD)+6)
822 001172 000000 $REG4: .WORD 0 ::CONTAINS (($REGAD)+10)
823 001174 000000 $REG5: .WORD 0 ::CONTAINS (($REGAD)+12)
824 001176 000000 $REG6: .WORD 0 ::CONTAINS (($REGAD)+14)
825 001200 000000 $REG7: .WORD 0 ::CONTAINS (($REGAD)+16)
826 001202 000000 $REG10: .WORD 0 ::CONTAINS (($REGAD)+20)
827 001204 000000 $REG11: .WORD 0 ::CONTAINS (($REGAD)+22)
828 001206 000000 $REG12: .WORD 0 ::CONTAINS (($REGAD)+24)
829 001210 000000 $REG13: .WORD 0 ::CONTAINS (($REGAD)+26)
830 001212 000000 $REG14: .WORD 0 ::CONTAINS (($REGAD)+30)
831 001214 000000 $REG15: .WORD 0 ::CONTAINS (($REGAD)+32)
832 001216 000000 $REG16: .WORD 0 ::CONTAINS (($REGAD)+34)
833 001220 000000 $REG17: .WORD 0 ::CONTAINS (($REGAD)+36)
834 001222 000000 $REG20: .WORD 0 ::CONTAINS (($REGAD)+40)
```

835	001224	000000		\$REG21: .WORD	0	::CONTAINS ((\$REGAD)+42)
836	001226	000000		\$REG22: .WORD	0	::CONTAINS ((\$REGAD)+44)
837	001230	000000		\$REG23: .WORD	0	::CONTAINS ((\$REGAD)+46)
838	001232	000000		\$REG24: .WORD	0	::CONTAINS ((\$REGAD)+50)
839	001234	000000		\$REG25: .WORD	0	::CONTAINS ((\$REGAD)+52)
840	001236	000000		\$TMP0: .WORD	0	::USER DEFINED
841	001240	000000		\$TMP1: .WORD	0	::USER DEFINED
842	001242	000000		\$TMP2: .WORD	0	::USER DEFINED
843	001244	000000		\$TMP3: .WORD	0	::USER DEFINED
844	001246	000000		\$TMP4: .WORD	0	::USER DEFINED
845	001250	000000		\$TMP5: .WORD	0	::USER DEFINED
846	001252	000000		\$TMP6: .WORD	0	::USER DEFINED
847	001254	000000		\$TMP7: .WORD	0	::USER DEFINED
848	001256	000000		\$TMP10: .WORD	0	::USER DEFINED
849	001260	000000		\$ESCAPE:0		::ESCAPE ON ERROR ADDRESS
850	001262	177607	000377	\$BELL: .ASCIZ	<207><377><377>	::CODE FOR BELL
851	001266	077		\$QUES: .ASCII	/?/	::QUESTION MARK
852	001267	015		\$CRLF: .ASCII	<15>	::CARRIAGE RETURN
853	001270	000012		\$LF: .ASCIZ	<12>	::LINE FEED
854				::*****		::*****

855
856
857
858
859
860
861
862
863
864
865
866
867
868
869
870
871
872
873
874
875
876
877
878
879
880
881
882
883
884
885
886
887
888
889
890
891
892
893
894
895
896
897
898
899
900
901
902
903
904
905
906
907
908
909
910

.SBTTL ERROR POINTER TABLE

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

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

\$ERRTB:

;	ERROR 1	;	UNIBUS PARITY ERROR
EM1			
DH100			
DT100			
DF01			
;	ERROR 2	;	NON-EXISTANT MEMORY
EM2			
DH100			
DT100			
DF01			
;	ERROR 3	;	NON-EXISTANT DRIVE
EM3			
DH100			
DT100			
DF01			
;	ERROR 4	;	UNIT FIELD ERROR
EM4			
DH100			
DT100			
DF01			
;	ERROR 5	;	SUBSYSTEM TIMEOUT
EM5			
DH100			
DT100			
DF02			
;	ERROR 6	;	SERCON PARITY ERROR
EM6			
DH100			
DT100			
DF03			
;	ERROR 7	;	DRIVE DETECTED PARITY ERROR
EM7			
DH100			
DT100			

001272

001272 043012
001274 046533
001276 050032
001300 050130

001302 043042
001304 046533
001306 050032
001310 050130

001312 043100
001314 046533
001316 050032
001320 050130

001322 043135
001324 046533
001326 050032
001330 050130

001332 043162
001334 046533
001336 050032
001340 050154

001342 043210
001344 046533
001346 050032
001350 050204

001352 043243
001354 046533
001356 050032

911	001360	050204	DF03	
912				
913			:ERROR 10	
914	001362	043303	EM10	:AC LOW
915	001364	046533	DH100	
916	001366	050032	DT100	
917	001370	050154	DF02	
918				
919			:ERROR 11	
920	001372	043316	EM11	:SPEED LOSS
921	001374	046533	DH100	
922	001376	050032	DT100	
923	001400	050154	DF02	
924				
925			:ERROR 12	
926	001402	043335	EM12	:ILLEGAL FUNCTION
927	001404	046533	DH100	
928	001406	050032	DT100	
929	001410	050154	DF02	
930				
931			:ERROR 13	
932	001412	043370	EM13	:PROGRAMMING ERROR
933	001414	046533	DH100	
934	001416	050032	DT100	
935	001420	050130	DF01	
936				
937			:ERROR 14	:NON-EXISTANT FUNCTION
938	001422	043416	EM14	
939	001424	046533	DH100	
940	001426	050032	DT100	
941	001430	050154	DF02	
942				
943			:ERROR 15	
944	001432	043456	EM15	:DRIVE TYPE ERROR
945	001434	046533	DH100	
946	001436	050032	DT100	
947	001440	050154	DF02	
948				
949			:ERROR 16	
950	001442	043503	EM16	:FORMAT ERROR
951	001444	046533	DH100	
952	001446	050032	DT100	
953	001450	050154	DF02	
954				
955			:ERROR 17	
956	001452	043524	EM17	:WRITE LOCK ERROR
957	001454	046533	DH100	
958	001456	050032	DT100	
959	001460	050154	DF02	
960				
961			:ERROR 20	
962	001462	043551	EM20	:DRIVE UNSAFE
963	001464	046533	DH100	
964	001466	050032	DT100	
965	001470	050154	DF02	
966				

967			:ERROR 21	
968	001472	043600	EM21	:SEEK INCOMPLETE
969	001474	046533	DH100	
970	001476	050032	DT100	
971	001500	050154	DF02	
972				
973			:ERROR 22	
974	001502	043632	EM22	:CYLINDER OVERFLOW
975	001504	046533	DH100	
976	001506	050032	DT100	
977	001510	050154	DF02	
978				
979			:ERROR 23	
980	001512	043666	EM23	:ILLEGAL CYLINDER
981	001514	046533	DH100	
982	001516	050032	DT100	
983	001520	050154	DF02	
984				
985			:ERROR 24	
986	001522	043731	EM24	:DRIVE OFF TRACK
987	001524	046533	DH100	
988	001526	050032	DT100	
989	001530	050154	DF02	
990				
991			:ERROR 25	
992	001532	043755	EM25	:DRIVE TIMING ERROR
993	001534	046533	DH100	
994	001536	050032	DT100	
995	001540	050154	DF02	
996				
997			:ERROR 26	
998	001542	044004	EM26	:DATA LATE
999	001544	046533	DH100	
1000	001546	050032	DT100	
1001	001550	050154	DF02	
1002				
1003			:ERROR 27	
1004	001552	044030	EM27	:CONTROLLER TIMEOUT
1005	001554	046533	DH100	
1006	001556	050032	DT100	
1007	001560	050154	DF02	
1008				
1009			:ERROR 30	
1010	001562	044065	EM30	:OPERATION INCOMPLETE
1011	001564	046533	DH100	
1012	001566	050032	DT100	
1013	001570	050240	DF05	
1014				
1015			:ERROR 31	
1016	001572	044124	EM31	:HEADER CRC ERROR
1017	001574	046533	DH100	
1018	001576	050032	DT100	
1019	001600	050270	DF06	
1020				
1021			:ERROR 32	
1022	001602	044151	EM32	:DATA CHECK ERROR

1023	001604	046533	DH100	
1024	001606	050032	DT100	
1025	001610	050320	DF07	
1026				
1027			:ERROR 33	
1028	001612	044176	EM33	:WRITE CHECK ERROR
1029	001614	046533	DH100	
1030	001616	050032	DT100	
1031	001620	050350	DF10	
1032				
1033			:ERROR 34	
1034	001622	044224	EM34	:DATA MISCOMPARE
1035	001624	047327	DH604	
1036	001626	050114	DT601	
1037	001630	050370	DF11	
1038				
1039			:ERROR 35	
1040	001632	044250	EM35	:NO DRIVE RESPONSE-UFE & NED
1041	001634	046533	DH100	
1042	001636	050032	DT100	
1043	001640	050130	DF01	
1044				
1045			:ERROR 36	
1046	001642	044310	EM36	:DRIVE ERROR WILL NOT CLEAR
1047	001644	000000	0	
1048	001646	000000	0	
1049	001650	000000	0	
1050				
1051			:ERROR 37	
1052	001652	044347	EM37	:DRIVE STATUS CHANGE WILL NOT CLEAR
1053	001654	000000	0	
1054	001656	000000	0	
1055	001660	000000	0	
1056				
1057			:ERROR 40	
1058	001662	044416	EM40	:ATTENTION BUT NO STATUS CHANGE OR FAULT
1059	001664	046533	DH100	
1060	001666	050032	DT100	
1061	001670	050154	DF02	
1062				
1063			:ERROR 41	
1064	001672	044472	EM41	:ATTENTION BUT DRIVE NOT AVAILABLE
1065	001674	046533	DH100	
1066	001676	050032	DT100	
1067	001700	050154	DF02	
1068				
1069			:ERROR 42	
1070	001702	044540	EM42	:ATTENTION WHEN NOT EXPECTED
1071	001704	046533	DH100	
1072	001706	050032	DT100	
1073	001710	050154	DF02	
1074				
1075			:ERROR 43	
1076	001712	044600	EM43	:ERROR WHILE GATHERING DRIVE STATUS
1077	001714	046533	DH100	
1078	001716	050032	DT100	

1079	001720	050404	DF12	
1080				
1081			:ERROR 44	
1082	001722	045126	EM63	:CLEAR CONTROLLER DID NOT CLEAR ERROR
1083	001724	046533	DH100	
1084	001726	050032	DT100	
1085	001730	050404	DF12	
1086				
1087			:ERROR 45	
1088	001732	045177	EM64	:NO ATTENTION IN ATTENTION SUMMARY REG
1089	001734	046533	DH100	
1090	001736	050032	DT100	
1091	001740	050404	DF12	
1092				
1093			:ERROR 46	
1094	001742	045256	EM65	:UNSOLICITED ATTENTION
1095	001744	046533	DH100	
1096	001746	050032	DT100	
1097	001750	050404	DF12	
1098				
1099			:ERROR 47	
1100	001752	045310	EM66	:UNEXPECTED DATA TYPE ERROR
1101	001754	046533	DH100	
1102	001756	050032	DT100	
1103	001760	050404	DF12	
1104				
1105			:ERROR 50	
1106	001762	045347	EM67	:ATTENTION DID NOT RESET WITH CLEAR
1107	001764	046533	DH100	
1108	001766	050032	DT100	
1109	001770	050404	DF12	
1110				
1111			:ERROR 51	
1112	001772	045416	EM70	:SUBSYSTEM CLEAR DID NOT CLEAR DRIVE ATTENTION
1113	001774	046533	DH100	
1114	001776	050032	DT100	
1115	002000	050404	DF12	
1116				
1117			:ERROR 52	
1118	002002	044647	EM52	:MULTIPLE DRIVE SELECT
1119	002004	046533	DH100	
1120	002006	050032	DT100	
1121	002010	050404	DF12	
1122				
1123			:ERROR 53	
1124	002012	044701	EM53	:ABREVIATED HCE ERROR
1125	002014	046533	DH100	
1126	002016	050032	DT100	
1127	002020	050430	DF13	
1128				
1129			:ERROR 54	
1130	002022	044065	EM30	:OPERATION INCOMPLETE ERROR
1131	002024	046533	DH100	
1132	002026	050032	DT100	
1133	002030	050454	DF14	
1134				

1135			:ERROR 55	
1136	002032	044124	EM31	:ABREVIATED HCRC ERROR
1137	002034	046533	DH100	
1138	002036	050032	DT100	
1139	002040	050430	DF13	
1140				
1141			:ERROR 56	
1142	002042	044732	EM56	:2 TIMEOUT ERROR
1143	002044	046533	DH100	
1144	002046	050032	DT100	
1145	002050	050500	DF15	
1146				
1147			:ERROR 57	:2ND LEVEL IN SUBSYSTEM TIMEOUT
1148	002052	044732	EM56	
1149	002054	046533	DH100	
1150	002056	050032	DT100	
1151	002060	050540	DF16	
1152				
1153			:ERROR 60	
1154	002062	044760	EM60	:ERROR IN RECAL FOR RECOVERY
1155	002064	000000	0	
1156	002066	000000	0	
1157	002070	000000	0	
1158				
1159			:ERROR 61	
1160	002072	045026	EM61	:ABORT MESSAGE
1161	002074	000000	0	
1162	002076	000000	0	
1163	002100	000000	0	
1164				
1165			:ERROR 62	
1166	002102	045100	EM62	:HEADER MISCOMPARE
1167	002104	047212	DH601	
1168	002106	050114	DT601	
1169	002110	050600	DF17	
1170				
1171			:ERROR 63	:DATA ERROR WORDS
1172	002112	000000	0	
1173	002114	000000	0	
1174	002116	050122	DT602	
1175	002120	050640	DF25	
1176				
1177			:ERROR 64	
1178	002122	045126	EM63	:CLEAR CONTROLLER DID NOT CLEAR ERROR
1179	002124	046533	DH100	
1180	002126	050032	DT100	
1181	002130	050154	DF02	
1182				
1183			:ERROR 65	
1184	002132	045177	EM64	:NO ATTENTION IN ATTENTION SUMMARY REG
1185	002134	046533	DH100	
1186	002136	050032	DT100	
1187	002140	050154	DF02	
1188				
1189			:ERROR 66	
1190	002142	045256	EM65	:UNSOLICITED ATTENTION

1191	002144	046533	DH100	
1192	002146	050032	DT100	
1193	002150	050154	DF02	
1194				
1195			:ERROR 67	
1196	002152	045310	EM66	:UNEXPECTED DATA TYPE ERROR
1197	002154	046533	DH100	
1198	002156	050032	DT100	
1199	002160	050154	DF02	
1200				
1201			:ERROR 70	
1202	002162	045347	EM67	:ATTENTION DID NOT RESET WITH CLEAR
1203	002164	046533	DH100	
1204	002166	050032	DT100	
1205	002170	050154	DF02	
1206				
1207			:ERROR 71	
1208	002172	045416	EM70	:SUBSYSTEM CLEAR DID NOT CLEAR ATT
1209	002174	046533	DH100	
1210	002176	050032	DT100	
1211	002200	050154	DF02	
1212				
1213			:ERROR 72	
1214	002202	045500	EM71	:DATA LATE WHEN UNLOADING HEADER
1215	002204	046533	DH100	
1216	002206	050032	DT100	
1217	002210	050154	DF02	
1218				
1219			:ERROR 73	
1220	002212	045544	EM72	:CONTROLLER ERROR DURING DRIVER SERVICE
1221	002214	046533	DH100	
1222	002216	050032	DT100	
1223	002220	050154	DF02	
1224				
1225			:ERROR 74	
1226	002222	045617	EM73	:DRIVE DETECTED PARITY ERROR
1227	002224	046533	DH100	
1228	002226	050032	DT100	
1229	002230	050154	DF02	
1230				
1231			:ERROR 75	
1232	002232	045657	EM74	:UNDEFINED ERROR
1233	002234	046533	DH100	
1234	002236	050032	DT100	
1235	002240	050154	DF02	
1236				
1237			:ERROR 76	
1238	002242	045703	EM75	:MARKING SECTOR BAD MESSAGE
1239	002244	000000	0	
1240	002246	000000	0	
1241	002250	000000	0	
1242				
1243			:ERROR 77	
1244	002252	045737	EM76	:BAD DATA VERIFICATION WITH READ
1245	002254	047426	DH605	
1246	002256	050114	DT601	

1247	002260	050620	DF21	
1248				
1249			:ERROR 100	
1250	002262	046032	EM77	:RETRY SUCCESSFUL MESSAGE
1251	002264	047761	DH800	
1252	002266	050114	DT601	
1253	002270	050630	DF23	
1254				
1255			:ERROR 101	
1256	002272	046032	EM77	:ANOTHER RETRY SUCCESSFUL MESSAGE
1257	002274	047761	DH800	
1258	002276	050114	DT601	
1259	002300	050630	DF23	
1260				
1261			:ERROR 102	
1262	002302	046057	EM100	:RETRY UNSUCCESSFUL MESSAGE
1263	002304	047761	DH800	
1264	002306	050114	DT601	
1265	002310	050630	DF23	
1266				
1267			:ERROR 103	
1268	002312	046106	EM101	:NO VALID HEADERS IN TRACK JUST READ
1269	002314	047410	DH6042	
1270	002316	050114	DT601	
1271	002320	050634	DF24	
1272				
1273			:ERROR 104	
1274	002322	046170	EM102	:BSE ERROR ON SECTOR NOT LISTED AS BAD
1275	002324	046533	DH100	
1276	002326	050032	DT100	
1277	002330	050154	DF02	
1278				
1279			:ERROR 105	
1280	002332	046246	EM103	:WORD COUNT NOT CORRECT TO CONTINUE
1281	002334	050004	DH900	
1282	002336	050122	DT602	
1283	002340	050640	DF25	
1284				
1285			:ERROR 106	
1286	002342	046312	EM104	:DRIVE STATUS NOT VALID
1287	002344	000000	0	
1288	002346	000000	0	
1289	002350	000000	0	
1290				
1291			:ERROR 107	
1292	002352	046377	EM105	:BAD HEADER AREA
1293	002354	047327	DH604	
1294	002356	050114	DT601	
1295	002360	050644	DF26	

```
1296
1297
1298
1299          000000          RKCS1= 0          ;CONTROL AND STATUS REGISTER 1
1300          000002          RKWC= 2          ;WORD COUNT REGISTER
1301          000004          RKBA= 4          ;BUS ADDRESS REGISTER
1302          000006          RKDA= 6          ;DESIRED TRACK SECTOR REGISTER
1303          000010          RKCS2= 10         ;CONTROL AND STATUS REGISTER 2
1304          000012          RKDS= 12         ;DRIVE STATUS REGISTER
1305          000014          RKER= 14         ;ERROR REGISTER
1306          000016          RKASOF= 16        ;ATTENTION SUMMARY AND OFFSET REGISTER
1307          000020          RKDC= 20         ;DESIRED CYLINDER REGISTER
1308          000020          RKDCYL= 20        ;DESIRED CYLINDER REGISTER
1309          000024          RKDB= 24         ;DATA BUFFER
1310          000026          RKMR1= 26        ;MAINTENANCE REGISTER 1
1311          000034          RKMR2= 34        ;MAINTENANCE REGISTER 2
1312          000036          RKMR3= 36        ;MAINTENANCE REGISTER 3
1313          000030          RKPOS= 30        ;ECC POSITION INFORMATION
1314          000030          RKECPS= 30       ;ECC POSITION INFORMATION
1315          000032          RKPAT= 32       ;ECC PATTERN INFORMATION
1316          000032          RKECPT= 32      ;ECC PATTERN INFORMATION
1317
1318          .SBTTL  DRIVE COMMANDS
1319
1320          000101          SELDRV= 101        ;SELECT DRIVE
1321          000103          PACK= 103        ;PACK ACKNOWLEDGE
1322          000105          CLEAR= 105       ;DRIVE CLEAR
1323          000107          UNLOAD= 107      ;UNLOAD
1324          000111          SRTSPL= 111      ;START SPINDLE
1325          000113          RECAL= 113       ;RECALIBRATE
1326          000115          OFFSET= 115     ;OFFSET
1327          000117          SEEK= 117       ;SEEK
1328          000121          RDDATA= 121     ;READ DATA
1329          000123          WRDATA= 123     ;WRITE DATA
1330          000125          RDHEAD= 125     ;READ HEADER
1331          000127          WRHEAD= 127     ;WRITE HEADER AND DATA
1332          000131          WRTCHK= 131     ;WRITE CHECK
1333
1334          ;          THE FOLLOWING ARE NOT DRIVE COMMANDS BUT ARE USED BY THE DRIVER
1335          ;          TO SIMULATE A SPECIFIC DESIRED OPERATION
1336
1337          000140          RELEAS= 140      ;RELEASE DRIVE
1338          000141          RDSTAT= 141     ;GET ALL STATUS FROM DRIVE
1339          000164          RDALHD= 164     ;READ ALL HEADERS
1340          000176          CONCLR= 176     ;CONTROLLER CLEAR (BIT 15 OF CS1)
1341          000177          SUBCLR= 177     ;SUBSYSTEM CLEAR (BIT 5 OF CS2)
1342          000300          INTR= 300      ;GENERATE INTERRUPT TO CPU
1343
1344          ;          DRIVER ISSUED SERVICE COMMANDS
1345
1346          000001          DR.SEL= 001     ;DRIVE SELECT
1347          000005          DR.CLR= 005     ;DRIVE CLEAR
1348
1349          .SBTTL  CONTROL AND STATUS REGISTER 1 BITS
1350
1351          000001          GO= BIT0          ;GO BIT
```

1352	000100	IF=	BIT6	: INTERRUPT ENABLE
1353	000200	RDY=	BIT7	: CONTROLLER READY
1354	000400	BA16=	BIT8	: BUS ADDRESS BIT 16
1355	001000	BA17=	BIT9	: BUS ADDRESS BIT 17
1356	002000	CDT=	BIT10	: CONTROLLER DRIVE TYPE (0=RK06,1=RK07)
1357	004000	CTO=	BIT11	: CONTROLLER TIMED OUT WAITING FOR : DRIVE RESPONSE
1358				
1359	010000	CFMT=	BIT12	: CONTROLLER DRIVE FORMAT (0=22 SECTOR, 1=20 SECTOR)
1360	020000	SPAR=	BIT13	: DRIVE BUS PARITY ERROR DETECTED BY CONTROLLER
1361	040000	DI=	BIT14	: DRIVE INTERRUPT
1362	100000	CERR=	BIT15	: CONTROLLER ERROR
1363	100000	CCLR=	BIT15	: CONTROLLER CLEAR
1364				
1365		:		
1366		:		THESE BIT DEFINITIONS ARE USED FOR ADDRESS THE HIGH BYTE OF RKCS1
1367				
1368	000001	B.BA16=	BIT0	: BUS ADDRESS BIT 16
1369	000002	B.BA17=	BIT1	: BUS ADDRESS BIT 17
1370	000004	B.CDT=	BIT2	: CONTROLLER DRIVE TYPE (0=RK06,1=RK07)
1371	000020	B.CFMT=	BIT4	: CONTROLLER DRIVE FORMAT (0=22 SECTOR, 1=20 SECTOR)
1372				
1373		.SBTTL	CONTROL AND STATUS REGISTER 2 BITS	
1374				
1375	000007	DRVMSK=	7	: MASK FOR DRIVE SELECTION CODE
1376	000010	DESL=	BIT3	: DESELECT OR RELEASE DRIVE IN BITS 0-2
1377	000010	RLS=	BIT3	: DESELECT OR RELEASE DRIVE IN BITS 0-2
1378	000020	BAI=	BIT4	: BUS ADDRESS INCREMENT INHIBIT
1379	000040	CLR=	BIT5	: CLEAR CONTROLLER AND ALL DRIVES
1380	000040	SCLR=	BIT5	: CLEAR CONTROLLER AND ALL DRIVES
1381	000100	IR=	BIT6	: INPUT READY
1382	000200	OR=	BIT7	: OUTPUT READY
1383	000400	UFE=	BIT8	: UNIT FIELD ERROR
1384	001000	MDS=	BIT9	: MULTIPLE DRIVE SELECT
1385	002000	PGE=	BIT10	: PROGRAMMING ERROR
1386	004000	NEM=	BIT11	: NON-EXISTENT MEMORY
1387	010000	NED=	BIT12	: NON-EXISTENT DRIVE
1388	020000	UPE=	BIT13	: UNIBUS PARITY ERROR
1389	040000	WCE=	BIT14	: WRITE CHECK ERROR
1390	100000	DLT=	BIT15	: DATA LATE ERROR
1391				
1392		.SBTTL	ERROR REGISTER BIT DEFINITION	
1393				
1394	000001	ILC=	BIT0	: ILLEGAL FUNCTION CODE
1395		:*ILF=	BIT0	: ILLEGAL FUNCTION CODE
1396	000002	SKI=	BIT1	: SEEK INCOMPLETE
1397	000004	ILF=	BIT2	: ILLEGAL DRIVE FUNCTION
1398	000004	NXF=	BIT2	: ILLEGAL DRIVE FUNCTION
1399	000010	DRPAR=	BIT3	: DRIVE DETECTED DRIVE BUS PARITY ERROR
1400	000020	FMTE=	BIT4	: FORMAT ERROR
1401	000040	DTYE=	BIT5	: DRIVE TYPE ERROR
1402	000100	ECH=	BIT6	: ECC HARD
1403	000200	BSE=	BIT7	: BAD SECTOR ERROR
1404	000400	HCRC=	BIT8	: HEADER CRC ERROR
1405	000400	HVRC=	BIT8	: HEADER VRC ERROR
1406	001000	COE=	BIT9	: CYLINDER ADDRESS OVERFLOW ERROR
1407	002000	IDAE=	BIT10	: INVALID DISK ADDRESS ERROR

1408	004000	WLE=	BIT11	:WRITE LOCK ERROR
1409	010000	DTE=	BIT12	:DRIVE TIMING ERROR
1410	020000	OPI=	BIT13	:OPERATION (SEARCH) INCOMPLETE
1411	040000	UNS=	BIT14	:DRIVE UNSAFE
1412	100000	DCK=	BIT15	:DATA CHECK
1413				
1414		.SBTTL	STATUS REGISTER BIT DEFINITION	
1415				
1416	000001	DRA=	BIT0	:DRIVE AVAILABLE (CONTROLLER IS SET IF : THIS BIT IS RESET)
1417				
1418	000004	OFST=	BIT2	:DRIVE OFFSET
1419	000010	ACLO=	BIT3	:AC LOW
1420	000020	SPDLSS=	BIT4	:SPEED LOSS
1421	000020	DCLO=	BIT4	:DC LOW
1422	000040	DROT=	BIT5	:DRIVE OFF TRACK
1423	000100	VV=	BIT6	:VOLUME VALID
1424	000200	DRY=	BIT7	:DRIVE READY
1425	000200	DRDY=	BIT7	:DRIVE READY
1426	000400	DDT=	BIT8	:DRIVE TYPE (0=RK06,1=RK07)
1427	004000	WRL=	BIT11	:WRITE LOCK
1428	020000	PIP=	BIT13	:POSITIONING IN PROGRESS
1429	040000	DSC=	BIT14	:DRIVE STATUS CHANGE
1430	100000	SVAL=	BIT15	:STATUS VALID
1431				
1432		.SBTTL	MAINTENANCE REGISTER 1 BIT DEFINITION	
1433				
1434	000017	MESMSK=	17	:MESSAGE MASK
1435				
1436	000020	PAT=	BIT4	:FORCE EVEN PARITY ON DRIVE BUS MESSAGE LINES
1437	000040	DMD=	BIT5	:DIAGNOSTIC MODE
1438	000100	MSP=	BIT6	:MAINTENANCE SECTOR PULSE
1439	000200	MIND=	BIT7	:MAINTENANCE INDEX
1440	000400	MCLK=	BIT8	:MAINTENANCE CLOCK
1441	001000	MERD=	BIT9	:MAINTENANCE ENCODED READ DATA
1442	002000	MEWD=	BIT10	:MAINTENANCE ENCODED WRITE DATA
1443	004000	PCA=	BIT11	:PRECOMPENSATION ADVANCE
1444	010000	PCD=	BIT12	:PRECOMPENSATION DELAY
1445	020000	ECCW=	BIT13	:ECC WORD IS BEING READ OR WRITTEN
1446	040000	WRTGAT=	BIT14	:WRITE GATE
1447	100000	RDGATE=	BIT15	:READ GATE
1448				
1449		.SBTTL	DEFINITION OF DRIVE STATUS BYTE 00 MESSAGE A	
1450				
1451	000040	S.DRA=	BIT5	:DRIVE AVAILIABLE
1452	000100	S.VV=	BIT6	:VOLUME VALID
1453	000200	S.DRY=	BIT7	:DRIVE READY
1454	000400	S.TYPE=	BIT8	:DRIVE TYPE
1455	001000	S.FORM=	BIT9	:DRIVE FORMAT
1456	002000	S.OFF=	BIT10	:OFFSET
1457	004000	S.WRL=	BIT11	:WRITE LOCK
1458	010000	S.SPIN=	BIT12	:SPINDLE ON
1459	020000	S.PIP=	BIT13	:POSITIONING IN PROGRESS
1460	040000	S.DSC=	BIT14	:DRIVE STATUS CHANGE
1461				
1462		.SBTTL	DEFINITION OF DRIVE STATUS BYTE 00 MESSAGE B	
1463				

1464	000040	S.ICYL= BIT5	:ILLEGAL CYLINDER ADDRESS
1465	000100	S.ACLO= BIT6	:AC LOW
1466	000200	S.FLT= BIT7	:DRIVE FAULT
1467	000400	S.ILF= BIT8	:ILLEGAL FUNCTION
1468	001000	S.PAR= BIT9	:DRIVE DETECTED DRIVE BUS PARITY ERROR
1469	002000	S.SKI= BIT10	:SEEK INCOMPLETE
1470	004000	S.WLE= BIT11	:WRITE LOCK ERROR
1471	010000	S.SPLS= BIT12	:SPEED LOSS
1472	010000	S.DCLO= BIT12	:DC LOW
1473	020000	S.DROT= BIT13	:DRIVE OFF TRACK
1474	040000	S.UNS= BIT14	:DRIVE UNSAFE

.SBTTL DEFINITION OF DRIVE STATUS BYTE 01 MESSAGE A

1478	000020	S.XDOK= BIT4	:TRANSDUCER OK
1479	000040	S.HDHM= BIT5	:HEADS HOME
1480	000100	S.BRHM= BIT6	:BRUSHES HOME
1481	000200	S.DOOR= BIT7	:DOOR INTERLOCKED
1482	000400	S.CART= BIT8	:CARTRAGE INTERLOCK
1483	001000	S.SPOK= BIT9	:SPEED OK
1484	002000	S.FWD= BIT10	:FORWARD
1485	004000	S.REV= BIT11	:REVERSE
1486	010000	S.LOAD= BIT12	:HEADS LOADING
1487	020000	S.RTZ= BIT13	:RETURN TO ZERO
1488	040000	S.UNLD= BIT14	:HEADS UNLOADING

.SBTTL DEFINITION OF DRIVE STATUS BYTE 01 MESSAGE B

1492	000020	S.SECT= BIT4	:SECTOR ERROR
1493	000040	S.WCLK= BIT5	:WRITE CLOCK AND NO WRITE GATE
1494	000100	S.WGAT= BIT6	:WRITE GATE AND NO TRANSISTIONS
1495	000200	S.HDFL= BIT7	:HEAD FAULT
1496	000400	S.MHD= BIT8	:MULTIPLE HEAD SELECT
1497	001000	S.XERR= BIT9	:INDEX ERROR
1498	002000	S.DIB= BIT10	:DIBIT ERROR
1499	004000	S.PLO= BIT11	:PLO ERROR
1500	010000	S.NMOV= BIT12	:SEEK AND NO MOTION
1501	020000	S.LIMD= BIT13	:LIMIT DETECT ON SEEK
1502	040000	S.BRKE= BIT14	:SERVO-BRAKE

.SBTTL COMMON MASKS

1506	000007	M.DRV= 7	:DRIVE CODE
1507	100000	M.PAR= BIT15	:PARITY
1508	000003	M.ID= 3	:BYTE ID
1509	017760	M.CDIF= 17760	:CYLINDER DIFFERENCE/OFFSET
1510	017760	M.CADD= 17760	:CYLINDER ADDRESS
1511	077770	M.SER= 77770	:DRIVE SERIAL NUMBER
1512	000760	M.SECT= 760	:SECTOR COUNT
1513	007000	M.HEAD= 7000	:HEAD DECODE

```

1514 .SBTTL PARAMETER BLOCK ALLOCATION
1515
1516 : * *****
1517 : * 1 : COMMAND : DRIVE NO. : 0
1518 : * 3 : CYLINDER ADDRESS : 2
1519 : * 5 : TRACK : SECTOR : 4
1520 : * 7 : BA16-17,FORMAT,DRV TYPE: OFFSET : 6
1521 : * 11 : BUS ADDRESS (LOW 16 BITS) : 10
1522 : * 13 : WORD COUNT (2'S COMPLEMENT) : 12
1523 : * 15 : PROGRAM DRIVE STATUS INFORMATION : 14
1524 : * 17 : COMMAND AND STATUS REGISTER 1 : 16
1525 : * 21 : COMMAND AND STATUS REGISTER 2 : 20
1526 : * 23 : WORD COUNT REGISTER : 22
1527 : * 25 : BUS ADDRESS REGISTER : 24
1528 : * 27 : DESIRED TRACK AND SECTOR : 26
1529 : * 31 : DESIRED CYLINDER : 30
1530 : * 33 : ATTENTION SUMMARY AND DRIVE OFFSET : 32
1531 : * 35 : ERROR REGISTER : 34
1532 : * 37 : STATUS REGISTER : 36
1533 : * 41 : MESSAGE LINE A STATUS BYTE 00 : 40
1534 : * 43 : MESSAGE LINE B STATUS BYTE 00 : 42
1535 : * 45 : MESSAGE LINE A STATUS BYTE 01 : 44
1536 : * 47 : MESSAGE LINE B STATUS BYTE 01 : 46
1537 : * 51 : MESSAGE LINE A STATUS BYTE 10 : 50
1538 : * 53 : MESSAGE LINE B STATUS BYTE 10 : 52
1539 : * 55 : MESSAGE LINE A STATUS BYTE 11 : 54
1540 : * 57 : MESSAGE LINE B STATUS BYTE 11 : 56
1541 : * 61 : ECC POSITION INFORMATION : 60
1542 : * 63 : ECC PATTERN INFORMATION : 62
1543 : * *****
  
```

```

1544
1545 .SBTTL PARAMETERS PASSED TO THE DRIVER
1546
1547 : THE FOLLOWING DEFINITIONS ARE USED TO PASS PARAMETERS
1548 : TO THE RK06/RK07 DRIVER
1549
  
```

```

1550 000000 P.DRVN= 0 ;DRIVE NUMBER
1551 000001 P.CMND= 1 ;COMMAND
1552 000002 P.CYLN= 2 ;CYLINDER ADDRESS
1553 000004 P.SECT= 4 ;SECTOR
1554 000005 P.TRCK= 5 ;TRACK
1555 000006 P.OFSI= 6 ;OFFSET
1556 000007 P.CSIH= 7 ;RKCS1 BITS 8-15
1557 000007 P.BAHI= 7 ;BUS ADDRESS (BITS 16 AND 17)
1558 000010 P.BALO= 10 ;BUS ADDRESS (BITS 0-15)
1559 000012 P.WC= 12 ;WORD COUNT (2'S COMPLEMENT)
1560 000014 P.PRST= 14 ;PROGRAM DRIVE STATUS INFORMATION
  
```

```

1561
1562 .SBTTL PROGRAM DEVICE STATUS REGISTER DEFINITION
1563
1564 000001 DRVUSE= BIT0 ;DRIVE IN USE
1565 000002 DRVPOS= BIT1 ;DRIVE POSITIONING
1566 000004 DRVPDT= BIT2 ;DRIVE POSITIONED FOR DATA TRANSFER
1567 000010 UEXATT= BIT3 ;UNEXPECTED ATTENTION
1568 000020 DRVHRD= BIT4 ;DRIVE HAS HARD ERROR
1569 000040 DRVDSC= BIT5 ;DRIVE STATUS CHANGE DID NOT CLEAR
  
```

```

1570      000100      CMDTO= BIT6      ;NO TERMINATION TO COMMAND FOR AT
1571      ;          ;          ;          ;          ;          ;          ;
1572      000200      W.WCK= BIT7      ;WRITE FOR WRITE WRITE CHECK
1573      000400      NOCHK= BIT8      ;NO CHECK, DO NOT SET INTERRUPT ENABLE
1574      001000      PBSVAL= BIT9      ;PARAMETER STATUS WORDS VALID
1575      ;          ;          ;          ;          ;          ;
1576      ;          ;          ;          ;          ;          ;
1577      002000      DRPDRV= BIT10     ;DROP DRIVE FROM TEST SEQUENCE
1578      004000      NODSC= BIT11     ;ATTENTION SET BUT DCS AND FAULT RESET
1579      010000      DRVSZD= BIT12     ;DRIVE SEIZED BY OTHER PORT
1580      020000      E.UNLD= BIT13     ;DRIVE UNLOADED DUE TO ERROR
1581      040000      Q.INIT= BIT14     ;PARAMETER BLOCK ENQUEUED IN INITIATION QUEUE
1582      100000      DTBAII= BIT15     ;INHIBIT BUS ADDRESS INCREMENT
1583

```

.SBTTL PARAMETERS PASSED FROM DRIVER TO PROGRAM

; THE FOLLOWING DEFINITIONS ARE USED FOR REGISTER RETURNS
; FROM THE DRIVER TO THE CALLING PROGRAM

```

1589      000016      P.CS1= 16      ;COMMAND AND STATUS REGISTER 1
1590      000020      P.CS2= 20      ;COMMAND AND STATUS REGISTER 2
1591      000022      P.WCR= 22      ;WORD COUNT REGISTER
1592      000024      P.BAR= 24      ;BUS ADDRESS REGISTER
1593      000026      P.DTS= 26      ;DESIRED TRACK SECTOR REGISTER
1594      000030      P.DCYL= 30      ;DESIRED CYLINDER REGISTER
1595      000032      P.ASOF= 32      ;ATTENTION SUMMARY/OFFSET REGISTER
1596      000034      P.ER= 34      ;ERROR REGISTER
1597      000036      P.DS= 36      ;STATUS REGISTER
1598      000040      P.A00= 40      ;MESSAGE A STATUS BYTE 00
1599      000042      P.B00= 42      ;MESSAGE B STATUS BYTE 00
1600      000044      P.A01= 44      ;MESSAGE A STATUS BYTE 01
1601      000046      P.B01= 46      ;MESSAGE B STATUS BYTE 01
1602      000050      P.A10= 50      ;MESSGGE A STATUS BYTE 10
1603      000052      P.B10= 52      ;MESSAGE B STATUS BYTE 10
1604      000054      P.A11= 54      ;MESSAGE A STATUS BYTE 11
1605      000056      P.B11= 56      ;MESSAGE B STATUS BYTE 11
1606      000060      P.EPOS= 60      ;ECC POSITION INFORMATION
1607      000062      P.EPAT= 62      ;ECC PATTERN INFORMATION
1608

```

.SBTTL PARAMETER BLOCK 0 FOR DRIVE

```

1611      002362      000      PARM0: .BYTE 0      ;DRIVE NUMBER
1612      002363      000      ;.BYTE 0      ;COMMAND
1613      002364      000000    ;.WORD 0      ;CYLINDER ADDRESS
1614      002366      000      ;.BYTE 0      ;SECTOR ADDRESS
1615      002367      000      ;.BYTE 0      ;TRACK ADDRESS
1616      002370      000      ;.BYTE 0      ;OFFSET VALUE
1617      002371      000      ;.BYTE 0      ;BUS ADDRESS (BITS 16 AND 17)
1618      002372      000000    ;.WORD 0      ;BUS ADDRESS (BITS 0 - 15)
1619      002374      000000    ;.WORD 0      ;WORD COUNT (2'S COMPLEMENT)
1620      002376      000000    ;.WORD 0      ;PROGRAM DRIVE STATUS INFORMATION
1621      002400      000000    ;.WORD 0      ;COMMAND AND STATUS REGISTER 1
1622      002402      000000    ;.WORD 0      ;COMMAND AND STATUS REGISTER 2
1623      002404      000000    ;.WORD 0      ;WORD COUNT REGISTER
1624      002406      000000    ;.WORD 0      ;BUS ADDRESS REGISTER
1625      002410      000000    ;.WORD 0      ;DESIRED TRACK AND SECTOR REGISTER

```


1626	002412	000000	.WORD	0	:DESIRED CYLINDER REGISTER
1627	002414	000000	.WORD	0	:ATTENTION SUMMARY/OFFSET REGISTER
1628	002416	000000	.WORD	0	:ERROR REGISTER
1629	002420	000000	.WORD	0	:STATUS REGISTER
1630	002422	000000	.WORD	0	:MESSAGE LINE A STATUS BYTE 00
1631	002424	000000	.WORD	0	:MESSAGE LINE B STATUS BYTE 00
1632	002426	000000	.WORD	0	:MESSAGE LINE A STATUS BYTE 01
1633	002430	000000	.WORD	0	:MESSAGE LINE B STATUS BYTE 01
1634	002432	000000	.WORD	0	:MESSAGE LINE A STATUS BYTE 10
1635	002434	000000	.WORD	0	:MESSAGE LINE B STATUS BYTE 10
1636	002436	000000	.WORD	0	:MESSAGE LINE A STATUS BYTE 11
1637	002440	000000	.WORD	0	:MESSAGE LINE B STATUS BYTE 11
1638	002442	000000	.WORD	0	:ECC POSITION INFORMATION
1639	002444	000000	.WORD	0	:ECC PATTERN INFORMATION

1640
 1641 .SBTTL PARAMETER BLOCK 1 FOR DRIVE

1642					
1643	002446	000	PARM1: .BYTE	0	:DRIVE NUMBER
1644	002447	000	.BYTE	0	:COMMAND
1645	002450	000000	.WORD	0	:CYLINDER ADDRESS
1646	002452	000	.BYTE	0	:SECTOR ADDRESS
1647	002453	000	.BYTE	0	:TRACK ADDRESS
1648	002454	000	.BYTE	0	:OFFSET VALUE
1649	002455	000	.BYTE	0	:BUS ADDRESS (BITS 16 AND 17)
1650	002456	000000	.WORD	0	:BUS ADDRESS (BITS 0 - 15)
1651	002460	000000	.WORD	0	:WORD COUNT (2'S COMPLEMENT)
1652	002462	000000	.WORD	0	:PROGRAM DRIVE STATUS INFORMATION
1653	002464	000000	.WORD	0	:COMMAND AND STATUS REGISTER 1
1654	002466	000000	.WORD	0	:COMMAND AND STATUS REGISTER 2
1655	002470	000000	.WORD	0	:WORD COUNT REGISTER
1656	002472	000000	.WORD	0	:BUS ADDRESS REGISTER
1657	002474	000000	.WORD	0	:DESIRED TRACK AND SECTOR REGISTER
1658	002476	000000	.WORD	0	:DESIRED CYLINDER REGISTER
1659	002500	000000	.WORD	0	:ATTENTION SUMMARY/OFFSET REGISTER
1660	002502	000000	.WORD	0	:ERROR REGISTER
1661	002504	000000	.WORD	0	:STATUS REGISTER
1662	002506	000000	.WORD	0	:MESSAGE LINE A STATUS BYTE 00
1663	002510	000000	.WORD	0	:MESSAGE LINE B STATUS BYTE 00
1664	002512	000000	.WORD	0	:MESSAGE LINE A STATUS BYTE 01
1665	002514	000000	.WORD	0	:MESSAGE LINE B STATUS BYTE 01
1666	002516	000000	.WORD	0	:MESSAGE LINE A STATUS BYTE 10
1667	002520	000000	.WORD	0	:MESSAGE LINE B STATUS BYTE 10
1668	002522	000000	.WORD	0	:MESSAGE LINE A STATUS BYTE 11
1669	002524	000000	.WORD	0	:MESSAGE LINE B STATUS BYTE 11
1670	002526	000000	.WORD	0	:ECC POSITION INFORMATION
1671	002530	000000	.WORD	0	:ECC PATTERN INFORMATION

1672
 1673 .SBTTL TEMPORARY CONTROLLER REGISTER STORAGE

1674					
1675	002532	000000	T.CS1: .WORD	0	:TEMPORARY STORAGE FOR COMMAND AND STATUS REGISTER 1
1676					
1677	002534	000000	T.CS2: .WORD	0	:TEMPORARY STORAGE FOR COMMAND AND STATUS REGISTER 2
1678					
1679	002536	000000	T.WCR: .WORD	0	:TEMPORARY STORAGE FOR WORD COUNT REGISTER
1680	002540	000000	T.BA: .WORD	0	:TEMPORARY STORAGE FOR BUS ADDRESS REGISTER
1681	002542	000000	T.DA: .WORD	0	:TEMPORARY STORAGE FOR DISK TRACK AND SECTOR

1682	002544	000000	T.DC:	.WORD	0	: TEMPORARY STORAGE FOR DRIVE CYLINDER
1683	002546	000000	T.ASOF:	.WORD	0	: TEMPORARY STORAGE FOR ATTENTION SUMMARY
1684						: AND OFFSET
1685	002550	000000	T.ER:	.WORD	0	: TEMPORARY STORAGE FOR ERROR REGISTER
1686	002552	000000	T.DS:	.WORD	0	: TEMPORARY STORAGE FOR DRIVE STATUS REGISTER
1687	002554	000000	T.MR1:	.WORD	0	: TEMPORARY STORAGE FOR MAINTENANCE REGISTER 1
1688	002556	000000	T.MR2:	.WORD	0	: TEMPORARY STORAGE FOR MAINTENANCE REGISTER 2
1689	002560	000000	T.MR3:	.WORD	0	: TEMPORARY STORAGE FOR MAINTENANCE REGISTER 3
1690	002562	000000	T.POS:	.WORD	0	: TEMPORARY STORAGE FOR ECC POSITION
1691	002564	000000	T.PAT:	.WORD	0	: TEMPORARY STORAGE FOR ECC PATTERN
1692	002566	000000	T.DB:	.WORD	0	: TEMPORARY STORAGE FOR DATA BUFFER REGISTER
1693						
1694			.SBTTL	DRIVER	PARAMETERS	
1695						
1696	002570	177440	RKBAS:	.WORD	177440	: ADDRESS OF RK611 UNIBUS ADDRESS BLOCK
1697	002572	000210	RKVEC:	.WORD	210	: ADDRESS OF R611 VECTOR
1698	002574	000240	RKPRI:	.WORD	PR5	: RK611 INTERRUPT PRIORITY
1699	002576	025524	A.NORM:	ERRFRE		: ADDRESS OF NORMAL RETURN FROM DRIVER
1700	002600	026462	A.ABNL:	ERRHDL		: ADDRESS OF ABNORMAL RETURN FROM DRIVER
1701	002602	026040	A.CONT:	CONERR		: ADDRESS OF CONTROLLER ERROR RETURN
1702	002604	000000	E.CONT:	.WORD	0	: CONTROLLER ERROR STATUS
1703						: THIS LOCATION IS CLEARED WHEN EVERY COMMAND
1704						: IS INITIATED. IF A CONTROLLER ERROR
1705						: OCCURS THE FOLLOWING BIT ASSIGNMENT IS
1706						: USED:
1707						
1708		000001	E.CCLR=	BIT0		: CLEAR CONTROLLER DID NOT CLEAR ERROR
1709		000002	E.NOAT=	BIT1		: NO ATTENTION IN ATTENTION SUMMARY REG
1710		000004	E.UATT=	BIT2		: UNSOLICATED ATTENTION (SEQUENTIAL ONLY)
1711		000010	E.UDAT=	BIT3		: UNEXPECTED DATA TYPE ERROR
1712		000020	E.CLAT=	BIT4		: ATTENTION DID NOT RESET WITH CLEAR
1713		000040	E.SCLR=	BIT5		: SUBSYSTEM CLEAR DID NOT CLEAR DRIVE
1714						: ATTENTION
1715		000100	E.ILLD=	BIT6		: ILLEGAL DRIVER COMMAND
1716		000400	E.DLT=	BIT8		: DATA LATE WHEN UNLOADING HEADER
1717		001000	E.CERR=	BIT9		: CONTROLLER ERROR DURING DRIVER SERVICING
1718		002000	E.DPAR=	BIT10		: DRIVE DETECTED PARITY ERROR
1719		040000	E.CMTO=	BIT14		: CONTROLLER COMMAND TIME OUT (QUEUED ONLY)
1720		100000	E.MDS=	BIT15		: MULTIPLE DRIVE SELECT
1721						
1722	002606	000000	O.WAIT:	.WORD	0	: PARAMETER BLOCK OF THE DRIVE
1723						: WAITING FOR COMMAND COMPLETION
1724	002610	000400	W.MTIM:	.WORD	400	: LOOP COUNTER FOR MILLISECOND SCAN OF DRIVE
1725	002612	000400	W.MILI:	.WORD	400	: 16 MILLISECOND TIME FOR PROGRAM
1726						
1727						
1728						
1729						
1730						
1731						
1732						
1733						
1734						
1735						
1736						
1737						

CPU	VALUE
---	-----
11/05	100
11/10	
11/20	
11/34	
11/40	
11/45	400
11/50	
11/70	

```

1738 002614 000300          W.SEC:  .WORD  300          ;SECOND COUNT COUNT FOR ALL COMMANDS
1739                                     ; EXCEPT START SPINDLE
1740 002616 003000          W.8SEC:  .WORD  3000         ;8 SECOND FOR DRIVE CYCLE DOWN
1741 002620 030000          W.MIN:   .WORD  30000        ;MINUTE TIME FOR START SPINDLE
1742 002622 000000          HDR.AD:  .WORD   0          ;ADDRESS USED FOR READ ALL HEADERS
1743 002624 000000          HDR.CT:  .WORD   0          ;NUMBER OF HEADERS LEFT TO READ FOR READ
1744                                     ; ALL HEADERS
1745 002626      000          I.ISRL:  .BYTE   0          ;INTERRUPT OR RELEASED COMMAND ISSUED
1746 002627      002          H.HEAD:  .BYTE  2,4,10        ;HEAD DECODES
1747 002632      000          W.TIME:  .BYTE   0          ;DRIVES BEING WATCH-DOG TIMED
1748
1749          .SBTTL  INTERRUPT MASKS
1750
1751 002633      000          INTMSK:  .BYTE   0          ;INTERRUPT MASKS FOR DRIVE IN PARAMETER BLOCK
1752
1753          ;      INTERRUPT MASK TABLE
1754
1755 002634      001          I.DRV:   .BYTE   1          ;INTERRUPT MASK FOR DRIVE 0
1756 002635      002          .BYTE   2          ;INTERRUPT MASK FOR DRIVE 1
1757 002636      004          .BYTE   4          ;INTERRUPT MASK FOR DRIVE 2
1758 002637      010          .BYTE  10          ;INTERRUPT MASK FOR DRIVE 3
1759 002640      020          .BYTE  20          ;INTERRUPT MASK FOR DRIVE 4
1760 002641      040          .BYTE  40          ;INTERRUPT MASK FOR DRIVE 5
1761 002642      100          .BYTE 100          ;INTERRUPT MASK FOR DRIVE 6
1762 002643      200          .BYTE 200          ;INTERRUPT MASK FOR DRIVE 7
1763
1764          .SBTTL  PARAMETER BLOCK TABLE
1765
1766 002644 002362          PBLKT:  PARM0          ;ADDRESS OF PARAMETER BLOCK GIVEN WITH
1767                                     ;DRIVE CALL. MUST BE LOADED INTO PBLKT
1768
1769
1770          .SBTTL  TIME FOR WATCH-DOG TIMER
1771
1772 002646 000000          W.DRV:   .WORD   0          ;TIME FOR INSTRUCTION IN PARAMETER BLOCK
1773          .SBTTL  PROGRAM SPECIFIC RESERVED LOCATIONS
1774 002650      000          DERCNT:  .BYTE   0          ;DATA ERROR COUNT
1775 002651      000          OPCOMP:  .BYTE   0          ;OPERATION COMPLETE FLAG
1776 002652      000          DONE:   .BYTE   0          ;DONE SWITCH
1777 002653      000          TYPFMT:  .BYTE   0          ;DRIVE TYPE & FORMAT CONTROL
1778 002654      000          ERRCNT:  .BYTE   0          ;ERROR COUNT
1779 002655      004          ERRLMF:  .BYTE   4          ;ERROR LIMIT
1780 002656      000          OPCONT:  .BYTE   0          ;OPERATION CONTROL SWITCHES
1781          WHDSW=BIT0          ;WRITE HEADER & DATA SWITCH
1782          VFHDSW=BIT1         ;VERIFY HEADERS SWITCH
1783          WCDASW=BIT2         ;WRITE CHECK DATA SWITCH
1784          RCDASW=BIT3         ;READ CHECK DATA SWITCH
1785          OREQSW=BIT4         ;OFFSET REQUIRED SWITCH
1786          PSDBSF= BIT5         ;SAVE SOFTWARE DETECTED BAD SECTOR FILES SWITCH
1787          .EVEN
1788
1789 002660 000400          RDBUF:   .BLKW  ^D256        ;READ BUFFER
1790 003660 000400          BSSOFT:  .BLKW  ^D256        ;RECORD OF BAD SECTORS FROM SOFTWARE
1791 004660 000400          BSFACT:  .BLKW  ^D256        ;RECORD OF BAD SECTORS FROM FACTORY
1792 005660 000006          COMSTR:  .BLKW   6          ;COMMAND STORAGE
1793 005674 000102          BUFFO:  .BLKW  ^D66        ;OUTPUT BUFFER 1

```

1794	006100	000102			BUFF1: .BLKW	^D66		:OUTPUT BUFFER 2
1795	006304	000000			BUFPRT: .WORD	0		:BUFFER POINTER
1796	006306	000000			TKWDCT: .WORD	0		:FULL TRACK WORD COUNT STORAGE
1797	006310	000006			INITTP: .WORD	6		:DEFAULT DRIVE TYPE TO RK06
1798	006312	000000			INITDR: .WORD	0		:DEFAULT DRIVE
1799	006314	000026			INITSE: .WORD	26		:DEFAULT SECTOR/TRACK
1800	006316	000002			INITMD: .WORD	2		:DEFAULT MODE
1801	006320	135143			INITWE: .WORD	135143		:DEFAULT DATA FOR EVEN CYL
1802	006322	072307			INITWO: .WORD	072307		:DEFAULT DATA FOR ODD CYL
1803	006324	000000			INITST: .WORD	0		:DEFAULT STARTING TRACK
1804	006326	000002			INITET: .WORD	2		:DEFAULT ENDING TRACK
1805	006330	000000			INITSC: .WORD	0		:DEFAULT STARTING CYLINDER
1806	006332	000000			INITEC: .WORD	0		:DEFAULT END CYLINDER, LOADED IN PARM2
1807	006334	000000			INITOF: .WORD	0		:DEFAULT OFFSET (+0)
1808	006336	000000			TPINUS: .WORD	0		:DRIVE TYPE IN USE
1809	006340	000000			DRINUS: .WORD	0		:DRIVE IN USE
1810	006342	000000			SEINUS: .WORD	0		:SECTOR/TRACK IN USE
1811	006344	000000			MDINUS: .WORD	0		:MODE IN USE
1812	006346	000000			WEINUS: .WORD	0		:DATA IN USE FOR EVEN CYL
1813	006350	000000			WOINUS: .WORD	0		:DATA IN USE FOR ODD CYL
1814	006352	000000			STINUS: .WORD	0		:STARTING TRACK IN USE
1815	006354	000000			ETINUS: .WORD	0		:ENDING TRACK IN USE
1816	006356	000000			SCINUS: .WORD	0		:STARTING CYLINDER IN USE
1817	006360	000000			ECINUS: .WORD	0		:END CYLINDER IN USE
1818	006362	000000			OFINUS: .WORD	0		:OFFSET VALUE IN USE
1819	006364	000000			RECODE: .WORD	0		:RECOVERY CODE WORD
1820	006366	000000			CTINUS: .WORD	0		:CURRENT TRACK IN USE
1821	006370	000000			CCINUS: .WORD	0		:CURRENT CYLINDER IN USE
1822	006372	000000			ERRCOM: .WORD	0		:ERROR COMMAND
1823		000002			BSERR=BIT1			:BSE ERROR
1824		000004			HVRCER=BIT2			:HVRC ERROR
1825		000010			OPIERR=BIT3			:OPI ERROR
1826		000020			DCKERR=BIT4			:DATA CHECK ERROR
1827		000040			ECCNC=BIT5			:ECC NON-CORRECTABLE
1828		000100			WCERR=BIT6			:WRITE CHECK ERROR
1829		000200			ABORT=BIT7			:ABORT
1830		000400			LEV2ER=BIT8			:LEVEL TWO ERROR
1831		001000			BADSEC=BIT9			:BAD SECTOR FLAG
1832		002000			TWOTOS=BIT10			:TWO TIME OUTS
1833		004000			RCLREQ=BIT11			:RECALIBRATE REQUIRED
1834		100000			ANYDER=BIT15			:ANY ERROR DETECTED FLAG
1835								
1836	006374	037477	000		QUES: .ASCII	/??/		
1837	006377	015	044012	046105	HELPFL: .ASCII	<15><12>/HELP FILE/<15><12><12>		
1838	006404	020120	044506	042514				
1839	006412	005015	012					
1840	006415	115	052517	052116	.ASCII	/MOUNT AN RK06K-RK07K DISK CARTRIDGE/<15><12>		
1841	006422	040440	020116	045522				
1842	006430	033060	026513	045522				
1843	006436	033460	020113	044504				
1844	006444	045523	041440	051101				
1845	006452	051124	042111	042507				
1846	006460	005015						
1847	006462	042522	042523	020124	.ASCII	/RESET WRITE LOCK IF WRITE HEADERS OR DATA OPERATION/<15><12>		
1848	006470	051127	052111	020105				
1849	006476	047514	045503	044440				

1850	006504	020106	051127	052111
1851	006512	020105	042510	042101
1852	006520	051105	020123	051117
1853	006526	042040	052101	020101
1854	006534	050117	051105	052101
1855	006542	047511	006516	012
1856	006547	122	051505	047520
1857	006554	042116	052040	020117
1858	006562	044124	020105	047506
1859	006570	046114	053517	047111
1860	006576	020107	040520	040522
1861	006604	042515	042524	020122
1862	006612	042522	052521	051505
1863	006620	051524	041040	035131
1864	006626	005015	047503	052116
1865	006634	047522	020114	020132
1866	006642	057050	024532	024040
1867	006650	051103	020051	047524
1868	006656	040440	046114	053517
1869	006664	040440	046114	047440
1870	006672	020122		
1871	006674	042522	040515	047111
1872	006702	047111	020107	040520
1873	006710	040522	042515	042524
1874	006716	051522	052040	020117
1875	006724	042504	040506	046125
1876	006732	020124	043101	042524
1877	006740	020122	051104	053111
1878	006746	020105	054524	042520
1879	006754	005015		
1880	006756	040503	051122	040511
1881	006764	042507	051040	052105
1882	006772	051125	020116	047524
1883	007000	042040	043105	052501
1884	007006	052114	052040	040510
1885	007014	020124	050123	041505
1886	007022	043111	041511	050040
1887	007030	052101	046501	052105
1888	007036	051105	005015	
1889	007042	047503	052116	047522
1890	007050	020114	020103	057050
1891	007056	024503	052040	020117
1892	007064	042522	052524	047122
1893	007072	052040	020117	044506
1894	007100	051522	020124	040520
1895	007106	040522	042515	042524
1896	007114	020122	042522	052521
1897	007122	051505	006524	012
1898	007127	103	047117	051124
1899	007134	046117	041440	024040
1900	007142	041536	020051	046101
1901	007150	047523	040440	047502
1902	007156	052122	020123	044124
1903	007164	020105	051120	043517
1904	007172	040522	020115	047117
1905	007200	042503	040	

.ASCII /RESPOND TO THE FOLLOWING PARAMETER REQUESTS BY:/

.ASCII <15><12>/CONTROL Z (^Z) (CR) TO ALLOW ALL OR /

.ASCII /REMAINING PARAMETERS TO DEFAULT AFTER DRIVE TYPE/<15><12>

.ASCII /CARRIAGE RETURN TO DEFAULT THAT SPECIFIC PARAMETER/<15><12>

.ASCII /CONTROL C (^C) TO RETURN TO FIRST PARAMETER REQUEST/<15><12>

.ASCII /CONTROL C (^C) ALSO ABORTS THE PROGRAM ONCE /

1906	007203	106	051117	040515	.ASCII /FORMATTING HAS STARTED/<15><12><12>
1907	007210	052124	047111	020107	
1908	007216	040510	020123	052123	
1909	007224	051101	042524	006504	
1910	007232	005012			
1911	007234	051117	005015	012	.ASCII /OR/<15><12><12>
1912	007241	105	052116	051105	.ASCII /ENTER THE DESIRED PARAMETER OPTION SELECTED /
1913	007246	052040	042510	042040	
1914	007254	051505	051111	042105	
1915	007262	050040	051101	046501	
1916	007270	052105	051105	047440	
1917	007276	052120	047511	020116	
1918	007304	042523	042514	052103	
1919	007312	042105	040		
1920	007315	106	047522	020115	.ASCII /FROM THE LIST BELOW./<15><12>
1921	007322	044124	020105	044514	
1922	007330	052123	041040	046105	
1923	007336	053517	006456	012	
1924	007343	101	046114	053040	.ASCII /ALL VALUES TO BE ENTERED ARE OCTAL/<15><12><12>
1925	007350	046101	042525	020123	
1926	007356	047524	041040	020105	
1927	007364	047105	042524	042522	
1928	007372	020104	051101	020105	
1929	007400	041517	040524	006514	
1930	007406	005012			
1931	007410	051104	053111	020105	.ASCII /DRIVE TYPE= 6 FOR RK06, 7 FOR RK07/<15><12>
1932	007416	054524	042520	020075	
1933	007424	020066	047506	020122	
1934	007432	045522	033060	020054	
1935	007440	020067	047506	020122	
1936	007446	045522	033460	005015	
1937	007454	051104	053111	020105	.ASCII /DRIVE NUM=(0-7) DEFAULTS TO 0/<15><12><12>
1938	007462	052516	036515	030050	
1939	007470	033455	020051	042504	
1940	007476	040506	046125	051524	
1941	007504	052040	020117	006460	
1942	007512	005012			
1943	007514	042523	052103	051117	.ASCII &SECTORS/TRACK=(24,26) DEFAULTS TO 26&<15><12><12>
1944	007522	027523	051124	041501	
1945	007530	036513	031050	026064	
1946	007536	033062	020051	042504	
1947	007544	040506	046125	051524	
1948	007552	052040	020117	033062	
1949	007560	005015	012		
1950	007563	115	042117	036505	.ASCII /MODE=(0-4) DEFAULTS TO 2/<15><12>
1951	007570	030050	032055	020051	
1952	007576	042504	040506	046125	
1953	007604	051524	052040	020117	
1954	007612	006462	012		
1955	007615	040	020040	030040	.ASCII / 0=WRITE HEADERS & DATA/<15><12>
1956	007622	053475	044522	042524	
1957	007630	044040	040505	042504	
1958	007636	051522	023040	042040	
1959	007644	052101	006501	012	
1960	007651	040	020040	030440	.ASCII / 1=WRITE HEADERS & DATA; VERIFY HEADERS/<15><12>
1961	007656	053475	044522	042524	

1962	007664	044040	040505	042504
1963	007672	051522	023040	042040
1964	007700	052101	035501	053040
1965	007706	051105	043111	020131
1966	007714	042510	042101	051105
1967	007722	006523	012	
1968	007725	040	020040	031040
1969	007732	053475	044522	042524
1970	007740	023040	053040	051105
1971	007746	043111	020131	042510
1972	007754	042101	051105	020123
1973	007762	020046	040504	040524
1974	007770	005015		
1975	007772	020040	020040	036463
1976	010000	042526	044522	054506
1977	010006	044040	040505	042504
1978	010014	051522	005015	
1979	010020	020040	020040	036464
1980	010026	042526	044522	054506
1981	010034	044040	040505	042504
1982	010042	051522	040440	042116
1983	010050	053040	051105	043111
1984	010056	020131	040504	040524
1985	010064	005015	012	
1986	010067	105	042526	020116
1987	010074	054503	044514	042116
1988	010102	051105	050040	052101
1989	010110	042524	047122	024075
1990	010116	040504	040524	053440
1991	010124	051117	024504	042040
1992	010132	043105	052501	052114
1993	010140	020123	047524	030440
1994	010146	032463	032061	006463
1995	010154	012		
1996	010155	117	042104	041440
1997	010162	046131	047111	042504
1998	010170	020122	040520	052124
1999	010176	051105	036516	042050
2000	010204	052101	020101	047527
2001	010212	042122	020051	042504
2002	010220	040506	046125	051524
2003	010226	052040	020117	033460
2004	010234	031462	033460	005015
2005	010242	051124	041501	020113
2006	010250	044514	044515	051524
2007	010256	036440	024040	026460
2008	010264	026062	026460	024462
2009	010272	042040	043105	052501
2010	010300	052114	020123	047524
2011	010306	030040	031054	005015
2012	010314	054503	044514	042116
2013	010322	051105	046040	046511
2014	010330	052111	020123	020075
2015	010336	030050	033055	031063
2016	010344	030054	033055	031063
2017	010352	020051	042504	040506

.ASCII / 2=WRITE & VERIFY HEADERS & DATA/<15><12>

.ASCII / 3=VERIFY HEADERS/<15><12>

.ASCII / 4=VERIFY HEADERS AND VERIFY DATA/<15><12><12>

.ASCII /EVEN CYLINDER PATTERN=(DATA WORD) DEFAULTS TO 135143/<15><12>

.ASCII /ODD CYLINDER PATTERN=(DATA WORD) DEFAULTS TO 072307/<15><12>

.ASCII /TRACK LIMITS = (0-2,0-2) DEFAULTS TO 0,2/<15><12>

.ASCII /CYLINDER LIMITS = (0-632,0-632) DEFAULTS TO 0,632/<15><12><12>

2018	010360	046125	051524	052040
2019	010366	020117	026060	031466
2020	010374	006462	005012	
2021	010400	043117	051506	052105
2022	010406	024075	041517	040524
2023	010414	020114	043117	051506
2024	010422	052105	053040	046101
2025	010430	042525	020051	042504
2026	010436	040506	046125	051524
2027	010444	052040	020117	020060
2028	010452	030050	047440	043106
2029	010460	042523	024524	005015
2030	010466	020040	020040	020040
2031	010474	030060	020060	047101
2032	010502	020104	030062	020060
2033	010510	020075	047502	044124
2034	010516	040440	042522	055040
2035	010524	051105	020117	043117
2036	010532	051506	052105	005015
2037	010540	020040	020040	020040
2038	010546	033062	020060	020075
2039	010554	030455	030062	020060
2040	010562	044515	051103	020117
2041	010570	047111	044103	051505
2042	010576	005015		
2043	010600	020040	020040	020040
2044	010606	033060	020060	020075
2045	010614	030453	030062	020060
2046	010622	044515	051103	020117
2047	010630	047111	044103	051505
2048	010636	005015		
2049	010640	020040	020040	020040
2050	010646	040505	044103	041440
2051	010654	052517	052116	044440
2052	010662	020123	047101	047440
2053	010670	043106	042523	020124
2054	010676	047111	051103	046505
2055	010704	047105	020124	043117
2056	010712	031040	020065	044515
2057	010720	051103	020117	047111
2058	010726	044103	051505	005015
2059	010734	012		
2060	010735	101	054516	051440
2061	010742	041505	047524	051522
2062	010750	052040	020117	042502
2063	010756	043040	040514	043507
2064	010764	042105	041040	042101
2065	010772	020077	052050	050131
2066	011000	020105	020131	051117
2067	011006	047040	006451	012
2068	011013	040	020040	020040
2069	011020	044440	020106	020116
2070	011026	051511	052040	050131
2071	011034	042105	020054	044124
2072	011042	020105	047506	046522
2073	011050	052101	047440	042520

.ASCII /OFFSET=(OCTAL OFFSET VALUE) DEFAULTS TO 0 (0 OFFSET)/<15><12>

.ASCII / 000 AND 200 = BOTH ARE ZERO OFFSET/<15><12>

.ASCII / 260 = -1200 MICRO INCHES/<15><12>

.ASCII / 060 = +1200 MICRO INCHES/<15><12>

.ASCII / EACH COUNT IS AN OFFSET INCREMENT OF 25 MICRO INCHES/<15><12><12>

.ASCII /ANY SECTORS TO BE FLAGGED BAD? (TYPE Y OR N)/<15><12>

.ASCII / IF N IS TYPED, THE FORMAT OPERATION IS INITIALIZED/<15><12>

2074	011056	040522	044524	047117
2075	011064	044440	020123	047111
2076	011072	052111	040511	044514
2077	011100	042532	006504	012
2078	011105	040	020040	020040
2079	011112	040440	042116	047440
2080	011120	046116	020131	044124
2081	011126	051517	020105	042523
2082	011134	052103	051117	020123
2083	011142	047506	047125	020104
2084	011150	040502	020104	054502
2085	011156	052040	042510	043040
2086	011164	051117	040515	006524
2087	011172	012		
2088	011173	040	020040	020040
2089	011200	050040	047522	051107
2090	011206	046501	040440	042116
2091	011214	052040	047510	042523
2092	011222	046040	051511	042524
2093	011230	020104	047111	052040
2094	011236	042510	043040	041501
2095	011244	047524	054522	052040
2096	011252	041101	042514	005015
2097	011260	020040	020040	020040
2098	011266	051101	020105	046106
2099	011274	043501	042105	041040
2100	011302	042101	020056	043111
2101	011310	054440	044440	020123
2102	011316	054524	042520	026104
2103	011324	052040	042510	052440
2104	011332	042523	020122	040510
2105	011340	020123	044124	006505
2106	011346	012		
2107	011347	040	020040	020040
2108	011354	041440	050101	041101
2109	011362	046111	052111	020131
2110	011370	043117	044440	051516
2111	011376	051124	041525	044524
2112	011404	043516	052040	042510
2113	011412	043040	051117	040515
2114	011420	052124	051105	052040
2115	011426	020117	046106	043501
2116	011434	005015		
2117	011436	020040	020040	020040
2118	011444	050123	041505	043111
2119	011452	042511	020104	042523
2120	011460	052103	051117	020123
2121	011466	051501	041040	042101
2122	011474	020056	044124	020105
2123	011502	051120	043517	040522
2124	011510	020115	044527	046114
2125	011516	005015		
2126	011520	020040	020040	020040
2127	011526	044504	042522	052103
2128	011534	052040	042510	052440
2129	011542	042523	020122	047111

.ASCII / AND ONLY THOSE SECTORS FOUND BAD BY THE FORMAT/<15><12>

.ASCII / PROGRAM AND THOSE LISTED IN THE FACTORY TABLE/<15><12>

.ASCII / ARE FLAGED BAD. IF Y IS TYPED, THE USER HAS THE/<15><12>

.ASCII / CAPABILITY OF INSTRUCTING THE FORMATTER TO FLAG/<15><12>

.ASCII / SPECIFIED SECTORS AS BAD. THE PROGRAM WILL/<15><12>

.ASCII / DIRECT THE USER IN HOW THIS IS DONE./<15><12><12>

2130	011550	044040	053517	052040
2131	011556	044510	020123	051511
2132	011564	042040	047117	027105
2133	011572	005015	012	
2134	011575	120	042522	042523
2135	011602	053122	020105	047523
2136	011610	052106	040527	042522
2137	011616	041040	042101	051440
2138	011624	041505	047524	020122
2139	011632	044506	042514	024077
2140	011640	054524	042520	054440
2141	011646	047440	020122	024516
2142	011654	041450	024522	005015
2143	011662	044411	020106	020131
2144	011670	051511	052040	050131
2145	011676	042105	020054	044124
2146	011704	020105	047506	046522
2147	011712	052101	042524	020122
2148	011720	044527	046114	050040
2149	011726	042522	042523	053122
2150	011734	020105	044124	006505
2151	011742	012		
2152	011743	011	050101	046120
2153	011750	041511	041101	042514
2154	011756	041040	042101	051440
2155	011764	041505	047524	020122
2156	011772	044506	042514	020054
2157	012000	040515	045522	047111
2158	012006	020107	044124	020105
2159	012014	042523	052103	051117
2160	012022	006523	012	
2161	012025	011	040502	020104
2162	012032	044124	052101	040440
2163	012040	042522	046040	051511
2164	012046	042524	020104	047111
2165	012054	052040	042510	043040
2166	012062	046111	027105	040440
2167	012070	054516	040440	042104
2168	012076	052111	047511	040516
2169	012104	006514	012	
2170	012107	011	042523	052103
2171	012114	051117	020123	047506
2172	012122	047125	020104	040502
2173	012130	020104	054502	052040
2174	012136	042510	050040	047522
2175	012144	051107	046501	040440
2176	012152	042522	040440	050120
2177	012160	047105	042504	006504
2178	012166	012		
2179	012167	011	047524	052040
2180	012174	042510	043040	046111
2181	012202	027105	044440	020106
2182	012210	020116	051511	052040
2183	012216	050131	042105	052040
2184	012224	042510	041040	042101
2185	012232	051440	041505	047524

.ASCII /PRESERVE SOFTWARE BAD SECTOR FILE?(TYPE Y OR N)(CR)/<15><12>

.ASCII / IF Y IS TYPED, THE FORMATTER WILL PRESERVE THE/<15><12>

.ASCII / APPLICABLE BAD SECTOR FILE, MARKING THE SECTORS/<15><12>

.ASCII / BAD THAT ARE LISTED IN THE FILE. ANY ADDITIONAL/<15><12>

.ASCII / SECTORS FOUND BAD BY THE PROGRAM ARE APPENDED/<15><12>

.ASCII / TO THE FILE. IF N IS TYPED THE BAD SECTOR FILE/<15><12>

CZR6LDO RK06L CTG FMTR MACY11 30(1046)
CZR6LD.P11 27-AUG-81 11:08

28-AUG-81 13:57 PAGE 44
PROGRAM SPECIFIC RESERVED LOCATIONS

SEQ 0042

2186	012240	020122	044506	042514
2187	012246	005015		
2188	012250	044411	020123	046103
2189	012256	040505	042522	020104
2190	012264	042502	047506	042522
2191	012272	052040	042510	050040
2192	012300	047522	051107	046501
2193	012306	041040	043505	047111
2194	012314	027123	005015	012
2195	012321	123	052105	051440
2196	012326	044527	041524	020110
2197	012334	020067	047524	050040
2198	012342	044522	052116	040440
2199	012350	046040	051511	020124
2200	012356	043117	040440	046114
2201	012364	041040	042101	051440
2202	012372	041505	047524	051522
2203	012400	006456	005012	
2204	012404	042523	020124	053523
2205	012412	052111	044103	030440
2206	012420	052040	020117	047506
2207	012426	041522	020105	042522
2208	012434	047520	052122	047111
2209	012442	020107	043117	040440
2210	012450	046114	042040	052101
2211	012456	020101	047111	042440
2212	012464	051122	051117	005015
2213	012472	012		
2214	012473	123	052105	051440
2215	012500	044527	041524	020110
2216	012506	020060	047524	051040
2217	012514	050105	040505	020124
2218	012522	044124	020105	047105
2219	012530	044524	042522	043040
2220	012536	051117	040515	020124
2221	012544	050117	051105	052101
2222	012552	047511	006516	005012
2223	012560	043111	052040	042510
2224	012566	051440	043117	053524
2225	012574	051101	020105	053523
2226	012602	052111	044103	051040
2227	012610	043505	051511	042524
2228	012616	020122	051511	044440
2229	012624	020116	051525	026105
2230	012632	006440	012	
2231	012635	124	050131	047111
2232	012642	020107	047503	052116
2233	012650	047522	020114	020107
2234	012656	057050	024507	041440
2235	012664	052501	042523	020123
2236	012672	044124	020105	051120
2237	012700	043517	040522	006515
2238	012706	012		
2239	012707	124	020117	042522
2240	012714	052521	051505	020124
2241	012722	042516	020127	053523

.ASCII / IS CLEARED BEFORE THE PROGRAM BEGINS./<15><12><12>

.ASCII /SET SWITCH 7 TO PRINT A LIST OF ALL BAD SECTORS./<15><12><12>

.ASCII /SET SWITCH 1 TO FORCE REPORTING OF ALL DATA IN ERROR/<15><12><12>

.ASCII /SET SWITCH 0 TO REPEAT THE ENTIRE FORMAT OPERATION/<15><12><12>

.ASCII /IF THE SOFTWARE SWITCH REGISTER IS IN USE, /<15><12>

.ASCII /TYPING CONTROL G (^G) CAUSES THE PROGRAM/<15><12>

.ASCII /TO REQUEST NEW SWR SETTING/<15><12><12>

2242	012730	020122	042523	052124	
2243	012736	047111	006507	005012	
2244	012744	047105	020104	042510	.ASCIZ /END HELP FILE/<15><12><12>
2245	012752	050114	043040	046111	
2246	012760	006505	005012	000	
2247	012765	040	000040		SPACE2: .ASCIZ / /
2248					
2249	012770	005015	025052	051040	PROGID: .ASCII <15><12>/** RK06K-RK07K CARTRIDGE FORMATER **/<15><12>
2250	012776	030113	045466	051055	
2251	013004	030113	045467	041440	
2252	013012	051101	051124	042111	
2253	013020	042507	043040	051117	
2254	013026	040515	042524	020122	
2255	013034	025052	005015		
2256	013040	055103	033122	042114	.ASCIZ /CZR6LDO/<15><12>
2257	013046	006460	000012		
2258	013052	054524	042520	044040	HELPO: .ASCIZ /TYPE HELP FOR OPERATING INFO, ELSE CR/<15><12>
2259	013060	046105	020120	047506	
2260	013066	020122	050117	051105	
2261	013074	052101	047111	020107	
2262	013102	047111	047506	020054	
2263	013110	046105	042523	041440	
2264	013116	006522	000012		
2265	013122	051104	053111	020105	TPQ: .ASCIZ /DRIVE TYPE(6 OR 7)/
2266	013130	054524	042520	033050	
2267	013136	047440	020122	024467	
2268	013144	000			
2269	013145	104	044522	042526	DRVQ: .ASCIZ /DRIVE NUM= /
2270	013152	047040	046525	000075	
2271	013160	042523	052103	051117	SECTQ: .ASCIZ &SECTOR/TRACK=&
2272	013166	052057	040522	045503	
2273	013174	000075			
2274	013176	047515	042504	000075	MODEQ: .ASCIZ /MODE= /
2275	013204	053105	047105	041440	WDEQ: .ASCIZ /EVEN CYLINDER PATTERN= /
2276	013212	046131	047111	042504	
2277	013220	020122	040520	052124	
2278	013226	051105	036516	000	
2279	013233	117	042104	041440	WDOD: .ASCIZ /ODD CYLINDER PATTERN= /
2280	013240	046131	047111	042504	
2281	013246	020122	040520	052124	
2282	013254	051105	036516	000	
2283	013261	124	040522	045503	TRKLIM: .ASCIZ /TRACK LIMITS= /
2284	013266	046040	046511	052111	
2285	013274	036523	000		
2286	013277	103	046131	047111	CYLLIM: .ASCIZ /CYLINDER LIMITS= /
2287	013304	042504	020122	044514	
2288	013312	044515	051524	000075	
2289	013320	043117	051506	052105	OFSETQ: .ASCIZ /OFFSET= /
2290	013326	000075			
2291	013330	051120	051505	051105	PSDQ: .ASCIZ /PRESERVE SOFTWARE BAD SECTOR FILES?(TYPE Y OR N)(CR)/
2292	013336	042526	051440	043117	
2293	013344	053524	051101	020105	
2294	013352	040502	020104	042523	
2295	013360	052103	051117	043040	
2296	013366	046111	051505	024077	
2297	013374	054524	042520	054440	

CZR6LDG RK06L CTG FMTR MACY11 30(1046)
CZR6LD.P11 27-AUG-81 11:08

28-AUG-81 13:57 PAGE 46
PROGRAM SPECIFIC RESERVED LOCATIONS

SEQ 0044

2298	013402	047440	020122	024516	
2299	013410	041450	024522	000	
2300	013415	123	043117	053524	SDBSFP: .ASCIZ /SOFTWARE DETECTED BAD SECTOR FILES /
2301	013422	051101	020105	042504	
2302	013430	042524	052103	042105	
2303	013436	041040	042101	051440	
2304	013444	041505	047524	020122	
2305	013452	044506	042514	020123	
2306	013460	000			
2307	013461	120	042522	042523	PRSVD: .ASCIZ /PRESERVED/<15><12>
2308	013466	053122	042105	005015	
2309	013474	000			
2310	013475	116	052117	050040	NPRSVD: .ASCIZ /NOT PRESERVED/<15><12>
2311	013502	042522	042523	053122	
2312	013510	042105	005015	000	
2313	013515	101	054516	051440	UBSQ: .ASCIZ /ANY SECTORS TO BE FLAGED BAD? (TYPE Y OR N)(CR)/
2314	013522	041505	047524	051522	
2315	013530	052040	020117	042502	
2316	013536	043040	040514	042507	
2317	013544	020104	040502	037504	
2318	013552	024040	054524	042520	
2319	013560	054440	047440	020122	
2320	013566	024516	041450	024522	
2321	013574	000			
2322	013575	105	052116	051105	UBSFMT: .ASCIZ /ENTER OCTAL ADDRESS TO BE FLAGED BAD IN FORMAT CCC,T,SS/<15><12>
2323	013602	047440	052103	046101	
2324	013610	040440	042104	042522	
2325	013616	051523	052040	020117	
2326	013624	042502	043040	040514	
2327	013632	042507	020104	040502	
2328	013640	020104	047111	043040	
2329	013646	051117	040515	020124	
2330	013654	041503	026103	026124	
2331	013662	051523	005015	000	
2332	013667	105	052116	051105	NEXTQ: .ASCIZ /ENTER ADDRESS OR (CR) TO TERMINATE/<15><12>
2333	013674	040440	042104	042522	
2334	013702	051523	047440	020122	
2335	013710	041450	024522	052040	
2336	013716	020117	042524	046522	
2337	013724	047111	052101	006505	
2338	013732	000012			
2339	013734	047111	040526	044514	BADENT: .ASCIZ /INVALID ENTRY/<15><12>
2340	013742	020104	047105	051124	
2341	013750	006531	000012		
2342	013754	047111	042126	051440	ILLSEC: .ASCIZ /INVD SECTOR NUMBER/<15><12>
2343	013762	041505	047524	020122	
2344	013770	052516	041115	051105	
2345	013776	005015	000		
2346	014001	111	053116	020104	INVDSE: .ASCIZ /INVD STARTING OR ENDING PACK ADDRESS/<15><12>
2347	014006	052123	051101	044524	
2348	014014	043516	047440	020122	
2349	014022	047105	044504	043516	
2350	014030	050040	041501	020113	
2351	014036	042101	051104	051505	
2352	014044	006523	000012		
2353	014050	047111	042126	041440	INVDCY: .ASCIZ /INVD CYLINDER NUMBER /<15><12>

2354	014056	046131	047111	042504	
2355	014064	020122	052516	041115	
2356	014072	051105	006440	000012	
2357	014100	047111	042126	052040	INVDTK: .ASCIZ /INVD TRACK NUMBER /<15><12>
2358	014106	040522	045503	047040	
2359	014114	046525	042502	020122	
2360	014122	005015	000		
2361	014125	111	053116	020104	INVDDR: .ASCIZ /INVD DRIVE NUMBER/<15><12>
2362	014132	051104	053111	020105	
2363	014140	052516	041115	051105	
2364	014146	005015	000		
2365	014151	111	053116	020104	INVDTP: .ASCIZ /INVD DRIVE TYPE/<15><12>
2366	014156	051104	053111	020105	
2367	014164	054524	042520	005015	
2368	014172	000			
2369	014173	111	053116	020104	INVDMD: .ASCIZ /INVD MODE SELECTION (MAX OF 4)/<15><12>
2370	014200	047515	042504	051440	
2371	014206	046105	041505	044524	
2372	014214	047117	024040	040515	
2373	014222	020130	043117	032040	
2374	014230	006451	000012		
2375	014234	047111	042126	047440	INVD OF: .ASCIZ /INVD OFFSET VALUE/<15><12>
2376	014242	043106	042523	020124	
2377	014250	040526	052514	006505	
2378	014256	000012			
2379	014260	046120	040505	042523	SWLREQ: .ASCIZ /PLEASE SET WRITE LOCK FOR VERIFY OPERATIONS/<15><12>
2380	014266	051440	052105	053440	
2381	014274	044522	042524	046040	
2382	014302	041517	020113	047506	
2383	014310	020122	042526	044522	
2384	014316	054506	047440	042520	
2385	014324	040522	044524	047117	
2386	014332	006523	000012		
2387	014336	046120	040505	042523	RWLREQ: .ASCIZ /PLEASE RESET WRITE LOCK FOR FORMAT OPERATIONS/<15><12>
2388	014344	051040	051505	052105	
2389	014352	053440	044522	042524	
2390	014360	046040	041517	020113	
2391	014366	047506	020122	047506	
2392	014374	046522	052101	047440	
2393	014402	042520	040522	044524	
2394	014410	047117	006523	000012	
2395	014416	051120	051505	020123	CONREQ: .ASCIZ /PRESS CONTINUE WHEN READY/<15><12>
2396	014424	047503	052116	047111	
2397	014432	042525	053440	042510	
2398	014440	020116	042522	042101	
2399	014446	006531	000012		
2400	014452	040503	047116	052117	BAD632: .ASCIZ /CANNOT READ BAD SECTOR TRACK/<15><12>
2401	014460	051040	040505	020104	
2402	014466	040502	020104	042523	
2403	014474	052103	051117	052040	
2404	014502	040522	045503	005015	
2405	014510	000			
2406	014511	123	043117	053524	TDMSOF: .ASCIZ /SOFTWARE BAD SECTOR FILE NOT FORMATTED PROPERLY OR/<15><12>
2407	014516	051101	020105	040502	
2408	014524	020104	042523	052103	
2409	014532	051117	043040	046111	

2410	014540	020105	047516	020124
2411	014546	047506	046522	052101
2412	014554	042524	020104	051120
2413	014562	050117	051105	054514
2414	014570	047440	006522	000012
2415	014576	044506	042514	044440
2416	014604	020123	047524	020117
2417	014612	040514	043522	020105
2418	014620	047506	020122	044124
2419	014626	020105	051120	043517
2420	014634	040522	020115	020050
2421	014642	030476	030060	042440
2422	014650	052116	044522	051505
2423	014656	006451	000012	
2424	014662	040506	052103	051117
2425	014670	020131	040502	020104
2426	014676	042523	052103	051117
2427	014704	043040	046111	020105
2428	014712	047516	020124	047506
2429	014720	046522	052101	042524
2430	014726	020104	051120	050117
2431	014734	051105	054514	047440
2432	014742	006522	000012	
2433	014746	040503	052122	044522
2434	014754	043504	020105	042523
2435	014762	044522	046101	047040
2436	014770	046525	042502	020122
2437	014776	025052	025052	020052
2438	015004	000040		
2439	015006	040503	052122	044522
2440	015014	043504	020105	047515
2441	015022	047125	042524	020104
2442	015030	051511	040440	020116
2443	015036	046101	043511	046516
2444	015044	047105	020124	040503
2445	015052	052122	044522	043504
2446	015060	027105	005015	
2447	015064	047506	046522	052101
2448	015072	044440	020123	041101
2449	015100	051117	042524	006504
2450	015106	005012	000	
2451	015111	116	020117	040504
2452	015116	040524	041440	042510
2453	015124	045503	020040	047117
2454	015132	051440	041505	047524
2455	015140	020122	044124	052101
2456	015146	053440	046111	020114
2457	015154	047516	020124	040520
2458	015162	051523	053440	044522
2459	015170	042524	041440	042510
2460	015176	045503	005015	000
2461	015203	104	052101	020101
2462	015210	044103	041505	020113
2463	015216	051105	047522	020122
2464	015224	047111	053440	044522
2465	015232	042524	020056	046511

TDMSUF: .ASCIZ /FILE IS TOO LARGE FOR THE PROGRAM (>100 ENTRIES)/<15><12>

TDMFAC: .ASCIZ /FACTORY BAD SECTOR FILE NOT FORMATTED PROPERLY OR/<15><12>

SERNUM: .ASCIZ /CARTRIDGE SERIAL NUMBER ***** /

ALNPAC: .ASCII /CARTRIDGE MOUNTED IS AN ALIGNMENT CARTRIDGE./<15><12>

.ASCIZ /FORMAT IS ABORTED/<15><12><12>

IMPER2: .ASCIZ /NO DATA CHECK ON SECTOR THAT WILL NOT PASS WRITE CHECK/<15><12>

IMPERR: .ASCIZ /DATA CHECK ERROR IN WRITE. IMPOSSIBLE!!!/<15><12>

2466	015240	047520	051523	041111	
2467	015246	042514	020441	006441	
2468	015254	000012			
2469	015256	041505	020103	047503	CORABL: .ASCIZ /ECC CORRECTABLE/<15><12>
2470	015264	051122	041505	040524	
2471	015272	046102	006505	000012	
2472	015300	025012	020052	053440	PRGRS1: .ASCIZ <12>/** WRITE OPERATION COMPLETE **/<15><12><12>
2473	015306	044522	042524	047440	
2474	015314	042520	040522	044524	
2475	015322	047117	041440	046517	
2476	015330	046120	052105	020105	
2477	015336	025040	006452	005012	
2478	015344	000			
2479	015345	012	025052	020040	PRGRS2: .ASCIZ <12>/** VERIFY OPERATION COMPLETE **/<15><12><12>
2480	015352	042526	044522	054506	
2481	015360	047440	042520	040522	
2482	015366	044524	047117	041440	
2483	015374	046517	046120	052105	
2484	015402	020105	025040	006452	
2485	015410	005012	000		
2486	015413	040	025040	025052	STARS: .ASCIZ / *****/<15><12>
2487	015420	025052	005015	000	
2488	015425	012	051105	047522	BSWERR: .ASCIZ <12>/ERROR WRITING SOFTWARE DETECTED BAD SECTOR TABLES./<15><12>
2489	015432	020122	051127	052111	
2490	015440	047111	020107	047523	
2491	015446	052106	040527	042522	
2492	015454	042040	052105	041505	
2493	015462	042524	020104	040502	
2494	015470	020104	042523	052103	
2495	015476	051117	052040	041101	
2496	015504	042514	0?7123	005015	
2497	015512	000			
2498	015513	123	041505	047524	SECINE: .ASCIZ /SECTOR IN ERROR =/
2499	015520	020122	047111	042440	
2500	015526	051122	051117	036440	
2501	015534	000			
2502	015535	012	040502	020104	BSWTFI: .ASCIZ <12>/BAD SECTOR TABLE WRITTEN./<15><12>
2503	015542	042523	052103	051117	
2504	015550	052040	041101	042514	
2505	015556	053440	044522	051124	
2506	015564	047105	006456	000012	
2507	015572	052012	042510	041040	BSHEAD: .ASCIZ <12>/THE BAD SECTORS FOR THIS CARTRIDGE IN /
2508	015600	042101	051440	041505	
2509	015606	047524	051522	043040	
2510	015614	051117	052040	044510	
2511	015622	020123	040503	052122	
2512	015630	044522	043504	020105	
2513	015636	047111	000040		
2514	015642	051440	041505	047524	BSTAIL: .ASCIZ / SECTOR FORMAT ARE:/<15><12>
2515	015650	020122	047506	046522	
2516	015656	052101	040440	042522	
2517	015664	006472	000012		
2518	015670	041012	042101	051440	FBSLAB: .ASCIZ <12>/BAD SECTORS IN FACTORY TABLE:/<15><12>
2519	015676	041505	047524	051522	
2520	015704	044440	020116	040506	
2521	015712	052103	051117	020131	

2522	015720	040524	046102	035105	
2523	015726	005015	000		
2524	015731	015	041012	042101	SBSLAB: .ASCIZ <15><12>/BAD SECTORS IN SOFTWARE TABLE: /<15><12>
2525	015736	051440	041505	047524	
2526	015744	051522	044440	020116	
2527	015752	047523	052106	040527	
2528	015760	042522	052040	041101	
2529	015766	042514	006472	000012	
2530	015774	047516	042516	005015	NONE: .ASCIZ /NONE/<15><12>
2531	016002	000			
2532	016003	062	000060		TWENTY: .ASCIZ /20/
2533	016006	031062	000		TWENT2: .ASCIZ /22/
2534	016011	040	020040	041440	COLHD: .ASCIZ / CYLNR TRACK SECTOR/<15><12>
2535	016016	046131	042116	020122	
2536	016024	052040	040522	045503	
2537	016032	020040	051440	041505	
2538	016040	047524	006522	000012	
2539	016046	020040	020040	000	SPACE4: .ASCIZ / /
2540	016053	040	020040	000	SPACE3: .ASCIZ / /
2541	016057	040	020075	047524	TOTMES: .ASCIZ / = TOTAL NUMBER OF BAD SECTORS/<15><12>
2542	016064	040524	020114	052516	
2543	016072	041115	051105	047440	
2544	016100	020106	040502	020104	
2545	016106	042523	052103	051117	
2546	016114	006523	000012		
2547	016120	005015	044103	047101	XXDPMG: .ASCII <CR><LF>/CHANGE XXDP PACK/
2548	016126	042507	054040	042130	
2549	016134	020120	040520	045503	
2550	016142	005015	046103	040505	.ASCIZ <CR><LF>/CLEAR LOC 40,RESTART PROGRAM/
2551	016150	020122	047514	020103	
2552	016156	030064	051054	051505	
2553	016164	040524	052122	050040	
2554	016172	047522	051107	046501	
2555	016200	000			
2556					
2557		016202			.EVEN
2558	016202				START:
2559					.SBTTL INITIALIZE THE COMMON TAGS
2560					::CLEAR THE COMMON TAGS (\$CMTAG) AREA
2561	016202	012706	001100		MOV #CMTAG,R6 ::FIRST LOCATION TO BE CLEARED
2562	016206	005026			CLR (R6)+ ::CLEAR MEMORY LOCATION
2563	016210	022706	001140		CMP #SWR,R6 ::DONE?
2564	016214	001374			BNE -6 ::LOOP BACK IF NO
2565	016216	012706	001100		MOV #STACK,SP ::SETUP THE STACK POINTER
2566					::INITIALIZE A FEW VECTORS
2567	016222	012737	040632	000030	MOV #ERROR,@EMTVEC ::EMT VECTOR FOR ERROR ROUTINE
2568	016230	012737	000340	000032	MOV #340,@EMTVEC+2 ::LEVEL 7
2569	016236	012737	042730	000034	MOV #TRAP,@TRAPVEC ::TRAP VECTOR FOR TRAP CALLS
2570	016244	012737	000340	000036	MOV #340,@TRAPVEC+2;LEVEL 7
2571	016252	012737	042324	000024	MOV #SPWRDN,@PWRVEC ::POWER FAILURE VECTOR
2572	016260	012737	000340	000026	MOV #340,@PWRVEC+2 ::LEVEL 7
2573	016266	005037	001260		CLR \$ESCAPE ::CLEAR THE ESCAPE ON ERROR ADDRESS
2574	016272	112737	000001	001115	MOVB #1,\$ERMAX ::ALLOW ONE ERROR PER TEST
2575					::SIZE FOR A HARDWARE SWITCH REGISTER. IF NOT FOUND OR IT IS
2576					::EQUAL TO A "-1", SETUP FOR A SOFTWARE SWITCH REGISTER.
2577	016300	013746	000004		MOV @ERRVEC,-(SP) ::SAVE ERROR VECTOR

```

2578 016304 012737 016340 000004      MOV      #64$,@#ERRVEC      ;;SET UP ERROR VECTOR
2579 016312 012737 177570 001140      MOV      #DSWR,SWR        ;;SETUP FOR A HARDWARE SWICH REGISTER
2580 016320 012737 177570 001142      MOV      #DDISP,DISPLAY   ;;AND A HARDWARE DISPLAY REGISTER
2581 016326 022777 177777 162604      CMP      #-1,@SWR        ;;TRY TO REFERENCE HARDWARE SWR
2582 016334 001012                BNE      66$              ;;BRANCH IF NO TIMEOUT TRAP OCCURRED
2583                                ;;AND THE HARDWARE SWR IS NOT = -1
2584 016336 000403                BR       65$              ;;BRANCH IF NO TIMEOUT
2585 016340 012716 016346          64$:    MOV      #65$, (SP)      ;;SET UP FOR TRAP RETURN
2586 016344 000002                RTI
2587 016346 012737 000176 001140 65$:    MOV      #SWREG,SWR       ;;POINT TO SOFTWARE SWR
2588 016354 012737 000174 001142      MOV      #DISPREG,DISPLAY
2589 016362 012637 000004          66$:    MOV      (SP)+,@#ERRVEC   ;;RESTORE ERROR VECTOR
2590
2591 016366 004737 041002                JSR      PC,$TKINT        ;:INIT KEYBOARD
2592 016372 013701 002572                MOV      RKVEC,R1        ;:GET ADDRESS OF VECTOR STORAGE
2593 016376 012721 032442                MOV      #I.INTR,(R1)+   ;:SET IT TO INTERRUPT HNDLR
2594 016402 012711 000340                MOV      #PR7,(R1)       ;:SET INTERRUPT HANDLER PR7
2595 016406 013746 000000                MOV      0,-(SP)        ;:PUT NEW PS ON STACK
2596 016412 012746 016420                MOV      #67$,-(SP)     ;:PUT NEW PC ON STACK
2597 016416 000002                RTI                      ;:POP NEW PC AND PS
2598 016420          67$:
2599 016420 104401 012770                TYPE     ,PROGID         ;:TYPE IDENTIFICATION
2600 016424 104401 013052                TYPE     ,HELPO         ;:TYPE HELP QUESTION
2601 016430 104410                RDLIN    ;:GET ANSWER TO HELP Q.
2602 016432 012601                MOV      (SP)+,R1        ;:GET ADDRESS OF ANSWER
2603 016434 105711                TSTB    (R1)            ;:TEST FIRST CHAR
2604 016436 001402                BEQ     STPARM
2605 016440 104401 006377                TYPE     ,HELPL        ;:TYPE HELP FILE
2606
2607          ;:*****
2608          ;:SBTTL PARAMETER REQUEST & CHECK ROUTINE
2609          ;:*ENTRY: AUTOMATIC AT START OF PROGRAM.
2610          ;:*EXIT: GO TO MODE DECODE.
2611          ;:*
2612          ;:*THIS ROUTINE REQUESTS EACH PARAMETER IN TURN AND CALLS
2613          ;:*THE GETANS SUBROUTINE TO HANDLE THE ANSWERS AND STORE
2614          ;:*THEM. IT THEN CHECKS THE PARAMETERS FOR LEGALITY AND
2615          ;:*VALIDITY AND, IF AN ERROR IS FOUND, REQUESTS THE
2616          ;:*PARAMETERS AGAIN, STARTING WITH THE PARAMETER IN ERROR.
2617          ;:*****
2618
2619 016444          STPARM:
2620 016444 122737 000013 000041      CMPB    #13,@#41        ;:LOAD FROM XXDP PACK ?
2621 016452 001003                BNE     1$              ;:BRANCH IF NOT
2622 016454 104401 016120                TYPE     ,XXDPMG
2623 016460 000000                HALT
2624 016462          1$:
2625 016462 012702 006310                MOV      #INITTP,R2     ;:STARTING ADDRESS OF DEF. PARAM
2626 016466 012703 006336                MOV      #TPINUS,R3    ;:STARTING ADD OF ACTIVE PARAMS
2627 016472 104401 013122                TYPE     ,TPQ          ;:TYPE DRIVE TYPE QUESTION
2628 016476 004437 017766                JSR     R4,GETANS
2629 016502 023727 006336 000006  PARM2:  CMP      TPINUS,#6      ;:SEE IF RK06
2630 016510 001404                BEQ     PARM3           ;:BR IF YES
2631 016512 012737 001456 006332      MOV      #1456,INITEC  ;:ELSE LOAD DEFAULT END CYL FOR RK07
2632 016520 000403                BR      PARM4
2633 016522 012737 000632 006332  PARM3:  MOV      #632,INITEC   ;:LOAD DEFAULT END CYL FOR RK06

```

2634	016530	104401	013145		PARM4: TYPE	,DRVQ		:TYPE DRIVE QUESTION
2635	016534	004437	017766			JSR R4,GETANS		:GO GET PARAM
2636	016540	104401	013160			TYPE ,SECTQ		:TYPE SECTOR/TRACK QUESTION
2637	016544	004437	017766			JSR R4,GETANS		:GO GET PARAM
2638	016550	104401	013176		PARAM3: TYPE	,MODEQ		:TYPE MODE QUES
2639	016554	004437	017766			JSR R4,GETANS		:GO GET PARAM
2640	016560	104401	013204			TYPE ,WDEQ		
2641	016564	004437	017766			JSR R4,GETANS		
2642	016570	104401	013233			TYPE ,WDOD		
2643	016574	004437	017766			JSR R4,GETANS		
2644	016600	104401	013261		PARAM5: TYPE	,TRKLIM		:TYPE TRACK LIMITS QUESTION
2645	016604	004437	017766			JSR R4,GETANS		:GO GET PARAM
2646	016610	104401	013277			TYPE ,CYLLIM		:TYPE CYLINDER LIMITS QUESTION
2647	016614	004437	017766			JSR R4,GETANS		:GO GET PARAM
2648	016620	104401	013320		PARAM6: TYPE	,OFSETQ		:TYPE OFFSET QUESTION
2649	016624	004437	017766			JSR R4,GETANS		
2650	016630	023727	006336	000007	DONEPR: CMP	TPINUS,#7		:TEST IF DRIVE TYPE IS
2651	016636	001405				BEQ 4\$:6 OR 7
2652	016640	023727	006336	000006		CMP TPINUS,#6		:IF NOT TYPE ERROR
2653	016646	001401				BEQ 4\$		
2654	016650	000475				BR IVDTP		
2655	016652	023737	006360	006356	4\$:	CMP ECINUS,SCINUS		:COMP START & END CYL
2656	016660	002451				BLT RSTADD		:NG-GET PARAM AGAIN
2657	016662	023737	006354	006352		CMP ETINUS,STINUS		:COMP START & END TRK
2658	016670	002445				BLT RSTADD		:NG-GET PARAM AGAIN
2659	016672	105037	002653			CLRB TYPFMT		:CLEAR TYPE-FORMAT SWITCH, ASSUME RK06 HERE
2660	016676	023727	006336	000006		CMP TPINUS,#6		:TEST IF TRACK PARAMETERS ARE
2661	016704	001407				BEQ 2\$:VALID FOR THE DRIVE TYPE
2662	016706	152737	000004	002653		BISB #B.CDT,TYPFMT		
2663	016714	012737	001456	017312		MOV #1456,LCYL		:SET LAST CYL FOR RK07
2664	016722	000403				BR 3\$		
2665	016724	012737	000632	017312	2\$:	MOV #632,LCYL		:SET LAST CYL FOR RK06
2666	016732	023727	006354	000002	3\$:	CMP ETINUS,#2		:TEST FOR VALID TRACK
2667	016740	003027				BGT IVDTRK		:BR IF TOO BIG
2668	016742	023737	006360	017312		CMP ECINUS,LCYL		:TEST FOR VALID CYL
2669	016750	003020				BGT IVDCYL		:BR IF TOO BIG
2670	016752	023727	006340	000007		CMP DRINUS,#7		:TEST IF VALID DRIVE
2671	016760	003026				BGT IVDDRV		:TOO BIG-BRANCH
2672	016762	023727	006344	000004		CMP MDINUS,#4		:TEST IF VALID MODE
2673	016770	003031				BGT IVDMOD		:ERROR IN MODE SELECTION
2674	016772	023727	006362	000260		CMP OFINUS,#260		:TEST IF VALID OFFSET
2675	017000	003034				BGT IVDOF		:TO BIG, ERROR
2676	017002	000442				BR GETOUT		:YES-BR TO EXIT
2677	017004	104401	014001		RSTADD: TYPE	,INVDSE		:TYPE ERROR MESSAGE
2678	017010	000405				BR SETRS		
2679	017012	104401	014050		IVDCYL: TYPE	,INVDCY		:TYPE ERROR
2680	017016	000402				BR SETRS		
2681	017020	104401	014100		IVDTRK: TYPE	,INVDTK		:TYPE ERROR
2682	017024	012702	006324		SETR: MOV	#INITST,R2		:RESET REGISTERS TO REENTER ALL
2683	017030	012703	006352			MOV #STINUS,R3		:START & END ADDRESSES
2684	017034	000661				BR PARAM5		:GO GET NEW START & END ADD
2685	017036	104401	014125		IVDDRV: TYPE	,INVDDR		
2686	017042	000600				BR STPARM		:GO GET ALL PARAM AGAIN
2687	017044	104401	014151		IVDTP: TYPE	,INVDTP		
2688	017050	000137	016444			JMP STPARM		
2689	017054	104401	014173		IVDMOD: TYPE	,INVDMD		

```
2690 017060 012702 006316      MOV      #INITMD,R2      ;RESET REGISTERS FOR RE-REQUEST
2691 017064 012703 006344      MOV      #MDINUS,R3     ;OF MODE
2692 017070 000627              BR       PARAM3         ;RETURN TO MODE REQUEST
2693 017072 104401 014234      IVDOF:  TYPE      ,INVDOF ;TYPE INVALID OFFSET MESSAGE
2694 017076 012702 006334      MOV      #INITOF,R2    ;SET UP R2 AND R3 TO REDO OFFSET
2695 017102 012703 006362      MOV      #OFINUS,R3    ;PARAMETER
2696 017106 000644              BR       PARM6         ;GO GET OFFSET
2697 017110 105037 002656      GETOUT: CLR      OPCONT ;CLEAR OPERATION CONTRL SWITCHES
2698 017114 023727 006342 000026  CMP      SEINUS,#26     ;TEST IF 26 SECTOR FORMAT
2699 017122 001412              BEQ      3$            ;YES-SKIP
2700 017124 012737 000024 006342  MOV      #24,SEINUS    ;FORCE TO 24 SECTORS/TRACK
2701 017132 152737 000020 002653  BISB    #B.CFMT,TYPFMT ;SET BIT FOR FORMAT &
2702 017140 012737 012000 006306  MOV      #12000,TKWDCT ;SET WORD COUNT
2703 017146 000403              BR       4$            ;
2704 017150 012737 013000 006306  3$:     MOV      #13000,TKWDCT ;ELSE SET FOR 22 SECTOR WORD COUNT
2705 017156 005437 006306 4$:     NEG      TKWDCT        ;MAKE IT NEG FOR RK611
2706 017162 023737 006362 006334  CMP      OFINUS,INITOF ;TEST IF OFFSET REQUIRED
2707 017170 001403              BEQ      5$            ;NO-SKIP
2708 017172 152737 000020 002656  BISB    #OREQSW,OPCONT ;ELSE SET SWITCH
2709 017200 023727 006344 000004  5$:     CMP      MDINUS,#4     ;IF MODE 4,
2710 017206 001004              BNE     6$            ;NO-SKIP
2711 017210 152737 000012 002656  BISB    #RCDASW!VFHDSW,OPCONT ;ELSE SET READ CHECK & HEADER VERIFY
2712 017216 000433              BR       10$           ;
2713 017220 023727 006344 000003  6$:     CMP      MDINUS,#3     ;IF MODE 3
2714 017226 001004              BNE     7$            ;NO-SKIP
2715 017230 152737 000002 002656  BISB    #VFHDSW,OPCONT ;SET VERIFY HEADER
2716 017236 000423              BR       10$           ;
2717 017240 023727 006344 000002  7$:     CMP      MDINUS,#2     ;IF MODE 2
2718 017246 001004              BNE     8$            ;NO-SKIP
2719 017250 152737 000007 002656  BISB    #WHDSW!VFHDSW!WCDASW,OPCONT ;SET WRITE HEADER & DATA VERIFY HEADER,
2720                                ;AND WRITE CHECK DATA SWITCHES
2721 017256 000413              BR       10$           ;
2722 017260 023727 006344 000001  8$:     CMP      MDINUS,#1     ;IF MODE 1
2723 017266 001004              BNE     9$            ;NO-SKIP
2724 017270 152737 000003 002656  BISB    #WHDSW!VFHDSW,OPCONT ;SET VERIFY HEADER &
2725                                ;SET WRITE HEADER & DATA SWITCH
2726 017276 000403              BR       10$           ;
2727 017300 152737 000001 002656  9$:     BISB    #WHDSW,OPCONT ;SET WRITE HDR & DATA SWITCH
2728 017306 000137 017314 10$:    JMP      GETUBS        ;GET USER BAD SECTORS
2729
2730 017312 000000      LCTL:   0              ;LAST CYL 632 FOR RK06, 1456 FOR RK07
2731
2732      ;*****
2733      ;SBTTL USER BAD SECTORS INPUT ROUTINE
2734      ;*THIS ROUTINE PROVIDES THE CAPABILITY TO INDICATE SPECIFIC PACK
2735      ;*ADDRESSES THAT ARE TO BE FLAGED AS SOFTWARE BAD SECTORS. THE USER
2736      ;*IS DIRECTED IN HOW TO ENTER THE ADDRESSES AND HOW TO END THE
2737      ;*OPERATION.
2738      ;*****
2739 017314 132737 000001 002656  GETUBS: BITB    #WHDSW,OPCONT ;TEST IF WRITING IS TO BE DONE
2740 017322 001566              BEQ      14$          ;NO - SKIP OUT
2741 017324 012700 000400              MOV      #400,R0      ;SET UP TO CLEAR BSSOFT FILE
2742 017330 012701 003660              MOV      #BSSOFT,R1   ;TO ALL ONES
2743 017334 012721 177777      1$:     MOV      #177777,(R1)+
2744 017340 005300              DEC      R0
2745 017342 001374              BNE     1$
```

2746	017344	012701	002660		MOV	#RDBUF,R1	:GET ADDRESS OF READ BUFFER
2747	017350	012700	000400		MOV	#400,R0	:SET COUNT FOR CLEAR
2748	017354	012721	177777	21\$:	MOV	#177777,(R1)+	:CLEAR TO ALL ONES
2749	017360	005300			DEC	R0	:DECREMENT COUNT
2750	017362	001374			BNE	21\$:LOOP UNTIL 0
2751	017364	104401	013515		TYPE	,UBSQ	:ASK USER BAD SECTOR QUESTION
2752	017370	104410			RDLIN		:GET ANSWER
2753	017372	012601			MOV	(SP)+,R1	
2754	017374	121127	000032		CMPB	(R1),#032	:TEST IF ^Z
2755	017400	001002			BNE	3\$:NO - SKIP
2756	017402	000137	020154		JMP	DRINIT	:ELSE GO START PROGRAM
2757	017406	105711		3\$:	TSTB	(R1)	:TEST IF CR (NULL)
2758	017410	001531			BEQ	15\$:YES - EXIT TEST
2759	017412	121127	000116		CMPB	(R1),#'N	:TEST IF 'NO'
2760	017416	001526			BEQ	15\$:YES - EXIT TEST
2761	017420	012700	002660		MOV	#RDBUF,R0	:SET POINTER TO FIRST WORD
2762	017424	104401	013575	4\$:	TYPE	,UBSFMT	:TYPE USER BAD SECTOR FORMAT
2763	017430	104401	013667		TYPE	,NEXTQ	:TYPE NEXT ENTRY REQUEST
2764	017434	104410		12\$:	RDLIN		:GET BAD SECTOR ENTRY
2765	017436	012601			MOV	(SP)+,R1	
2766	017440	105711			TSTB	(R1)	:TEST IF CR (NULL)
2767	017442	001512			BEQ	13\$:YES - EXIT TEST
2768	017444	010146			MOV	R1,-(SP)	:SET TO CONVERT TO BINARY
2769	017446	004737	037636		JSR	PC,OCTBIN	
2770	017452	017652			30\$:ERROR RETURN
2771	017454	021637	006360		CMP	(SP),ECINUS	:TEST IF CYLINDER VALID
2772	017460	003003			BGT	16\$:YES - SKIP
2773	017462	021637	006356		CMP	(SP),SCINUS	:TEST IF CYLINDER VALID
2774	017466	002004			BGE	5\$:YES - SKIP
2775	017470	104401	014050	16\$:	TYPE	,INVDCY	:INVALID CYLINDER MESSAGE
2776	017474	005726			TST	(SP)+	:CLEAN STACK
2777	017476	000752			BR	4\$:GO TYPE FORMAT MESSAGE
2778	017500	012620		5\$:	MOV	(SP)+,(R0)+	:STORE CYLINDER
2779	017502	121127	000054	6\$:	CMPB	(R1),#',	:LOOP TO BUMP R1 PAST CYLINDER ENTRY
2780	017506	001406			BEQ	7\$:LOOK FOR COMMA BUT RESTART
2781	017510	105721			TSTB	(R1)+	:THIS ENTRY IF CR (NULL) IS
2782	017512	001373			BNE	6\$:IS FOUND
2783	017514	005740			TST	-(R0)	:RESET POINTER
2784	017516	104401	013734		TYPE	,BADENT	:TYPE BAD ENTRY MESSAGE
2785	017522	000740			BR	4\$:GO TYPE ENTRY FORMAT MESSAGE
2786	017524	105721		7\$:	TSTB	(R1)+	:BUMP PAST COMMA
2787	017526	010146			MOV	R1,-(SP)	:GO CONVERT TRACK TO BINARY
2788	017530	004737	037636		JSR	PC,OCTBIN	
2789	017534	017652			30\$:ERROR RETURN
2790	017536	012603			MOV	(SP)+,R3	:STORE TRACK
2791	017540	121127	000054	8\$:	CMPB	(R1),#',	:LOOP TO BUMP R1 PAST
2792	017544	001406			BEQ	9\$:TRACK ENTRY. LOOK FOR COMMA
2793	017546	105721			TSTB	(R1)+	:BUT RESTART THIS ENTRY IF CR
2794	017550	001373			BNE	8\$: (NULL) IS FOUND
2795	017552	005740			TST	-(R0)	:RESET POINTER
2796	017554	104401	013734		TYPE	,BADENT	:TYPE BAD ENTRY MESSAGE
2797	017560	000721			BR	4\$: AND TYPE FORMAT MESSAGE
2798	017562	105721		9\$:	TSTB	(R1)+	:BUMP PAST COMMA
2799	017564	010146			MOV	R1,-(SP)	:GO CONVERT SECTOR TO BINARY
2800	017566	004737	037636		JSR	PC,OCTBIN	
2801	017572	017652			30\$:ERROR RETURN

```

2802 017574 012604          MOV      (SP)+,R4      ;STORE SECTOR
2803 017576 020337 006354   CMP      R3,ETINUS    ;TEST IF TRACK VALID
2804 017602 003003          BGT      17$          ;YES - SKIP
2805 017604 020337 006352   CMP      R3,STINUS    ;TEST IF TRACK VALID
2806 017610 002004          BGE      10$          ;YES - SKIP
2807 017612 104401 014100   17$:    TYPE      ,INVDTK ;TYPE INVALID TRACK MESSAGE
2808 017616 005740          TST      -(R0)        ;RESET POINTER
2809 017620 000701          BR       4$           ;RESTART ENTRY
2810 017622 020437 006342   10$:    CMP      R4,SEINUS ;TEST IF SECTOR VALID
2811 017626 002404          BLT      11$          ;YES - SKIP
2812 017630 104401 013754   TYPE      ,ILLSEC     ;TYPE ILLEGAL SECTOR MESSAGE
2813 017634 005740          TST      -(R0)        ;RESET POINTER
2814 017636 000672          BR       4$           ;GO RESTART ENTRY
2815 017640 110420          11$:    MOV      R4,(R0)+   ;INSERT SECTOR NUMBER
2816 017642 110320          MOV      R3,(R0)+   ;INSERT TRACK NUMBER
2817 017644 104401 013667   TYPE      ,NEXTQ      ;TYPE NEXT QUESTION
2818 017650 000671          BR       12$          ;GO GET NEXT ENTRY
2819 017652 104401 006374   30$:    TYPE      ,QUES    ;TYPE BAD ENTRY LINE
2820 017656 104401 042170   TYPE      ,STTYIN
2821 017662 104401 001267   TYPE      ,$CRLF
2822 017666 000656          BR       4$           ;GET LINE AGAIN
2823 017670 012710 177777   13$:    MOV      #177777,(R0) ;MAKE SURE THE WORD AFTER THE LAST
2824                                ;VALID ENTRY IS ALL ONES.
2825 017674 000137 017704   15$:    JMP      CKPRES     ;GO CHECK IF PRESERVE SDBSF
2826 017700 000137 020154   14$:    JMP      DRINIT    ;GO INITIALIZE DRIVE
2827                                ;*****
2828                                ;SBTTL PRESERVE SDBSF QUESTION ROUTINE
2829                                ;* THIS ROUTINE DETERMINES IF SOFTWARE BAD SECTOR FILES ARE TO
2830                                ;* BE PRESERVED
2831
2832 017704 104401 013330   CKPRES: TYPE      ,PSDQ    ;TYPE PRESERVE SDBSF QUESTION
2833 017710 104410          RDLIN          ;GET ANSWER
2834 017712 012601          MOV      (SP)+,R1    ;GET POINTER TO ANSWER
2835 017714 104401 013415   TYPE      ,SDBSFP    ;TYPE START OF PRESERVE MESSAGE
2836 017720 121127 000032   CMP      (R1),#032   ;TEST IF ^Z
2837 017724 001001          BNE      2$          ;NO - SKIP
2838 017726 000405          BR       1$          ;ELSE SET BIT AND GO FORMAT
2839 017730 105711          2$:    TST      (R1)    ;TEST IF NULL
2840 017732 001403          BEQ      1$          ;YES - EXIT
2841 017734 121127 000116   CMP      (R1),#'N    ;TEST IF 'N'
2842 017740 001406          BEQ      3$          ;YES EXIT
2843 017742 152737 000040 002656 1$:    BIS      #PSDBSF,OPCONT ;SET FLAG TO PRESERVE SDBSF
2844 017750 104401 013461   TYPE      ,PRSV      ;REPORT PRESERVED
2845 017754 000402          BR       4$          ;SKIP
2846 017756 104401 013475   3$:    TYPE      ,NPRSV   ;REPORT NOT PRESERVED
2847 017762 000137 020154   4$:    JMP      DRINIT
2848
2849                                ;*****
2850                                ;SBTTL GET ANSWER (PARAMETER ENTRY) SUBROUTINE
2851                                ;*ENTRY JSR R4,GETANS WITH R2 POINTING TO THE DEFAULT PARAM
2852                                ;* R3 POINTING TO THE ACTIVE PARAM
2853                                ;*RETURN RTS R4
2854                                ;*
2855                                ;*THIS ROUTINE RECEIVES THE PARAMETER ENTRY AND ALTERS THE
2856                                ;*APPROPRIATE ACTIVE PARAMETER. IF THE ENTRY IS NULL THE DEFAULT
2857                                ;*VALUE IS PLACED IN THE ACTIVE ENTRY. THE CONTROL C & Z

```

2858
2859
2860
2861
2862
2863 017766
2864 017766 104410
2865 017770 012601
2866 017772 121127 000032
2867 017776 001434
2868 020000 105711
2869 020002 001002
2870 020004 011213
2871 020006 000405
2872 020010 010146
2873 020012 004737 037636
2874 020016 020106
2875 020020 012613
2876 020022 022703 006352
2877 020026 001403
2878 020030 022703 006356
2879 020034 001010
2880 020036 022223
2881 020040 105711
2882 020042 001404
2883 020044 122127 000054
2884 020050 001373
2885 020052 000756
2886 020054 011213
2887 020056 022223
2888 020060 000204
2889 020062 012704 016444
2890 020066 000204
2891 020070 012223
2892 020072 020327 006364
2893 020076 001374
2894 020100 012704 016630
2895 020104 000204
2896 020106 104401 006374
2897 020112 104401 042170
2898 020116 104401 001267
2899 020122 020327 006354
2900 020126 001403
2901 020130 020327 006360
2902 020134 001004
2903 020136 162702 000002
2904 020142 162703 000002
2905 020146 162704 000010
2906 020152 000204
2907
2908
2909
2910
2911
2912
2913

:(^C & ^Z) ARE DECODED AND THE APPROPRIATE ACTION TAKEN. SOME
:PARAMETER CHECKING IS DONE. ALL NUMERICS ARE TESTED TO
:INSURE THEY ARE OCTAL.
:*****

GETANS:
RDLIN ;READ ANSWER
MOV (SP)+,R1 ;GET START ADD OF MESSAGE
CMPB (R1),#032 ;TEST FOR CONTROL Z
BEQ 11\$;YES-BRANCH
TSTB (R1) ;TEST FOR NULL
BNE 4\$
MOV (R2),(R3) ;GET DEFAULT FOR THIS PARAM
BR 5\$
4\$: MOV R1,-(SP) ;PUT ADDRESS OF PARAM IN ON STACK
JSR PC,OCTBIN ;CALL CONVERSION
30\$;ERROR RETURN
MOV (SP)+,(R3) ;STORE RESULTS
5\$: CMP #STINUS,R3 ;START TRACK PARAM
BEQ 6\$;YES-BRANCH
CMP #SCINUS,R3 ;STARTING CYL PARAM
BNE 9\$;NO-GO TO EXIT
6\$: CMP (R2)+,(R3)+ ;BUMP R2 & R3
7\$: TSTB (R1) ;CHECK IF NULL
BEQ 8\$;YES-GO STORE DEFAULT & EXIT
CMPB (R1)+,#' ;TEST IF COMMA
BNE 7\$;NO-LOOP
BR 4\$;YES-GO BACK TO CONVERT PARAM
8\$: MOV (R2),(R3) ;STORE DEFAULT
9\$: CMP (R2)+,(R3)+ ;BUMP R2 & R3
RTS R4 ;RETURN
10\$: MOV #STPARM,R4 ;DUMMY R4 TO RESTART PARAM
RTS R4 ;RETURN
11\$: MOV (R2)+,(R3)+ ;MOV IN DEFAULT
CMP R3,#OFINUS+2 ;LAST PARAM SET?
BNE 11\$;NO-LOOP
MOV #DONEPR,R4 ;DUMMY RETURN FOR PARAM COMPLETE
RTS R4 ;RETURN
30\$: TYPE ,QUES ;TYPE QUESTION MARK
TYPE ,STTYIN ;TYPE BAD LINE
TYPE ,\$CRLF
CMP R3,#ETINUS ;IS THIS ENDING TRACK?
BEQ 31\$;YES-BRANCH
CMP R3,#ECINUS ;IS THIS ENDING CYLINDER?
BNE 32\$;NO-BRANCH
31\$: SUB #2,R2 ;BACK UP REGISTERS TO REDO
SUB #2,R3 ;PARAM REQUEST
32\$: SUB #10,R4 ;BACK UP TO REREQUEST PARAM
RTS R4 ;RETURN

:SBTTL DRIVE INITIALIZE ROUTINE
:BEFORE FORMATTING BEGINS, THIS ROUTINE INITIALIZES
:THE SUBSYSTEM, RECALIBRATES THE DRIVE, AND SETS VOLUME
:VALID. IT THEN CHECKS DRIVE STATUS TO INSURE IT IS
:AVAILABLE, READY, AND VALID. WRITE LOCK IS CHECKED TO

```

2914
2915
2916
2917
2918
2919
2920
2921
2922
2923
2924
2925 020154
2926 020154 012706 001100
2927 020160 012737 020154 001110
2928 020166 012737 026462 002600
2929 020174 012737 025524 002576
2930 020202 105037 002654
2931 020206 005037 006364
2932 020212 012703 002446
2933 020216 113763 006340 000000
2934 020224 105063 000007
2935 020230 153763 002653 000007
2936 020236 013702 002570
2937 020242 012705 002362
2938 020246 005065 000014
2939 020252 105065 000007
2940 020256 153765 002653 000007
2941 020264 112765 000177 000001
2942 020272 004737 025442
2943 020276 113765 006340 000000
2944 020304 112765 000113 000001
2945 020312 004737 025442
2946 020316 112765 000103 000001
2947 020324 004737 025442
2948 020330 112765 000141 000001
2949 020336 004737 025442
2950 020342 023727 006344 000003
2951 020350 002415
2952 020352 032765 004000 000040
2953 020360 001026
2954 020362 104401 014260
2955 020366 104401 014416
  
```

```

:*VERIFY IT IS RESET.
:*
:*REGISTER 5 IS LOADED WITH THE ADDRESS OF THE
:*PARAMETER BLOCK. R5 WILL CONTAIN THIS ADDRESS FOR
:*THE ENTIRE PROGRAM.
:*
:*THE DRIVE NUMBER PARAMETER IN THE PARAMETER
:*BLOCK IS SET TO THE SELECTED DRIVE. THIS ALSO REMAINS
:*VALID FOR THE DURATION OF THE PROGRAM.
:*****
  
```

```

DRINIT:
MOV #STACK,SP ;RESET THE STACK
MOV #DRINIT,$LPERR ;SET UP TO LOOP ON ERROR
MOV #ERRHDL,A.ABNL ;SET UP FOR ERROR HANDLING
MOV #ERRFRE,A.NORM ;SET UP FOR NO ERROR DRIVER RETURNS
CLRB ERRCNT ;CLEAR ERROR COUNT
CLR RECODE ;CLEAR RECOVERY CODE
MOV #PARM1,R3 ;SET UP PARAMETER BLOCK 1 FOR
MOVB DRINUS,P.DRVN(R3) ;ADDITIONAL INFO RECOVERY IF
CLRB P.CS1H(R3) ;CLEAR PREV. CONTENTS
BISB TYPFMT,P.CS1H(R3) ;NEEDED IN ERROR PROCESSING
MOV RKBAS,R2 ;SET R2 FOR RKBASE ADDRESS
MOV #PARM0,R5 ;GET PARAMETER BLOCK ADDRESS
CLR P.PRST(R5) ;CLEAR PROGRAM STATUS REG
CLRB P.CS1H(R5) ;CLR PREV CONTENTS
BISB TYPFMT,P.CS1H(R5)
MOVB #SUBCLR,P.CMND(R5) ;DO SUBSYSTEM CLEAR
JSR PC,DRVCAL
MOVB DRINUS,P.DRVN(R5) ;LOAD DRIVE NUMBER
MOVB #RECAL,P.CMND(R5) ;DO RECALIBRATE
JSR PC,DRVCAL
MOVB #PACK,P.CMND(R5) ;DO PACK ACK
JSR PC,DRVCAL
MOVB #RDSTAT,P.CMND(R5) ;DO RD STAT
JSR PC,DRVCAL
CMP MDINUS,#3 ;TEST IF ANY WRITE TO BE DONE
BLT 1$ ;NO-GO TEST WRITE LOCK ON
BIT #S.WRL,P.A00(R5) ;ELSE TEST IF WRITE LOCK OFF
BNE 2$ ;IF WRONG - TYPE MESSAGE
TYPE ,SWLREQ ;AND WAIT FOR RESPONSE
TYPE ,CONREQ
  
```



```

2956 020372 042762 000100 000000      BIC      #IE,RKCS1(R2)      ;PREVENT INTERRUPT WHEN WRITE LOCK CHANGED
2957 020400 000000                      HALT
2958 020402 000664                      BR        DRINIT          ;WRITE LOCK IS TESTED AGAIN
2959 020404 032765 004000 000040 1$:   BIT      #S.WRL,P.A00(R5) ;FOR CORRECT SETTING
2960 020412 001411                      BEQ      2$
2961 020414 104401 014336                      TYPE    ,RWLREQ
2962 020420 104401 014416                      TYPE    ,CONREQ
2963 020424 042762 000100 000000      BIC      #IE,RKCS1(R2)      ;PREVENT INTERRUPT WHEN WRITE LOCK CHANGE
2964 020432 000000                      HALT
2965 020434 000647                      BR        DRINIT
2966 020436 112765 000117 000001 2$:   MOV      #SEEK,P.CMND(R5) ;DO SEEK
2967 020444 013765 017312 000002      MOV      LCYL,P.CYLN(R5) ;TO LAST CYL
2968 020452 112765 000002 000005      MOV      #2,P.TRCK(R5)   ;TRACK 2
2969 020460 004737 025442                      JSR      PC,DRVCAL
2970 020464 012765 004660 000010      MOV      #BSFACT,P.BALO(R5) ;SET BUFFER ADDRESS FOR BAD SECTORS
2971 020472 005000                      CLR      R0
2972 020474 012703 000011                      MOV      #11,R3          ;SET UP MAX SECTOR VALUE
2973 020500 112765 000121 000001 8$:   MOV      #RDATA,P.CMND(R5) ;DO READ DATA FROM 0 THRU ?
2974 020506 012765 000400 000012      MOV      #400,P.WC(R5)   ;READ ENTIRE SECTOR
2975 020514 005465 000012                      NEG      P.WC(R5)
2976 020520 142765 000020 000007      BIC      #B.CFMT,P.CS1H(R5) ;SET TO 22 SECTOR FORMAT
2977 020526 132737 000020 002653      BIT      #B.CFMT,TYPFMT  ;TEST IF PACK TO BE FORMATTED 20 SECTOR
2978 020534 001401                      BEQ      3$              ;NO-BR. LEAVE R0=0
2979 020536 005200                      INC      R0              ;SET R0=1
2980 020540 110065 000004 3$:   MOV      R0,P.SECT(R5)   ;PUT IN BLOCK
2981 020544 004737 025442                      JSR      PC,DRVCAL
2982 020550 032737 100000 006364      BIT      #ANYDER,RECODE  ;TEST IF ERROR
2983 020556 001410                      BEQ      4$              ;BR IF NO
2984 020560 062700 000002                      ADD      #2,R0           ;BUMP TO NEXT SECTOR
2985 020564 020003                      CMP      R0,R3           ;CHECK IF STILL IN
2986 020566 003764                      BLE      3$              ;BAD SECTOR. IF YES, DO IT AGAIN
2987 020570 104401 014452                      TYPE    ,BAD632         ;TYPE MESSAGE
2988 020574 000000                      HALT
2989 020576 000776                      BR        -2
2990 020600 132737 000001 002656 4$:   BIT      #WHDSW,OPCONT   ;WILL THIS PASS WRITE?
2991 020606 001404                      BEQ      17$             ;NO - SKIP
2992 020610 132737 000040 002656      BIT      #PSDBSF,OPCONT  ;TEST IF PRESERVE FILE FLAG SET
2993 020616 001413                      BEQ      7$              ;NO SKIP
2994 020620 020027 000011 17$:   CMP      R0,#11         ;TEST IF R0 GREATER THAN 11
2995 020624 003010                      BGT      7$              ;IF YES, WE ALREADY HAVE A GOOD READ
2996 020626 012765 003660 000010      MOV      #BSSOFT,P.BALO(R5) ;ELSE SET UP TO READ SOFTWARE FILE
2997 020634 012703 000025                      MOV      #25,R3          ;SET UP MAX SECTOR VALUE
2998 020640 012700 000012                      MOV      #12,R0          ;SET UP STARTING SECTOR
2999 020644 000715                      BR        8$              ;GO READ FILE
3000 020646 104401 014746 7$:   TYPE    ,SERNUM         ;TYPE OUT SERIAL NUMBER
3001 020652 012746 004660                      MOV      #BSFACT,-(SP)   ;OF CARTIDGE TO BE FORMATTED
3002 020656 004737 040264                      JSR      PC,@#$DB20      ;CONVERT SERIAL NUMBER TO ASCII
3003 020662 012637 020670                      MOV      (SP)+,6$        ;GET RESULT
3004 020666 104401                      TYPE    ;PRINT IT
3005 020670 000000 6$:   .WORD   ;ADDRESS OF RESULT GOES HERE
3006 020672 104401 015413                      TYPE    ,STARS
3007 020676 005737 004666                      TST     BSFACT+6        ;TEST IF 3RD WORD ALL ZEROS
3008 020702 001404                      BEQ      10$             ;IF YES - SKIP
3009 020704 104401 015006                      TYPE    ,ALNPAC         ;ELSE TYPE ALIGNMENT PACK MESSAGE
3010                      ;AND ABORT
3011 020710 000137 016202                      JMP     START           ;JUMP TO START IF CONTINUE

```

```

3012 020714 153765 002653 000007 10$: BISB TYPFMT,P.CS1H(R5) ;SET TYPE & FORMAT AGAIN
3013 020722 132737 000001 002656 BITB #WHDSW,OPCONT ;TEST IF WE WILL BE WRITING
3014 020730 001441 BEQ 19$ ;NO - SKIP BSSOFT INIT
3015 020732 012700 003670 MOV #BSSOFT+10,R0 ;ELSE SET POINTER TO BAD SECTOR DATA
3016 020736 012701 000200 MOV #200,R1 ;SET A COUNT
3017 020742 132737 000040 002656 BITB #PSDBSF,OPCONT ;TEST IF PRESERVE FLAG SET
3018 020750 001413 BEQ 25$ ;NO - SKIP
3019 020752 005720 20$: TST (R0)+ ;TEST IF ALL ONES
3020 020754 100410 BMI 21$ ;YES - SKIP
3021 020756 005301 DEC R1 ;DECREMENT COUNT TO CHECK IT DOES NOT
3022 020760 001374 BNE 20$ ;OVERFLOW THE BUFFER
3023 020762 104401 014511 TYPE ,TDM$OF ;MESSAGE CAUSED BY OVERFLOW
3024 020766 104401 014576 TYPE ,TDM$UF
3025 020772 000137 016202 JMP START ;RESTART THE PROGRAM
3026 020776 005740 21$: TST -(R0) ;REPOSITION POINTER
3027 021000 012703 002660 25$: MOV #RDBUF,R3 ;SET POINTER TO SECTORS ENTERED AS BAD
3028 021004 012320 22$: MOV (R3)+,(R0)+ ;MOV IN THE ADDRESS
3029 021006 100412 BMI 19$ ;IF NEGATIVE, MOVE IS DONE
3030 021010 012320 MOV (R3)+,(R0)+ ;ELSE GET SECOND WORD
3031 021012 124141 CMPB -(R1),-(R1) ;DEC COUNT BY 2
3032 021014 005701 TST R1 ;CHECK IT DIDN'T OVERFLOW
3033 021016 100372 BPL 22$ ;NOT YET - LOOP
3034 021020 104401 014511 TYPE ,TDM$OF ;ELSE TYPE MESSAGE
3035 021024 104401 014576 TYPE ,TDM$UF
3036 021030 000137 016202 JMP START ;RESTART TEST
3037 021034 012700 003660 19$: MOV #BSSOFT,R0 ;GET ADDRESS OF BSSOFT
3038 021040 013720 004660 MOV BSFACT,(R0)+ ;GET FIRST WORD
3039 021044 013720 004662 MOV BSFACT+2,(R0)+ ;GET SECOND WORD
3040 021050 005020 CLR (R0)+ ;CLEAR NEXT TWO WORDS
3041 021052 005020 CLR (R0)+ ;AS REQUIRED
3042 021054 012701 000200 MOV #200,R1 ;SET COUNT
3043 021060 012700 004670 MOV #BSFACT+10,R0 ;GET ADDR OF START OF BAD SECTORS
3044 021064 005720 24$: TST (R0)+ ;TEST IF NEGATIVE
3045 021066 100410 BMI 23$ ;YES - SKIP
3046 021070 005301 DEC R1 ;ELSE DEC COUNT
3047 021072 100374 BPL 24$ ;IF NOT YET ZERO - LOOP
3048 021074 104401 014662 TYPE ,TDM$FAC ;MESSAGE REPORT
3049 021100 104401 014576 TYPE ,TDM$UF
3050 021104 000137 016202 JMP START ;GO RESTART PROGRAM
3051 021110 000137 021114 23$: JMP DSPTCH ;GO TO DISPATCHER
3052 *****
3053 .SBTTL DISPATCH ROUTINE
3054 *****
3055 DSPTCH:
3056 021114 132737 000001 002656 BITB #WHDSW,OPCONT ;TEST IF HDRS & DATA TO BE WRITTEN
3057 021122 001404 BEQ 1$
3058 021124 004737 022064 JSR PC,WRTOP ;GO TO WRITE HEADERS & DATA
3059 021130 104401 015300 TYPE ,PRGRS1 ;TYPE PROGRESS 1 MESSAGE
3060
3061 021134 132737 000016 002656 1$: BITB #VFHDSW!WCDASW!RCDASW,OPCONT ;ANY OTHER OPERATION?
3062 021142 001033 BNE 3$ ;YES-SKIP
3063 021144 104401 015345 4$: TYPE ,PRGRS2 ;TYPE DONE MESSAGE
3064 021150 132737 000001 002656 BITB #WHDSW,OPCONT ;TEST IF HEADERS AND DATA WERE WRITTEN
3065 021156 001402 BEQ 5$ ;NO - DON'T WRITE BAD SECTOR FILES
3066 021160 004737 031662 JSR PC,BDSTWT ;GO WRITE BAD SECTOR TABLES
3067 021164 032777 000200 157746 5$: BIT #SW7,@SWR ;TEST IF BAD SECTOR REPORT REQUESTED
  
```

```
3068 021172 001402          BEQ      2$
3069 021174 004737 032062    JSR      PC,BSREPT      ;GO REPORT BAD SECTORS
3070 021200 012706 001100    2$:     MOV      #STACK,SP ;RESET STACK
3071 021204 032777 000001 157726    BIT      #SW0,@SWR      ;TEST IF LOOP ON PROGRAM
3072 021212 001405          BEQ      6$              ;NO - GO GET NEXT PARAMETERS
3073 021214 012737 177777 002660    MOV      #177777,RDBUF
3074 021222 000137 020154    JMP      DRINIT          ;ELSE RESTART OPERATION
3075 021226 000137 016444    6$:     JMP      STPARM         ;GO GET NEXT PARAM SET
3076 021232 004737 023064    3$:     JSR      PC,VEROP    ;GO TO VERIFY HEADERS AND/OR DATA
3077 021236 000742          BR       4$
```

:SBTTL TRACK INCREMENT ROUTINE

```
3080 021240 032777 040000 157672 TRKINC: BIT      #SW14,@SWR ;TEST IF LOOP ON OPERATION
3081 021246 001011          BNE      2$              ;YES - BYPASS INCREMENT
3082 021250 020037 017312    CMP      R0,LCYL        ;CHECK IF NEXT INCREMENT WOULD
3083 021254 001007          BNE      1$              ;SELECT LAST TRACK ON THE
3084 021256 020127 000001    CMP      R1,#1          ;PACK. IF SO-DO NOT
3085 021262 001004          BNE      1$              ;INCREMENT
3086 021264 112737 000377 002651 3$:     MOVVB   #377,OPCOMP    ;SET DONE
3087 021272 000207          2$:     RTS      PC          ;& EXIT
3088 021274 005201          1$:     INC      R1          ;ELSE INCREMENT TRACK
3089 021276 020137 006354    CMP      R1,ETINUS      ;IF THIS BUMP DOES NOT
3090 021302 003773          BLE      2$              ;EXCEED THE SELECTED PACK
3091 021304 013701 006352    MOV      STINUS,R1      ;ADDRESS FORMAT-EXIT
3092 021310 005200          INC      R0              ;IF IT DOES, SET THE TRACK TO
3093 021312 020037 006360    CMP      R0,ECINUS      ;THE STARTING TRACK VALUES AND
3094 021316 101401          BLOS    4$              ;
3095 021320 000761          BR       3$              ;
3096 021322 020037 017312    4$:     CMP      R0,LCYL    ;SEE IF ON LAST CYL
3097 021326 001401          BEQ      5$              ;BR IF YES
3098 021330 000207          6$:     RTS      PC          ;ELSE RETURN
3099 021332 020127 000002    5$:     CMP      R1,#2      ;SEE IF ON TRACK 2
3100 021336 103774          BLO      6$              ;
3101 021340 000751          BR       3$
```

:SBTTL BUILD HEADERS ROUTINE

```
3102 021342          BLDHDR: SAVREG
3103 021342 104411          MOV      #26,R3          ;PRESET FOR 22 SECTOR FORMAT
3104 021344 012703 000026    MOV      #BUFF0,R2      ;GET BUFFER ADDRESS
3105 021350 012702 005674    ASL      R1              ;SHIFT TO ALIGN FOR HEADER
3106 021354 006301          ASL      R1
3107 021356 006301          ASL      R1
3108 021360 006301          ASL      R1
3109 021362 006301          ASL      R1
3110 021364 006301          ASL      R1
3111 021366 052701 140000    BIS      #140000,R1      ;SET GOOD SECTOR BITS
3112 021372 132737 000020 002653 BITB     #B.CFMT,TYPFMT ;TEST IF 20 SET FORMAT
3113 021400 001404          BEQ      1$              ;NO-SKIP
3114 021402 052701 001000    BIS      #BIT9,R1        ;ELSE SET FORMAT BIT
3115 021406 012703 000024    MOV      #24,R3          ;ADJUST SECTOR COUNT
3116 021412 010022          1$:     MOV      R0,(R2)+   ;INSERT CYLINDER WORD
3117 021414 010122          MOV      R1,(R2)+       ;INSERT TRK/SECT WORD
3118 021416 010004          MOV      R0,R4          ;COMPUTE VRC
```

```

3124 021420 010105      MOV      R1,R5
3125 021422 040005      BIC      R0,R5
3126 021424 040104      BIC      R1,R4
3127 021426 050504      BIS      R5,R4
3128 021430 010422      MOV      R4,(R2)+      ;INSERT VRC
3129 021432 005201      INC      R1            ;BUMP SECTOR COUNT
3130 021434 005303      DEC      R3            ;DEC COUNT
3131 021436 001365      BNE      1$           ;EXIT BUILD IF DONE
3132 021440 104412      RESREG
3133 021442 012737 100000 001250      MOV      #BIT15,$TMP5   ;SET BIT TO BE BAD SECTOR INDICATOR
3134 021450 012737 004660 021462      MOV      #BSFACT,2$    ;SET TABLE TO BE SEARCHED
3135 021456 004437 021556      JSR      R4,SRHTBS     ;GO SEARCH TABLE FOR BAD SECTORS
3136 021462 000000      .WORD
3137 021464 012603      MOV      (SP)+,R3      ;GET BAD SECTOR COUNT
3138 021466 001417      BEQ      4$           ;IF NO BAD SECTORS GO TEST BSSOFT
3139 021470 011602      MOV      (SP),R2      ;GET BAD SECTOR #
3140 021472 006302      ASL      R2            ;MULTIPLY BAD SECTOR # BY 6
3141 021474 006302      ASL      R2            ;AND ADD 2 TO GET INDEX
3142 021476 061602      ADD      (SP),R2      ;TO TRK/SEC WORD OF HEADER OF
3143 021500 062602      ADD      (SP)+,R2     ;BAD SECTOR
3144 021502 062702 000002      ADD      #2,R2
3145 021506 043762 001250 005674      BIC      $TMP5,BUFF0(R2) ;CLEAR BIT TO INDICATE BAD SECTOR
3146 021514 043762 001250 005676      BIC      $TMP5,BUFF0+2(R2) ;CLEAR BIT IN VRC FOR BAD SECTOR
3147 021522 005303      DEC      R3            ;DEC # OF BAD SECTOR COUNTER
3148 021524 001361      BNE      6$           ;IF 0
3149 021526 023727 021462 003660 4$:      CMP      2$,#BSSOFT   ;TEST IF BAD SECTOR SOFTWARE
3150 021534 001407      BEQ      5$           ;HAS BEEN TESTED. IF NOT, GO
3151 021536 012737 040000 001250      MOV      #BIT14,$TMP5
3152 021544 012737 003660 021462      MOV      #BSSOFT,2$   ;SET TABLE TO BE SEARCHED
3153 021552 000741      BR       3$
3154 021554      5$:
3155 021554 000207      RTS      PC           ;ELSE EXIT
3156
3157      ;*****
3158      .SBTTL SEARCH BAD SECTOR TABLES ROUTINE
3159      ;*THIS ROUTINE RETURNS A LIST OF BAD SECTORS FOUND IN
3160      ;*THE SPECIFIC TABLE (BAD SECTOR SOFTWARE OR BAD SECTOR FACTORY)
3161      ;*FOR THE SPECIFIC CYLINDER AND TRACK DESIRED. R0 AND R1 MUST
3162      ;*CONTAIN THE CYLINDER AND TRACK, RESPECTIVELY.
3163      ;*
3164      ;*THE LIST OF BAD SECTORS IS FOUND ON THE STACK WHEN THE ROUTINE
3165      ;*RETURNS TO THE CALLER. THE FIRST ENTRY ON THE STACK WILL BE THE
3166      ;*NUMBER OF BAD SECTORS FOUND FOR THIS CYLINDER AND TRACK.
3167      ;*****
3167 021556 010237 001242      SRHTBS: MOV      R2,$TMP2
3168 021562 010337 001244      MOV      R3,$TMP3
3169 021566 012637 001246      MOV      (SP)+,$TMP4   ;STORE RETURN CONTENTS OF R4
3170 021572 011402      MOV      (R4),R2      ;GET ADDRESS OF BAD SECTOR TABLE TO SEARCH
3171 021574 012437 001240      MOV      (R4)+,$TMP1   ;SETUP FOR LENGTH OF BAD SECTOR TABLE
3172 021600 062737 001000 001240      ADD      #1000,$TMP1
3173 021606 005003      CLR      R3            ;CLEAR R3 FOR COUNT
3174 021610 062702 000010      ADD      #10,R2       ;SET R2 FOR POINTER TO CYLINDER ENTRY
3175 021614 005712      1$:      TST      (R2)         ;TEST IF ALL ONES
3176 021616 100430      BMI      5$           ;YES-DONE
3177 021620 020012      CMP      R0,(R2)      ;TEST IF BAD SECTOR IN PRESENT CYL
3178 021622 001406      BEQ      3$           ;YES-GO CHECK TRACK
3179 021624 062702 000004      ADD      #4,R2       ;ELSE BUMP POINTER
    
```

```
3180 021630 020237 001240      CMP      R2,$TMP1      ;TEST IF OUT OF TABLE
3181 021634 002021      BGE      5$           ;YES-EXIT
3182 021636 000766      BR       1$           ;LOOP
3183 021640 005722      3$:     TST      (R2)+      ;TRACK CHECK - PUT POINTER AT TRK/SEC BYTE
3184 021642 011237 001256      MOV      (R2),$TMP10   ;GET TRK/SEC WORD
3185 021646 042737 174377 001256      BIC      #174377,$TMP10 ;CLEAR ALL BUT TRACK
3186 021654 123701 001257      CMPB    $TMP10+1,R1   ;CHECK IF BAD SECTOR IN THIS TRACK
3187 021660 001402      BEQ      4$           ;YES - GO PUT SECTOR NUMBER ON STACK
3188 021662 005722      TST      (R2)+      ;ELSE BUMP POINTER TO NEXT CYL WORD
3189 021664 000753      BR       1$           ;GO TEST NEXT CYL
3190 021666 005203      4$:     INC      R3           ;BUMP BAD SECTOR COUNT
3191 021670 012246      MOV      (R2)+,-(SP)  ;PUT TRK/SEC WORD ON STACK
3192 021672 042716 177700      BIC      #177700,(SP) ;CLEAR ALL BUT SECTOR NUMBER
3193 021676 000746      BR       1$           ;GO CHECK REST OF FILE
3194 021700 010346      5$:     MOV      R3,-(SP)   ;EXIT - PUT NUMBER OF BAD
3195 021702 013702 001242      MOV      $TMP2,R2     ;SECTORS ON STACK-RESTORE
3196 021706 013703 001244      MOV      $TMP3,R3     ;REGISTERS
3197 021712 013746 001246      MOV      $TMP4,-(SP)  ;PUT RETURN ON STACK
3198 021716 000204      RTS      R4           ;RETURN
```

```
::*****
:SBTTL BAD SECTOR CHECK ROUTINE
:*THIS ROUTINE WILL SEARCH BOTH TABLES TO DETERMINE IF A
:*SPECIFIC SECTOR IS LISTED AS BAD. IF THE SECTOR IS LISTED IN THE
:*TABLES THE ROUTINE SETS THE 'BADSEC' FLAG AND RETURNS. IF THE SECTOR
:*IS NOT LISTED THE FLAG IS RESET.
```

```
3206 021720 104411      BDSRCK: SAVREG
3207 021722 012737 004660 021734      MOV      #BSFACT,1$   ;SET TABLE TO SEARCH
3208 021730 004437 021556      JSR      R4,SRHTBS    ;GO SEARCH IT
3209 021734 000000      1$:     .WORD          ;TABLE ADDRESS GOES HERE
3210 021736 012603      MOV      (SP)+,R3     ;GET NUMBER OF BAD SECTORS, IF ANY
3211 021740 001015      BNE      6$           ;IF ANY, GO TEST WHICH ONES
3212 021742 023727 021734 003660 7$:     CMP      1$,#BSSOFT   ;ELSE TEST IF OTHER TABLE ALREADY SEARCHED
3213 021750 001404      BEQ      3$           ;IF YES-EXIT
3214 021752 012737 003660 021734      MOV      #BSSOFT,1$   ;SET OTHER TABLE FOR SEARCH
3215 021760 000763      BR       2$           ;GO SEARCH IT
3216 021762 042737 001000 006364 3$:     BIC      #BADSEC,RECODE ;CLEAR BAD SECTOR BIT
3217 021770 104412      4$:     RESREG
3218 021772 000207      RTS      PC           ;RETURN
3219 021774 022602      6$:     CMP      (SP)+,R2 ;THIS SECTOR IN TABLE?
3220 021776 001403      BEQ      8$           ;YES-GO SET BIT & EXIT
3221 022000 005303      DEC      R3           ;DECREMENT BAD SECTOR COUNT
3222 022002 001374      BNE      6$           ;IF NOT ZERO-CHECK NEXT ENTRY
3223 022004 000756      BR       7$           ;ELSE GO SEARCH OTHER TABLE
3224 022006 052737 001000 006364 8$:     BIS      #BADSEC,RECODE ;SET BAD SECTOR BIT
3225 022014 005303      9$:     DEC      R3           ;CLEAR STACK OF OTHER BAD SECTOR
3226 022016 001764      BEQ      4$           ;NUMBER
3227 022020 005726      TST      (SP)+
3228 022022 000774      BR       9$           ;EXIT
```

```
::*****
:SBTTL INSERT BAD SECTOR IN BAD SECTOR SOFTWARE TABLE ROUTINE
```

```
3232 022024 104411      BSSINS: SAVREG
3233 022026 012704 003670      MOV      #BSSOFT+10,R4 ;GET ADDRESS OF BAD SOFTWARE
3234 022032 005724      1$:     TST      (R4)+      ;SECTOR TABLE. FIND FIRST
3235 022034 100376      BPL      1$           ;EMPTY ENTRY
```

```
3236 022036 005744          TST      -(R4)          ;BACK UP POINTER
3237 022040 010024          MOV      R0,(R4)+      ;INSERT CYLINDER
3238 022042 012703 000010    MOV      #10,R3        ;SET COUNT FOR SHIFT TO ALIGN
3239 022046 006301          2$:     ASL      R1          ;SHIFT TRACK TO ALIGN FOR ENTRY
3240 022050 005303          DEC      R3            ;DEC SHIFT COUNT
3241 022052 001375          BNE      2$            ;TEST IF SHIFT DONE
3242 022054 060201          ADD      R2,R1         ;ADD IN SECTOR NUMBER
3243 022056 010114          MOV      R1,(R4)       ;INSERT TRACK/SECTOR
3244 022060 104412          RESREG
3245 022062 000207          RTS      PC
3246
3247          ;:*****
3248          ;SBTTL WRITE OPERATION CONTROL ROUTINE
3249          ;:*****
3249 022064 105037 002651          WRTOP:  CLR      OPCOMP      ;CLEAR DONE FLAG
3250 022070 012737 022146 001110    MOV      #WERROR,$LPERR ;SETUP FOR LOOP ON ERROR
3251 022076 013700 006356          MOV      SCINUS,R0     ;GET STARTING ADDRESS
3252 022102 013701 006352          MOV      STINUS,R1
3253 022106 010037 006370          WRTLUP: MOV      R0,CCINUS   ;STORE CURRENT CYL
3254 022112 010137 006366          MOV      R1,CTINUS    ;STORE CURRENT TRACK
3255 022116 032777 001000 157014    BIT      #SW9,@SWR     ;TEST IF LOOP ON ERROR
3256 022124 001002          BNE      1$            ;IF YES - DO NOT CLEAR ERROR FLAG
3257 022126 105037 001103          CLR      $ERFLG       ;ELSE CLEAR FLAG
3258 022132 004737 021342          1$:     JSR      PC,BLDHDR   ;GO BUILD HEADERS
3259 022136 004737 022206          JSR      PC,WRTHDR    ;GO WRITE HEADERS
3260 022142 004737 022312          JSR      PC,WRTTRK    ;GO WRITE DATA
3261 022146 032777 001000 156764    WERROR: BIT      #SW9,@SWR ;TEST IF LOOP ON ERROR
3262 022154 001406          BEQ      2$            ;IF NO - SKIP ERROR FLAG TEST
3263 022156 105737 001103          TSTB    $ERFLG       ;ELSE TEST ERROR FLAG
3264 022162 001403          BEQ      2$            ;NO - SKIP
3265 022164 004737 031336          JSR      PC,PREPLP    ;GO PREPARE FOR LOOP
3266 022170 000746          BR       WRTLUP        ;GO TO LOOP
3267 022172 004737 021240          2$:     JSR      PC,TRKINC    ;BUMP TO NEXT ADDRESS
3268 022176 105737 002651          TSTB    OPCOMP       ;WRITE OPERATION DONE?
3269 022202 001741          BEQ      WRTLUP       ;NO-LOOP
3270 022204 000207          RTS      PC            ;ELSE RETURN TO DISPATCH
3271
3272          ;:*****
3273          ;SBTTL WRITE HEADERS ROUTINE
3274          ;:*****
3274 022206          WRTHDR:
3275 022206 104411          SAVREG
3276 022210 005065 000014          CLR      P.PRST(R5)   ;CLEAR PROG STATUS REGISTER
3277 022214 112765 000127 000001    MOV      #WRHEAD,P.CMND(R5) ;LOAD PARAMETER BLOCK WITH
3278 022222 012765 005674 000010    MOV      #BUFF0,P.BALO(R5) ;VALUES TO DO THE WRITE HEADER.
3279 022230 010065 000002          MOV      R0,P.CYLN(R5)
3280 022234 012765 000102 000012    MOV      #102,P.WC(R5) ;LOAD THE WORD COUNT WITH
3281 022242 132737 000020 002653    BIT      #B.CFMT,TYPFMT ;THE PROPER VALUE FOR THE FORMAT
3282 022250 001403          BEQ      1$            ;TYPE
3283 022252 012765 000074 000012    MOV      #74,P.WC(R5)
3284 022260 005465 000012          1$:     NEG      P.WC(R5)
3285 022264 110165 000005          MOV      R1,P.TRCK(R5)
3286 022270 004737 025442          JSR      PC,DRVCAL    ;CALL DRIVER
3287 022274 032737 100000 006364    BIT      #ANYDER,RECODE ;TEST FOR ANY DATA ERRORS
3288 022302 001401          BEQ      2$
3289          ;INSERT CODE FOR ANY DATA TYPE ERROR ON WRITE HEADERS
3290 022304 000000          HALT
3291 022306 104412          2$:     RESREG
```

```

3292 022310 000207          RTS      PC          ;RETURN
3293                          ;*****
3294                          ;SBTTL WRITE ONE TRACK ROUTINE
3295                          ;*****
3296 022312          WRTRK:
3297 022312 104411          SAVREG
3298 022314 105037 002650 8$:  CLRB    DERCNT      ;CLEAR ERROR COUNT
3299 022320 013765 006306 000012  MOV    TKWDCT,P.WC(R5) ;SET WORD COUNT FOR ONE TRACK
3300 022326 105065 000004          CLRB    P.SECT(R5)   ;SET SECTOR ZERO
3301 022332 012765 006346 000010 4$:  MOV    #WEINUS,P.BALO(R5) ;SET BUFFER ADDRESS TO ODD
3302 022340 032700 000001          BIT    #BIT0,R0      ;OR EVEN CYLINDER DATA DEPENDING
3303 022344 001403          BEQ    1$           ;ON THE CYLINDER NUMBER
3304 022346 012765 006350 000010  MOV    #WOINUS,P.BALO(R5)
3305 022354 010065 000002 1$:  MOV    RO,P.CYLN(R5)   ;SET UP THE PARAMETER BLOCK
3306 022360 110165 000005          MOVB   R1,P.TRCK(R5)  ;TO WRITE THE ENTIRE TRACK
3307 022364 112765 000123 000001  MOVB   #WRDATA,P.CMND(R5)
3308 022372 005065 000014          CIR    P.PRST(R5)
3309 022376 052765 100000 000014  B.S    #DTBAIL,P.PRST(R5)
3310 022404 004737 025442          JSR    PC,DRVCAL     ;CALL DRIVER
3311 022410 032737 100000 006364  BIT    #ANYDER,RECODE ;ANY DATA ERROR?
3312 022416 001002          BNE   12$           ;YES - PROCESS ERROR
3313 022420 000137 023060          JMP    7$           ;NO-WRITE A-OK, EXIT
3314 022424 032737 000016 006364 12$: BIT    #HVR CER!OPIERR!BSERR,RECODE ;VRC OR OPI ERROR
3315 022432 001002          BNE   14$           ;YES-SKIP
3316 022434 000137 023046          JMP    6$           ;GO PRINT FUNNY MESSAGE
3317 022440 016537 000026 001252 14$: MOV    P.DTS(R5),$TMP6 ;STORE TRK/SEC
3318 022446 042737 174377 001252  BIC    #174377,$TMP6  ;CLEAR ALL BUT TRACK
3319 022454 113701 001253          MOVB   $TMP6+1,R1   ;STORE TRACK
3320 022460 016537 000022 001252  MOV    P.WCR(R5),$TMP6 ;STORE WORD COUNT
3321 022466 042737 000377 001252  BIC    #377,$TMP6   ;CLEAR PARTIAL COUNT
3322 022474 062737 070400 001252  ADD    #400,$TMP6   ;BUMP COUNT TO NEXT SECTOR
3323 022502 016502 000026          MOV    P.DTS(R5),R2 ;& GET BAD SECTOR
3324 022506 042702 177740          BIC    #177740,R2
3325 022512 032737 000002 006364  BIT    #BSERR,RECODE ;TEST IF BAD SECTOR
3326 022520 001076          BNE   2$:          ;GO SET UP TO WRITE REMAINDER IF TRACK
3327 022522 105237 002650 2$:  INCB   DERCNT      ;BUMP ERROR COUNT
3328 022526 112765 000123 000001  MOVB   #WRDATA,P.CMND(R5) ;SET UP PARAM BLOCK
3329 022534 012765 177400 000012  MOV    #177400,P.WC(R5) ;TO WRITE ONE SECTOR
3330 022542 010065 000002          MOV    RO,P.CYLN(R5) ;THIS IS THE RETRY
3331 022546 110165 000005          MOVB   R1,P.TRCK(R5)
3332 022552 110265 000004          MOVB   R2,P.SECT(R5)
3333 022556 052765 100000 000014  BIS    #DTBAIL,P.PRST(R5)
3334 022564 004737 025442          JSR    PC,DRVCAL
3335 022570 032737 100000 006364  BIT    #ANYDER,RECODE ;ERROR IN RETRY?
3336 022576 001441          BEQ    5$           ;NO-GO PRINT RETRY SUCCESSFUL
3337 022600 122737 000007 002650  CMPB   #7,DERCNT    ;TEST IF ERROR COUNT TO BIG
3338 022606 002345          BGE   2$           ;IF NO - GO RETRY AGAIN
3339 022610 005037 001174          CLR    $REG5
3340 022614 113737 002650 001174  MOVB   DERCNT,$REG5
3341 022622 104102          ERROR 102
3342 022624 004737 021720          JSR    PC,BDSRCK   ;GO TEST IF THIS SECTOR ALREADY MARKED BAD
3343 022630 032737 001000 006364  BIT    #BADSEC,RECODE ;TEST RESULTS OF TEST
3344 022636 001411          BEQ    11$          ;NOT BAD BEFORE - SKIP
3345 022640 010037 001174          MOV    RO,$REG5    ;SET UP TO PRINT ERROR
3346 022644 010137 001176          MOV    R1,$REG6
3347 022650 010237 001200          MOV    R2,$REG7
  
```

```

3348 022654 104107          ERROR 107          ;ERROR- BAD HEADER AREA
3349 022656 000137 016202    JMP     START
3350 022662 104076          11$:  ERROR 76          ;REPORT MARKING SECTOR BAD
3351 022664 004737 022024    JSR    PC,BSSINS    ;INSERT BAD SECTOR IN BSSOFT
3352 022670 004737 021342    JSR    PC,BLDHDR    ;BUILD NEW HEADERS
3353 022674 004737 022206    JSR    PC,WRTHDR    ;WRITE NEW HEADERS
3354 022700 000605          BR     8$           ;RESTART WRITE TRACK
3355 022702 005037 001174    5$:  CLR    $REG5      ;CLEAR REG 5 FOR COUNT
3356 022706 113737 002650 001174  MOVB   DEFCNT,$REG5 ;INSERT RETRY COUNT
3357 022714 104101          ERROR 101          ;PRINT RETRY SUCCESSFUL
3358 022716 005202          3$:  INC    R2         ;THIS CODE SETS UP THE
3359 022720 020237 006342    CMP    R2,SEINUS    ;PARAMETER BLOCK TO PICK
3360 022724 002055          BGE    7$           ;UP THE TRACK WRITE
3361 022726 010265 000004    MOV    R2,P.SECT(R5) ;AFTER THE BAD SECTOR
3362 022732 012704 000010    MOV    #10,R4
3363
3364
3365          ;THE FOLLOWING BLOCK OF CODE CHECKS THAT THE WORD COUNT IS
3366          ;CORRECTLY ADJUSTED TO PICK UP THE OPERATION AFTER THE SECTOR
3367          ;THAT IS CAUSING THE ERROR IS PROCESSED. IF THE WORD COUNT
3368          ;SOMEHOW GETS SCREWED UP THE PROGRAM HALTS. CONTINUE CAN BE
3369          ;DEPRESSED TO CAUSE THE WORD COUNT TO BE CORRECTED AND THE
3370          ;PROGRAM ADJUSTS THE COUNT AND CONTINUES THE OPERATION.
3370 022736 006302          9$:  ASL    R2
3371 022740 005304          DEC    R4
3372 022742 001375          BNE    9$
3373 022744 012704 013000    MOV    #13000,R4
3374 022750 023727 006342 000026  CMP    SEINUS,#26    ;ARE WE IN 26 SECTOR FORMAT?
3375 022756 001402          BEQ    13$          ;YES - SKIP
3376 022760 012704 012000    MOV    #12000,R4    ;ELSE CHANGE VALUE FOR 24 SECTOR FORMAT
3377 022764 160204          13$: SUB    R2,R4
3378 022766 005404          NEG    R4
3379 022770 016502 000004    MOV    P.SECT(R5),R2
3380 022774 020437 001252    CMP    R4,$TMP6
3381 023000 001413          BEQ    10$
3382 023002 010437 001204    MOV    R4,$REG11
3383 023006 013737 001252 001206  MOV    $TMP6,$REG12
3384 023014 010237 001202    MOV    R2,$REG10
3385 023020 104105          ERROR 105
3386 023022 000000          HALT
3387 023024 010437 001252    MOV    R4,$TMP6
3388
3389          ;END OF BLOCK OF CODE TO CHECK WORD COUNT
3390
3391 023030 013765 001252 000012 10$:  MOV    $TMP6,P.WC(R5)
3392 023036 105037 002650    CLRB   DEFCNT
3393 023042 000137 022332    JMP    4$
3394 023046 104401 015203    6$:  TYPE   ,IMPERR    ;TYPE IMPOSSIBLE ERROR
3395 023052 000000          HALT                ;HALT
3396 023054 000137 016202    JMP    START
3397 023060 104412          7$:  RESREG
3398 023062 000207          RTS    PC
3399          ;*****
3400          ;SBTTL VERIFY OPERATION CONTROL ROUTINE
3401          ;*****
3402 023064
3403 023064 012737 023200 001110  VEROP: MOV    #VERROR,$LPERR ;SET UP TO LOOP ON ERROR
  
```



```

3404 023072 013700 006356      MOV      SCINUS,R0      ;GET STARTING CYLINDER
3405 023076 013701 006352      MOV      STINUS,R1      ;AND TRACK
3406 023102 105037 002651      CLR      CLR      ;
3407 023106 010037 006370      VERLUP: MOV      RO,CCINUS ;STORE CURRENT CYL
3408 023112 010137 006366      MOV      R1,CTINUS      ;STORE CURRENT TRACK
3409 023115 032777 001000 156014 BIT      #SW9,@SWR      ;TEST IF LOOP ON ERROR
3410 023124 001002 001103      BNE      1$             ;IF YES - DON'T CLEAR ERROR FLAG
3411 023126 105037 001103      CLR      $ERFLG
3412 023132 004737 023240      1$:      JSR      PC,RDHDRS
3413 023136 004737 021342      JSR      PC,BLDHDR
3414 023142 004737 023576      JSR      PC,HDRCMP
3415 023146 132737 000010 002656 BIT      #RCDASW,OPCONT
3416 023154 001403 024740      BEQ      2$
3417 023156 004737 024740      JSR      PC,RDOP      ;GO DO READ FOR DATA VERIFICATION
3418 023162 000406 000004 002656 2$:      BR      VERROR
3419 023164 132737 000004 002656 2$:      BIT      #WCDASW,OPCONT ;CHECK IF WRITE CHECK DATA
3420 023172 001402 023714      BEQ      VERROR      ;IF NO - SKIP WRITE CHECK
3421 023174 004737 023714      JSR      PC,WCOPI      ;ELSE GO DO WRITE CHECK FOR DATA VERIF.
3422 023200 032777 001000 155732 VERROR: BIT      #SW9,@SWR ;TEST IF LOOP ON ERROR
3423 023206 001406 001103      BEQ      4$           ;IF NO - GO DO NEXT TRACK
3424 023210 105737 001103      TSTB     $ERFLG      ;ELSE CHECK ERROR FLAG
3425 023214 001403 031336      BEQ      4$           ;NO SKIP
3426 023216 004737 031336      JSR      PC,PREPLP    ;GO PREPARE FOR LOOPING
3427 023222 000731 021240      BR      VERLUP      ;LOOP
3428 023224 004737 021240      4$:      JSR      PC,TRKINC
3429 023230 105737 002651      TSTB     OPCOMP
3430 023234 001724 000207      BEQ      VERLUP
3431 023236 000207 000207      RTS      PC
3432
3433      ;*****
3434      ;SBTTL READ HEADERS ROUTINE
3435      ;*THE HEADERS FOR A GIVEN CYLINDER AND TRACK (SPECIFIED IN R0 AND R1,
3436      ;*RESPECTIVELY) ARE READ IN AND SORTED INTO THE REQUIRED SEQUENCE.
3437      ;*****
3438      RDHDRS:
3439      SAVREG
3440      JSR      PC,POSOFF ;GO POSITION AND OFFSET
3441      MOV      #RDBUF,P.BALO(R5) ;LOAD ADDRESS FOR HEADERS
3442      CLR      P,PRST(R5) ;CLEAR PROGRAM STATUS
3443      2$:      MOV      R0,P.CYLN(R5) ;SET UP TO READ HEADER
3444      MOV      R1,P.TRCK(R5) ;OF DESIRED TRACK & CYLINDER
3445      CLR      @STKS ;TURN OFF INTERRUPTS FROM KEYBOARD
3446      MOV      #RDALHD,P.CMND(R5)
3447      JSR      PC,DRVCAL ;DO IT
3448      MOV      #100,@STKS ;TURN ON KEYBOARD
3449      MOV      #204,$TMP0 ;SET UP TEMP STORE FOR THE NUMBER OF
3450      CMP      SEINUS,#26 ;HEADER WORDS THAT MUST BE STORED FOR
3451      BEQ      9$ ;THE FORMAT IN USE.
3452      MOV      #170,$TMP0
3453      9$:      CLR      R3
3454      4$:      CMP      RDBUF(R3),R0 ;TEST IF FIRST HEADER WORD IS RIGHT CYL
3455      BNE      3$ ;NO - GO BUMP TO NEXT
3456      MOV      RDBUF+2(R3),R4 ;GET TRACK/SECTOR WORD
3457      BIC      #177437,R4 ;CLEAR ALL BUT TRACK BITS
3458      ASR      R4 ;SHIFT TRACK TO ALIGN FOR TESTING
3459      ASR      R4
  
```

```

3460 023366 006204 ASR R4
3461 023370 006204 ASR R4
3462 023372 020401 CMP R4,R1 ;CHECK IF TRACK IS CORRECT
3463 023374 001014 BNE 3$ ;NO - GO BUMP TO NEXT HEADER
3464 023376 016304 002662 MOV RDBUF+2(R3),R4 ;GET SECOND WORD
3465 023402 016302 002660 MOV RDBUF(R3),R2 ;GET FIRST WORD
3466 023406 046304 002660 BIC RDBUF(R3),R4 ;COMPUTE VRC OF HEADER
3467 023412 046302 002662 BIC RDBUF+2(R3),R2
3468 023416 050402 BIS R4,R2
3469 023420 026302 002664 CMP RDBUF+4(R3),R2 ;CHECK IF VRC IS CORRECT
3470 023424 001414 BEQ 5$ ;YES - SKIP BUMP TO NEXT HEADER
3471 023426 062703 000006 3$: ADD #6,R3 ;BUMP TO NEXT HEADER
3472 023432 020337 001236 CMP R3,$TMP0 ;TEST IF END OF HEADERS REACHED
3473 023436 003741 BLE 4$ ;NO - GO TEST NEXT HEADER
3474 023440 010037 001202 MOV R0,$REG10 ;SET UP CYLINDER AND TRACK FOR REPORT
3475 023444 010137 001204 MOV R1,$REG11
3476 023450 104103 ERROR 103 ;REPORT ERROR
3477 023452 000137 016202 JMP START
3478 023456 016304 002662 5$: MOV RDBUF+2(R3),R4 ;GET TRACK/SECTOR AGAIN
3479 023462 042704 177740 BIC #177740,R4 ;CLEAR ALL BUT SECTOR
3480 023466 020437 006342 CMP R4,SEINUS ;TEST IF A VALID SECTOR NUMBER
3481 023472 002355 BGE 3$ ;NO - GO BUMP TO NEXT SECTOR
3482 023474 013702 006342 MOV SEINUS,R2 ;INITIALIZE COUNTER FOR MOV
3483 023500 010401 MOV R4,R1 ;SET THE INDEX VALUE BY MULT SECTOR READ BY 6
3484 023502 006304 ASL R4
3485 023504 006304 ASL R4
3486 023506 060104 ADD R1,R4
3487 023510 060104 ADD R1,R4
3488 023512 016364 002660 006100 6$: MOV RDBUF(R3),BUFF1(R4) ;MOV WORD ONE
3489 023520 016364 002662 006102 MOV RDBUF+2(R3),BUFF1+2(R4) ;MOV WORD TWO
3490 023526 016364 002664 006104 MOV RDBUF+4(R3),BUFF1+4(R4) ;MOV WORD THREE
3491 023534 005302 DEC R2 ;DECREMENT COUNT
3492 023536 001415 BEQ 8$ ;COUNT ZERO MOVE DONE, EXIT
3493 023540 062704 000006 ADD #6,R4 ;BUMP INDEX POINTERS TO NEXT HEADER
3494 023544 062703 000006 ADD #6,R3
3495 023550 020337 001236 CMP R3,$TMP0 ;TEST IF END OF RDBUF REACHED
3496 023554 002401 BLT 7$ ;NO - SKIP RESET TO START OF BUFFER
3497 023556 005003 CLR R3 ;ELSE RESET R3
3498 023560 020437 001236 7$: CMP R4,$TMP0 ;SAME TEST FOR BUFF1
3499 023564 002752 BLT 6$ ;IF NOT END OF BUFFER, GO DO NEXT MOVE
3500 023566 005004 CLR R4 ;ELSE RESET R4 TO START OF BUFFER
3501 023570 000750 BR 6$ ;THEN GO DO NEXT MOVE
3502 023572 104412 8$: RESREG
3503 023574 000207 RTS PC
3504 ::*****
3505 .SBTTL HEADER COMPARE ROUTINE
3506 ::*****
3507 HDRCMP:
3508 023576 104411 SAVREG
3509 023600 005002 CLR R2
3510 023602 013704 006342 MOV SEINUS,R4 ;GET NUM OF SECTORS
3511 023606 026262 005674 006100 1$: CMP BUFF0(R2),BUFF1(R2) ;COMPARE HDR WORD 1
3512 023614 001015 BNE 2$
3513 023616 026262 005676 006102 CMP BUFF0+2(R2),BUFF1+2(R2) ;COMPARE HDR WORD 2
3514 023624 001011 BNE 2$
3515 023626 026262 005700 006104 CMP BUFF0+4(R2),BUFF1+4(R2) ;COMPARE HDR WORD 3
  
```

```

3516 023634 001005
3517 023636 062702 000006 3$: BNE 2$
3518 023642 005304 ADD #6,R2 ;BUMP INDEX
3519 023644 001360 DEC R4 ;DEC COUNT
3520 023646 000420 BNE 1$ ;LOOP UNTIL ZERO
3521 023650 012703 001174 BR 4$ ;EXIT
3522 023654 016223 005674 2$: MOV #SREG5,R3 ;SET UP FOR ERROR REPORT
3523 023660 016223 005676 MOV BUFF0(R2),(R3)+
3524 023664 016223 005700 MOV BUFF0+2(R2),(R3)+
3525 023670 016223 006100 MOV BUFF0+4(R2),(R3)+
3526 023674 016223 006102 MOV BUFF1(R2),(R3)+
3527 023700 016223 006104 MOV BUFF1+2(R2),(R3)+
3528 023704 104062 ERROR 62 ;REPORT ERROR
3529 023706 000753 BR 3$ ;GO TEST NEXT HEADER
3530 023710 104412 4$: RESREG ;EXIT
3531 023712 000207 RTS PC
3532
3533 ;*****
3534 ;SBTTL DATA VERIFY WITH WRITE CHECK OPERATION ROUTINE
3535 ;*****
3536 023714 104411 WCOP:
3537 023716 013765 006306 000012 12$: SAVREG
3538 023724 005037 002650 MOV TKWDCT,P.WC(R5) ;SET WORD COUNT FOR 1 TRACK
3539 023730 105065 000004 CLR DERCNT ;CLEAR ERROR COUNT
3540 023734 004737 031262 5$: CLRB P.SECT(R5) ;SET TO SECTOR 0
3541 023740 012765 006346 000010 JSR PC,POSOFF ;GO POSITION AND OFFSET
3542 023746 032700 000001 MOV #WEINUS,P.BALO(R5) ;SET UP FOR EVEN OR ODD
3543 023752 001403 BIT #BIT0,R0 ;CYLINDER DATA
3544 023754 012765 006350 000010 BEQ 1$
3545
3546 023762 010065 000002 1$: MOV R0,P.CYLN(R5) ;SET UP PARAMETER BLOCK
3547 023766 110165 000005 MOV R1,P.TRCK(R5) ;FOR 1 TRACK WRITE CHECK
3548 023772 005065 000014 CLR P.PRST(R5)
3549 023776 052765 100000 000014 BIS #DTBAII,P.PRST(R5)
3550 024004 112765 000131 000001 MOV #WRTCHK,P.CMND(R5)
3551 024012 004737 025442 JSR PC,DRVCAL
3552 024016 032737 100000 006364 BIT #ANYDER,RECODE ;TEST IF ANY ERROR
3553 024024 001002 BNE 16$
3554 024026 000137 024566 JMP 15$ ;GO TO EXIT
3555 024032 016503 000022 16$: MOV P.WCR(R5),R3 ;STORE WORD COUNT & SECTOR
3556 024036 016502 000026 MOV P.DTS(R5),R2 ;NUMBER AT ERROR TIME
3557 024042 042702 177740 BIC #177740,R2 ;CLEAR ALL BUT SECTOR NUMBER
3558 024046 042703 000377 BIC #377,R3 ;MAKE THE WORD COUNT
3559 ;MODULO 400
3560 024052 032737 000016 006364 BIT #OPIERR!HVR CER!BSERR,RECODE ;TEST IF OPI OR HVRC
3561 024060 001013 BNE 4$ ;IF YES-SKIP
3562 024062 032737 000020 006364 BIT #DCKERR,RECODE ;TEST IF DATA CHECK
3563 024070 001402 BEQ 2$ ;IF NO, WD COUNT IS NOW CORRECT-SKIP
3564 024072 162703 000400 SUB #400,R3 ;ELSE INCREASE WORD COUNT
3565 024076 005702 2$: TST R2 ;TEST IF SECTOR NOW ZERO
3566 024100 001002 BNE 3$ ;IF NO-GO DECREMENT IT
3567 024102 013702 006342 MOV SEINUS,R2 ;ELSE GET MAX SECTOR +1 VALUE
3568 024106 005302 3$: DEC R2 ;NOW DECREMENT IT
3569 024110 032737 000002 006364 4$: BIT #BSERR,RECODE ;TEST IF BAD SECTOR
3570 024116 001453 BEQ 6$ ;IF YES - GO CONTINUE TRACK
3571 024120 005202 8$: INC R2 ;ELSE BUMP TO NEXT SECTOR

```

```

3572 024122 020237 006342          CMP      R2,SEINUS          ;IF NOT AT LAST SECTOR -
3573 024126 002402                   BLT      19$                ;GO TO EXIT IF LAST SECTOR IN ERROR
3574 024130 000137 024566          JMP      15$
3575 024134 105037 002650          19$:   CLRB   DERCNT            ;CLEAR ERROR COUNT
3576 024140 010265 000004          MOV      R2,P.SECT(R5)     ;SET UP FOR CONTINUE OF WC
3577 024144 062703 000400          ADD      #400,R3           ;BUMP COUNT TO NEXT SECTOR
3578 024150 012704 000010          MOV      #10,R4
3579
3580                                     ;THE FOLLOWING BLOCK OF CODE CHECKS THAT THE WORD COUNT IS
3581                                     ;CORRECTLY ADJUSTED TO PICK UP THE OPERATION AFTER THE SECTOR
3582                                     ;THAT IS CAUSING THE ERROR IS PROCESSED. IF THE WORD COUNT
3583                                     ;SOMEHOW GETS SCREWED UP THE PROGRAM HALTS. CONTINUE CAN BE
3584                                     ;DEPRESSED TO CAUSE THE WORD COUNT TO BE CORRECTED AND THE
3585                                     ;PROGRAM ADJUSTS THE COUNT AND CONTINUES THE OPERATION.
3586 024154 006302          17$:   ASL      R2
3587 024156 005304          DEC      R4
3588 024160 001375          BNE     17$
3589 024162 012704 013000          MOV      #13000,R4
3590 024166 023727 006342 000026          CMP      SEINUS,#26       ;TEST IF WE ARE IN 26 SECTOR FORMAT
3591 024174 001402          BEQ     20$                ;YES - SKIP
3592 024176 012704 012000          MOV      #12000,R4       ;ELSE SET COUNT FOR 24 SECTOR FORMAT
3593 024202 160204          20$:   SUB      R2,R4
3594 024204 005404          NEG      R4
3595 024206 016502 000004          MOV      P.SECT(R5),R2
3596 024212 020403          CMP      R4,R3
3597 024214 001411          BEQ     18$
3598 024216 010437 001204          MOV      R4,$REG11
3599 024222 010337 001206          MOV      R3,$REG12
3600 024226 010237 001202          MOV      R2,$REG10
3601 024232 104105          ERROR   105
3602 024234 000000          HALT
3603 024236 010403          MOV      R4,R3
3604
3605                                     ;END OF CODE TO CHECK WORD COUNT
3606
3607 024240 010365 000012          18$:   MOV      R3,P.WC(R5)     ;AT NEXT SECTOR
3608 024244 000633          BR      5$                 ;GO DO IT
3609 024246 105237 002650          6$:   INCB   DERCNT            ;RETRY SEQUENCE-
3610 024252 004737 031262          JSR     PC,POSOFF          ;GO POSITION AND OFFSET
3611 024256 112765 000131 000001          MOVB   #WRTCHK,P.CMND(R5) ;COUNT THE ERROR
3612 024264 010065 000002          MOV      R0,P.CYLN(R5)   ;SET UP PARAM BLOCK
3613 024270 110165 000005          MOVB   R1,P.TRCK(R5)
3614 024274 110265 000004          MOVB   R2,P.SECT(R5)
3615 024300 012765 177400 000012          MOV      #177400,P.WC(R5)
3616 024306 052765 100000 000014          BIS      #DTBAII,P.PRST(R5)
3617 024314 004737 025442          JSR     PC,DRVCAL
3618 024320 032737 100000 006364          BIT      #ANYDER,RECODE   ;TEST IF ANY ERROR
3619 024326 001507          BEQ     14$                ;GO TYPE RETRY SUCCESSFUL IF NO
3620 024330 022737 000007 002650          CMP      #7,DERCNT        ;ELSE TEST IF ENOUGH RETRIES
3621 024336 002343          BGE     6$                 ;NO-LOOP
3622 024340 032737 000100 006364          BIT      #WCERR,RECODE    ;TEST IF WCE, IF
3623 024346 001446          BEQ     10$                ;NO GO CHECK FOR DATA CHECK
3624 024350 004737 031262          JSR     PC,POSOFF          ;GO POSITION AND OFFSET
3625 024354 112765 000121 000001          MOVB   #RDDATA,P.CMND(R5) ;ELSE DO A READ OF SECTOR
3626 024362 012765 177400 000012          MOV      #177400,P.WC(R5) ;IN ERROR
3627 024370 010065 000002          MOV      R0,P.CYLN(R5)

```

```

3628 024374 110165 000005      MOVB   R1,P.TRCK(R5)
3629 024400 110265 000004      MOVB   R2,P.SECT(R5)
3630 024404 005065 000014      CLR    P.PRST(R5)      ;CLEAR PROGRAM STATUS WORD (BAII)
3631 024410 012765 002660 000010      MOV    #RDBUF,P.BALO(R5)
3632 024416 004737 025442      JSR    PC,DRVCAL
3633 024422 032737 100000 006364      BIT    #ANYDER,RECODE  ;IF NO ERROR ON READ
3634 024430 001442      BEQ    13$
3635 024432 032737 000020 006364      BIT    #DCKERR,RECODE  ;OR NO DATA CHECK
3636 024440 001436      BEQ    13$      ;GO PRINT FUNNY MESSAGE
3637 024442 004737 024572      JSR    PC,DACOMP     ;ELSE COMPARE DATA & PRINT ERRORS
3638 024446 032737 000040 006364 9$:      BIT    #ECCNC,RECODE  ;CHECK IF NON-CORRECTABLE
3639 024454 001007      BNE    11$      ;IF YES-GO TO BAD SECTOR PROCESSING
3640 024456 104401 015256      TYPE   ,CORABL      ;ELSE TYPE CORRECTABLE MESSAGE
3641 024462 000616      BR     8$      ;& GO WC REST OF TRACK
3642 024464 032737 000020 006364 10$:     BIT    #DCKERR,RECODE  ;THIS IS TEST IF DAK WITH NO WCE
3643 024472 001365      BNE    9$      ;IF YES-GO TEST FOR CORRECTABLE
3644 024474 005037 001174 11$:     CLR    $REG5
3645 024500 113737 002650 001174      MOVB   DERCNT,$REG5
3646 024506 104102      ERROR  102
3647 024510 104076      ERROR  76      ;REPORT MARKING SECTOR BAD
3648 024512 004737 022024      JSR    PC,BSSINS     ;PUT BAD SECTOR IN BAD SECTOR
3649 024516 004737 021342      JSR    PC,BLDHDR     ;TABLE, COMPUTE NEW HEADERS,
3650 024522 004737 022206      JSR    PC,WRTHDR     ;WRITE THEM, REWRITE THE
3651 024526 004737 022312      JSR    PC,WRTTRK    ;TRACK
3652 024532 000137 023716      JMP    12$      ;& REDO THE WRITE CHECK
3653 024536 104401 015111 13$:     TYPE   ,IMPER2     ;TYPE FUNNY MESSAGE
3654 024542 000137 023716      JMP    12$      ;GO RESTART THE WRITE CHECK
3655 024546 005037 001174 14$:     CLR    $REG5
3656 024552 113737 002650 001174      MOVB   DERCNT,$REG5  ;GET RETRY COUNT
3657 024560 104101      ERROR  101      ;RETRY SUCCESSFUL MESSAGE
3658 024562 000137 024120      JMP    8$
3659 024566 104412 15$:     RESREG
3660 024570 000207      RTS    PC      ;EXIT
  
```

```

:*****
.SBTTL DATA COMPARE ROUTINE
:*****
  
```

```

3664 024572      DACOMP:
3665 024572 104411      SAVREG
3666 024574 005005      CLR    R5      ;CLEAR R5 FOR ERROR COUNTING
3667 024576 012703 006346      MOV    #WEINUS,R3  ;GET THE WORD THAT SHOULD BE
3668 024602 032700 000001      BIT    #BIT0,R0    ;WRITTEN THROUGHOUT THE CYLINDER
3669 024606 001402      BEQ    1$
3670 024610 012703 006350      MOV    #WOINUS,R3
3671 024614 012704 002660 1$:      MOV    #RDBUF,R4   ;SET POINTER TO READ DATA
3672 024620 010037 001174      MOV    R0,$REG5    ;STORE PACK ADDRESS FOR REPORT
3673 024624 010137 001176      MOV    R1,$REG6
3674 024630 010237 001200      MOV    R2,$REG7
3675 024634 012701 000001      MOV    #1,R1      ;PRESET REGISTERS
3676 024640 012702 000001      MOV    #1,R2      ;FOR COUNTING
3677 024644 012700 000400      MOV    #400,R0
3678 024650 021324 2$:      CMP    (R3),(R4)+  ;COMPARE DATA
3679 024652 001005      BNE    3$      ;GO REPORT IF NOT EQUAL
3680 024654 005202 5$:      INC    R2      ;ELSE BUMP COUNTERS
3681 024656 005300      DEC    R0
3682 024660 001373      BNE    2$
3683 024662 104412 6$:      RESREG
  
```

```
3684 024664 000207          RTS      PC
3685 024666 005205          3$: INC   R5          ;BUMP ERROR COUNT
3686 024670 005301          DEC   R1          ;BUMP R1 TO ZERO
3687 024672 011337 001202    MOV   (R3), $REG10 ;STORE GOOD, BAD, & WORD NUM
3688 024676 014437 001204    MOV   -(R4), $REG11
3689 024702 010237 001206    MOV   R2, $REG12
3690 024706 005724          TST   (R4)+       ;BUMP R4 TO NEXT DATA WORD
3691 024710 005701          TST   R1          ;TEST R1 SWITCH
3692 024712 001001          BNE   4$          ;IF NOT ZERO, THIS IS NOT FIRST PASS
3693 024714 104034          ERROR  34        ;TYPE HEADING
3694 024716 032777 000002 154214 4$: BIT   #SW1, @SWR   ;TEST IF NO LIMIT ON ERRORS
3695 024724 001003          BNE   7$          ;SWITCH 1 OFF - LIMIT TO 10
3696 024726 020527 000010    CMP   R5, #10    ;ELSE TEST IF 10 ERRORS REPORTED
3697 024732 001753          BEQ   6$          ;YES - EXIT
3698 024734 104063          7$: ERROR  63        ;TYPE DATA
3699 024736 000746          BR    5$          ;GO CONTINUE COMPARE
3700
3701          ;*****
3702          .SBTTL DATA VERIFY WITH READ OPERATION ROUTINE
3703          ;*****
3704          RDOP:
3704 024740 104411          SAVREG
3705 024742 105065 000004    CLRB  P.SECT(R5)  ;SET UP TO READ A FULL TRACK
3706 024746 105037 002650    CLRB  DERCNT      ;WITH BUS INCREMENT INHIBIT
3707 024752 013765 006306 000012    MOV   TKWDCT, P.WC(R5)
3708 024760 004737 031262          JSR   PC, POSOFF  ;GO POSITION AND OFFSET
3709 024764 010065 000002    MOV   R0, P.CYLN(R5)
3710 024770 110165 000005    MOV   R1, P.TRCK(R5)
3711 024774 005065 000014    CLR   P.PRST(R5)
3712 025000 052765 100000 000014    BIS   #DTBAII, P.PRST(R5)
3713 025006 112765 000121 000001    MOV   #RDDATA, P.CMND(R5)
3714 025014 004737 025442          JSR   PC, DRVCAL  ;DO IT
3715 025020 032737 100000 006364    BIT   #ANYDER, RECODE ;TEST IF ANY DATA ERROR
3716 025026 001002          BNE   11$        ;READ A-OK, EXIT
3717 025030 000137 025436          JMP   8$
3718 025034 016537 000022 001254 11$: MOV   P.WCR(R5), $TMP7 ;STORE WORD COUNT AT ERROR
3719 025042 016502 000026          MOV   P.DTS(R5), R2 ;STORE SECTOR & TRACK
3720 025046 042702 177400          BIC   #177400, R2  ;CLEAR ALL BUT SECTOR
3721 025052 062737 000400 001254    ADD   #400, $TMP7  ;BUMP STORED WORD COUNT BY 1 SECTOR
3722 025060 032737 000002 006364    BIT   #BSERR, RECODE ;CHECK IF BAD SECTOR
3723 025066 001106          BNE   5$          ;YES - GO CONTINUE READ
3724 025070 032737 000020 006364    BIT   #DCKERR, RECODE ;CHECK IF DATA CHECK
3725 025076 001410          BEQ   3$          ;IF YES, DECREMENT SECTOR COUNT BY
3726 025100 005702          TST   R2          ;1 TO GET SECTOR IN ERROR BUT BE
3727 025102 001002          BNE   1$          ;SURE SECTOR COUNT IS NOT ZERO
3728 025104 013702 006342          MOV   SEINUS, R2
3729 025110 005302          1$: DEC   R2
3730 025112 162737 000400 001254    SUB   #400, $TMP7  ;IF DCK, ADD WORD COUNT BACK IN
3731 025120 105237 002650          3$: INCB  DERCNT      ;BUMP ERROR COUNT
3732 025124 004737 031262          JSR   PC, POSOFF  ;GO POSITION AND OFFSET
3733 025130 112765 000121 000001    MOV   #RDDATA, P.CMND(R5) ;ELSE SET UP TO RETRY
3734 025136 010065 000002          MOV   R0, P.CYLN(R5)
3735 025142 110165 000005          MOV   R1, P.TRCK(R5)
3736 025146 110265 000004          MOV   R2, P.SECT(R5)
3737 025152 012765 177400 000012    MOV   #177400, P.WC(R5)
3738 025160 012765 001240 000010    MOV   #$TMP1, P.BALO(R5)
3739 025166 052765 100000 000014    BIS   #DTBAII, P.PRST(R5)
```

```
3740 025174 004737 025442 JSR PC,DRVCL
3741 025200 032737 100000 006364 BIT #ANYDER,RECODE ;ANY DATA ERROR IN RETRY?
3742 025206 001504 BEQ 7$ ;NO-GO TYPE RETRY SUCCESSFUL
3743 025210 122737 000007 002650 CMPB #7,DERCNT ;ELSE CHECK IF ERROR COUNT TO BIG
3744 025216 002340 BGE 3$ ;IF YES-TYPE UNSUCCESSFUL
3745 025220 005037 001174 CLR $REG5
3746 025224 113737 002650 001174 MOVB DERCNT,$REG5
3747 025232 104102 ERROR 102
3748 025234 032737 000020 006364 BIT #DCKERR,RECODE ;TEST IF THIS IS A DATA ERROR
3749 025242 001420 BEQ 5$ ;IF NO - SKIP ECC PRINTOUT
3750 025244 016537 000060 001174 MOV P.EPOS(R5),$REG5 ;STORE ECC PATTERN AND POSITION
3751 025252 016537 000062 001174 MOV P.EPAT(R5),$REG6 ;FOR ERROR REPORT
3752 025260 005037 001200 CLR $REG7 ;SET $REG7 FOR ECC UNCORRECTABLE
3753 025264 032737 000040 006364 BIT #ECCNC,RECODE ;WAS THAT CORRECT?
3754 025272 001003 BNE 4$ ;YES - GO REPORT
3755 025274 052737 000001 001200 BIS #1,$REG7 ;NO - SET FOR CORRECTABLE
3756 025302 104077 4$: ERROR 77 ;REPORT ERROR
3757 025304 005202 5$: INC R2 ;SET UP TO CONTINUE ON NEXT SECTOR
3758 025306 020237 006342 CMP R2,SEINUS ;CHECK IF ERROR SECTOR WAS LAST SECTOR
3759 025312 002051 BGE 8$ ;EXIT IF YES
3760 025314 010265 000004 MOV R2,P.SECT(R5) ;READ OF REST OF TRACK BY
3761 025320 012704 000010 MOV #10,R4
3762 025324 006302 9$: ASL R2
3763 025326 005304 DEC R4
3764 025330 001375 BNE 9$
3765 025332 012704 013000 MOV #13000,R4
3766 025336 160204 SUB R2,R4
3767 025340 005404 NEG R4
3768 025342 016502 000004 MOV P.SECT(R5),R2
3769 025346 020437 001254 CMP R4,$TMP7
3770 025352 001413 BEQ 10$
3771 025354 010437 001204 MOV R4,$REG11
3772 025360 013737 001254 001206 MOV $TMP7,$REG12
3773 025366 010237 001202 MOV R2,$REG10
3774 025372 104105 ERROR 105
3775 025374 000000 HALT
3776 025376 010437 001254 MOV R4,$TMP7
3777 025402 013765 001254 000012 10$: MOV $TMP7,P.WC(R5) ;BUMPING TO THE NEXT SECTOR
3778 025410 105037 002650 CLR B DERCNT ;INSERTING THE WORD COUNT &
3779 025414 000137 024760 JMP 6$ ;GO START RD DATA
3780 025420 005037 001174 7$: CLR $REG5 ;CLEAR REG 5 FOR RETRY COUNT BYTE
3781 025424 113737 002650 001174 MOVB DERCNT,$REG5 ;GET RETRY COUNT
3782 025432 104101 ERROR 101 ;REPORT RETRY SUCCESSFUL
3783 025434 000723 BR 5$
3784 025436 104412 8$: RESREG
3785 025440 000207 RTS PC
3786 *****
3787 .SBTTL CALL DRIVER ROUTINE
3788 ;*ENTRY JSR PC,DRVCL
3789 ;* WITH R5 POINTING TO PARAMETER BLOCK
3790 ;*RETURN RTS PC
3791 ;*
3792 ;*THIS ROUTINE IS USED TO INITIATE A SUBSYSTEM OPERATION BY
3793 ;*CALLING THE DRIVER. THE PARAMETER BLOCK MUST BE SET UP
3794 ;*BY THE CALLER AND R5 MUST POINT TO THE PARAMETER
3795 ;*BLOCK WHEN THE ROUTINE IS CALLED.
```


3852	025604	104411			TYPERR: SAVREG		
3853	025606	032777	020000	153324	BIT	#SW13,@SWR	:INHIBIT ERROR TYPEOUTS?
3854	025614	001107			BNE	20\$:YES-BRANCH
3855	025616	113700	001114		MOVB	\$ITEMB,R0	:ENTER ERROR NUMBER
3856	025622	042700	177400		BIC	#177400,R0	:CLEAR UPPER BITS
3857	025626	005300			DEC	R0	:FORM INDEX FOR ERROR TABLE
3858	025630	006300			ASL	R0	
3859	025632	006300			ASL	R0	
3860	025634	006300			ASL	R0	
3861	025636	062700	001272		1\$: ADD	#\$ERRTB,R0	:FORM ADDRESS OF ERROR ENTRY
3862	025642	012037	025656		MOV	(R0)+,2\$:GET EM POINTER
3863	025646	001404			BEQ	3\$:BRANCH IF THERE ISN'T ONE
3864	025650	104401	001267		TYPE	,\$CRLF	:TYPE CARRIAGE RETURN LINE FEED
3865	025654	104401			TYPE		:TYPE ERROR MESSAGE (EM)
3866	025656	000000			2\$: .WORD	0	:EM POINTER GOES HERE
3867	025660	012037	025674		3\$: MOV	(R0)+,4\$:GET DH POINTER
3868	025664	001404			BEQ	5\$:BRANCH IF THERE ISN'T ONE
3869	025666	104401	001267		TYPE	,\$CRLF	:TYPE CR-LF
3870	025672	104401			TYPE		:TYPE DATA HEADER
3871	025674	000000			4\$: .WORD	0	:DH POINTER GOES HERE
3872	025676	012001			5\$: MOV	(R0)+,R1	:GET DT POINTER
3873	025700	001455			BEQ	20\$:BRANCH IF THERE ARE NONE
3874	025702	005004			CLR	R4	:SET INDENT SWITCH
3875	025704	012000			MOV	(R0)+,R0	:GET DF POINTER
3876	025706	012002			MOV	(R0)+,R2	:STORE NUMBER OF DH'S
3877	025710	001446			BEQ	17\$:DH NUM IS 0-BRANCH
3878	025712	005104			COM	R4	:NO INDENT
3879	025714	104401	001267		TYPE	,\$CRLF	
3880	025720	112003			10\$: MOVB	(R0)+,R3	:GET & STORE NUMBER OF DATA WORDS
3881	025722	105720			TSTB	(R0)+	:BUMP PAST FORMAT WORD
3882	025724	005703			TST	R3	:TEST IF ANY DATA FOR THIS HEADER
3883	025726	001407			BEQ	14\$:NO - SKIP DATA PRINT
3884	025730	013146			11\$: MOV	@(R1)+,-(SP)	:PUT FIRST DATA WORD ON STACK
3885	025732	104402			TYPOC		:TYPE IT
3886	025734	005303			DEC	R3	:MORE DATA WORDS
3887	025736	001403			BEQ	14\$:NO-BRANCH
3888	025740	104401	012765		TYPE	,\$SPACE2	:TYPE SEPARATORS
3889	025744	000771			BR	11\$:LOOP
3890	025746	005302			14\$: DEC	R2	:MORE DH'S?
3891	025750	003431			BLE	20\$:NO-BRANCH
3892	025752	104401	001267		TYPE	,\$CRLF	
3893	025756	005760	000002		TST	2(R0)	:ONLY A DH IN THIS REQUEST?
3894	025762	001404			BEQ	15\$:YES-BRANCH BYPASS INDENT
3895	025764	005104			COM	R4	:INDENT?
3896	025766	001002			BNE	15\$:NO-BRANCH
3897	025770	104401	012765		TYPE	,\$SPACE2	:YES-TYPE SPACES
3898	025774	012037	026002		15\$: MOV	(R0)+,16\$:GET NEXT DH POINTER
3899	026000	104401			TYPE		:TYPE DH
3900	026002	000000			16\$: .WORD	0	:DH POINTER GOES HERE
3901	026004	105710			TSTB	(R0)	:TYPE A DT?
3902	026006	001003			BNE	21\$:YES-BRANCH
3903	026010	062700	000002		ADD	#2,R0	:INCREMENT DF POINTER
3904	026014	000754			BR	14\$:SEE IF END OF DF BLOCK
3905	026016	104401	001267		21\$: TYPE	,\$CRLF	
3906	026022	005704			TST	R4	:INDENT?
3907	026024	001335			BNE	10\$:NO-BRANCH

```

3908 026026 104401 012765 17$: TYPE SPACE2 ;YES-TYPE SPACES
3909 026032 000732 BR 10$ ;LOOP
3910 026034 104412 20$: RESREG
3911 026036 000207 RTS PC
3912
3913
3914
3915
3916
3917
3918
3919
3920
3921
3922
3923 026040 104411
3924 026042 152737 000377 002652
3925 026050 105237 002654
3926 026054 032765 001000 000014
3927 026062 001403
3928 026064 004737 026302
3929 026070 000402
3930 026072 004737 030560 12$: JSR PC, TOPROC ;LOAD RK REGS INTO $REGS
3931 026076 032737 000001 002604 13$: BIT #BIT0,E.CONT ;ERROR 0:
3932 026104 001402 BEQ 1$ ;NO-BRANCH
3933 026106 104064 ERROR 64 ;CLEAR CONT DID NOT CLEAR ERROR
3934 026110 000463 BR 7$
3935 026112 032737 000002 002604 1$: BIT #BIT1,E.CONT ;ERROR 1?
3936 026120 001402 BEQ 2$ ;NO-BRANCH
3937 026122 104065 ERROR 65 ;NO ATTENTION IN ATTENTION SUM REG
3938 026124 000455 BR 7$
3939 026126 032737 000004 002604 2$: BIT #BIT2,E.CONT ;ERROR 2?
3940 026134 001402 BEQ 3$ ;NO-BRANCH
3941 026136 104066 ERROR 66 ;UNSOLICITED ATTENTION
3942 026140 000447 BR 7$
3943 026142 032737 000010 002604 3$: BIT #BIT3,E.CONT ;ERROR 3?
3944 026150 001402 BEQ 4$ ;NO-BRANCH
3945 026152 104067 ERROR 67 ;UNEXPECTED DATA TYPE ERROR
3946 026154 000441 BR 7$
3947 026156 032737 000020 002604 4$: BIT #BIT4,E.CONT ;ERROR 4?
3948 026164 001402 BEQ 5$ ;NO-BRANCH
3949 026166 104070 ERROR 70 ;ATTENTION DID NOT RESET WITH CLEAR
3950 026170 000433 BR 7$
3951 026172 032737 000040 002604 5$: BIT #BIT5,E.CONT ;ERROR 5?
3952 026200 001402 BEQ 6$ ;NO-BRANCH
3953 026202 104071 ERROR 71 ;SUBSYS CLEAR DIDN'T CLEAR DRIVE ATTENTION
3954 026204 000425 BR 7$

```

```

*****
:SBTTL CONTROLLER ERROR REPORTER ROUTINE
:*ENTRY: JSR PC, CONERR
:*RETURN: RTS PC
:*
:*THIS ROUTINE DECODES THE CONTROLLER ERROR WORD AND
:*REPORTS THE APPROPRIATE MESSAGE. THE RK611 REGISTERS ARE
:*RETRIEVED FROM THE RK611 AND PLACED IN THE PARAMETER
:*BLOCK. THIS IS DONE BECAUSE PARM 0 MAY NOT BE VALID
:*AT THIS TIME.
*****

```

```

CONERR: SAVREG
BISB #377,DONE ;SET DONE FLAG
INCB ERRCNT ;BUMP ERROR COUNTER
BIT #PBSVAL,P.PRST(R5) ;TEST IF STATUS VALID
BEQ 12$ ;YES - SKIP
JSR PC,REPSUP ;CALL FOR GETTING STATUS
BR 13$
12$: JSR PC, TOPROC ;LOAD RK REGS INTO $REGS
13$: BIT #BIT0,E.CONT ;ERROR 0:
BEQ 1$ ;NO-BRANCH
ERROR 64 ;CLEAR CONT DID NOT CLEAR ERROR
BR 7$
1$: BIT #BIT1,E.CONT ;ERROR 1?
BEQ 2$ ;NO-BRANCH
ERROR 65 ;NO ATTENTION IN ATTENTION SUM REG
BR 7$
2$: BIT #BIT2,E.CONT ;ERROR 2?
BEQ 3$ ;NO-BRANCH
ERROR 66 ;UNSOLICITED ATTENTION
BR 7$
3$: BIT #BIT3,E.CONT ;ERROR 3?
BEQ 4$ ;NO-BRANCH
ERROR 67 ;UNEXPECTED DATA TYPE ERROR
BR 7$
4$: BIT #BIT4,E.CONT ;ERROR 4?
BEQ 5$ ;NO-BRANCH
ERROR 70 ;ATTENTION DID NOT RESET WITH CLEAR
BR 7$
5$: BIT #BIT5,E.CONT ;ERROR 5?
BEQ 6$ ;NO-BRANCH
ERROR 71 ;SUBSYS CLEAR DIDN'T CLEAR DRIVE ATTENTION
BR 7$

```

```

3955 026206 032737 000400 002604 6$: BIT #BIT8,E.CONT
3956 026214 001401 BEQ 8$
3957 026216 104072 ERROR 72 ;DATA LATE WHEN UNLOADING HEADER
3958 026220 032737 001000 002604 8$: BIT #BIT9,E.CONT
3959 026226 001401 BEQ 9$
3960 026230 104073 ERROR 73 ;CONTROLLER ERROR DURING DRIVER SERVICE
3961 026232 032737 002000 002604 9$: BIT #BIT10,E.CONT
3962 026240 001401 BEQ 10$
3963 026242 104074 ERROR 74 ;DRIVE DETECTED PARITY ERROR
3964 026244 032737 100000 002604 10$: BIT #BIT15,E.CONT
3965 026252 001401 BEQ 11$
3966 026254 104052 ERROR 52 ;MULTIPLE DRIVE SELECT
3967 026256 104075 ERROR 75 ;UNDEFINED ERROR
3968 026260 032737 000400 006364 7$: BIT #LEV2ER,RECODE ;TEST IF LEVEL 2 ERROR
3969 026266 001401 BEQ 14$ ;NO - SKIP
3970 026270 104106 ERROR 106
3971 026272 005037 002604 14$: CLR E.CONT ;CLEAR CONTROLLER ERROR WORD
3972 026276 000137 030416 JMP CERTY ;GO DO RETRY
  
```

```

;*****
.SBTTL REPORT SUPPORT ROUTINE
;*THIS ROUTINE LOADS ALL THE PARAMETER BLOCK DATA TO BE REPORTED
;*INTO THE PROPER TEMPORARY REGISTERS FOR REPORTING. ALL THESE MAY
;*NOT BE INCLUDED IN THE REPORT, BUT THEY ARE LOADED ANYWAY.
  
```

```

REPSUP:
3978 026302 SAVREG
3979 026302 104411 CLR ERRCOM ;CLEAR ERROR COMMAND STORE
3980 026304 005037 006372 MOV P.CMND(R5),ERRCOM ;STORE COMMAND START VALUES
3981 026310 116537 000001 006372 MOV #SREG0,R0 ;FOR REPORTING
3982 026316 012700 001162 MOV P.CYLN(R5),(R0)+
3983 026322 016520 000002 MOV P.TRCK(R5),(R0)+
3984 026326 116520 000005 CLRB (R0)+
3985 026332 105020 CLRB (R0)+
3986 026334 116520 000004 MOV P.SECT(R5),(R0)+
3987 026340 105020 CLRB (R0)+
3988 026342 016520 000012 MOV P.WC(R5),(R0)+
3989 026346 016520 000010 MOV P.BALO(R5),(R0)+
3990 026352 016520 000016 MOV P.CS1(R5),(R0)+ ;GET ALL THE VALUES FROM THE
3991 026356 016520 000020 MOV P.CS2(R5),(R0)+ ;PARAMETER BLOCK AND LOAD
3992 026362 016520 000030 MOV P.DCYL(R5),(R0)+ ;THE TEMPORARY REGISTERS
3993 026366 016520 000026 MOV P.DTS(R5),(R0)+ ;FOR REPORTING. ALL THIS
3994 026372 016520 000022 MOV P.WCR(R5),(R0)+ ;DATA MAY NOT BE VALID
3995 ;FOR ALL REPORTS (TO BE
3996 026376 016520 000024 MOV P.BAR(R5),(R0)+ ;DETERMINED LATER) BUT IT IS
3997 026402 016520 000032 MOV P.ASOF(R5),(R0)+ ;STORED ANY WAY.
3998 026406 016520 000036 MOV P.DS(R5),(R0)+
3999 026412 016520 000034 MOV P.ER(R5),(R0)+
4000 026416 016520 000040 MOV P.A00(R5),(R0)+
4001 026422 016520 000042 MOV P.B00(R5),(R0)+
4002 026426 016520 000044 MOV P.A01(R5),(R0)+
4003 026432 016520 000046 MOV P.B01(R5),(R0)+
4004 026436 016520 000050 MOV P.A10(R5),(R0)+
4005 026442 016520 000052 MOV P.B10(R5),(R0)+
4006 026446 016520 000054 MOV P.A11(R5),(R0)+
4007 026452 016520 000056 MOV P.B11(R5),(R0)+
4008 026456 104412 RESREG
4009 026460 000207 RTS PC
4010
  
```

```

;*****
  
```

```

4011 .SBTTL REPORT ERROR ROUTINE
4012 :* ENTRY JSR PC, ERRHDL
4013 :* RETURN RTS PC
4014 :*
4015 :* THIS ROUTINE IS CALLED BY THE DRIVER WHEN AN ERROR IS DETECTED
4016 :* IN THE OPERATION. THE ROUTINE DETERMINES WHICH COMMAND WAS
4017 :* BEING EXECUTED AND GENERATES THE PROPER ERROR MESSAGE.
4018 :*****
4019
4020 026462 104411 ERRHDL: SAVREG
4021 026464 152737 000377 002652 BISB #377,DONE ;SET DONE FLAG
4022 026472 105237 002654 INCB ERRCNT ;INCREMENT ERROR COUNT
4023 026476 005037 006364 CLR RECODE ;CLEAR RECOVERY CODE WORD
4024 026502 032737 000400 006364 ER2ENT: BIT #LEV2ER,RECODE ;TEST IF 2ND LEVEL ERROR
4025 026510 001402 BEQ 52$ ;NO - SKIP PARAM BLOCK CHANGE
4026 026512 012705 002446 MOV #PARM1,R5 ;ELSE SET R5 TO PARAMETER BLOCK 1
4027 026516 012737 030532 002600 52$: MOV #RETANL,A.ABNL ;SET NOW ABNORMAL AND NORMAL RETURN FOR
4028 026524 012737 030550 002576 MOV #RETNML,A.NORM ;DRIVER OPERATIONS IN ERROR PROCESSING
4029 026532 004737 026302 JSR PC,REPSUP ;GO SET UP REGISTERS FOR REPORT
4030 ;NOW BEGIN TESTING THE ERROR
4031 ;BITS. THE SEQUENCE IN
4032 ;WHICH THEY ARE TESTED IS
4033 ;CONSIDERED SIGNIFICANT IN
4034 ;THAT ERRORS OF A MORE
4035 ;CATASTROPHIC NATURE ARE FIRST
4036 ;TESTED.
4037 ;
4038 ;IF AN ERROR IS FOUND SET,
4039 ;THAT ERROR IS REPORTED AND
4040 ;THE REPORTING IS TERMINATED.
4041 ;IF ADDITIONAL ERRORS ARE SET,
4042 ;THE RK611 REGISTER PRINTOUTS
4043 ;WILL SHOW THIS BUT THE
4044 ;REGISTER CONTENTS MUST BE
4045 ;MANUALLY DECODED TO LOCATE THE
4046 ;SECOND ERROR
4047 026536 016504 000034 MOV P.ER(R5),R4 ;SET R4 TO ERROR REGISTER
4048 026542 032765 020000 000020 BIT #UPE,P.CS2(R5) ;TEST UPE. IF UES-SET
4049 026550 001406 BEQ 1$ ;REPORT ERROR
4050 026552 104001 ERROR 1
4051 026554 052737 000200 006364 BIS #ABORT,RECODE
4052 026562 000137 030106 JMP 37$
4053 026566 032765 004000 000020 1$: BIT #NEM,P.CS2(R5) ;TEST NON-EXISTANT MEMORY
4054 026574 001406 BEQ 2$
4055 026576 104002 ERROR 2
4056 026600 052737 000200 006364 BIS #ABORT,RECODE
4057 026606 000137 030106 JMP 37$
4058 026612 032765 010000 000020 2$: BIT #NED,P.CS2(R5) ;TEST NON-EXISTANT DRIVE
4059 026620 001412 BEQ 3$
4060 026622 032765 000400 000020 BIT #UFE,P.CS2(R5) ;TEST IF NED & UFE BOTH SET
4061 026630 001403 BEQ 38$
4062 026632 104035 ERROR 35 ;NED & UFE BOTH SET ERROR
4063 026634 000137 030106 JMP 37$
4064 026640 104003 38$: ERROR 3 ;NED ONLY
4065 026642 000137 030106 JMP 37$
4066 026646 032765 000400 000020 3$: BIT #UFE,P.CS2(R5) ;TEST UNIT FIELD ERROR
  
```

4067	026654	001412				BEQ	4\$	
4068	026656	032765	010000	000020		BIT	#NED,P.CS2(R5)	;TEST IF UFE & NED BOTH SET
4069	026664	001403				BEQ	39\$;NO-SKIP
4070	026666	104035				ERROR	35	;REPORT NED & UFE BOTH SET
4071	026670	000137	030106			JMP	37\$	
4072	026674	104004			39\$:	ERROR	4	;REPORT UFE ONLY
4073	026676	000137	030106			JMP	37\$	
4074	026702	032765	000100	000014	4\$:	BIT	#CMDTO,P.PRST(R5)	;
4075	026710	001414				BEQ	5\$	
4076	026712	004737	030560			JSR	PC, TOPROC	;GO PROCESS TIMEOUT
4077	026716	032737	000400	006364		BIT	#LEV2ER,RECODE	;TEST IF LEVEL 2 ERROR
4078	026724	001003				BNE	41\$;YES - SKIP TO ERROR 57
4079	026726	104005				ERROR	5	;ELSE MAKE FULL TIMEOUT REPORT
4080	026730	000137	030106			JMP	37\$	
4081	026734	104057			41\$:	ERROR	57	;2ND LEVEL ERROR REPORT
4082	026736	000137	030106			JMP	37\$	
4083	026742	032765	020000	000016	5\$:	BIT	#SPAR,P.CS1(R5)	;TEST DRIVE BUS PARITY ERROR
4084	026750	001406				BEQ	6\$	
4085	026752	104006				ERROR	6	
4086	026754	052737	004000	006364		BIS	#RCLREQ,RECODE	
4087	026762	000137	030106			JMP	37\$	
4088	026766	032704	000010		6\$:	BIT	#DRPAR,R4	;TEST DRIVE DETECTED PARITY ERROR
4089	026772	001406				BEQ	7\$	
4090	026774	104007				ERROR	7	
4091	026776	052737	004000	006364		BIS	#RCLREQ,RECODE	
4092	027004	000137	030106			JMP	37\$	
4093	027010	032765	000010	000036	7\$:	BIT	#ACLO,P.DS(R5)	;TEST AC LOW
4094	027016	001403				BEQ	8\$	
4095	027020	104010				ERROR	10	
4096	027022	000137	030106			JMP	37\$	
4097	027026	032765	000020	000036	8\$:	BIT	#SPDLSS,P.DS(R5)	;TEST SPEED LOSS
4098	027034	001403				BEQ	24\$	
4099	027036	104011				ERROR	11	
4100	027040	000137	030106			JMP	37\$	
4101	027044	032765	004000	000016	24\$:	BIT	#CTO,P.CS1(R5)	;TEST FOR CONTROLLER TIMEOUT
4102	027052	001401				BEQ	25\$	
4103	027054	104027				ERROR	27	
4104	027056	032704	000001		25\$:	BIT	#ILC,R4	;TEST ILLEGAL FUNCTION CODE
4105	027062	001403				BEQ	10\$	
4106	027064	104012				ERROR	12	
4107	027066	000137	030106			JMP	37\$	
4108	027072	032765	002000	000020	10\$:	BIT	#PGE,P.CS2(R5)	;TEST PROGRAMMING ERROR
4109	027100	001403				BEQ	11\$	
4110	027102	104013				ERROR	13	
4111	027104	000137	030106			JMP	37\$	
4112	027110	032704	000004		11\$:	BIT	#NXF,R4	;TEST ILLEGAL DRIVE FUNCTION
4113	027114	001403				BEQ	12\$	
4114	027116	104014				ERROR	14	
4115	027120	000137	030106			JMP	37\$	
4116	027124	032704	000040		12\$:	BIT	#DIYE,R4	;TEST DRIVE TYPE ERROR
4117	027130	001403				BEQ	13\$	
4118	027132	104015				ERROR	15	
4119	027134	000137	030106			JMP	37\$	
4120	027140	032704	000020		13\$:	BIT	#FMTE,R4	;TEST FORMAT ERROR
4121	027144	001403				BEQ	14\$	
4122	027146	104016				ERROR	16	

4123	027150	000137	030106			JMP	37\$	
4124	027154	032704	004000	14\$:		BIT	#WLE,R4	;TEST WRITE LOCK ERROR
4125	027160	001403				BEQ	15\$	
4126	027162	104017				ERROR	17	
4127	027164	000137	030106			JMP	37\$	
4128	027170	032704	040000	15\$:		BIT	#UNS,R4	;TEST DRIVE UNSAFE
4129	027174	001406				BEQ	16\$	
4130	027176	104020				ERROR	20	
4131	027200	052737	000200	006364		BIS	#ABORT,RECODE	
4132	027206	000137	030206			JMP	ALLTRM	
4133	027212	032704	000002	16\$:		BIT	#SKI,R4	;TEST SEEK INCOMPLETE
4134	027216	001406				BEQ	17\$	
4135	027220	104021				ERROR	21	
4136	027222	052737	004000	006364		BIS	#RCLREQ,RECODE	
4137	027230	000137	030106			JMP	37\$	
4138	027234	032704	001000	17\$:		BIT	#COE,R4	;TEST CYLINDER OVERFLOW
4139	027240	001406				BEQ	18\$	
4140	027242	104022				ERROR	22	
4141	027244	052737	004000	006364		BIS	#RCLREQ,RECODE	
4142	027252	000137	030106			JMP	37\$	
4143	027256	032704	002000	18\$:		BIT	#IDAE,R4	;TEST ILLEGAL CYLINDER
4144	027262	001406				BEQ	19\$	
4145	027264	104023				ERROR	23	
4146	027266	052737	004000	006364		BIS	#RCLREQ,RECODE	
4147	027274	000137	030106			JMP	37\$	
4148	027300	032765	000040	000036	19\$:	BIT	#DROT,P.DS(R5)	;TEST DRIVE OFF TRACK
4149	027306	001406				BEQ	20\$	
4150	027310	104024				ERROR	24	
4151	027312	052737	004000	006364		BIS	#RCLREQ,RECODE	
4152	027320	000137	030106			JMP	37\$	
4153	027324	032704	010000	20\$:		BIT	#DTE,R4	;TEST DRIVE TIMING ERROR
4154	027330	001406				BEQ	21\$	
4155	027332	104025				ERROR	25	
4156	027334	052737	000200	006364		BIS	#ABORT,RECODE	
4157	027342	000137	030106			JMP	37\$	
4158	027346	032765	100000	000020	21\$:	BIT	#DLT,P.CS2(R5)	;TEST DATA LATE
4159	027354	001403				BEQ	22\$	
4160	027356	104026				ERROR	26	
4161	027360	000137	030106			JMP	37\$	
4162	027364	032704	020000	22\$:		BIT	#OPI,R4	;TEST IF OPI ERROR
4163	027370	001457				BEQ	29\$	
4164	027372	052737	000010	006364		BIS	#OPIERR,RECODE	
4165	027400	105737	002650			TSTB	DERCNT	;TEST IF FIRST ERROR
4166	027404	001402				BEQ	50\$	
4167	027406	000137	030106			JMP	37\$;NO - SKIP REPORT
4168	027412	032737	000400	006364	50\$:	BIT	#LEV2ER,RECODE	;TEST IF A SECOND LEVEL 2 ERROR
4169	027420	001403				BEQ	26\$;HAS ALREADY OCCURRED
4170	027422	104054			27\$:	ERROR	54	
4171	027424	000137	030106			JMP	37\$;GET OUT OF ERROR REPORT
4172	027430	004737	031534	26\$:		JSR	PC,BLDEXH	;GO BUILD EXPECTED HEADER
4173	027434	004737	031122			JSR	PC,RDHDO	;GET HEADER OF SECTOR 0
4174	027440	032737	000400	006364		BIT	#LEV2ER,RECODE	;TEST IF ERROR GETTING HDR
4175	027446	001025				BNE	28\$;IF YES-GO MAKE ABBREVIATED REPORT
4176	027450	013702	002570			MOV	RKBAS,R2	;STORE HEADER 0 INTO REGISTERS
4177	027454	042762	000100	000000		BIC	#IE,RKCS1(R2)	;RESET INTERRUPT ENABLE
4178	027462	016237	000024	001202		MOV	RKDB(R2),\$REG10	;FOR REPORTING

4179	027470	016237	000024	001204		MOV	RKDB(R2), \$REG11	
4180	027476	016237	000024	001206		MOV	RKDB(R2), \$REG12	
4181	027504	032762	100000	000000		BIT	#CERR, RKCS1(R2)	;TEST IF ERROR DURING STORAGE
4182	027512	001343				BNE	27\$	
4183	027514	104030				ERROR	30	;MAKE OPI REPORT
4184	027516	000137	030106			JMP	37\$	
4185	027522	104054			28\$:	ERROR	54	
4186	027524	000137	026502			JMP	ER2ENT	;GO MAKE 2ND LEVEL REPORT
4187	027530	032704	000400		29\$:	BIT	#HVRC, R4	;TEST IF HVRC ERROR
4188	027534	001457				BEQ	23\$	
4189	027536	052737	000004	006364		BIS	#HVR CER, RECODE	
4190	027544	105737	002650			TSTB	DERCNT	;TEST IF FIRST ERROR
4191	027550	001402				BEQ	30\$;YES - REPORT
4192	027552	000137	030106			JMP	37\$;JUMP TO RETURN
4193	027556	032737	000400	006364	30\$:	BIT	#LEV2ER, RECODE	;TEST IF A 2ND LEVEL ERROR HAS ALREADY
4194	027564	001403				BEQ	31\$;OCCURRED. NO-SKIP EXIT
4195	027566	104055			51\$:	ERROR	55	
4196	027570	000137	030106			JMP	37\$;GET OUT OF ERROR REPORT
4197	027574	004737	031534		31\$:	JSR	PC, BLDEXH	;GO BUILD EXPECTED HEADER
4198	027600	004737	031122			JSR	PC, RDHDO	;GO GET HDR 0
4199	027604	032737	000400	006364		BIT	#LEV2ER, RECODE	;TEST IF ERROR IN GETTING HDR
4200	027612	001025				BNE	32\$;IF YES-GO MAKE ABBREVIATED REPORT
4201	027614	013702	002570			MOV	RKBAS, R2	;GET RK611 BASE ADDRESS
4202	027620	042762	000100	000000		BIC	#IE, RKCS1(R2)	;CLEAR INTERRUPT ENABLE
4203	027626	016237	000024	001202		MOV	RKDB(R2), \$REG10	;STORE OFF HEADER
4204	027634	016237	000024	001204		MOV	RKDB(R2), \$REG11	
4205	027642	016237	000024	001206		MOV	RKDB(R2), \$REG12	
4206	027650	032762	100000	000000		BIT	#CERR, RKCS1(R2)	;TEST IN ANY ERROR IN UNLOAD
4207	027656	001343				BNE	51\$;IY YES - GO MAKE SHORT REPORT
4208	027660	104031				ERROR	31	;MAKE FULL REPORT
4209	027662	000137	030106			JMP	37\$	
4210	027666	104055			32\$:	ERROR	55	;ABBREVIATED HVRC ERROR RPORT
4211	027670	000137	026502			JMP	ER2ENT	;GO REPORT 2ND LEVEL ERROR
4212	027674	032704	000200		23\$:	BIT	#BSE, R4	;TEST FOR BAD SECTOR ERROR
4213	027700	001422				BEQ	33\$;NO - SKIP
4214	027702	052737	000002	006364		BIS	#BSERR, RECODE	;SET ERROR FLAG
4215	027710	016502	000026			MOV	P.DTS(R5), R2	;GET SECTOR IN ERROR
4216	027714	042702	177740			BIC	#177740, R2	;CLEAR ALL BITS EXCEPT SECTOR
4217	027720	004737	021720			JSR	PC, BDRCK	;GO SEE IF THIS SECTOR LISTED
4218	027724	032737	001000	006364		BIT	#BADSEC, RECODE	;TEST RESULT
4219	027732	001065				BNE	37\$;YES - EXIT, NO ERROR OR REPORT
4220	027734	104104				ERROR	104	;ELSE REPORT BAD BSE
4221	027736	042737	000002	006364		BIC	#BSERR, RECODE	;RESET BSE ERROR FLAG
4222	027744	000460				BR	37\$;GO EXIT
4223	027746	032704	100000		33\$:	BIT	#DCK, R4	;TEST IF DATA CHECK
4224	027752	001435				BEQ	36\$	
4225	027754	052737	000020	006364		BIS	#DCKERR, RECODE	;SET DATA CHECK ERROR IN RECOVERY CODE
4226	027762	032704	000100			BIT	#ECH, R4	;TEST IF ECC IS HARD. IF
4227	027766	001406				BEQ	34\$;YES SET UNCORRECTABLE IN
4228	027770	052737	000040	006364		BIS	#ECCNC, RECODE	;RECOVERY FLAG AND A 0 IN
4229	027776	005037	001206			CLR	\$REG12	;REG12 TO INDICATE UNCORRECTABLE,
4230	030002	000403				BR	35\$;A 1 IN REG 12 FOR CORRECTABLE
4231	030004	012737	000001	001206	34\$:	MOV	#1, \$REG12	
4232	030012	105737	002650		35\$:	TSTB	DERCNT	;TEST IF FIRST ERROR
4233	030016	001033				BNE	37\$;NO SKIP REPORT
4234	030020	004737	031410			JSR	PC, GTPKAD	;GO GET PACK ADDRESS OF ERROR

```

4235 030024 016537 000060 001202 MOV P.EPOS(R5), $REG10 ;STORE ECC POSITION &
4236 030032 016537 000062 001204 MOV P.EPAT(R5), $REG11 ;PATTERN
4237 030040 104032 ERROR 32 ;REPORT DCK ERROR
4238 030042 000137 030106 JMP 37$
4239 030046 032765 040000 000020 36$: BIT #WCE, P.CS2(R5) ;TEST WRITE CHECK ERROR
4240 030054 001414 BEQ 37$
4241 030056 042737 000200 006364 BIC #ABORT, RECODE ;CLEAR ABORT & SET WRITE
4242 030064 052737 000100 006364 BIS #WCERR, RECODE ;CHECK ERROR IN RECODE
4243 030072 105737 002650 TSTB DERCNT ;TEST IF FIRST ERROR
4244 030076 001003 BNE 37$ ;NO - SKIP
4245 030100 004737 031410 JSR PC, GTPKAD ;GO GET ADDRESS OF ERROR
4246 030104 104033 ERROR 33 ;REPORT WCE
4247
4248 030106 032765 000020 000014 37$: BIT #DRVHRD, P.PRST(R5) ;TEST HARD ERROR
4249 030114 001404 BEQ 43$
4250 030116 104036 ERROR 36
4251 030120 052737 000200 006364 BIS #ABORT, RECODE
4252
4253 030126 032765 000040 000014 43$: BIT #DRVDSC, P.PRST(R5) ;TEST STATUS CHANGE NOT CLEARED
4254 030134 001404 BEQ 44$
4255 030136 104037 ERROR 37
4256 030140 052737 000200 006364 BIS #ABORT, RECODE
4257
4258 030146 032765 004000 000014 44$: BIT #NODSC, P.PRST(R5) ;IFST ATTENTION BUT NO FAULT OR DSC
4259 030154 001404 BEQ 46$
4260 030156 104040 ERROR 40
4261 030160 052737 000200 006364 BIS #ABORT, RECODE
4262 030166 032765 000010 000014 46$: BIT #UEXATT, P.PRST(R5) ;TEST UNEXPECTED ATTENTION
4263 030174 001404 BEQ ALLTRM
4264 030176 104042 ERROR 42
4265 030200 052737 000200 006364 BIS #ABORT, RECODE
4266
4267
4268 ;ALL ERRORS MUST EXIT THROUGH THIS POINT
4269
4270 030206 012705 002362 ALLTRM: MOV #PARMO, R5 ;RESTORE PARAMETER BLOCK SELECTION
4271 030212 012737 026462 002600 MOV #ERRHDL, A.ABNL ;RESTORE INTERRUPT VECTORS FOR RETRY
4272 030220 012737 025524 002576 MOV #ERRFRE, A.NORM ;DRIVER OPERATIONS, IF ANY
4273 030226 032737 000200 006364 BIT #ABORT, RECODE ;IF ABORT IS NOT SET AND
4274 030234 001050 BNE 48$ ;THE DRIVE IS READY (HAS NOT
4275 ;CYCLED DOWN)
4276 030236 013702 002570 MOV RKBAS, R2 ;GET BASE ADDRESS
4277 030242 032762 000200 000012 BIT #DRDY, RKDS(R2) ;TEST IF DRIVE READY SET
4278 030250 001004 BNE 47$ ;RECALIBRATE REQUIRED BIT IS SET
4279 030252 052737 000200 006364 BIS #ABORT, RECODE ;ELSE ABORT WITH ABORT MESSAGE
4280 030260 000436 BR 48$
4281 030262 032737 004000 006364 47$: BIT #RCLREQ, RECODE ;IF RECALIBRATE REQUIRED
4282 030270 001436 BEQ BGNRTY ;FOR RETRY SET UP PARAM
4283 030272 012705 002446 MOV #PARM1, R5 ;SET TO PARAMETER BLOCK ONE
4284 030276 112765 000113 000001 MOV #RECAL, P.CMND(R5) ;BLOCK TO DO IT.
4285 030304 012737 030532 002600 MOV #RETANL, A.ABNL
4286 030312 004737 025442 JSR PC, DRVCAL
4287 030316 012737 026462 002600 MOV #ERRHDL, A.ABNL ;RESTORE ERROR RETURN
4288 030324 032737 000400 006364 BIT #LEV2ER, RECODE ;IF AN ERROR OCCURRED IN THE
4289 030332 001003 BNE 49$ ;RECAL ATTEMP SET ABORT,
4290 030334 012705 002362 MOV #PARMO, R5 ;RESET PARAM TO BLOCK 0

```



```
4291 030340 000412 BR BGNRTY
4292 030342 052737 000200 006364 49$: BIS #ABORT,RECODE ;PRINT THE RECAL ERROR MESSAGE, AND
4293 030350 104060 ERROR 60 ;GO REPORT DETAILS
4294 030352 000137 026502 JMP ER2ENT
4295 030356 104061 48$: ERROR 61 ;REPORT ABORT-RETRY FAILED
4296
4297 ;THE PROGRAM WILL HALT HERE IF THE ABORT FLAG HAS BEEN SET OR
4298 ;IF THE DRIVE READY BIT IS RESET.
4299
4300 030360 000000 HLTPRG: HALT
4301 030362 000137 016202 JMP START
4302
4303
4304 ;THE FOLLOWING CODE WILL DETERMINE IF AND DATA TYPE ERROR
4305 ;HAS OCCURRED. IF YES, THE RETRY IS NOT DONE HERE BUT RETURNS TO
4306 ;THE INITIATING ROUTINE FOR RETRY. ANY OTHER ERROR IS TO BE
4307 ;RETRIED HERE. IF RETRY IS UNSUCCESSFUL AFTER 4 ATTEMPTS, THE ABORT
4308 ;FLAG IS SET AND PROGRAM HALTS.
4309
4310 030366 032737 000136 006364 BGNRTY: BIT #BSERR!HVR CER!OPIERR!DCKERR!WCERR,RECODE ;TEST IF ANY DATA ERROR
4311 030374 001404 BEQ DOL3
4312 030376 052737 100000 006364 BIS #ANYDER,RECODE
4313 030404 000450 BR DOL2
4314 030406
4315 030406 032737 001000 006364 DOL3: BIT #BADSEC,RECODE ;TEST IF BAD SECTOR FLAG SET
4316 030414 001044 BNE DOL2 ;IF YES-EXIT TO CALLER
4317 030416 123737 002654 002655 CERTY: CMPB ERRCNT,ERRLMT ;BEGIN RETRY IF ERROR COUNT HAS
4318 030424 001012 BNE DOL1 ;NOT BEEN EXCEEDED
4319 030426 005037 CLR $REG5
4320 030432 113737 002654 001174 MOVB ERRCNT,$REG5 ;GET ERROR RETRY COUNT
4321 030440 104102 ERROR 102 ;REPORT RETRY UNSUCCESSFUL
4322 030442 052737 000200 006364 BIS #ABORT,RECODE ;SET ABORT & QUIT
4323 030450 000743 BR HLTPRG
4324 030452 013702 002570 DOL1: MOV RKBAS,R2 ;GET BASE ADDRESS
4325 030456 012762 000040 000010 MOV #SCLR,RKCS2(R2) ;CLEAR SUBSYSTEM
4326 030464 013746 000000 MOV 0,-(SP) ;PUT NEW PS ON STACK
4327 030470 012746 030476 MOV #64$,-(SP) ;PUT NEW PC ON STACK
4328 030474 000002 RTI ;POP NEW PC AND PS
4329 030476
4330 030476 012700 005660 64$: MOV #COMSTR,R0 ;GO AND REESTABLISH THE COMMAND
4331 030502 012025 MOV (R0)+,(R5)+ ;AS IT WAS ENTERED INTO THE
4332 030504 012025 MOV (R0)+,(R5)+ ;PARAMETER BLOCK
4333 030506 012025 MOV (R0)+,(R5)+
4334 030510 012025 MOV (R0)+,(R5)+
4335 030512 012025 MOV (R0)+,(R5)+
4336 030514 012025 MOV (R0)+,(R5)+
4337 030516 012705 002362 MOV #PARM0,R5
4338 030522 004737 025442 JSR PC,DRV CAL ;CALL DRIVER
4339 030526 104412 DOL2: RESREG ;IF RETURN GETS HERE, NO ERROR
4340 030530 000207 RTS PC ;OCCURRED, RECOVERY WAS SUCCESSFUL
4341 030532 152737 000377 002652 RETANL: BISB #377,DONE ;SET DONE
4342 030540 052737 000400 006364 BIS #LEV2ER,RECODE ;SET LEVEL TWO ERROR
4343 030546 000207 RTS PC
4344 030550 152737 000377 002652 RETNML: BISB #377,DONE ;SET DONE
4345 030556 000207 RTS PC
4346 ;:*****
```

4347
 4348
 4349
 4350
 4351 030560
 4352 030560 104411
 4353 030562 013702 002570
 4354 030566 012701 001162
 4355 030572 005037 006372
 4356 030576 116537 000001 006372
 4357 030604 016521 000002
 4358 030610 116521 000005
 4359 030614 105021
 4360 030616 116521 000004
 4361 030622 105021
 4362 030624 016521 000012
 4363 030630 016521 000010
 4364 030634 016221 000000
 4365 030640 016221 000010
 4366 030644 016221 000020
 4367 030650 016221 000006
 4368 030654 016221 000002
 4369 030660 016221 000004
 4370 030664 016221 000016
 4371 030670 016221 000012
 4372 030674 016221 000014
 4373
 4374
 4375
 4376 030700 004737 030710
 4377 030704 104412
 4378 030706 000207
 4379
 4380
 4381
 4382 030710 104411
 4383 030712 005000
 4384 030714 012762 100000 000000
 4385 030722 010146
 4386 030724 012703 000010
 4387 030730 012721 123456
 4388 030734 005303
 4389 030736 001374
 4390 030740 012601
 4391 030742 152737 000377 002632
 4392 030750 013746 002600
 4393 030754 012737 030532 002600
 4394 030762 013762 006340 000010
 4395 030770 032762 100000 000000
 4396 030776 001037
 4397 031000 010062 000026
 4398 031004 012762 000001 000000
 4399 031012 016204 000000
 4400 031016 032704 000200
 4401 031022 001007
 4402 031024 004737 032312

.SBTTL TIME OUT PROCESSOR ROUTINE
 :*THIS ROUTINE SUPPORTS THE ERROR HANDLER BY PROCESSING TIME OUT STATUS
 :*GATHERING DUTIES.
 :*****

```

TOPROC:
  SAVREG
  MOV RKBAS,R2
  MOV #SREG0,R1 ;SET UP R1 FOR RK REGISTER STORAGE
  CLR ERRCOM ;CLEAR FOR COMMAND
  MOV P.CMND(R5),ERRCOM ;STORE COMMAND
  MOV P.CYLN(R5),(R1)+ ;STORE CYLINDER
  MOV P.TRCK(R5),(R1)+ ;STORE TRACK
  CLRB (R1)+
  MOV P.SECT(R5),(R1)+ ;STORE SECTOR
  CLRB (R1)+
  MOV P.WC(R5),(R1)+ ;STORE WORD COUNT
  MOV P.BALO(R5),(R1)+ ;STORE BUS ADDRESS
  MOV RKCS1(R2),(R1)+ ;STORE ALL PARAMETERES AND VALID RK611
  MOV RKCS2(R2),(R1)+ ;REGISTERS
  MOV RKDC(R2),(R1)+
  MOV RKDA(R2),(R1)+
  MOV RKWC(R2),(R1)+
  MOV RKBA(R2),(R1)+
  MOV RKASOF(R2),(R1)+
  MOV RKDS(R2),(R1)+
  MOV RKER(R2),(R1)+

  ;THIS CODE WILL ATTEMPT TO
  ;RETRIEVE THE STATUS FROM THE
  ;DRIVE.
  JSR PC,GETDRST ;GO GET DRIVE STATUS
  RESREG
  RTS PC
  
```

:*****
 .SBTTL RETRIEVE DRIVE STATUS ROUTINE

```

GETDRST:
  SAVREG
  CLR R0
  MOV #CCLR,RKCS1(R2) ;CLEAR CONTROLLER
  MOV R1,-(SP) ;STORE R5
  MOV #10,R3 ;SET COUNT FOR STATUS CLEAR
  1$: MOV #123456,(R1)+ ;INSERT CLEAR WORD
  DEC R3 ;BUMP COUNTER
  BNE 1$ ;LOOP UNTIL 0
  MOV (SP)+,R1 ;RESTORE R5
  BISB #377,W.TIME ;GIVE WATCH DOG SOMETHING TO TIME
  MOV A.ABNL,-(SP) ;STORE ERROR RETURN ADDRESS
  MOV #RETANL,A.ABNL ;SET UP NEW ERROR RETURN
  MOV DRINUS,RKCS2(R2) ;LOAD DRIVE NUMBER
  BIT #CERR,RKCS1(R2) ;TEST IF CONT ERROR SET
  BNE 4$ ;YES - GET OUT
  3$: MOV R0,RKMR1(R2) ;SET UP STATUS WORD TO READ
  MOV #1,RKCS1(R2) ;DO DRIVE SELECT
  MOV RKCS1(R2),R4 ;GET CS1
  BIT #RDY,R4 ;TEST IF READY
  BNE 2$ ;YES - SKIP
  JSR PC,W.WTCH ;CALL WATCH DOG
  
```

```

4403 031030 032765 000100 000014      BIT      #CMDTO,P.PRST(R5)      ;TEST IF TIMEOUT
4404 031036 001360      BNE      3$                  ;NO - LOOP
4405 031040 000416      BR       4$                  ;ELSE GET OUT
4406 031042 016204 000000      MOV      RKCS1(R2),R4        ;GET CS1
4407 031046 032704 100000      BIT      #CERR,R4           ;TEST IF ERROR
4408 031052 001011      BNE      4$                  ;YES - GET OUT
4409 031054 016221 000034      MOV      RKMR2(R2),(R1)+     ;ELSE STORE STATUS WORDS JUST READ
4410 031060 016221 000036      MOV      RKMR3(R2),(R1)+
4411 031064 005200      INC      R0                  ;BUMP TO NEXT STATUS PAIR
4412 031066 020027 000004      CMP      R0,#4              ;TEST IF ALL HAVE BEEN READ
4413 031072 001342      BNE      3$                  ;YES - GOOD EXIT
4414 031074 000406      BR       5$
4415 031076 052737 000400 006364 4$:  BIS      #LEV2ER,RECODE      ;SET LEVEL 2 ERROR
4416 031104 012762 000040 000010      MOV      #SCLR,RKCS2(R2)    ;CLEAR SUBSYSTEM
4417 031112 012637 002600      MOV      (SP)+,A.ABNL       ;RESTORE OLD ERROR RETURN
4418 031116 104412      RESREG
4419 031120 000207      RTS      PC
4420
4421      ;*****
4422      ;SBTTL READ HEADER 0 ROUTINE
4423      ;*****
4423 031122      RDHDO:
4424 031122 104411      SAVREG
4425 031124 016501 000026      MOV      P.DTS(R5),R1       ;STORE TRACK AND SECTOR
4426 031130 016500 000052      MOV      P.B10(R5),R0       ;GET THE CYLINDER ADDRESS
4427 031134 042700 160013      BIC      #160013,R0         ;FROM THE DRIVE STATUS. CLEAR
4428 031140 006200      ASR      R0                  ;OFF UNUSED BITS AND POSITION
4429 031142 006200      ASR      R0                  ;FOR USE AS THE DESIRED
4430 031144 006200      ASR      R0                  ;CYLINDER IN THE READ
4431 031146 006200      ASR      R0                  ;HEADER COMMAND
4432 031150 012705 002446      MOV      #PARM1,R5          ;SET UP TO USE PARM 1
4433 031154 010165 070004      MOV      R1,P.SECT(R5)      ;INSERT TRK/SECT FOR READ HEADER
4434 031160 010065 000002      MOV      R0,P.CYLN(R5)
4435 031164 132737 000020 002653      BITB     #B.CFMT,TYPFMT     ;TEST PRESENT FORMAT & CHANGE
4436 031172 001404      BEQ      1$                  ;TO THE OPPOSITE. THIS WILL CAUSE
4437 031174 142765 000020 000007      BICB     #B.CFMT,P.CS1H(R5) ;A READ OF SECTOR 0 HEADER ON
4438 031202 000403      BR       2$                  ;THE READ HEADER COMMAND
4439 031204 152765 000020 000007 1$:  BISB     #B.CFMT,P.CS1H(R5)
4440 031212 112765 000125 000001 2$:  MOVB     #RDHEAD,P.CMND(R5) ;SET UP TO READ HEADER
4441 031220 013746 000000      MOV      0,-(SP)           ;PUT NEW PS ON STACK
4442 031224 012746 031232      MOV      #64$,-(SP)        ;PUT NEW PC ON STACK
4443 031230 000002      RTI                          ;POP NEW PC AND PS
4444 031232
4445 031232 004737 025442      JSR      PC,DRVCL          ;CALL DRIVER
4446 031236 142765 000020 000007      BICB     #B.CFMT,P.CS1H(R5) ;CLEAR THE FORMAT BIT
4447 031244 153765 002653 000007      BISB     TYPFMT,P.CS1H(R5) ;RESTORE TYPE AND FORMAT IN USE
4448 031252 012705 002362      MOV      #PARM0,R5          ;RESTORE PARM 0 BLOCK
4449 031256 104412      RESREG      ;RETURN
4450 031260 000207      RTS      PC
4451
4452      ;*****
4453      ;SBTTL POSITION AND OFFSET ROUTINE
4454      ;*THIS ROUTINE CHECKS IF OFFSET IS REQUIRED. IF IT IS, THE HEADS
4455      ;*POSITIONED TO THE CYLINDER SPECIFIED IN R0 AND THE HEADS ARE OFFSET
4456      ;*TO THE AMOUNT SPECIFIED IN "OFINUS".
4457      ;*****
4458 031262 104411      POSOFF: SAVREG
  
```

```
4459 031264 132737 000020 002656 BITB #OREQSW,OPCONT ;TEST IF OFFSET REQUIRED
4460 031272 001417 BEQ 1$ ;IF NO - SKIP TO EXIT
4461 031274 112765 000117 000001 MOVB #SEEK,P.CMND(R5) ; SET UP TO DO THE SEEK
4462 031302 010065 000002 MOV R0,P.CYLN(R5) ;SET CYLINDER
4463 031306 004737 025442 JSR PC,DRVCAL ;GO DO IT
4464 031312 112765 000115 000001 MOVB #OFFSET,P.CMND(R5) ;SET UP TO OFFSET
4465 031320 113765 006362 000006 MOVB OFINUS,P.OFST(R5) ;SET OFFSET VALUE
4466 031326 004737 025442 JSR PC,DRVCAL ;GO DO IT
4467 031332 104412 1$: RESREG
4468 031334 000207 RTS PC
4469
4470
4471
4472
4473 031336 105037 002654 .SBTTL PREPARE FOR LOOP ON ERROR
4474 031342 005037 006364 PREPLP: CLR ERRCNT ;CLEAR ERROR COUNT
4475 031346 013700 006370 CLR RECODE ;CLEAR RECODE
4476 031352 013701 006366 MOV CCINUS,R0 ;SET CURRENT CYLINDER
4477 031356 012737 026462 002600 MOV CTINUS,R1 ;SET CURRENT TRACK
4478 031364 012737 025524 002576 MOV #ERRHDL,A.ABNL ;SET UP ERROR RETURN
4479 031372 012704 001074 MOV #ERRFRE,A.NORM ;SET UP ERROR FREE RETURN
4480 031376 011624 MOV #STACK-4,R4 ;SET R4 TO STACK BASE -4
4481 031400 005014 MOV (SP),(R4)+ ;STORE RETURN ON STACK
4482 031402 012706 001074 CLR (R4) ;CLEAR FOR 0 PRIORITY
4483 031406 000002 MOV #STACK-4,SP ;SET STACK POINTER FOR RETURN
4484 RTI ;DO RTI TO RETURN TO LOOPING ROUTINE
4485
4486 .SBTTL GET PACK ADDRESS ROUTINE
4487 031410 016537 000030 001174 GTPKAD: MOV P.DCYL(R5),$REG5 ;GET CYLINDER NUMBER
4488 031416 005037 001176 CLR $REG6 ;CLEAR REGISTERS FOR
4489 031422 005037 001200 CLR $REG7 ;TRACK & SECTOR STORAGE
4490 031426 116537 000026 001200 MOVB P.DTS(R5),$REG7 ;STORE THE TRACK AND SECTOR
4491 031434 116537 000027 001176 MOVB P.DTS+1(R5),$REG6
4492 031442 005737 001200 TST $REG7
4493 ;ADJUST THE ADDRESS CONTAINED IN
4494 031446 001403 BEQ 1$ ;THE RK REGISTERS FOR THE AUTOMATIC
4495 031450 005337 001200 DEC $REG7 ;INCREMENT
4496 031454 000426 BR 5$
4497 031456 032765 010000 000016 1$: BIT #CFMT,P.CS1(R5)
4498 031464 001404 BEQ 2$
4499 031466 012737 000023 001200 MOV #19,,$REG7
4500 031474 000403 BR 3$
4501 031476 012737 000025 001200 2$: MOV #21,,$REG7
4502 031504 005737 001176 3$: TST $REG6
4503 031510 001403 BEQ 4$
4504 031512 005337 001176 DEC $REG6
4505 031516 000405 BR 5$
4506 031520 012737 000002 001176 4$: MOV #2,$REG6
4507 031526 005337 001174 DEC $REG5
4508 031532 000207 5$: RTS PC
4509
4510 .SBTTL BUILD EXPECTED HEADER
4511 ;*USES DESIRED CYLINDER, TRACK AND SECTOR REGISTERS TO DETERMINE
4512 ;*WHICH HEADER WAS EXPECTED. LOADS EXPECTED VALUES IN $REG5, 6, AND
4513 ;*7 FOR REPORTING.
4514 ;*
```

```

4515 031534 104411          BLDEXH: SAVREG
4516 031536 016537 000030 001174  MOV      P.DCYL(R5), $REG5 ;CONSTRUCT EXPECTED HDR
4517 031544 016501 000026          MOV      P.DTS(R5), R1 ;DESIRED CYLINDER & DESIRED TRACK
4518 031550 042701 174377          BIC      #174377, R1 ;CLEAR ALL BUT TRACK BITS
4519 031554 006201          ASR      R1 ;AND SECTOR. SHIFT THE TRACK
4520 031556 006201          ASR      R1 ;OVER TO CONFORM TO HEADER FORMAT
4521 031560 006201          ASR      R1 ;CHECK THE FORMAT BIT AND
4522                                     ;IF SET, SET THE HEADER FORMAT
4523 031562 016537 000026 001176  MOV      P.DTS(R5), $REG6 ;BIT.
4524 031570 042737 177740 001176  BIC      #177740, $REG6 ;CLEAR ALL BUT SECTOR
4525 031576 060137 001176          ADD      R1, $REG6 ;ADD TRACK AND SECTOR TOGETHER
4526 031602 052737 140000 001176  BIS      #140000, $REG6 ;INSERT BSE BITS
4527 031610 032765 010000 000016  BIT      #CFMT, P.CS1(R5)
4528 031616 001403          BEQ      23$
4529 031620 052737 001000 001176  BIS      #1000, $REG6
4530 031626 013737 001174 001200 23$:  MOV      $REG5, $REG7 ;COMPUTE THE HEADER VRC
4531 031634 013701 001176          MOV      $REG6, R1
4532 031640 043737 001176 001200  BIC      $REG6, $REG7
4533 031646 043701 001174          BIC      $REG5, R1
4534 031652 050137 001200          BIS      R1, $REG7
4535 031656 104412          RESREG
4536 031660 000207          RTS      PC
4537                                     ;*****
4538                                     ;SBTTL WRITE BAD SECTOR FILE ROUTINE
4539                                     ;*THIS ROUTINE WRITES THE BSSOFT (BAD SECTOR FILE DETECTED BY SOFTWARE)
4540                                     ;*INTO THE APPROPRIATE BAD SECTOR FILE. IT DOES NOT WRITE THE FACTORY
4541                                     ;*DETECTED FILES.
4542                                     ;*****
4543 BDSTWT: SAVREG
4544 031662 104411          CLRB     DERCNT ;CLEAR ERROR COUNT
4545 031664 105037 002650          MOV      #6, R3 ;PRESET FOR 6 SECTORS TO BE WRITTEN
4546 031670 012703 000006          MOV      #12, R2 ;PRESET TO 22 SECTOR/TRACK
4547 031674 012702 000012          CLR      P.PRST(R5) ;CLEAR PROG STAT REG
4548 031700 005065 000014          BITB    #B.CFMT, TYPFMT ;TEST IF 20 OR 22 SEC/TRACK
4549 031704 132737 000020 002653  BEQ      1$ ;22 SECTOR - SKIP
4550 031712 001404          INC      R2 ;20 SECTOR - BUMP TO NEXT SECTOR (13)
4551 031714 005202          BICB    #B.CFMT, P.CS1H(R5) ;CLEAR FOR 22 SECTOR WRITE
4552 031716 142765 000020 000007 1$:  MOV      #BSSOFT, P.BALO(R5) ;SET UP PARAMETER BLOCK
4553 031724 012765 003660 000010  MOVB    #WRDATA, P.CMND(R5) ;TO WRITE THE SECTORS OF BAD
4554 031732 112765 000123 000001  MOV      L.CYL, P.CYLN(R5) ;SECTOR TABLE (SOFTWARE DETECTED)
4555 031740 013765 017312 000002  MOVB    #2, P.TRCK(R5)
4556 031746 112765 000002 000005  MOVB    R2, P.SECT(R5) ;INSERT SECTOR NUMBER
4557 031754 110265 000004          MOV      #177400, P.WC(R5)
4558 031760 012765 177400 000012  JSR      PC, DRVCAL
4559 031766 004737 025442          BIT      #ANYDER, RECODE ;CHECK IF ANY ERROR IN WRITE
4560 031772 032737 100000 006364  BNE     3$ ;IF YES - SKIP
4561 032000 001005          ADD      #2, R2 ;ELSE BUMP SECTOR BY TWO
4562 032002 062702 000002 2$:  DEC      R3 ;DECREMENT COUNTER
4563 032006 005303          BNE     1$ ;EXIT IF DONE
4564 032010 001345          BR      4$
4565 032012 000417          INCB    DERCNT ;COUNT ERROR FOR RETRY
4566 032014 105237 002650 000012 3$:  CMPB    #7, DERCNT ;IF TO MANY TYPE MESSAGE
4567 032020 122737 000007 002650  BLT     1$ ;ELSE RETRY
4568 032026 002736          TYPE    ,BSWERR ;ERROR MESSAGE
4569 032030 104401 015425          TYPE
4570 032034 104401 015513          TYPE    ,SECINE

```

```
4571 032040 010246      MOV      R2,-(SP)
4572 032042 104402      TYPOC
4573 032044 104401 001267      TYPE    , $CRLF
4574 032050 000754      BR      2$
4575 032052 104401 015535      4$:    TYPE    ,BSWTFI      ;TYPE FINISHED MESSAGE
4576 032056 104412      RESREG
4577 032060 000207      RTS     PC
4578
4579      ;*****
4580      .SBTTL  BAD SECTOR REPORT ROUTINE
4581      ;*****
4581 032062      BSREPT:
4582 032062 104411      SAVREG
4583 032064 005037 001242      CLR     $TMP2      ;CLEAR TEMP 2 FOR PASS SWITCH
4584 032070 104401 015572      TYPE    ,BSHEAD    ;TYPE HEADER LINE
4585 032074 005000      CLR     R0         ;CLEAR R0 FOR COUNTER
4586 032076 012737 016006 032124      MOV     #TWENTY,2$ ;CHOSE THE MODE TO REPORT
4587 032104 132737 000020 002653      BITB   #B.CFMT,TYPFMT ;20 OR 22 SECTOR FORMAT
4588 032112 001403      BEQ    1$         ;QUESSSED RIGHT - SKIP
4589 032114 012737 016003 032124      MOV     #TWENTY,2$ ;QUESSSED WRONG, SET TO 22 SECTOR
4590 032122 104401      1$:    TYPE
4591 032124 000000      2$:    .WORD
4592 032126 104401 015642      TYPE    ,BSTAIL
4593 032132 104401 015670      TYPE    ,FBSLAB    ;TYPE FACTORY LABEL
4594 032136 012704 004670      MOV     #BSFACT+10,R4 ;SET R4 TO POINT TO FACTORY TABLE
4595 032142 104401 016011      6$:    TYPE    ,COLHD  ;TYPE COLUMN HEADER
4596 032146 104401 016046      3$:    TYPE    ,SPACE4 ;SET FORMAT OF REPORT
4597 032152 012446      MOV     (R4)+,-(SP) ;GET FIRST CYLINDER WORD
4598 032154 100417      BMI    4$         ;IF ALL 1'S EXIT THIS LOOP
4599 032156 104402      TYPOC      ;ELSE TYPE VALUE
4600 032160 104401 012765      TYPE    ,SPACE2    ;FORMAT
4601 032164 105724      TSTB   (R4)+      ;BUMP R4 TO THE TRACK IN TRK/SEC WORD
4602 032166 111446      MOV     (R4),-(SP) ;PUT TRACK ON THE STACK
4603 032170 104402      TYPOC      ;TYPE TRACK NUMBER
4604 032172 104401 012765      TYPE    ,SPACE2    ;FORMAT
4605 032176 114446      MOV     -(R4),-(SP) ;PUT SECTOR ON STACK
4606 032200 104402      TYPOC      ;TYPE SECTOR NUMBER
4607 032202 104401 001267      TYPE    , $CRLF    ;RETURN THE CARRIAGE
4608 032206 005724      TST    (R4)+      ;PUT R4 AT NEXT FILE ENTRY
4609 032210 005200      INC    R0         ;COUNT BAD SECTORS
4610 032212 000755      BR      3$         ;LOOP
4611 032214 005726      4$:    TST    (SP)+    ;CLEAN OFF STACK
4612 032216 005737 001242      TST    $TMP2      ;TEST IF 2ND PASS, IF YES
4613 032222 001016      BNE    5$         ;EXIT
4614 032224 052737 000001 001242      BIS    #1,$TMP2    ;ELSE SET 2ND PASS SWITCH
4615 032232 012704 003670      MOV     #BSSOFT+10,R4 ;SWITCH TO OTHER TABLE
4616 032236 005700      TST    R0
4617 032240 001004      BNE    7$         ;
4618 032242 104401 001267      TYPE    , $CRLF
4619 032246 104401 015774      TYPE    ,NONE
4620 032252 104401 015731      7$:    TYPE    ,SBSLAB  ;TYPE SOFTWARE LABEL
4621 032256 000731      BR      6$         ;GO BACK TO LOOP
4622 032260 104401 001267      5$:    TYPE    , $CRLF
4623 032264 005700      TST    R0
4624 032266 001405      BEQ    9$
4625 032270 010046      MOV     R0,-(SP)  ;EXIT ROUTINE - TYPE THE
4626 032272 104402      TYPOC      ;NUMBER OF BAD SECTORS
```

4627	032274	104401	016057		TYPE	TOTMES		
4628	032300	000402			BR	8\$:AND RETURN
4629	032302	104401	015774	9\$:	TYPE	,NONE		
4630	032306	104412		8\$:	RESREG			
4631	032310	000207			RTS	PC		

4632
4633
4634
4635
4636
4637
4638
4639
4640
4641
4642
4643
4644
4645
4646
4647
4648
4649
4650
4651
4652
4653
4654
4655
4656
4657
4658
4659
4660
4661
4662
4663
4664
4665
4666
4667
4668
4669
4670
4671
4672
4673

.SBTTL RK611/RK06-RK07 UNIBUS DRIVER FOR SEQUENTIAL OPERATIONS (REV. 0.10)

;*COPYRIGHT (C) 1975,1976,1977
;*DIGITAL EQUIPMENT CORP.
;*MAYNARD, MA. 01754
;*AUTHOR: ROY SPITZER

.SBTTL *WATCH-DOG TIMER

*
* THE WATCH-DOG TIMER DOES A PSEUDO-TIMING OF RK06-RK07 UNIBUS
* SUBSYSTEM COMMAND. SINCE ONE CAN NOT GUARANTEE THAT A
* REAL-TIME CLOCK (KW11-P OR KW11-L) IS ON THE SYSTEM
* THE RK06-RK07 DRIVER WILL USE THE LOCATION W.MTIM FOR
* MILLI-SECOND TIMING. WHEN W.MTIM REACHES ZERO THE
* WATCH-DOG TIMER WILL SCAN THE DRIVES IN USE AS
* DETERMINED BY THE LOCATION W.TIME. THE TIMER COUNTS
* (ONE FOR EACH DRIVE) ARE KEPT IN THE TABLE W.DRV.
* IF ANY COUNT IN THE TABLE W.DRV REACHES ZERO A COMMAND
* TIME-OUT WILL BE DESIGNATED IN THE PROGRAM DEVICE STATUS
* REGISTER OF THAT DRIVE'S PARAMETER BLOCK.
*
* THE DRIVER WILL USE THE LOCATION W.MIN AS THE NUMBER
* OF MILLISECONDS FOR AN UNLOAD OR START SPINDLE COMMAND.
* THE DRIVER WILL USE THE LOCATION W.SEC AS THE TIME
* LIMIT FOR ALL OTHER COMMANDS.
*
* FOR QUEUED OPERATIONS THE WATCH-DOG TIMER WILL
* WATCH UP TO 8 OPERATIONS SIMULTEOUSLY. FOR SEQUENTIAL
* OPERATIONS ONLY ONE OPERATION WILL BE WATCHED.
*
*CALL JSR PC,W.WTCH
* RETURN IF NO DRIVE ORDER EXCEEDED ITS TIME LIMIT
*
* OTHERWISE AN ABNORMAL RETURN TO THE ROUTINE ADDRESS
* BY LOCATION A.ABNL WILL OCCUR AND THE CMDTO FLAG
* IN THE PROGRAM DEVICE STATUS REGISTER OF THE
* APPROPRIATE PARAMETER BLOCK WILL BE SET.
*

4674 032312 010546
4675 032314 010446
4676 032316 010346
4677 032320 010246
4678 032322 013746 177776
4679 032326 005337 002610
4680 032332 001034
4681 032334 013737 002612 002610
4682 032342 105737 002632
4683 032346 001426
4684 032350 013737 002574 177776
4685 032356 013702 002570
4686 032362 005337 002646
4687 032366 001016

W.WTCH: MOV R5,-(SP) ;SAVE R5 ON THE STACK
MOV R4,-(SP) ;SAVE R4 ON THE STACK
MOV R3,-(SP) ;SAVE R3 ON THE STACK
MOV R2,-(SP) ;SAVE R2 ON STACK
MOV PS,-(SP) ;SAVE PROGRAM STATUS WORD ON STACK
DEC W.MTIM ;DECREMENT MILLISECOND TIMER
BNE 20\$;IF NOT ZERO RETURN
MOV W.MILI,W.MTIM ;REINITIALIZE MILLISECOND TIMER
TSTB W.TIME ;CHECK IF DRIVE IS BEING TIMED
BEQ 20\$;NO, RETURN
MOV RKPRI,PS ;LOCK OUT RK06-RK07 INTERRUPTS
MOV RKBAS,R2 ;LOAD BASE OF RK06-RK07 REGISTERS
DEC W.DRV ;DECREMENT COMMAND TIMER
BNE 20\$;RETURN IF NO TIME OUT

468P	032370	105037	002632		CLRB	W.TIME	:RESET TIMING INDICATOR
46E9	032374	013705	002644		MOV	PBLKT,R5	:LOAD ADDRESS OF PARAMETER BLOCK
4690							:TABLE FOR INDEXING
4691	032400	052765	000100	000014	BIS	#CMDTO,P.PRST(R5)	:SET COMMAND TIME OUT
4692	032406	020537	002606		CMP	R5,O.WAIT	:CHECK IF DRIVER IS WAITING FOR
4693							:COMMAND COMPLETION
4694	032412	001002			BNE	5\$:NO, DO NOT ALTER WAITING FOR
4695							:COMMAND COMPLETION
4696	032414	005037	002606		CLR	O.WAIT	:CLEAR WAIT FOR COMMAND COMPLETION
4697	032420	004737	035674	5\$:	JSR	PC,R.ABNL	:BRANCH TO ERROR ROUTINE
4698	032424	012637	177776	20\$:	MOV	(SP)+,PS	:RESTORE PSW
4699	032430	012602			MOV	(SP)+,R2	:RESTORE R2
4700	032432	012603			MOV	(SP)+,R3	:RESTORE R3
4701	032434	012604			MOV	(SP)+,R4	:RESTORE R4
4702	032436	012605			MOV	(SP)+,R5	:RESTORE R5
4703	032440	000207			RTS	PC	:RETURN

4704
4705
4706
4707
4708
4709
4710
4711
4712
4713
4714
4715
4716
4717
4718
4719
4720
4721
4722
4723
4724
4725
4726
4727
4728
4729
4730
4731
4732
4733
4734
4735
4736
4737
4738
4739
4740
4741
4742
4743
4744
4745
4746
4747

.SBTTL *RK06 INTERRUPT SERVICE ROUTINE

THIS ROUTINE WILL SERVICE ALL RK06 INTERRUPTS.

UPON RECEIVING AN INTERRUPT, THIS ROUTINE WILL PERFORM ONE OF THE FOLLOWING SERVICES:

- 1.) SERVICE PORT WAS SEIZED BY OTHER PORT
- 2.) SERVICE DRIVER IS WAIT FOR COMMAND COMPLETION
- 3.) SERVICE POSITIONING COMPLETION
- 4.) REQUEUE COMMAND IF DRIVE WAS RELEASED FOR THE QUEUED RK06 DRIVER.
- 5.) IF NO SERVICE IS REQUIRED, THE COMMAND WILL BE ISSUED FOR THE QUEUED RK06 DRIVER.

THREE LINKS ARE PROVIDED TO THE DRIVING PROGRAM. THEY ARE:

- 1.) A.NORM ADDRESS OF NORMAL RETURN (SUCESSFUL COMPLETION OF COMMAND)
- 2.) A.ABNL ADDRESS OF ABNORMAL RETURN (UNSUCESSFUL COMPLETION OF COMMAND)
- 3.) A.CONT ADDRESS OF CONTROL ERROR RETURN

FOR NORMAL AND ABNORMAL RETURNS, THE ADDRESS OF THE APPROPRIATE PARAMETER BLOCK WILL BE IN R5.

FOR THE CONTROLLER ERROR RETURN, THE LOCATION E.CONT CONTAINS THE REASON FOR THE CONTROLLER ERROR.

ROUTINES USED:

- C.OPT (QUEUED ONLY)
- Q.PUSH (QUEUED ONLY)
- Q.RMOV (QUEUED ONLY)
- R.CONT (SEQUENTIAL ONLY)
- R.NORM (SEQUENTIAL ONLY)
- R.ABNL (SEQUENTIAL ONLY)
- I.CSTS
- I.STAT
- I.ISSU
- I.CCLR

4748 032442 010546
4749 032444 010446
4750 032446 010346
4751 032450 010246
4752 032452 010146
4753 032454 010046
4754 032456 013702 002570
4755 032462 016237 000010 002534
4756 032470 032737 001000 002534
4757 032476 001407
4758 032500 052737 100000 002604
4759 032506 004737 035720

I.INTR: MOV R5,-(SP) ;STORE R5 ON THE STACK
MOV R4,-(SP) ;STORE R4 ON THE STACK
MOV R3,-(SP) ;STORE R3 ON THE STACK
MOV R2,-(SP) ;STORE R2 ON THE STACK
MOV R1,-(SP) ;STORE R1 ON THE STACK
MOV R0,-(SP) ;STORE R0 ON THE STACK
MOV RKBAS,R2 ;LOAD R2 TO ADDRESS RK06 REGISTER
MOV RKCS2(R2),T.CS2 ;STORE CS2
BIT #MDS,T.CS2 ;CHECK IF MULTIPLE DRIVE SELECT
BEQ 1\$;NO, CONTINUE PROCESSING
BIS #E.MDS,E.CONT ;SET MULTIPLE DRIVE SELECT
JSR PC,R.CONT ;REPORT ERROR

```

4760 032512 000137 034672          JMP      I.RTRN          ;RETURN
4761
4762 032516 105737 002626          1$:     TSTB     I.ISRL          ;CHECK IF INTERRUPT OR RELEASE
4763 032522 001410          BEQ      6$              ;NO, CHECK IF DRIVE AVAILABLE
4764 032524 100403          BMI      5$              ;CHECK IF RELEASE COMMAND
4765 032526 105037 002626          CLRB    I.ISRL          ;YES,CLEAR FLAG
4766 032532 000473          BR       I.I00          ;CONTINUE PROCESSING INTERRUPT
4767
4768 032534 105037 002626          5$:     CLRB    I.ISRL          ;CLEAR FLAG
4769 032540 000137 033654          JMP     I.ATTN          ;GO PROCESS DRIVE ATTENTIONS
4770
4771 032544 032737 010400 002534 6$:     BIT      #NED!UFE,T.CS2 ;CHECK FOR NON-EXISTENT DRIVE OR
4772                                     ; UNIT FIELD ERROR
4773 032552 001413          BEQ      7$              ;NO, WAIT FOR DUAL ACCESS INTERRUPT
4774 032554 013704 002534          MOV     T.CS2,R4        ;LOAD R4 FOR DRIVE NUMBER
4775 032560 042704 177770          BIC     #^C<DRVMSK>,R4 ;KEEP DRIVE BITS
4776 032564 013705 002644          MOV     PBLKT,R5        ;STORE PARAMETER BLOCK ADDRESS
4777 032570 016237 000000 002532          MOV     RKCS1(R2),T.CS1 ;LOAD TEMPORARY CS1 FG: STATUS REPORT
4778 032576 000137 033064          JMP     I.ERRC          ;REPORT ERROR
4779
4780 032602 016237 000012 002552 7$:     MOV     RKDS(R2),T.DS   ;STORE STATUS REGISTER FOR COMPARISON
4781 032610 032737 000001 002552          BIT     #DRA,T.DS      ;CHECK IF DRIVE SEIZED BY OTHER
4782                                     ; PORT
4783 032616 001041          BNE     I.I00          ;NO, CONTINUE PROCESSING INTERRUPT
4784
4785                                     ;CHECK IF ANY DATA TRANSFER ERROR EXISTS
4786 032620 032737 164000 002534          BIT     #DLT!WCE!UPE!NEM,T.CS2
4787
4788 032626 001007          BNE     10$             ;INDICATE ERROR
4789 032630 016237 000014 002550          MOV     RKER(R2),T.ER   ;STORE ERROR REGISTER
4790
4791                                     ; CHECK FOR DATA TRANSFER ERROR TYPE ERROR
4792 032636 032737 125700 002550          BIT     #DCK!OPI!WLE!COE!HVRC!BSE!ECH,T.ER
4793
4794 032644 001407          BEQ     11$             ;NO, WAIT FOR RELEASE OF RK06 DRIVE
4795
4796 032646 052737 000010 002604 10$:    BIS     #E.UDAT,E.CONT  ;SET UNEXPECTED DATA TYPE ERROR
4797 032654 004737 035720          JSR     PC,R.CONT       ;REPORT ERROR
4798 032660 000137 034672          JMP     I.RTRN          ;RESTORE REGISTERS
4799
4800 032664 105037 002632          11$:    CLRB    W.TIME        ;RESET TIMING ON THIS DRIVE
4801 032670 005037 002646          CLR     W.DRV           ;CLEAR TIMING COUNT FOR THIS DRIVE
4802 032674 013705 002644          MOV     PBLKT,R5        ;LOAD R5 WITH PARAMETER BLOCK
4803                                     ; ADDRESS
4804 032700 052765 010000 000014          BIS     #DRVSZD,P.PRST(R5) ;SET DRIVE SEIZED IN THE
4805                                     ; PROGRAM DRIVE STATUS REGISTER
4806 032706 005037 002606          CLR     O.WAIT          ;CLEAR WAIT FOR COMMAND COMPLETION
4807 032712 004737 035674          JSR     PC,R.ABNL       ;INDICATE ABNORMAL TERMINATION
4808 032716 000137 034672          JMP     I.RTRN          ;GO RESTORE REGISTERS
4809
4810 032722 013705 002606          I.I00: MOV     O.WAIT,R5     ;LOAD PARAMETER BLOCK ADDRESS INTO R5
4811 032726 001002          BNE     2$              ;IS COMMAND WAITING PROCESSING
4812                                     ; YES, DO PROCESSING
4813 032730 000137 033654          JMP     I.ATTN          ;NO, PROCESS ATTENTION
4814
4815 032734 013704 002534          2$:     MOV     T.CS2,R4        ;STORE RKCS2 FOR DRIVE NUMBER

```

```

4816 032740 042704 177770          BIC      #^C<DRVMSK>,R4 ;MASK OUT UNNECESSARY BITS
4817
4818
4819 032744 126504 000000          CMPB    P.DRVN(R5),R4 ;CHECK IF DRIVE NUMBER IS EXPECTED
4820 032750 001401          BEQ     3$ ;YES, CONTINUE
4821 032752 000000          HALT   ;NO, DRIVER ERROR
4822 032754 122765 000164 000001 3$:    CMPB    #RDALHD,P.CMND(R5) ;CHECK IF READ ALL HEADERS
4823 032762 001002          BNE    10$ ;NO, EXECUTE NORMAL DATA TRANSFER
4824 032764 000137 033322          JMP     I.HDAL ;GO EXECUTE SPECIAL HEADER SEQUENCE
4825
4826 032770 005037 002606          10$:   CLR     O.WAIT ;CLEAR WAIT FOR COMMAND COMPLETION
4827 032774 005037 002646          CLR     W.DRV ;CLEAR WATCH-DOG TIME
4828 033000 105037 002632          CLRB   W.TIME ;RESET TIMING ON THIS DRIVE
4829 033004 016237 000000 002532    MOV     RKCS1(R2),T.CS1 ;STORE COMMAND AND STATUS REGISTER 1
4830 033012 032737 100000 002532    BIT     #CERR,T.CS1 ;CHECK IF CONTROLLER ERROR
4831 033020 001021          BNE    I.ERRC ;YES, PROCESS ERROR
4832 033022 016237 000016 002546    MOV     RKASOF(R2),T.ASOF ;STORE ATTENTION SUMMARY
4833 033030 133737 002633 002547    BITB   INTMSK,T.ASOF+1 ;CHECK IF DRIVE ATTENTION SET
4834 033036 001004          BNE    15$ ;YES, REPORT ERROR
4835 033040 004737 035706          JSR    PC,R.NORM ;INDICATE NORMAL RETURN
4836 033044 000137 034672          JMP     I.RTRN ;RESTORE REGISTERS
4837
4838 033050 052765 000017 000014 15$:   BIS     #UEXATT,P.PRST(R5) ;SET UNEXPECTED ATTENTION
4839
4840 033056 004737 035342          I.ERRA: JSR    PC,I.CSTS ;STORE CONTROLLER STATUS
4841 033062 000405          BR     I.ERR ;STORE PATTERN AND POSITION INFORMATION
4842
4843 033064 013765 002532 000016 I.ERRC: MOV     T.CS1,P.CS1(R5) ;GET ERROR RKCS1
4844 033072 004737 035364          JSR    PC,I.CST1 ;GET REST OF CONTROLLER STATUS
4845 033076 016265 000032 000062 I.ERR:  MOV     RKECPT(R2),P.EPAT(R5) ;STORE ECC PATTERN
4846 033104 016265 000030 000060    MOV     RKECPS(R2),P.EPOS(R5) ;STORE ECC POSITION
4847 033112 004037 034710          JSR    R0,I.CCLR ;CLEAR CONTROLLER
4848 033116 034672          I.RTRN ;ERROR RETURN
4849 033120 032765 010400 000020    BIT     #NED!UFE,P.CS2(R5) ;CHECK IF IT WAS NON-EXISTENT DRIVE OR
4850                                     ; UNIT FIELD ERROR
4851 033126 001046          BNE    5$ ;YES, REPORT ERROR
4852 033130 004037 035446          JSR    R0,I.STAT ;GATHER DRIVE STATUS
4853 033134 034672          I.RTRN ;ERROR RETURN
4854 033136 112737 000005 002532    MOVB   #DR.CLR,T.CS1 ;LOAD COMMAND
4855 033144 004037 034772          JSR    R0,I.ISSU ;ISSUE DRIVE CLEAR
4856 033150 034672          I.RTRN ;ERROR RETURN
4857 033152 133737 002633 002547    BITB   INTMSK,T.ASOF+1 ;CHECK IF ATTENTION RESET
4858 033160 001407          BEQ     2$ ;NO, INDICATE DRIVE ERROR
4859 033162 052737 000020 002604    BIS     #E.CLAT,E.CONT ;SET ATTENTION DID NOT RESET
4860                                     ; WITH CLEAR
4861 033170 004737 035720          JSR    PC,R.CONT ;REPORT CONTROLLER ERROR
4862 033174 000137 034672          JMP     I.RTRN ;GO RESTORE REGISTERS
4863
4864 033200 032737 040000 002556 2$:   BIT     #S.DSC,T.MR2 ;CHECK IF DRIVE STATUS CHANGE CLEARED
4865 033206 001403          BEQ     3$ ;YES, CHECK FAULT
4866 033210 052765 000040 000014    BIS     #DRVDSC,P.PRST(R5) ;SET DSC DID NOT CLEAR
4867 033216 032737 001000 002560 3$:   BIT     #S.PAR,T.MR3 ;CHECK IF DRIVE PARITY ERROR
4868 033224 001407          BEQ     5$ ;NO, INDICATE ABNORMAL TERMINATION
4869 033226 052737 002000 002604    BIS     #E.DPAR,E.CONT ;SET DRIVE PARITY ERROR
4870 033234 004737 035720          JSR    PC,R.CONT ;INDICATE CONTROLLER ERROR
4871 033240 000137 034672          JMP     I.RTRN ;RETURN

```

```

4872
4873 033244 032765 000020 000014 5$: BIT #DRVHRD,P.PRST(R5) ;CHECK IF HARD DRIVE ERROR
4874 033252 001017 BNE 10$ ;YES, GO REPORT ERROR
4875 033254 032737 020000 002556 BIT #S.PIP,T.MR2 ;CHECK IF DRIVE IS CYCLING DOWN
4876 033262 001413 BEQ 10$ ;NO, REPORT ERROR
4877 033264 052765 020000 000014 BIS #E.UNLD,P.PRST(R5) ;SET DRIVE UNLOADING
4878 033272 113737 002633 002632 MOV# INTMSK,W.TIME ;SET UP 8 SECONDS FOR DRIVE TO CYCLE UP
4879 033300 013737 002616 002646 MOV W.8SEC,W.DRV
4880 033306 000137 034672 JMP I.RTRN ;GO RESTORE REGISTERS
4881
4882 033312 004737 035674 10$: JSR PC,R.ABNL ;GO REPORT ERROR
4883 033316 000137 034672 JMP I.RTRN ;GO RESTORE REGISTERS
4884
4885 .SBTTL *READ ALL HEADERS INTERRUPT SEQUENCE
4886
4887 033322 016237 000000 002532 I.HDAL: MOV RKCS1(R2),T.CS1 ;STORE CS1 TO CHECK CONTROLLER
4888 ; ERROR
4889 033330 032737 100000 002532 BIT #CERR,T.CS1 ;CHECK IF CONTROLLER ERROR
4890 033336 001422 BEQ 5$ ;NO, CHECK FOR ATTENTION
4891
4892 033340 005037 002606 CLR O.WAIT ;CLEAR WAITING FOR COMMAND COMPLETE
4893 033344 105037 002632 CLR# W.TIME ;RESET TIMING ON DRIVE
4894 033350 005037 002646 CLR W.DRV ;CLEAR TIME OUT COUNT
4895 033354 013765 002532 000016 MOV T.CS1,P.CS1(R5) ;STORE ERROR RKCS1
4896 033362 004737 035364 JSR PC,I.CST1 ;STORE CONTROLLER REGISTERS
4897 033366 004037 034710 JSR RO,I.CCLR ;CLEAR CONTROLLER
4898 033372 034672 I.RTRN ;ERROR RETURN
4899 033374 004737 035674 JSR PC,R.ABNL ;INDICATE ERROR RETURN
4900 033400 000137 034672 JMP I.RTRN ;RESTORE REGISTERS
4901
4902 033404 016237 000016 002546 5$: MOV RKASOF(R2),T.ASOF ;STORE ATTENTION SUMMARY
4903 033412 133737 002633 002547 BIT# INTMSK,T.ASOF+1 ;CHECK IF DRIVE ATTENTION IS SET
4904 033420 001410 BEQ 7$ ;NO, CHECK IF READ ALL HEADERS
4905 033422 005037 002606 CLR O.WAIT ;CLEAR WAITING FOR COMMAND COMPLETION
4906 033426 105037 002632 CLR# W.TIME ;RESET TIMING ON DRIVE
4907 033432 005037 002646 CLR W.DRV ;CLEAR TIME OUT COUNT
4908 033436 000137 033056 JMP I.ERRA ;GO REPORT ERROR
4909
4910 033442 013701 002622 7$: MOV HDR.AD,R1 ;GET MAIN MEMORY ADDRESS
4911 033446 016221 000024 MOV RKDB(R2),(R1)+ ;GET FIRST WORD OF HEADER
4912 033452 016221 000024 MOV RKDB(R2),(R1)+ ;GET SECOND WORD OF HEADER
4913 033456 016221 000024 MOV RKDB(R2),(R1)+ ;GET THIRD WORD OF HEADER
4914 033462 010137 002622 MOV R1,HDR.AD ;STORE ADDRESS FOR NEXT HEADER
4915 033466 016237 000010 002534 MOV RKCS2(R2),T.CS2 ;STORE CS2 TO CHECK FOR DATA LATE
4916 033474 032737 100000 002534 BIT #DLT,T.CS2 ;CHECK FOR DATA LATE
4917 033502 001055 BNE 35$ ;YES, REPORT ERROR
4918 033504 005337 002624 DEC HDR.CT ;DECREMENT NUMBER OF HEADER YET TO READ
4919 033510 001026 BNE 25$ ;IF NON-ZERO, GO ISSUE NEXT READ HEADER
4920 033512 005037 002606 CLR O.WAIT ;CLEAR DRIVER WAITING FOR COMMAND COMPLETION
4921 033516 005037 002646 CLR W.DRV ;CLEAR TIME OUT COUNT FOR THIS DRIVE
4922 033522 105037 002632 CLR# W.TIME ;CLEAR WATCH DOG TIME ON THIS DRIVE
4923 033526 012762 000003 000026 MOV #3,RKMR1(R2) ;LOAD MAINTENANCE REGISTER FOR SECTOR COUNT
4924 033534 112737 000001 002532 MOV# #DR.SEL,T.CS1 ;LOAD SELECT COMMAND
4925 033542 004037 034772 JSR RO,I.ISSU ;GET SECTOR COUNT
4926 033546 034672 I.RTRN ;ERROR RETURN
4927 033550 013765 002560 000056 MOV T.MR3,P.B11(R5) ;LOAD SECTOR COUNT

```

```

4928 033556 004737 035706      JSR    PC,R.NORM      ;INDICATE NORMAL TERMINATION
4929 033562 000137 034672      JMP    I.RTRN        ;RESTORE REGISTERS
4930
4931 033566 016562 000002 000020 25$:  MOV    P.CYLN(R5),RKDCYL(R2) ;LOAD CYLINDER ADDRESS REGISTER
4932 033574 016562 000004 000006      MOV    P.SECT(R5),RKDA(R2) ;LOAD SECTOR AND TRACK
4933 033602 116565 000007 000017      MOVB   P.CS1H(R5),P.CS1+1(R5) ;STORE BITS 8-15 OF CS1
4934 033610 042765 165777 000016      BIC    #^C<CDT!CFMT>,P.CS1(R5) ;CLEAR ALL BITS EXCEPT FORMAT AND
4935                                     ; DRIVE TYPE
4936 033616 112765 000125 000016      MOVB   #RDHEAD,P.CS1(R5) ;STORE COMMAND ISSUED
4937 033624 016562 000016 000000      MOV    P.CS1(R5),RKCS1(R2) ;ISSUE READ HEADER
4938 033632 000137 034672      JMP    I.RTRN        ;RESTORE REGISTERS
4939
4940 033636 052737 000400 002604 35$:  BIS    #E.DLT,E.CONT    ;SET DATA LATE WHILE UNLOADING HEADER
4941 033644 004737 035720      JSR    PC,R.CONT     ;REPORT ERROR
4942 033650 000137 034672      JMP    I.RTRN        ;RESTORE REGISTERS
4943
4944                                     .SBTTL  *DRIVE ATTENTION SCANNER
4945
4946 033654 016237 000000 002532 I.ATTN: MOV    RKCS1(R2),T.CS1 ;STORE COMMAND AND STATUS
4947                                     ; REGISTER 1 FOR COMPARISON
4948 033662 032737 100000 002532      BIT    #CERR,T.CS1    ;CHECK IF CONTROLLER ERROR OCCURRED
4949 033670 001441                                     BEQ    5$             ;NO, CHECK IF ATTENTION
4950
4951                                     ;CHECK IF ANY DATA TRANSFER TYPE ERROR EXISTS
4952 033672 032737 164000 002534      BIT    #DLT!WCE!UPE!NEM,T.CS2
4953
4954 033700 001007      BNE    1$             ;INDICATE ERROR
4955 033702 016237 000014 002550      MOV    RKER(R2),T.ER  ;STORE ERROR REGISTER
4956
4957                                     ;
4958 033710 032737 125700 002550      BIT    #DCK!OPI!WLE!COE!HVRC!BSE!ECH,T.ER
4959
4960 033716 001407      BEQ    2$             ;NO DATA TRANSFER ERROR
4961
4962 033720 052737 000010 002604 1$:  BIS    #E.UDAT,E.CONT ;SET UNEXPECTED DATA TYPE ERROR
4963 033726 004737 035720      JSR    PC,R.CONT     ;REPORT ERROR
4964 033732 000137 034672      JMP    I.RTRN        ;RESTORE REGISTERS
4965
4966 033736 013704 002534 2$:  MOV    T.CS2,R4       ;SAVE CS2 FOR REGISTER NUMBER
4967 033742 042704 177770      BIC    #^C<DRVMSK>,R4 ;STRIP OFF JUNK
4968 033746 105037 002632      CLRB   W.TIME        ;CLEAR WATCH DOG TIMER
4969 033752 005037 002646      CLR    W.DRV         ;RESET TIMER VALUE
4970 033756 013705 002644      MOV    PBLKT,R5      ;STORE PARAMETER BLOCK ADDRESS IN R5
4971
4972                                     ;
4973                                     ; CLEAR DRIVE POSITIONING AND DRIVE POSITIONED FOR DATA TRANSFER
4974 033762 042765 000006 000014      BIC    #DRVPOS!DRVPDT,P.PRST(R5)
4975
4976 033770 000137 033064      JMP    I.ERRC        ;GO REPORT ERROR
4977
4978 033774 032737 040000 002532 5$:  BIT    #DI,T.CS1     ;CHECK IF ANY DRIVE ATTENTION
4979 034002 001002      BNE    6$             ;YES, PROCESS INTERRUPT
4980 034004 000137 034672      JMP    I.RTRN        ;RESTORE REGISTERS
4981
4982 034010 016237 000016 002546 6$:  MOV    RKASOF(R2),T.ASOF ;STORE ATTENTION SUMMARY
4983 034016 105737 002547      TSTB   T.ASOF+1     ;CHECK IF ANY ATTENTIONS SET

```



```

5040 .SBTTL *ATTENTION ERROR HANDLER
5041
5042 034324 042765 000004 000014 I.AERR: BIC #DRV PDT,P.PRST(R5) ;RESET POSITIONING IN PROGRESS BECAUSE
5043 ; OF DATA TRANSFER
5044 034332 105037 002632 CLRB W.TIME ;CLEAR TIMING FOR THIS DRIVE
5045 034336 005037 002646 CLR W.DRV ;RESET WATCH-DOG TIME
5046 034342 042765 177741 000016 BIC #177741,P.CS1(R5) ;KEEP COMMAND ISSUED
5047 034350 042737 000036 002532 BIC #36,T.CS1 ;KEEP CURRENT CONTROLLER STATUS
5048 034356 053765 002532 000016 BIS T.CS1,P.CS1(R5) ;MAKE GOOD MESSAGE
5049 034364 013765 002534 000020 MOV T.CS2,P.CS2(R5) ;STORE CONTROLLER REGISTERS
5050 034372 013765 002536 000022 MOV T.WCR,P.WCR(R5)
5051 034400 013765 002540 000024 MOV T.BA,P.BAR(R5)
5052 034406 013765 002542 000026 MOV T.DA,P.DTS(R5)
5053 034414 013765 002544 000030 MOV T.DC,P.DCYL(R5)
5054 034422 013765 002546 000032 MOV T.ASOF,P.ASOF(R5)
5055 034430 013765 002550 000034 MOV T.ER,P.ER(R5)
5056 034436 013765 002552 000036 MOV T.DS,P.DS(R5)
5057 034444 004037 035446 JSR RO,I.STAT ;GATHER DRIVE STATUS
5058 034450 034672 I.RTRN ;ERROR RETURN
5059 034452 112737 000005 002532 MOVB #DR.CLR,T.CS1 ;LOAD COMMAND
5060 034460 004037 034772 JSR RO,I.ISSU ;CLEAR DRIVE ERRORS
5061 034464 034672 I.RTRN ;ERROR RETURN
5062 034466 133737 002633 002547 BITB INTMSK,T.ASOF+1 ;CHECK IF ATTENTION RESET
5063 034474 001407 BEQ 2$ ;YES, FLAG DRIVE ERROR
5064 034476 052737 000020 002604 BIS #E.CLAT,E.CONT ;SET ATTENTION DID NOT RESET
5065 034504 004737 035720 JSR PC,R.CONT ;REPORT ERROR
5066 034510 000137 034672 JMP I.RTRN ;RESTORE REGISTERS
5067
5068 034514 032765 000020 000014 2$: BIT #DRVHRD,P.PRST(R5) ;CHECK IF A HARD DRIVE ERROR
5069 034522 001017 BNE 10$ ;YFS, REPORT ERROR
5070 034524 032737 000000 002556 BIT #S.PIP,T.MR2 ;CHECK IF DRIVE IS UNLOADING
5071 034532 001413 BEQ 10$ ;NO, REPORT ERROR
5072 034534 052765 020000 000014 BIS #E.UNLD,P.PRST(R5) ;SET DRIVE UNLOADING DUE TO ERROR
5073 034542 113737 002633 002632 MOVB INTMSK,W.TIME ;SET TIMING ON THIS DRIVE
5074 034550 013737 002616 002646 MOV W.8SEC,W.DRV ;LOAD 8 SECONDS FOR CYCLE UP TIME
5075 034556 000137 034672 JMP I.RTRN ;RESTORE REGISTERS
5076
5077 034562 004737 035674 10$: JSR PC,R.ABNL ;REPORT ERROR
5078 034566 000137 034672 JMP I.RTRN ;RESTORE REGISTERS
5079
5080 .SBTTL *ERROR CAUSING DRIVE TO UNLOAD
5081
5082 034572 052765 020000 000014 I.UNLD: BIS #E.UNLD,P.PRST(R5) ;CLEAR DRIVE UNLOADING BECAUSE OF ERROR
5083 034600 112737 000005 002532 MOVB #DR.CLR,T.CS1 ;LOAD IN DRIVE CLEAR
5084 034606 004037 034772 JSR RO,I.ISSU ;GO ISSUE DRIVE CLEAR
5085 034612 034672 I.RTRN ;ERROR RETURN
5086 034614 136437 002633 002547 BITB INTMSK(R4),T.ASOF+1 ;CHECK IF ATTENTION CLEARED
5087 034622 001406 BEQ 15$ ;YES, CONTINUE
5088 034624 012737 000020 002604 MOV #E.CLAT,E.CONT ;SET ATTENTION DID NOT RESET
5089 034632 004737 035720 JSR PC,R.CONT ;REPORT ERROR
5090 034636 000415 BR I.RTRN ;RESTORE REGISTERS
5091
5092 034640 032737 040000 002556 15$: BIT #S.DSC,T.MR2 ;CHECK IF DRIVE STAU CHANGE RESET
5093 034646 001403 BEQ 20$ ;YES, CONTINUE
5094 034650 052765 000040 000014 BIS #DRV DSC,P.PRST(R5) ;SET DRIVE STAU CHANGE DID NOT CLEAR
5095 034656 105037 002632 20$: CLRB W.TIME ;RESET TIMING ON THIS DRIVE

```


5096	034662	005037	002646	CLR	W.DRV	:CLEAR TIME COUNT
5097	034666	004737	035674	JSR	PC,R.ABNL	:REPORT ERROR
5098						
5099	034672	012600		I.RTRN: MOV	(SP)+,R0	:RESTORE R0
5100	034674	012601		MOV	(SP)+,R1	:RESTORE R1
5101	034676	012602		MOV	(SP)+,R2	:RESTORE R2
5102	034700	012603		MOV	(SP)+,R3	:RESTORE R3
5103	034702	012604		MOV	(SP)+,R4	:RESTORE R4
5104	034704	012605		MOV	(SP)+,R5	:RESTORE R5
5105	034706	000002		RTI		:RETURN
5106						

.SBTTL *CONTROLLER CLEAR ROUTINE

5107
5108
5109
5110
5111
5112
5113
5114
5115
5116
5117
5118
5119
5120
5121
5122
5123
5124
5125
5126
5127
5128
5129
5130
5131
5132
5133
5134
5135
5136
5137
5138
5139
5140
5141

```
*****  
* THIS ROUTINE WILL BE USED BY THE DRIVER TO CLEAR THE CONTROLLER  
* AND CHECK IF THE CONTROLLER ERRORS ARE RESET. IF THE ERROR IS NOT  
* CLEARED, THE ROUTINE AS SPECIFIED IN A.CONT WILL BE CALLED WITH  
* E.CCLR SET IN E.CONT.  
*  
* REGISTER      USE  
* -----      ---  
*  
* R2            ADDRESS OF RK06 REGISTERS  
* R5            ADDRESS OF PARAMETER BLOCK  
*  
*CALL JSR      R0,I.CCLR  
*          <ADDRESS OF ERROR RETURN>  
*          RETURN  
*****
```

```
I.CCLR: MOV      #CCLR,RKCS1(R2) ;CLEAR CONTROLLER  
        MOV      RKCS1(R2),T.CS1 ;STORE COMMAND AND STATUS REGISTER 1  
        BIT      #CERR,T.CS1    ;CHECK IF CONTROLLER CLEAR DID  
        ;        CLEAR ERROR  
        BEQ      5$            ;YES, RETURN TO DRIVER PROCESSING  
        BIS      #E.CCLR,E.CONT ;SET CLEAR CONTROLLER DID NOT CLEAR EPROR  
        JSR      PC,R.CONT      ;REPORT CONTROLLER ERROR  
        MOV      (R0),R0        ;SET UP ERROR RETURN  
        RTS      R0            ;RETURN  
  
5$:     MOV      #IE,RKCS1(R2)  ;SET INTERRUPT ENABLE  
        MOVB     #-1,I.ISRL     ;SET INTERRUPT ENABLE ISSUED  
        TST      (R0)+          ;ADJUST FOR NORMAL RETURN  
        RTS      R0            ;RETURN
```

SBTTL *COMMAND ISSUED BY DRIVER SERVICE ROUTINE

5142
5143
5144
5145
5146
5147
5148
5149
5150
5151
5152
5153
5154
5155
5156
5157
5158
5159
5160
5161
5162
5163
5164
5165
5166
5167
5168
5169
5170
5171
5172
5173
5174
5175
5176
5177
5178
5179
5180
5181
5182
5183
5184
5185
5186
5187
5188
5189
5190
5191
5192
5193
5194
5195
5196
5197

```
*****  
* THIS ROUTINE WILL ISSUE THE COMMAND AS SPECIFIED IN T.CS1  
* AND CHECK IF A CONTROLLER ERROR OCCURRED. IF A CONTROLLER  
* ERROR OCCURRED, E.CERR WILL BE SET IN E.CONT AND  
* CONTROL WILL BE TURN OVER TO THE ROUTINE SPECIFIED BY THE  
* ADDRESS IN A.CONT.  
*  
* REGISTER      USE  
* -----      ---  
*  
* R2            ADDRESS OF RK06 REGISTERS  
* R5            ADDRESS OF PARAMETER BLOCK  
*  
*CALL JSR      R0,I.ISSU  
*          <ADDRESS OF ERROR RETURN>  
*          RETURN  
*  
* ROUTINES USED:  
* -----  
*  
*          I.CCLR  
*          I.STOR  
*****
```

```
I.ISSU: MOV      T.CS1,-(SP)      ;STORE COMMAND ISSUED  
        CLR      T.CS2          ;CLEAR TEMPORARY CS2  
        MOVB     P.DRVN(R5),T.CS2 ;LOAD IN DRIVE NUMBER  
        MOV      T.CS2,RKCS2(R2) ;LOAD DRIVE NUMBER FOR COMMAND  
        MOVB     P.CS1H(R5),T.CS1+1 ;STORE BITS 8-15 OF CS1  
        BICB     #^C<B.CDT!B.CFMT>,T.CS1+1 ;CLEAR ALL BITS EXCEPT  
        ;        FORMAT AND DRIVE TYPE  
        MOV      T.CS1,RKCS1(R2) ;ISSUE COMMAND  
1$:     TSTB     RKCS1(R2)      ;WAIT FOR READY  
        BPL      1$  
        JSR      PC,I.STOR      ;GO STORE REGISTERS  
        BIT      #CERR,T.CS1    ;CHECK IF CONTROLLER ERROR OCCURED  
        BEQ      5$            ;NO, RETURN  
        BIT      #MDS,T.CS2     ;CHECK IF MULTIPLE DRIVE SELECT  
        BEQ      2$            ;NO, CHECK FOR OTHER CONTROLLER ERRORS  
        BIS      #E.MDS,E.CONT  ;SET MULTIPLE DRIVE SELECT FLAG  
        JSR      PC,R.CONT      ;REPORT CONTROLLER ERROR  
        BR       10$           ;RETURN  
        ;CHECK IF ANY CONTROLLER ERROR IS SET  
2$:     BIT      #CTO!SPAR,T.CS1  
        BNE      7$  
        BIT      #UFE!PGE!NEM!NED!UPE!WCE!DLT,T.CS2  
        BNE      7$  
        BIT      #ILC!DTYE!FMTE!ECH!BSE!HVRC!COE!DTE!OP!DCK,T.ER  
        BNE      7$  
        CMPB     #DR.CLR,(SP)   ;CHECK IF CLEAR DRIVE
```

```

5198 035142 001003          BNE      3$      ;NO, DO NOT SET DRIVE HARD ERROR
5199 035144 052765 000020 000014  BIS      #DRVHRD,P.PRST(R5) ;SET HARD DRIVE ERROR
5200 035152 004037 034710          JSR      RO,I.CCLR      ;GO ISSUE A CONTROLLER CLEAR
5201 035156 035206          10$      ;ERROR RETURN
5202 035160 012762 000100 000000  5$:     MOV      #IE,RKCS1(R2) ;SET INTERRUPT ENABLE
5203 035166 005726          TST      (SP)+      ;ADJUST STACK
5204 035170 005720          TST      (RO)+      ;ADJUST RO FOR NORMAL RETURN
5205 035172 000200          RTS      RO          ;RETURN
5206
5207 035174 052737 001000 002604  7$:     BIS      #E.CERR,E.CONT ;SET CONTROLLER ERROR DURING
5208                                     ; DRIVER SERVICING
5209 035202 004737 035720          JSR      PC,R.CONT   ;REPORT ERROR
5210 035206 005726          10$:     TST      (SP)+      ;ADJUST STACK
5211 035210 011000          MOV      (RO),RO    ;ADJUST RO FOR ERROR RETURN
5212 035212 000200          RTS      RO          ;RETURN
  
```

5213
5214
5215
5216
5217
5218
5219
5220
5221
5222
5223
5224
5225
5226
5227
5228
5229
5230
5231
5232
5233
5234
5235
5236
5237
5238
5239
5240
5241
5242
5243
5244

.SBTTL *STORE RK611 UNIBUS REGISTERS

```
*****  
*  
* THIS SUBROUTINE IS CALLED BY THE RK06 DRIVER TO STORE ALL  
* RK611 REGISTER IN TEMPORARY LOCATIONS.  
*  
*CALL JSR PC,I.STOR  
* RETURN  
*  
* REGISTER USE  
* -----  
* R2 ADDRESS OF RK611 REGISTERS  
*****
```

035214	016237	000000	002532
035222	016237	000010	002534
035230	016237	000002	002536
035236	016237	000004	002540
035244	016237	000006	002542
035252	016237	000012	002552
035260	016237	000014	002550
035266	016237	000016	002546
035274	016237	000020	002544
035302	016237	000026	002554
035310	016237	000034	002556
035316	016237	000036	002560
035324	016237	000030	002562
035332	016237	000032	002564
035340	000207		

```
I.STOR: MOV RKCS1(R2),T.CS1 ;STORE ALL CONTROLLER REGISTERS  
MOV RKCS2(R2),T.CS2 ; EXCEPT DATA BUFFER  
MOV RKWC(R2),T.WCR  
MOV RKBA(R2),T.BA  
MOV RKDA(R2),T.DA  
MOV RKDS(R2),T.DS  
MOV RKER(R2),T.ER  
MOV RKASOF(R2),T.ASOF  
MOV RKDCYL(R2),T.DC  
MOV RKMR1(R2),T.MR1  
MOV RKMR2(R2),T.MR2  
MOV RKMR3(R2),T.MR3  
MOV RKECPS(R2),T.POS  
MOV RKECPT(R2),T.PAT  
RTS PC ;RETURN
```

5245
5246
5247
5248
5249
5250
5251
5252
5253
5254
5255
5256
5257
5258
5259
5260
5261
5262
5263
5264
5265
5266
5267
5268
5269
5270
5271
5272
5273
5274
5275
5276
5277
5278
5279
5280
5281
5282
5283
5284
5285
5286
5287
5288
5289

.SBTTL *STORE CONTROLLER STATUS

```

:*****
:
: THIS SUBROUTINE IS CALLED BY THE RK06 DRIVER AT PRIORITY 7.
: THE FOLLOWING REGISTERS WILL BE STORED:
:

```

```

:
: COMMAND AND STATUS REGISTER 2
: WORD COUNT REGISTER
: BUS ADDRESS REGISTER
: DESIRED TRACK AND SECTOR
: STATUS REGISTER
: ERROR REGISTER
: ATTENTION SUMMARY/OFFSET REGISTER
: CYLINDER ADDRESS REGISTER
:

```

```

:CALL JSR PC,I.CSTS
:      RETURN
:

```

```

: THIS ROUTINE ASSUMES THE FOLLOWING REGISTERS CONTAIN:
:

```

REGISTER	CONTENTS
-----	-----
R2	RK06 BASE ADDRESS
R5	ADDRESS OF PARAMETER BLOCK

```

:*****
:
: I.CSTS: BIC #177741,P.CS1(R5) ;CLEAR ALL BITS EXCEPT FUNCTION
: ;OF LAST COMMAND ISSUED
: BIC #36,T.CS1 ;CLEAR FUNCTION OF CS1 STATUS
: BIS T.CS1,P.CS1(R5) ;GENERATE CS1 STATUS INFORMATION
: I.CST1: MOV RKCS2(R2),P.CS2(R5) ;STORE COMMAND AND STATUS REGISTER 2
: MOV RKWC(R2),P.WCR(R5) ;STORE WORD COUNT REGISTER
: MOV RKBA(R2),P.BAR(R5) ;STORE BUS ADDRESS REGISTER
: MOV RKDA(R2),P.DTS(R5) ;STORE DESIRED TRACK AND SECTOR
: MOV RKDS(R2),P.DS(R5) ;STORE DRIVE STATUS REGISTER
: MOV RKER(R2),P.ER(R5) ;STORE ERROR REGISTER
: MOV RKASOF(R2),P.ASOF(R5) ;STORE ATTENTION SUMMARY AND
: ; OFFSET
: MOV RKDCYL(R2),P.DCYL(R5) ;STORE CYLINDER ADDRESS
: RTS PC ;RETURN
:

```

5290
5291
5292
5293
5294
5295
5296
5297
5298
5299
5300
5301
5302
5303
5304
5305
5306
5307
5308
5309
5310
5311
5312
5313
5314
5315
5316
5317
5318
5319
5320
5321
5322
5323
5324
5325
5326
5327
5328
5329
5330
5331
5332
5333
5334
5335
5336
5337
5338
5339
5340
5341
5342
5343
5344
5345

.SBTTL *GATHER DRIVE STATUS

THIS SUBROUTINE WILL BE USED TO GATHER DRIVE STATUS
 BYTE 01, 10, AND 11. IT IS ASSUMED THAT THE DRIVE
 HAS PREVIOUSLY BEEN SEIZED. IT RUNS AT PRIORITY 7.

*CALL JSR R0,I.STAT
 <ADDRESS OF ERROR RETURN>
 RETURN

THIS ROUTINE ASSUMES THE FOLLOWING REGISTERS CONTAIN:

REGISTER	CONTENTS
R2	RK06 BASE ADDRESS
R5	ADDRESS OF PARAMETER BLOCK

ROUTINES USED:
 I.ISSU

```

5316 035446 012762 000001 000026 I.STAT: MOV #1,RKMR1(R2) ;LOAD MAINTENANCE REGISTER 1
5317 ; FOR STATUS BYTE 01
5318 035454 112737 000001 002532 MOVB #DR.SEL,T.CS1 ;LOAD COMMAND
5319 035462 004037 034772 JSR R0,I.ISSU ;GET STATUS BYTES 01
5320 035466 035656 3$ ;ERROR RETURN
5321 035470 013765 002556 000044 MOV T.MR2,P.A01(R5) ;STORE STATUS BYTE 01 MESS A
5322 035476 013765 002560 000046 MOV T.MR3,P.B01(R5) ;STORE STATUS BYTE 01 MESS B
5323 035504 012762 000002 000026 MOV #2,RKMR1(R2) ;LOAD MAINTENANCE REGISTER 1
5324 ; FOR STATUS BYTE 10
5325 035512 112737 000001 002532 MOVB #DR.SEL,T.CS1 ;LOAD COMMAND
5326 035520 004037 034772 JSR R0,I.ISSU ;GET STATUS BYTES 10
5327 035524 035656 3$ ;ERROR RETURN
5328 035526 013765 002556 000050 MOV T.MR2,P.A10(R5) ;STORE STATUS BYTE 10 MESS A
5329 035534 013765 002560 000052 MOV T.MR3,P.B10(R5) ;STORE STATUS BYTE 10 MESS B
5330 035542 012762 000003 000026 MOV #3,RKMR1(R2) ;LOAD MAINTENANCE REGISTER
5331 ; FOR STATUS BYTE 11
5332 035550 112737 000001 002532 MOVB #DR.SEL,T.CS1 ;LOAD COMMAND
5333 035556 004037 034772 JSR R0,I.ISSU ;GET STATUS BYTES 11
5334 035562 035656 3$ ;ERROR RETURN
5335 035564 013765 002556 000054 MOV T.MR2,P.A11(R5) ;STORE STATUS BYTE 11 MESS A
5336 035572 013765 002560 000056 MOV T.MR3,P.B11(R5) ;STORE STATUS BYTE 11 MESS B
5337 035600 005062 000026 CLR RKMR1(R2) ;LOAD MAINTENANCE REGISTER 1
5338 ; FOR STATUS BYTE 00
5339 035604 112737 000001 002532 MOVB #DR.SEL,T.CS1 ;LOAD COMMAND
5340 035612 004037 034772 JSR R0,I.ISSU ;GET STATUS BYTES 00
5341 035616 035656 3$ ;ERROR RETURN
5342 035620 013765 002556 000040 MOV T.MR2,P.A00(R5) ;STORE STATUS BYTE 00 MESS A
5343 035626 013765 002560 000042 MOV T.MR3,P.B00(R5) ;STORE STATUS BYTE 00 MESS B
5344 035634 032737 001000 002560 BIT #S.PAR,T.MR3 ;CHECK IF BAD PARITY DETECTED BY DRIVE
5345 035642 001407 BEQ 5$ ;NO, RETURN NORMALLY
    
```

5346	035644	052737	002000	002604		BIS	#E.DPAR,E.CONT	:INDICATE BAD PARITY DETECTED BY DRIVE
5347	035652	004737	035720			JSR	PC,R.CONT	:REPORT ERROR
5348	035656	011000			3\$:	MOV	(R0),R0	:LOAD R0 FOR ERROR RETURN
5349	035660	000200				RTS	R0	:RETURN
5350								
5351	035662	052765	001000	000014	5\$:	BIS	#PBSVAL,P.PRST(R5)	:SET PARAMETER BLOCK STATUS VALID
5352	035670	005720				TST	(R0)+	:ADJUST R0 FOR NORMAL RETURN
5353	035672	000200				RTS	R0	:RETURN
5354								


```
5355 .SBTTL *COMMON DRIVER RETURNS
5356
5357 035674 105037 002633 R.ABNL: CLR B INTMSK :INHIBIT FUTURE DRIVE INTERRUPT REPORTING
5358 035700 004777 144674 JSR PC,@A.ABNL :INDICATE ABNORMAL RETURN
5359 035704 000207 RTS PC :RETURN
5360
5361 035706 105037 002633 R.NORM: CLR B INTMSK :INHIBIT FUTURE DRIVE INTERRUPT REPORTING
5362 035712 004777 144660 JSR PC,@A.NORM :INDICATE NORMAL RETURN
5363 035716 000207 RTS PC :RETURN
5364
5365 035720 105037 002633 R.CONT: CLR B INTMSK :INHIBIT FUTURE DRIVE INTERRUPT REPORTING
5366 035724 105037 002632 CLR B W.TIME :RESET WATCH DOG TIMING ON THIS DRIVE
5367 035730 005037 002646 CLR W.DRV :CLEAR TIMING COUNT FOR THIS DRIVE
5368 035734 004777 144642 JSR PC,@A.CONT :INDICATE CONTROLLER ERROR RETURN
5369 035740 000207 RTS PC :RETURN
```

5370
5371
5372
5373
5374
5375
5376
5377
5378
5379
5380
5381
5382
5383
5384
5385
5386
5387
5388
5389
5390
5391
5392
5393
5394
5395
5396
5397
5398
5399
5400
5401
5402
5403
5404
5405
5406
5407
5408
5409
5410
5411
5412
5413
5414
5415
5416
5417
5418
5419
5420
5421
5422
5423
5424
5425

```
.SBTTL *COMMAND INITATOR

*****
:
: THIS SUBROUTINE WILL INITIATE ALL COMMANDS AS SPECIFIED
: BY THE COMMAND FIELD OF THE PARAMETER BLOCK. THE FOLLOWING
: SPECIAL COMMAND ARE ALSO EXECUTED:
:
:         RELEASE
:         CONROLLER CLEAR
:         SUBSYSTEM CLEAR
:         READ ALL DRIVE STATUS
:         READ SPECIFIED HEADER
:
: THE ABOVE COMMANDS ARE TRANSLATED INTO A SEQUENCE OF COMMANDS
:
: *CALL JSR    PC,C.INIT
:        <ADDRESS OF PARAMETER BLOCK>
:        RETURN
:
: FOR THE SEQUENTIAL OPERATIONS, THE DRIVER WILL LOAD THE
: LOCATIONS, PBLKT AND INTMSK.
:
: ROUTINES USED:
:         W.WTCH
:         I.CSTS
:         I.STAT
:         I.CCLR
:
: *****

C.INIT: MOV    R5,-(SP)      ;STORE R5 ON STACK
        MOV    R4,-(SP)      ;STORE R4 ON STACK
        MOV    R3,-(SP)      ;STORE R3 ON STACK
        MOV    R2,-(SP)      ;STORE R2 ON STACK
        MOV    R1,-(SP)      ;STORE R1 ON STACK
        MOV    R0,-(SP)      ;STORE R0 ON STACK
        MOV    PS,-(SP)      ;STORE PSW ON STACK
        MOV    RKPRI,PS      ;LOCK OUT RK06 INTERRUPTS
        MOV    @16(SP),R5     ;STORE PARAMETER BLOCK ADDRESS
        ADD    #2,16(SP)     ;ADJUST RETURN
        MOV    P.DRVN(R5),R4  ;STORE DRIVE NUMBER
        BIC    #^C<DRVMSK>,R4 ;MASK OUT JUNK
        MOV    R5,PBLKT      ;LOAD PARAMETER BLOCK TABLE
        MOVB   I.DRV(R4),INTMSK ;LOAD INTERRUPT MASK
        MOVB   I.DRV(R4),W.TIME ;SET WATCH-DOG TIMER FLAG
        MOV    W.SEC,W.DRV    ;LOAD WATCH-DOG TIME

        MOV    RKBAS,R2      ;LOAD R2 WITH RK06 ADDRESS BASE

:
: RESET ALL BITS IN PROGRAM DEVICE STATUS REGISTER EXCEPT
: DRIVE IN USE
: WRITE FOR WRITE CHECK
: NO CHECK
: DROP DRIVE FROM TEST SEQUENCE
: INHIBIT BUS ADDRESS INCREMENT
```

035742	010546		
035744	010446		
035746	010346		
035750	010246		
035752	010146		
035754	010046		
035756	013746	177776	
035762	013737	002574	177776
035770	017605	000016	
035774	062766	000002	000016
036002	016504	000000	
036006	042704	177770	
036012	010537	002644	
036016	116437	002634	002633
036024	116437	002634	002632
036032	013737	002614	002646
036040	013702	002570	

```

5426 036044 042765 075176 000014      BIC      #*C<DRVUSE!W.WCK!NOCHK!DPPDRV!DTBAII>,P.PRST(R5)
5427
5428 036052 010500      MOV      R5,R0          ;STORE PARAMETER BLOCK ADDRESS
5429 036054 062700 000016      ADD      #P.CS1,R0      ;CALCULATE FIRST LOCATION TO BE CLEARED
5430 036060 010501      MOV      R5,R1          ;STORE PARAMETER BLOCK ADDRESS
5431 036062 062701 000062      ADD      #P.EPAT,R1     ;CALCULATE LAST LOCATION TO BE CLEARED
5432
5433 036066 005020      1$:     CLR      (R0)+          ;CLEAR RETURN PARAMETER
5434 036070 020001      CMP      R0,R1          ;CHECK IF FINISHED
5435 036072 101775      BLOS     1$             ;NO, CLEAR NEXT RETURN PARAMETER
5436 036074 105037 002626      CLRB     I.ISRL         ;CLEAR RELEASE OR INTERRUPT ISSUED
5437 036100 010465 000020      MOV      R4,P.CS2(R5)   ;STORE DRIVE NUMBER
5438 036104 005062 000026      CLR      RKMRI(R2)      ;CLEAR RK06 MAINTENANCE REGISTER 1
5439 036110 132765 000040 000001      BITB     #BIT5,P.CMND(R5);CHECK IF SPECIAL COMMAND
5440 036116 001402      BEQ      3$             ;NO, PROCESS
5441 036120 000137 036634      JMP      C.SPEC         ;JUMP TO SPECIAL COMMAND PROCESSOR
5442
5443 036124 122765 000107 000001 3$:     CMPB     #UNLOAD,P.CMND(R5);CHECK IF POSITIONING COMMAND
5444                                     ; START SPINDLE
5445                                     ; RECALIBRATE
5446                                     ; OFFSET
5447                                     ; SEEK
5448                                     ; UNLOAD
5449
5450 036132 101174      BHI      25$            ;NO, DRIVE COMMAND
5451                                     ; SELECT DRIVE
5452                                     ; PACK ACKNOWLEDGE
5453                                     ; CLEAR
5454
5455 036134 122765 000117 000001      CMPB     #SEEK,P.CMND(R5);CHECK IF DATA TRANSFER
5456 036142 103540      BLO      20$            ;YES, DATA TRANSFER COMMAND
5457                                     ; READ DATA
5458                                     ; WRITE DATA
5459                                     ; READ HEADER
5460                                     ; WRITE HEADER
5461                                     ; WRITE CHECK
5462 036144 016562 000020 000010      MOV      P.CS2(R5),RKCS2(R2);LOAD DRIVE NUMBER
5463 036152 052765 000002 000014      BIS      #DRVPOS,P.PRST(R5);SET DRIVE POSITIONING
5464 036160 005037 002606      CLR      O.WAIT         ;CLEAR WAIT FOR COMMAND
5465 036164 122765 000117 000001      CMPB     #SEEK,P.CMND(R5);CHECK IF SEEK
5466 036172 001007      BNE      5$             ;NO, CHECK FOR OFFSET
5467 036174 016562 000002 000020      MOV      P.CYLN(R5),RKDCYL(R2);LOAD CYLINDER ADDRESS
5468 036202 016562 000004 000006      MOV      P.SECT(R5),RKDA(R2);LOAD SECTOR AND TRACK
5469 036210 000431      BR       8$             ;GO ISSUE COMMAND
5470
5471 036212 122765 000115 000001 5$:     CMPB     #OFFSET,P.CMND(R5);CHECK IF OFFSET
5472 036220 001007      BNE      6$             ;NO, CHECK FOR UNLOAD
5473 036222 116565 000006 000032      MOVB     P.OFST(R5),P.ASOF(R5);STORE OFFSET
5474 036230 016562 000032 000016      MOV      P.ASOF(R5),RKASOF(R2);LOAD OFFSET REGISTER
5475 036236 000416      BR       8$             ;GO ISSUE COMMAND
5476
5477 036240 122765 000111 000001 6$:     CMPB     #SRTSPL,P.CMND(R5);CHECK IF START SPINDLE
5478 036246 001003      BNE      7$             ;NO, CHECK IF RECAL
5479 036250 013737 002620 002646      MOV      W.MIN,W.DRV     ;LOAD WATCH DOG TIME FOR 1 MINUTE
5480 036256 122765 000113 000001 7$:     CMPB     #RECAL,P.CMND(R5);CHECK IF RECAL
5481 036264 001003      BNE      8$             ;NO, CONTINUE

```

```

5482 036266 013737 002616 002646      MOV      W.8SEC,W.DRV      ;LOAD RECAL TIME FOR 8 SECONDS
5483 036274 116565 000007 000017 8$:      MOVVB   P.CS1H(R5),P.CS1+1(R5) ;STORE BITS 8-15 OF CS1
5484 036302 042765 165777 000016      BIC     #^C<CFMT!CDT>,P.CS1(R5) ;CLEAR ALL BITS EXCEPT FORMAT
5485                                     ; AND DRIVE TYPE
5486 036310 116565 000001 000016      MOVVB   P.CMND(R5),P.CS1(R5) ;MOVE COMMAND INTO CS1
5487 036316 042765 000200 000014      BIC     #W.WCK,P.PRST(R5) ;RESET WRITE FOR WRITE CHECK
5488 036324 032765 000400 000014      BIT     #NOCHK,P.PRST(R5) ;CHECK IN NO CHECK MODE
5489 036332 001533                                     BEQ     30$                ;NO, SKIP CLEAR OF INTERRUPT ENABLE
5490 036334 042765 000100 000016      BIC     #IE,P.CS1(R5)      ;CLEAR INTERRUPT ENABLE
5491 036342 016562 000016 000000      MOV     P.CS1(R5),RKCS1(R2) ;ISSUE COMMAND
5492 036350 004737 032312 000000 10$:     JSR     PC,W.WTCH          ;CALL WATCH DOG TIMER
5493 036354 016237 000000 002532      MOV     RKCS1(R2),T.CS1    ;STORE COMMAND AND STATUS REGISTER 1
5494 036362 032737 000200 002532      BIT     #RDY,T.CS1        ;WAIT FOR READY
5495 036370 001767                                     BEQ     10$
5496 036372 032737 100000 002532      BIT     #CERR,T.CS1       ;CHECK FOR ERROR
5497 036400 001011                                     BNE     15$                ;YES, GIVE NORMAL RETURN
5498 036402 004737 032312 000016 11$:     JSR     PC,W.WTCH          ;CALL WATCH DOG TIMER
5499 036406 016237 000016 002546      MOV     RKASOF(R2),T.ASOF  ;STORE ATTENTION SUMMARY
5500 036414 133737 002633 002547      BITB   INTMSK,T.ASOF+1    ;CHECK IF INTERRUPT HAS OCCURRED
5501 036422 001767                                     BEQ     11$                ;WAIT FOR DRIVE INTERRUPT
5502 036424 105037 002632 000016 15$:     CLR     W.TIME            ;RESET TIMING ON THIS DRIVE
5503 036430 005037 002646                                     CLR     W.DRV              ;CLEAR DRIVE TIMING COUNT
5504 036434 004737 035706                                     JSR     PC,R.NORM          ;INDICATE COMMAND IS FINISHED
5505 036440 000137 037614                                     JMP     C.RTRN             ;RESTORE REGISTERS
5506
5507 036444 016562 000010 000004 20$:     MOV     P.BALO(R5),RKBA(R2) ;LOAD BUS ADDRESS REGISTER
5508 036452 016562 000012 000002      MOV     P.WC(R5),RKWC(R2)  ;LOAD WORD COUNT REGISTER
5509 036460 016562 000002 000020      MOV     P.CYLN(R5),RKDCYL(R2) ;LOAD CYLINDER ADDRESS REGISTER
5510 036466 016562 000004 000006      MOV     P.SECT(R5),RKDA(R2) ;LOAD SECTOR AND TRACK NUMBER
5511 036474 122765 000131 000001      CMPB   #WRTCHK,P.CMND(R5) ;CHECK IF WRITE CHECK COMMAND
5512 036502 001010                                     BNE     25$                ;NO, GO ISSUE THE COMMAND
5513 036504 032765 000200 000014      BIT     #W.WCK,P.PRST(R5) ;CHECK IF WRITE COMMAND SHOULD BE ISSUED
5514 036512 001404                                     BEQ     25$                ;NO, GO ISSUE THE COMMAND
5515 036514 012765 000123 000016      MOV     #WRDATA,P.CS1(R5) ;ISSUE WRITE COMMAND
5516 036522 000406                                     BR      26$                ;GO ISSUE COMMAND
5517
5518 036524 116565 000001 000016 25$:     MOVVB   P.CMND(R5),P.CS1(R5) ;MOVE COMMAND INTO CS1
5519 036532 042765 000200 000014      BIC     #W.WCK,P.PRST(R5) ;RESET WRITE FOR WRITE CHECK
5520 036540 116565 000007 000017 26$:     MOVVB   P.CS1H(R5),P.CS1+1(R5) ;STORE BITS 8-15 OF CS1
5521 036546 142765 177750 000017      BICB   #^C<B.CFMT!B.CDT!B.BA16!B.BA17>,P.CS1+1(R5) ;CLEAR ALL BITS EXCEPT
5522                                     ; FORMAT, DRIVE TYPE, AND BUS ADDRESS
5523                                     ; BITS 16-17
5524 036554 010537 002606                                     MOV     R5,O.WAIT          ;LOAD WAITING FOR COMMAND
5525 036560 032765 100000 000014      BIT     #DTBAII,P.PRST(R5) ;CHECK IF INHIBIT BUS ADDRESS INCREMENT
5526 036566 001403                                     BEQ     27$                ;NO, LOAD CS2
5527 036570 052765 000020 000020      BIS     #BAI,P.CS2(R5)     ;SET INHIBIT BUS ADDRESS INCREMENT
5528 036576 016562 000020 000010 27$:     MOV     P.CS2(R5),RKCS2(R2) ;LOAD CS2
5529 036604 032765 000400 000014      BIT     #NOCHK,P.PRST(R5) ;CHECK IN NO CHECK MODE
5530 036612 001403                                     BEQ     30$                ;NO, SKIP CLEAR OF INTERRUPT ENABLE
5531 036614 042765 000100 000016      BIC     #IE,P.CS1(R5)     ;CLEAR INTERRUPT ENABLE
5532 036622 016562 000016 000000 30$:     MOV     P.CS1(R5),RKCS1(R2) ;ISSUE COMMAND
5533 036630 000137 037614                                     JMP     C.RTRN             ;RESTORE REGISTERS
5534
5535                                     .SBTTL  *SPECIAL COMMAND PROCESSING
5536
5537 036634 122765 000141 000001 C.SPEC: CMPB   #RDSTAT,P.CMND(R5) ;CHECK IF READ DRIVE STATUS

```



```

5594 037202 112765 000101 000016      MOV#SELDRV,P.CS1(R5) ;STORE COMMAND
5595 037210 032765 000400 000014      BIT#NOCHK,P.PRST(R5) ;CHECK IF NO CHECK MODE
5596 037216 001403          BEQ11$ ;NO, DO NOT RESET INTERRUPT ENABLE
5597 037220 042765 000100 000016      BIC#IE,P.CS1(R5) ;RESET INTERRUPT ENABLE
5598 037226 016562 000016 000000 11$:  MOV P.CS1(R5),RKCS1(R2) ;ISSUE COMMAND
5599 037234 000137 037614          JMP C.RTRN ;RESTORE REGISTERS
5600
5601 037240 122765 000164 000001 13$:  CMPB #RDALHD,P.CMND(R5) ;CHECK IF READ ALL HEADERS
5602 037246 001053          BNE30$ ;NO, CHECK IF CONTROLLER CLEAR
5603 037250 010537 002606          MOV R5,O.WAIT ;SET WAITING FOR COMMAND COMPLETION
5604 037254 016537 000010 002622  MOV P.BALO(R5),HDR.AD ;LOAD HEADER ADDRESS
5605 037262 132765 000020 000007  BITB #B.CFMT,P.CS1H(R5) ;CHECK IF 22 SECTOR FORMANT
5606 037270 001404          BEQ14$ ;YES, LOAD 22 IN HEADER COUNT
5607 037272 012737 000024 002624  MOV #20.,HDR.CT ;LOAD 20 IN SECTOR COUNT
5608 037300 000403          BR22$ ;GO ISSUE READ HEADER COMMAND
5609
5610 037302 012737 000026 002624 14$:  MOV #22.,HDR.CT ;LOAD 22 IN SECTOR COUNT
5611 037310 016562 000002 000020 22$:  MOV P.CYLN(R5),RKDCYL(R2) ;LOAD CYLINDER ADDRESS
5612 037316 016562 000004 000006  MOV P.SECT(R5),RKDA(R2) ;LOAD TRACK NUMBER
5613 037324 016562 000020 000010  MOV P.CS2(R5),RKCS2(R2) ;LOAD DRIVE NUMBER
5614 037332 116565 000007 000017  MOV#P.CS1H(R5),P.CS1+1(R5) ;STORE BITS 8-15 OF CS1
5615 037340 042765 165777 000016  BIC #^C<CFMT!CDT>,P.CS1(R5) ;CLEAR ALL BITS EXCEPT DRIVE TYPE
5616          ; AND FORMAT
5617 037346 112765 000125 000016  MOV#RDHEAD,P.CS1(R5) ;STORE READ HEADER COMMAND
5618 037354 032765 000400 000014  BIT#NOCHK,P.PRST(R5) ;CHECK IF NO CHECK MODE
5619 037362 001027          BNE34$ ;YES, INDICATE ILLEGAL DRIVER COMMAND
5620 037364 016562 000016 000000  MOV P.CS1(R5),RKCS1(R2) ;ISSUE READ HEADER
5621 037372 000137 037614          JMP C.RTRN ;RESTORE REGISTERS
5622
5623 037376 122765 000176 000001 30$:  CMPB #CONCLR,P.CMND(R5) ;CHECK IF CONTROLLER CLEAR
5624 037404 001012          BNE32$ ;NO, CHECK IF SUBSYSTEM CLEAR
5625 037406 004037 034710          JSR R0,I.CCLR ;CLEAR CONTROLLER
5626 037412 037614          C.RTRN ;ERROR RETURN
5627 037414 032765 000400 000014  BIT#NOCHK,P.PRST(R5) ;CHECK IF NO CHECK MODE
5628 037422 001472          BEQ40$ ;NO, INDICATE NORMAL RETURN
5629 037424 005062 000000          CLR RKCS1(R2) ;RESET INTERRUPT ENABLE
5630 037430 000467          BR40$ ;INDICATE NORMAL RETURN
5631
5632 037432 122765 000177 000001 32$:  CMPB #SUBCLR,P.CMND(R5) ;CHECK IF SUBSYSTEM CLEAR
5633 037440 001406          BEQ36$ ;YES, CLEAR SUBSYSTEM
5634 037442 052737 000100 002604 34$:  BIS #E.ILLD,E.CONT ;SET ILLEGAL DRIVER COMMAND
5635 037450 004737 035720          JSR PC,R.CONT ;REPORT ERROR
5636 037454 000457          BR C.RTRN ;RESTORE REGISTERS
5637
5638 037456 012762 000040 000010 36$:  MOV #SCLR,RKCS2(R2) ;ISSUE SUBSYSTEM CLEAR
5639 037464 016265 000000 000016  MOV RKCS1(R2),P.CS1(R5) ;STORE COMMAND AND STATUS REGISTER 1
5640 037472 032765 100000 000016  BIT#CERR,P.CS1(R5) ;CLEAR IF CONTROLLER ERROR RESET
5641 037500 001406          BEQ37$ ;NO, FINISH COMMAND
5642 037502 052737 000001 002604  BIS #BIT0,E.CONT ;SET CLEAR SUBSYSTEM DID NOT CLEAR
5643          ; CONTROLLER ERROR
5644 037510 004737 035720          JSR PC,R.CONT ;REPORT ERROR
5645 037514 000437          BR C.RTRN ;RESTORE REGISTERS
5646
5647 037516 013746 002612          37$:  MOV W.MILI,-(SP) ;LOAD 16 MILI-SECOND COUNT FOR ATTENTION
5648          ; TO DISAPPEAR
5649 037522 016265 000000 000016 38$:  MOV RKCS1(R2),P.CS1(R5) ;STORE CS1

```

```

5650 037530 032765 040000 000016 BIT #DI,P.CS1(R5) ;CHECK IF ATTENTIONS CLEARED
5651 037536 001411 BEQ 39$ ;YES, FINISH COMMAND
5652 037540 005316 DEC (SP) ;DECREMENT 16 MILLISECOND COUNT
5653 037542 001367 BNE 38$ ;CHECK DRIVE INTERRUPT AGAIN
5654 037544 005726 TST (SP)+ ;ADJUST STACK
5655 037546 052737 000040 002604 BIS #E.SCLR,E.CONT ;SET SUBSYSTEM CLEAR DID NOT CLEAR
5656 ; DRIVE ATTENTIONS
5657 037554 004737 035720 JSR PC,R.CONT ;REPORT ERROR
5658 037560 000415 BR C.RTRN ;RESTORE REGISTER
5659
5660 037562 005726 39$: TST (SP)+ ;ADJUST STACK
5661 037564 032765 000400 000014 BIT #NOCHK,P.PRST(R5) ;CHECK IF NO CHECK MODE
5662 037572 001010 BNE C.RTRN ;YES, RESTORE REGISTERS
5663 037574 112737 177777 002626 MOVB #-1,I.ISRL ;SET INTERRUPT ENABLE SET
5664 037602 012762 000100 000000 MOV #IE,RKCS1(R2) ;SET INTERRUPT ENABLE
5665 037610 004737 035706 40$: JSR PC,R.NORM ;INDICATE NORMAL TERMINATION
5666
5667 037614 012637 177776 C.RTRN: MOV (SP)+,PS ;RESTORE PSW
5668 037620 012600 MOV (SP)+,R0 ;RESTORE R0
5669 037622 012601 MOV (SP)+,R1 ;RESTORE R1
5670 037624 012602 MOV (SP)+,R2 ;RESTORE R2
5671 037626 012603 MOV (SP)+,R3 ;RESTORE R3
5672 037630 012604 MOV (SP)+,R4 ;RESTORE R4
5673 037632 012605 MOV (SP)+,R5 ;RESTORE R5
5674 037634 000207 RTS PC ;RETURN
5675
5676 .SBTTL OCTAL TO BINARY CONVERSION ROUTINE
5677
5678 ;*****
5679 ;
5680 ; THIS ROUTINE WILL CHECK A STRING OF ASCII CHARACTERS TERMINATED
5681 ; WITH A NULL <000> OR COMMA. IF THE CHARACTERS ARE LEGAL
5682 ; IT WILL GENERATE TWO BINARY WORDS PLACING THE LOW 16 BITS
5683 ; ON THE STACK AND THE HIGH 16 BITS IN LOCATION $HIOCT.
5684 ;
5685 ;*CALL
5686 ;* MOV <ADDRESS OF ASCII STRING>,-(SP)
5687 ;* JSR PC,OCTBIN
5688 ;* <ADDRESS OF ERROR RETURN>
5689 ;* RETURN
5690 ;*
5691 ;*****
5692 037636 010046 OCTBIN: MOV R0,-(SP) ;SAVE R0
5693 037640 010146 MOV R1,-(SP) ;SAVE R1
5694 037642 010246 MOV R2,-(SP) ;SAVE R2
5695 037644 016600 000010 MOV 10(SP),R0 ;GET ADDRESS OF ASCII STRING
5696 037650 005001 CLR R1 ;CLEAR DATA WORDS
5697 037652 005002 CLR R2
5698 037654 112046 2$: MOVB (R0)+,-(SP) ;PICK THIS CHARACTER
5699 037656 001423 BEQ 3$ ;IF ZERO GET OUT
5700 037660 121627 000054 CMPB (SP),#', ;CHECK IF COMMA
5701 037664 001420 BEQ 3$ ;IF COMMA GET OUT
5702 037666 122716 000060 CMPB #'0,(SP) ;MAKE SURE THIS CHARACTER IS
5703 037672 003030 BGT 4$ ; AN OCTAL DIGIT
5704 037674 122716 000067 CMPB #'7,(SP)
5705 037700 002425 BLT 4$

```

```

5706 037702 006301      ASL      R1      ; *2
5707 037704 006102      ROL      R2
5708 037706 006301      ASL      R1      ; *4
5709 037710 006102      ROL      R2
5710 037712 006301      ASL      R1      ; *8
5711 037714 006102      ROL      R2
5712 037716 042716 177770 BIC      #^C7,(SP, ;STRIP THE ASCII JUNK
5713 037722 062601      ADD      (SP)+,R1 ;ADD THIS DIGIT
5714 037724 000753      BR       2$      ;LOOP
5715 037726 005726      3$:     TST      (SP)+ ;CLEAN PARTIAL FROM STACK
5716 037730 010166 000010 MOV      R1,10(SP) ;SAVE RESULT
5717 037734 010237 037770 MOV      R2,$HIOCT
5718 037740 012602      MOV      (SP)+,R2 ;RESTORE R2
5719 037742 012601      MOV      (SP)+,R1 ;RESTORE R1
5720 037744 012600      MOV      (SP)+,R0 ;RESTORE R0
5721 037746 062716 000002 ADD      #2,(SP)  ;ADJUST RETURN
5722 037752 000207      RTS      PC      ;RETURN
5723
5724 037754 005726      4$:     TST      (SP)+ ;CLEAN UP PARTIAL FROM STACK
5725 037756 012602      MOV      (SP)+,R2 ;RESTORE R2
5726 037760 012601      MOV      (SP)+,R1 ;RESTORE R1
5727 037762 012600      MOV      (SP)+,R0 ;RESTORE R0
5728 037764 013616      MOV      @ (SP)+,(SP) ;PUT ADDRESS OF ERROR ROUTINE ON STACK
5729 037766 000207      RTS      PC      ;GO PROCESS ERROR
5730 037770 000000      $HIOCT: .WORD 0 ;HIGH ORDER BITS GO HERE
5731
5732      .SBTTL TYPE ROUTINE
5733
5734      ;*****
5735      ;ROUTINE TO TYPE ASCIZ MESSAGE. MESSAGE MUST TERMINATE WITH A 0 BYTE.
5736      ;THE ROUTINE WILL INSERT A NUMBER OF NULL CHARACTERS AFTER A LINE FEED.
5737      ;*NOTE1:      $NULL CONTAINS THE CHARACTER TO BE USED AS THE FILLER CHARACTER.
5738      ;*NOTE2:      $FILLS CONTAINS THE NUMBER OF FILLER CHARACTERS REQUIRED.
5739      ;*NOTE3:      $FILLC CONTAINS THE CHARACTER TO FILL AFTER.
5740      ;*
5741      ;*CALL:
5742      ;*1) USING A TRAP INSTRUCTION
5743      ;*      TYPE      ,MESADR      ;;MESADR IS FIRST ADDRESS OF AN ASCIZ STRING
5744      ;*OR
5745      ;*      TYPE
5746      ;*      MESADR
5747      ;*
5748 037772 105737 001157      $TYPE:  TSTB     $TPFLG      ;;IS THERE A TERMINAL?
5749 037776 100002      BPL      1$      ;;BR IF YES
5750 040000 000000      HALT
5751 040002 000407      BR       3$      ;;HALT HERE IF NO TERMINAL
5752 040004 010046      1$:     MOV      R0,-(SP)      ;;LEAVE
5753 040006 017600 000002      MOV      @2(SP),R0      ;;SAVE R0
5754 040012 112046      2$:     MOVB     (R0)+,-(SP) ;GET ADDRESS OF ASCIZ STRING
5755 040014 001005      BNE      4$      ;;PUSH CHARACTER TO BE TYPED ONTO STACK
5756 040016 005726      TST      (SP)+      ;;BR IF IT ISN'T THE TERMINATOR
5757 040020 012600      60$:   MOV      (SP)+,R0      ;;IF TERMINATOR POP IT OFF THE STACK
5758 040022 062716 000002      ADD      #2,(SP)      ;;RESTORE R0
5759 040026 000002      RTI
5760 040030 122716 000011      4$:     CMPB     #HT,(SP)      ;;ADJUST RETURN PC
5761 040034 001430      BEQ      8$      ;;RETURN
                    ;:BRANCH IF <HT>

```



```

5762 040036 122716 000200      (MPB  #CRLF,(SP)      ;;BRANCH IF NOT <CRLF>
5763 040042 001006      BNE 5$
5764 040044 005726      TST (SP)+      ;;POP <CR><LF> EQUIV
5765 040046 104401      TYPE      ;;TYPE A CR AND LF
5766 040050 001267      $CRLF
5767 040052 105037 040260      CLRB $CHARCNT      ;;CLEAR CHARACTER COUNT
5768 040056 000755      BR 2$      ;;GET NEXT CHARACTER
5769 040060 004737 040142      5$: JSR PC,$TYPEC      ;;GO TYPE THIS CHARACTER
5770 040064 123726 001156      6$: (MPB $FILLC,(SP)+      ;;IS IT TIME FOR FILLER CHARS.?
5771 040070 001350      BNE 2$      ;;IF NO GO GET NEXT CHAR.
5772 040072 013746 001154      MOV $NULL,-(SP)      ;;GET # OF FILLER CHARS. NEEDED
5773      ;;AND THE NULL CHAR.
5774 040076 105366 000001      7$: DECB 1(SP)      ;;DOES A NULL NEED TO BE TYPED?
5775 040102 002770      BLT 6$      ;;BR IF NO--GO POP THE NULL OFF OF STACK
5776 040104 004737 040142      JSR PC,$TYPEC      ;;GO TYPE A NULL
5777 040110 105337 040260      DECB $CHARCNT      ;;DO NOT COUNT AS A COUNT
5778 040114 000770      BR 7$      ;;LOOP
5779
5780      ;HORIZONTAL TAB PROCESSOR
5781
5782 040116 112716 000040      8$: MOVB #' ,(SP)      ;;REPLACE TAB WITH SPACE
5783 040122 004737 040142      9$: JSR PC,$TYPEC      ;;TYPE A SPACE
5784 040126 132737 000007 040260      BITB #7,$CHARCNT      ;;BRANCH IF NOT AT
5785 040134 001372      BNE 9$      ;;TAB STOP
5786 040136 005726      TST (SP)+      ;;POP SPACE OFF STACK
5787 040140 000724      BR 2$      ;;GET NEXT CHARACTER
5788 040142
5789 040142 105777 140776      $TYPEC: TSTB @$TKS      ;;CHAR IN KYBD BUFFER? ;MJD001
5790 040146 100022      BPL 10$      ;;BR IF NOT ;MJD001
5791 040150 017746 140772      MOV @$TKB,-(SP)      ;;GET CHAR ;MJD001
5792 040154 042716 177600      BIC #177600,(SP)      ;;STRIP EXTRANEIOUS BITS ;MJD001
5793 040160 122716 000023      (MPB #$XOFF,(SP)      ;;WAS CHAR XOFF ;MJD001
5794 040164 001012      BNE 102$      ;;BR IF NOT ;MJD001
5795 040166
5796 040166 105777 140752      101$: TSTB @$TKS      ;;WAIT FOR CHAR ;MJD001
5797 040172 100375      BPL 101$      ;MJD001
5798 040174 117716 140746      MOVB @$TKB,(SP)      ;;GET CHAR ;MJD001
5799 040200 042716 177600      BIC #177600,(SP)      ;;STRIP IT ;MJD001
5800 040204 122716 000021      (MPB #$XON,(SP)      ;;WAS IT XON? ;MJD001
5801 040210 001366      BNE 101$      ;;BR IF NOT ;MJD001
5802 040212
5803 040212 005726      102$: TST (SP)+      ;;FIX STACK ;MJD001
5804 040214
5805 040214 105777 140730      10$: TSTB @$TPS      ;;WAIT UNTIL PRINTER IS READY ;MJD001
5806 040220 100375      BPL 10$
5807 040222 116677 000002 140722      MOVB 2(SP),@$TPB      ;;LOAD CHAR TO BE TYPED INTO DATA REG.
5808 040230 122766 000015 000002      (MPB #CR,2(SP)      ;;IS CHARACTER A CARRIAGE RETURN?
5809 040236 001003      BNE 1$      ;;BRANCH IF NO
5810 040240 105037 040260      CLRB $CHARCNT      ;;YES--CLEAR CHARACTER COUNT
5811 040244 000406      BR $TYPEX      ;;EXIT
5812 040246 122766 000012 000002      1$: (MPB #LF,2(SP)      ;;IS CHARACTER A LINE FEED?
5813 040254 001402      BEQ $TYPEX      ;;BRANCH IF YES
5814 040256 105227      INCB (PC)+      ;;COUNT THE CHARACTER
5815 040260 000000      $CHARCNT: .WORD 0      ;;CHARACTER COUNT STORAGE
5816 040262 000207      $TYPEX: RTS PC
5817

```

.SBTTL DOUBLE LENGTH BINARY TO OCTAL ASCII CONVERT ROUTINE

*THIS ROUTINE WILL CONVERT A 32-BIT UNSIGNED BINARY NUMBER TO AN
*UNSIGNED OCTAL ASCII NUMBER.

*CALL
* MOV #PNTR,-(SP) ;; POINTER TO LOW WORD OF BINARY NUMBER
* JSR PC,@#\$DB20 ;; CALL THE ROUTINE
* RETURN ;; THE ADDRESS OF THE FIRST ASCII CHAR. IS ON THE STACK

\$DB20: SAVREG ;; SAVE ALL REGISTERS
MOV 2(SP),R1 ;; PICKUP THE POINTER TO LOW WORD
MOV #SOCTVL+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\$: MOV B R2,-(R5) ;; PUT CHARACTER IN DATA TABLE
MOV R0,R2 ;; GET THIS DIGIT
DEC R4 ;; COUNT THIS CHARACTER
BGT 3\$;; BR IF NOT THE LAST DIGIT
BEQ 2\$;; BR IF IT IS THE LAST DIGIT
INC R5 ;; ALL DIGITS DONE-ADJUST POINTER FOR FIRST
MOV R5,2(SP) ;; ASCII CHAR. & PUT IT ON THE STACK
RESREG ;; RESTORE ALL REGISTERS
RTS PC ;; RETURN TO USER
2\$: ASR R3 ;; POSITION THE MASK FOR THE LAST DIGIT
3\$: ROR R1 ;; POSITION THE BINARY NUMBER FOR
ROR R0 ;; THE NEXT OCTAL DIGIT
ROR R1
ROR R0
ROR R1
ROR R0
BIC R3,R2 ;; MASK OUT ALL JUNK
ADD #'0,R2 ;; MAKE THIS CHAR. ASCII
BR 1\$;; GO PUT IT IN THE DATA TABLE
\$SOCTVL: .BLKB 14. ;; RESERVE DATA TABLE

.SBTTL BINARY TO OCTAL (ASCII) AND TYPE

*THIS ROUTINE IS USED TO CHANGE A 16-BIT BINARY NUMBER TO A 6-DIGIT
*OCTAL (ASCII) NUMBER AND TYPE IT.
*\$TYPOS---ENTER HERE TO SETUP SUPPRESS ZEROS AND NUMBER OF DIGITS TO TYPE

*CALL:
* MOV NUM,-(SP) ;; NUMBER TO BE TYPED
* TYPOS ;; CALL FOR TYPEOUT
* .BYTE N ;; N=1 TO 6 FOR NUMBER OF DIGITS TO TYPE
* .BYTE M ;; M=1 OR 0
* ;; 1=TYPE LEADING ZEROS
* ;; 0=SUPPRESS LEADING ZEROS

*\$TYPON----ENTER HERE TO TYPE OUT WITH THE SAME PARAMETERS AS THE LAST
*\$TYPOS OR \$TYPOC
*CALL:

5818
5819
5820
5821
5822
5823
5824
5825
5826
5827
5828
5829 040264 104411
5830 040266 016601 000002
5831 040272 012705 040403
5832 040276 012704 000014
5833 040302 012703 177770
5834 040306 012100
5835 040310 012101
5836 040312 005002
5837 040314 110245
5838 040316 010002
5839 040320 005304
5840 040322 003007
5841 040324 001405
5842 040326 005205
5843 040330 010566 000002
5844 040334 104412
5845 040336 000207
5846 040340 006203
5847 040342 006001
5848 040344 006000
5849 040346 006001
5850 040350 006000
5851 040352 006001
5852 040354 006000
5853 040356 040302 000060
5854 040360 062702
5855 040364 000753
5856 040366 000016
5857
5858
5859
5860
5861
5862
5863
5864
5865
5866
5867
5868
5869
5870
5871
5872
5873

```

5874      ;*      MOV      NUM,-(SP)      ;;NUMBER TO BE TYPED
5875      ;*      TYPON      ;;CALL FOR TYPEOUT
5876      ;*
5877      ;*$TYPOC---ENTER HERE FOR TYPEOUT OF A 16 BIT NUMBER
5878      ;*CALL:
5879      ;*      MOV      NUM,-(SP)      ;;NUMBER TO BE TYPED
5880      ;*      TYPOC      ;;CALL FOR TYPEOUT
5881
5882 040404 017646 000000 $TYPOS: MOV @ (SP),-(SP)      ;;PICKUP THE MODE
5883 040410 116637 000001 040627 MOVB 1(SP),$OFILL      ;;LOAD ZERO FILL SWITCH
5884 040416 112637 040631 MOVB (SP)+,$OMODE+1      ;;NUMBER OF DIGITS TO TYPE
5885 040422 062716 000002 ADD #2,(SP)      ;;ADJUST RETURN ADDRESS
5886 040426 000406 BR $TYPON
5887 040430 112737 000001 040627 $TYPOC: MOVB #1,$OFILL      ;;SET THE ZERO FILL SWITCH
5888 040436 112737 000006 040631 MOVB #6,$OMODE+1      ;;SET FOR SIX(6) DIGITS
5889 040444 112737 000005 040626 $TYPON: MOVB #5,$OCNT      ;;SET THE ITERATION COUNT
5890 040452 010346 MOV R3,-(SP)      ;;SAVE R3
5891 040454 010446 MOV R4,-(SP)      ;;SAVE R4
5892 040456 010546 MOV R5,-(SP)      ;;SAVE R5
5893 040460 113704 040631 MOVB $OMODE+1,R4      ;;GET THE NUMBER OF DIGITS TO TYPE
5894 040464 005404 NEG R4
5895 040466 062704 000006 ADD #6,R4      ;;SUBTRACT IT FOR MAX. ALLOWED
5896 040472 110437 040630 MOVB R4,$OMODE      ;;SAVE IT FOR USE
5897 040476 113704 040627 MOVB $OFILL,R4      ;;GET THE ZERO FILL SWITCH
5898 040502 016605 000012 MOV 12(SP),R5      ;;PICKUP THE INPUT NUMBER
5899 040506 005003 CLR R3      ;;CLEAR THE OUTPUT WORD
5900 040510 006105 1$: ROL R5      ;;ROTATE MSB INTO 'C'
5901 040512 000404 BR 3$      ;;GO DO MSB
5902 040514 006105 2$: ROL R5      ;;FORM THIS DIGIT
5903 040516 006105 ROL R5
5904 040520 006105 ROL R5
5905 040522 010503 MOV R5,R3
5906 040524 006103 3$: ROL R3      ;;GET LSB OF THIS DIGIT
5907 040526 105337 040630 DECB $OMODE      ;;TYPE THIS DIGIT?
5908 040532 100016 BPL 7$      ;;BR IF NO
5909 040534 042703 177770 BIC #177770,R3      ;;GET RID OF JUNK
5910 040540 001002 BNE 4$      ;;TEST FOR 0
5911 040542 005704 TST R4      ;;SUPPRESS THIS 0?
5912 040544 001403 BEQ 5$      ;;BR IF YES
5913 040546 005204 4$: INC R4      ;;DON'T SUPPRESS ANYMORE 0'S
5914 040550 052703 000060 BIS #'0,R3      ;;MAKE THIS DIGIT ASCII
5915 040554 052703 000040 5$: BIS #' ,R3      ;;MAKE ASCII IF NOT ALREADY
5916 040560 110337 040624 MOVB R3,8$      ;;SAVE FOR TYPING
5917 040564 104401 040624 TYPE ,8$      ;;GO TYPE THIS DIGIT
5918 040570 105337 040626 7$: DECB $OCNT      ;;COUNT BY 1
5919 040574 003347 BGT 2$      ;;BR IF MORE TO DO
5920 040576 002402 BLT 6$      ;;BR IF DONE
5921 040600 005204 INC R4      ;;INSURE LAST DIGIT ISN'T A BLANK
5922 040602 000744 BR 2$      ;;GO DO THE LAST DIGIT
5923 040604 012605 6$: MOV (SP)+,R5      ;;RESTORE R5
5924 040606 012604 MOV (SP)+,R4      ;;RESTORE R4
5925 040610 012603 MOV (SP)+,R3      ;;RESTORE R3
5926 040612 016666 000002 000004 MOV 2(SP),4(SP)      ;;SET THE STACK FOR RETURNING
5927 040620 012616 MOV (SP)+,(SP)
5928 040622 000002 RTI      ;;RETURN
5929 040624 000 8$: .BYTE 0      ;;STORAGE FOR ASCII DIGIT

```

```

5930 040625 000
5931 040626 000
5932 040627 000
5933 040630 000000
5934
5935
5936
5937
5938
5939
5940
5941
5942
5943
5944
5945
5946
5947
5948 040632
5949 040632 104406
5950 040634 105237 001103
5951 040640 001775
5952 040642 013777 001102 140272
5953 040650 032777 002000 140262
5954 040656 001402
5955 040660 104401 001262
5956 040664 005237 001112
5957 040670 011637 001116
5958 040674 162737 000002 001116
5959 040702 117737 140210 001114
5960 040710 032777 0?0000 140222
5961 040716 001004
5962 040720 004737 025604
5963 040724 104401 001267
5964 040730
5965 040730 005777 140204
5966 040734 100002
5967 040736 000000
5968 040740 104406
5969 040742 032777 001000 140170
5970 040750 001402
5971 040752 013716 001110
5972 040756 005737 001260
5973 040762 001402
5974 040764 013716 001260
5975 040770
5976 040770 000002
5977
5978
5979
5980
5981 040772 000000
5982 040774 000000
5983 040776 000000
5984 041000 000001
5985 041001

      .BYTE 0          ;; TERMINATOR FOR TYPE ROUTINE
SOCNT: .BYTE 0      ;; OCTAL DIGIT COUNTER
$OFILL: .BYTE 0     ;; ZERO FILL SWITCH
$OMODE: .WORD 0     ;; NUMBER OF DIGITS TO TYPE
.SBTTL  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

$ERROR:
7$:  CKSWR          ;; TEST FOR CHANGE IN SOFT-SWR
     INCB          $ERFLG   ;; SET THE ERROR FLAG
     BEQ           7$      ;; DON'T LET THE FLAG GO TO ZERO
     MOV          $TSTNM,@DISPLAY ;; DISPLAY TEST NUMBER AND ERROR FLAG
     BIT          #BIT10,@SWR ;; BELL ON ERROR?
     BEQ          1$      ;; NO - SKIP
     TYPE        ,$BELL    ;; RING BELL
1$:  INC          $ERTTL   ;; 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         20$      ;; SKIP TYPEOUTS
     JSR         PC,TYPERR ;; GO TO USER ERROR ROUTINE
     TYPE        ,$CRLF

20$:
2$:  TST          @SWR     ;; HALT ON ERROR
     BPL         3$      ;; SKIP IF CONTINUE
     HALT        ;; HALT ON ERROR!
     CKSWR      ;; TEST FOR CHANGE IN SOFT-SWR
3$:  BIT          #BIT09,@SWR ;; LOOP ON ERROR SWITCH SET?
     BEQ         4$      ;; BR IF NO
     MOV          $LPERR,(SP) ;; FUDGE RETURN FOR LOOPING
4$:  TST          $ESCAPE  ;; CHECK FOR AN ESCAPE ADDRESS
     BEQ         5$      ;; BR IF NONE
     MOV          $ESCAPE,(SP) ;; FUDGE RETURN ADDRESS FOR ESCAPE
5$:
     RTI          ;; RETURN
.SBTTL  TTY INPUT ROUTINE

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

```

5986          041002          .EVEN
5987
5988          ;*TK INITIALIZE ROUTINE
5989          ;*THIS ROUTINE WILL INITIALIZE THE TTY KEYBOARD INPUT QUEUE
5990          ;*SETUP THE INTERRUPT VECTOR AND TURN ON THE KEYBOARD INTERRUPT
5991
5992          ;*CALL:
5993          ;*      JSR      PC,$TKINT
5994          ;*      RETURN
5995
5996 041002 005037 040772 $TKINT: CLR $TKCNT          ;;CLEAR COUNT OF ITEMS IN QUEUE
5997 041006 012737 041000 040774 MOV $TKQSR,$TKQIN ;;MOVE THE STARTING ADDRESS OF THE
5998 041014 013737 040774 040776 MOV $TKQIN,$TKQOUT ;;QUEUE INTO THE INPUT & OUTPUT POINTERS.
5999 041022 012737 041052 000060 MOV $TKSRV,$TKVEC ;;INITIALIZE THE KEYBOARD VECTOR
6000 041030 012737 000200 000062 MOV #200,$TKVEC+2 ;;'BR' LEVEL 4
6001 041036 005777 140104 TST @TKB          ;;CLEAR DONE FLAG
6002 041042 012777 000100 140074 MOV #100,$TKS     ;;ENABLE TTY KEYBOARD INTERRUPT
6003 041050 000207          RTS PC          ;;RETURN TO CALLER
6004
6005          ;*TK SERVICE ROUTINE
6006          ;*THIS ROUTINE WILL SERVICE THE TTY KEYBOARD INTERRUPT
6007          ;*BY READING THE CHARACTER FROM THE INPUT BUFFER AND PUTTING
6008          ;*IT IN THE QUEUE.
6009          ;*IF THE CHARACTER IS A "CONTROL-C" (^C) $TKINT IS CALLED AND
6010          ;*UPON RETURN EXIT IS MADE TO THE "CONTROL-C" RESTART ADDRESS (START)
6011
6012 041052 117746 140070 $TKSRV: MOVB @TKB,-(SP) ;;PICKUP THE CHARACTER
6013 041056 042716 177600 BIC #^C177,(SP) ;;STRIP THE JUNK
6014 041062 021627 000021 CMP (SP),#$XON ;;IS IT A RANDOM XON?
6015 041066 001002          BNE 30$ ;;BRANCH IF NO
6016 041070 005726          TST (SP)+ ;;CLEAN RANDOM XON OFF STACK
6017 041072 000002          RTI ;;RETURN
6018 041074          30$:
6019 041074 021627 000003 CMP (SP),#3 ;;IS IT A CONTROL C?
6020 041100 001007          BNE 1$ ;;BRANCH IF NO
6021 041102 104401 042262 TYPE ,SCNTLC ;;TYPE A CONTROL-C (^C)
6022 041106 004737 041002 JSR PC,$TKINT ;;INIT THE KEYBOARD
6023 041112 005726          TST (SP)+ ;;CLEAN UP STACK
6024 041114 000137 016202 JMP START ;;CONTROL C RESTART
6025 041120 021627 000007 1$: CMP (SP),#7 ;;IS IT A CONTROL G?
6026 041124 001004          BNE 2$ ;;BRANCH IF NO
6027 041126 022737 000176 001140 CMP #SWREG,SWR ;;IS SOFT-SWR SELECTED?
6028 041134 001500          BEQ 6$ ;;GO TO SWR CHANGE
6029
6030 041136          2$:
6031 041136 022737 000001 040772 CMP #1,$TKCNT ;;IS THE QUEUE FULL?
6032 041144 001004          BNE 3$ ;;BRANCH IF NO
6033 041146 104401 001262 TYPE ,SBELL ;;RING THE TTY BELL
6034 041152 005726          TST (SP)+ ;;CLEAN CHARACTER OFF OF STACK
6035 041154 000451          BR 5$ ;;EXIT
6036 041156 021627 000023 3$: CMP (SP),#23 ;;IS IT A CONTROL-S?
6037 041162 001021          BNE 32$ ;;BRANCH IF NO
6038 041164 005077 137754 CLR @TKS ;;DISABLE TTY KEYBOARD INTERRUPTS
6039 041170 005726          TST (SP)+ ;;CLEAN CHAR OFF STACK
6040 041172 105777 137746 31$: TSTB @TKS ;;WAIT FOR A CHAR
6041 041176 100375          BPL 31$ ;;LOOP UNTIL ITS THERE

```

:RAN001
 :RAN001
 :RAN001
 :RAN001
 :RAN001

```

6042 041200 117746 137742      MOVB   @TKB,-(SP)      ;;GET THE CHARACTER
6043 041204 042716 177600      BIC    #^C177,(SP)   ;;MAKE IT 7-BIT ASCII
6044 041210 022627 000021      CMP    (SP)+,#21     ;;IS IT A CONTROL-Q?
6045 041214 001366             BNE    31$           ;;BRANCH IF NO
6046 041216 012777 000100 137720  MOV    #100,@TKS     ;;REENABLE TTY KEYBOARD INTERRUPTS
6047 041224 000002             RTI                    ;;RETURN
6048 041226 005237 040772      32$:  INC    $TKCNT     ;;COUNT THIS CHARACTER
6049 041232 021627 000140      CMP    (SP),#140    ;;IS IT UPPER CASE?
6050 041236 002405             BLT    4$           ;;BRANCH IF YES
6051 041240 021627 000175      CMP    (SP),#175    ;;IS IT A SPECIAL CHAR?
6052 041244 003002             BGT    4$           ;;BRANCH IF YES
6053 041246 042716 000040      BIC    #40,(SP)     ;;MAKE IT UPPER CASE
6054 041252 112677 177516      4$:  MOVB   (SP)+,@TKQIN ;;AND PUT IT IN QUEUE
6055 041256 005237 040774      INC    $TKQIN       ;;UPDATE THE POINTER
6056 041262 023727 040774 041001  CMP    $TKQIN,$TKQEND ;;GO OFF THE END?
6057 041270 001003             BNE    5$           ;;BRANCH IF NO
6058 041272 012737 041000 040774  MOV    #$TKQSR,$TKQIN ;;RESET THE POINTER
6059 041300 000002      5$:  RTI                    ;;RETURN

```

```

6060
6061
6062
6063
6064
6065
6066
6067
6068
6069
6070
6071
6072
6073
6074
6075
6076
6077
6078
6079

```

```

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

```

```

6066 041302 022737 000176 001140 $CKSWR: CMP    #SWREG,SWR    ;;IS THE SOFT-SWR SELECTED
6067 041310 001124             BNE    15$          ;;EXIT IF NOT
6068 041312 105777 137626      TSTB   @TKS         ;;IS A CHAR WAITING?
6069 041316 100121             BPL    15$          ;;IF NOT, EXIT
6070 041320 117746 137622      MOVB   @TKB,-(SP)   ;;YES
6071 041324 042716 177600      BIC    #^C177,(SP) ;;MAKE IT 7-BIT ASCII
6072 041330 021627 000007      CMP    (SP),#7     ;;IS IT A CONTROL-G?
6073 041334 001300             BNE    2$           ;;IF NOT, PUT IT IN THE TTY QUEUE
6074
6075
6076
6077
6078
6079

```

```

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

```

```

6080 041336 123727 001134 000001 6$:  CMPB   $AUTOB,#1    ;;ARE WE RUNNING IN AUTO-MODE?
6081 041344 001674             BEQ    2$           ;;BRANCH IF YES
6082 041346 005726             TST    (SP)+       ;;CLEAR CONTROL-G OFF STACK
6083 041350 004737 041002      JSR    PC,$TKINT   ;;FLUSH THE TTY INPUT QUEUE
6084 041354 005077 137564      CLR    @TKS        ;;DISABLE TTY KEYBOARD INTERRUPTS
6085 041360 112737 000001 001135  MOVB   #1,$INTAG   ;;SET INTERRUPT MODE INDICATOR
6086
6087 041366 104401 042274      TYPE   ,$CNTLG     ;;ECHO THE CONTROL-G (^G)
6088 041372 104401 042301      $GTSWR: TYPE   $MSWR  ;;TYPE CURRENT CONTENTS
6089 041376 013746 000176      MOV    SWREG,-(SP) ;;SAVE SWREG FOR TYPEOUT
6090 041402 104402             TYPOC             ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
6091 041404 104401 042312      TYPE   ,$MNEW     ;;PROMPT FOR NEW SWR
6092 041410 005046      19$:  CLR    -(SP)      ;;CLEAR COUNTER
6093 041412 005046             CLR    -(SP)      ;;THE NEW SWR
6094 041414 105777 137524      7$:  TSTB   @TKS        ;;CHAR THERE?
6095 041420 100375             BPL    7$         ;;IF NOT TRY AGAIN
6096
6097 041422 117746 137520      MOVB   @TKB,-(SP) ;;PICK UP CHAR

```

```

6098 041426 042716 177600          BIC      #^C177,(SP)      ;;MAKE IT 7-BIT ASCII
6099
6100 041432 021627 000003          CMP      (SP),#3        ;;IS IT A CONTROL-C?
6101 041436 001015          BNE      9$             ;;BRANCH IF NOT
6102 041440 104401 042262          TYPE    ,SCNTLC        ;;YES, ECHO CONTROL-C (^C)
6103 041444 062706 000006          ADD      #6,SP         ;;CLEAN UP STACK
6104 041450 123727 001135 000001  CMPB    $INTAG,#i      ;;REENABLE TTY KEYBOARD INTERRUPTS?
6105 041456 001003          BNE      8$             ;;BRANCH IF NO
6106 041460 012777 000100 137456  MOV     #100,@$TKS     ;;ALLOW TTY KEYBOARD INTERRUPTS
6107 041466 000137 016202 8$:      JMP     START          ;;CONTROL-C RESTART
6108
6109
6110 041472 021627 000025          9$:      CMP      (SP),#25     ;;IS IT A CONTROL-U?
6111 041476 001005          BNE      10$           ;;BRANCH IF NOT
6112 041500 104401 042267          TYPE    ,SCNTLU        ;;YES, ECHO CONTROL-U (^U)
6113 041504 062706 000006          20$:    ADD      #6,SP         ;;IGNORE PREVIOUS INPUT
6114 041510 000737          BR      19$           ;;LET'S TRY IT AGAIN
6115
6116
6117 041512 021627 000015          10$:    CMP      (SP),#15     ;;IS IT A <CR>?
6118 041516 001022          BNE      16$           ;;BRANCH IF NO
6119 041520 005766 000004          TST     4(SP)          ;;YES, IS IT THE FIRST CHAR?
6120 041524 001403          BEQ     11$           ;;BRANCH IF YES
6121 041526 016677 000002 137404  MOV     2(SP),@SWR     ;;SAVE NEW SWR
6122 041534 062706 000006          11$:    ADD      #6,SP         ;;CLEAR UP STACK
6123 041540 104401 001267          14$:    TYPE    ,SCRLF        ;;ECHO <CR> AND <LF>
6124 041544 123727 001135 000001  CMPB    $INTAG,#1      ;;RE-ENABLE TTY KBD INTERRUPTS?
6125 041552 001003          BNE      15$           ;;BRANCH IF NOT
6126 041554 012777 000100 137362  MOV     #100,@$TKS     ;;RE-ENABLE TTY KBD INTERRUPTS
6127 041562 000002          15$:    RTI                    ;;RETURN
6128 041564 004737 040142          16$:    JSR     PC,$TYPEC     ;;ECHO CHAR
6129 041570 021627 000060          CMP     (SP),#60       ;;CHAR < 0?
6130 041574 002420          BLT     18$           ;;BRANCH IF YES
6131 041576 021627 000067          CMP     (SP),#67       ;;CHAR > 7?
6132 041602 003015          BGT     18$           ;;BRANCH IF YES
6133 041604 042726 000060          BIC     #60,(SP)+      ;;STRIP-OFF ASCII
6134 041610 005766 000002          TST     2(SP)          ;;IS THIS THE FIRST CHAR
6135 041614 001403          BEQ     17$           ;;BRANCH IF YES
6136 041616 006316          ASL     (SP)           ;;NO, SHIFT PRESENT
6137 041620 006316          ASL     (SP)           ;;CHAR OVER TO MAKE
6138 041622 006316          ASL     (SP)           ;;ROOM FOR NEW ONE.
6139 041624 005266 000002          17$:    INC     2(SP)        ;;KEEP COUNT OF CHAR
6140 041630 056616 177776          BIS     -2(SP),(SP)    ;;SET IN NEW CHAR
6141 041634 000667          BR      7$             ;;GET THE NEXT ONE
6142 041636 104401 001266          18$:    TYPE    ,SQUES      ;;TYPE ?<CR><LF>
6143 041642 000720          BR      20$           ;;SIMULATE CONTROL-U
6144          .DSABL  LSB
6145
6146
6147
6148
6149
6150
6151
6152
6153

```

```

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

```

```

6154
6155 041644 011646 $RDCHR: MOV (SP),-(SP) ;;PUSH DOWN THE PC AND
6156 041646 016666 000004 000002 MOV 4(SP),2(SP) ;;THE PS
6157 041654 005066 000004 CLR 4(SP) ;;GET READY FOR A CHARACTER
6158 041660 005046 CLR -(SP) ;;PUT NEW PS ON STACK
6159 041662 012746 041670 MOV #64$,-(SP) ;;PUT NEW PC ON STACK
6160 041666 000002 RTI ;;POP NEW PC AND PS
6161 041670
6162 041670 005737 040772 64$: TST $TKCNT ;;WAIT ON A CHARACTER
6163 041674 001775 1$: BEQ 1$
6164 041676 005337 040772 DEC $TKCNT ;;DECREMENT THE COUNTER
6165 041702 117766 177070 000004 MOVB @STKQOUT,4(SP) ;;GET ONE CHARACTER
6166 041710 005237 040776 INC $TKQOUT ;;UPDATE THE POINTER
6167 041714 023727 040776 041001 CMP $TKQOUT,$$TKQEND ;;DID IT GO OFF OF THE END?
6168 041722 001003 BNE 2$ ;;BRANCH IF NO
6169 041724 012737 041000 040776 MOV $$TKQSRRT,$TKQOUT ;;RESET THE POINTER
6170 041732 000002 2$: RTI ;;RETURN
6171 *****
6172 *THIS ROUTINE WILL INPUT A STRING FROM THE TTY
6173 *CALL:
6174 * RDLIN ;;INPUT A STRING FROM THE TTY
6175 * RETURN HERE ;;ADDRESS OF FIRST CHARACTER WILL BE ON THE STACK
6176 * ;;TERMINATOR WILL BE A BYTE OF ALL 0'S
6177
6178 041734 010346 $RDLIN: MOV R3,-(SP) ;;SAVE R3
6179 041736 005046 CLR -(SP) ;;CLEAR THE RUBOUT KEY
6180 041740 012703 042170 1$: MOV $$TTYIN,R3 ;;GET ADDRESS
6181 041744 022703 042262 2$: CMP $$TTYIN+72,R3 ;;BUFFER FULL?
6182 041750 101456 BLOS 4$ ;;BR IF YES
6183 041752 104407 RDCHR ;;GO READ ONE CHARACTER FROM THE TTY
6184 041754 112613 MOVB (SP)+,(R3) ;;GET CHARACTER
6185 041756 122713 000177 10$: CMPB #177,(R3) ;;IS IT A RUBOUT
6186 041762 001022 BNE 5$ ;;BR IF NO
6187 041764 005716 TST (SP) ;;IS THIS THE FIRST RUBOUT?
6188 041766 001007 BNE 6$ ;;BR IF NO
6189 041770 112737 000134 042166 MOVB #' \,9$ ;;TYPE A BACK SLASH
6190 041776 104401 042166 TYPE ,9$
6191 042002 012716 177777 MOV #-1,(SP) ;;SET THE RUBOUT KEY
6192 042006 005303 6$: DEC R3 ;;BACKUP BY ONE
6193 042010 020327 042170 CMP R3,$$TTYIN ;;STACK EMPTY?
6194 042014 103434 BLO 4$ ;;BR IF YES
6195 042016 111337 042166 MOVB (R3),9$ ;;SETUP TO TYPEOUT THE DELETED CHAR.
6196 042022 104401 042166 TYPE ,9$ ;;GO TYPE
6197 042026 000746 BR 2$ ;;GO READ ANOTHER CHAR.
6198 042030 005716 5$: TST (SP) ;;RUBOUT KEY SET?
6199 042032 001406 BEQ 7$ ;;BR IF NO
6200 042034 112737 000134 042166 MOVB #' \,9$ ;;TYPE A BACK SLASH
6201 042042 104401 042166 TYPE ,9$
6202 042046 005016 CLR (SP) ;;CLEAR THE RUBOUT KEY
6203 042050 122713 000025 7$: CMPB #25,(R3) ;;IS CHARACTER A CTRL U?
6204 042054 001003 BNE 8$ ;;BR IF NO
6205 042056 104401 042267 TYPE ,9$CNTLU ;;TYPE A CONTROL 'U'
6206 042062 000726 BR 1$ ;;GO START OVER
6207 042064 122713 000022 8$: CMPB #22,(R3) ;;IS CHARACTER A '^R'?
6208 042070 001011 BNE 3$ ;;BRANCH IF NO
6209 042072 105013 CLRB (R3) ;;CLEAR THE CHARACTER
  
```



```

6210 042074 104401 001267          TYPE      , $CRLF          ;; TYPE A 'CR' & 'LF'
6211 042100 104401 042170          TYPE      , $TTYIN        ;; TYPE THE INPUT STRING
6212 042104 000717                    BR          2$           ;; GO PICKUP ANOTHER CHACTER
6213 042106 104401 001266          4$: TYPE      , $QUES        ;; TYPE A '?'
6214 042112 000712                    BR          1$           ;; CLEAR THE BUFFER AND LOOP
6215 042114 111337 042166          3$: MOV      (R3), 9$      ;; ECHO THE CHARACTER
6216 042120 104401 042166          TYPE      , 9$
6217 042124 122723 000015          CMP      #15, (R3)+      ;; CHECK FOR RETURN
6218 042130 001305                    BNE         2$           ;; LOOP IF NOT RETURN
6219 042132 105063 177777          CLR      -1(R3)         ;; CLEAR RETURN (THE 15)
6220 042136 104401 001270          TYPE      , $LF          ;; TYPE A LINE FEED
6221 042142 005726                    TST        (SP)+         ;; CLEAN RUBOUT KEY FROM THE STACK
6222 042144 012603                    MOV        (SP)+, R3      ;; RESTORE R3
6223 042146 011646                    MOV        (SP), -(SP)    ;; ADJUST THE STACK AND PUT ADDRESS OF THE
6224 042150 016666 000004 000002    MOV        4(SP), 2(SP)   ;; FIRST ASCII CHARACTER ON IT
6225 042156 012766 042170 000004    MOV        #$TTYIN, 4(SP)
6226 042164 000002                    RTI                     ;; RETURN
6227 042166          000          9$: .BYTE      0          ;; STORAGE FOR ASCII CHAR. TO TYPE
6228 042167          000                    .BYTE      0          ;; TERMINATOR
6229 042170 000072                    $TTYIN: .BLKB      72    ;; RESERVE 72 BYTES FOR TTY INPUT
6230 042262 041536 005015          000          $CNTLC: .ASCIZ / ^C / <15><12> ;; CONTROL 'C'
6231 042267          136 006525 000012    $CNTLU: .ASCIZ / ^U / <15><12> ;; CONTROL 'U'
6232 042274 043536 005015          000          $CNTLG: .ASCIZ / ^G / <15><12> ;; CONTROL 'G'
6233 042301          015 051412 051127    $MSWR: .ASCIZ <15><12> / SWR = /
6234 042306 036440 000040
6235 042312 020040 042516 020127    $MNEW: .ASCIZ / NEW = /
6236 042320 020075          000
6237          042324
6238          .EVEN
6239          .SBTTL POWER DOWN AND UP ROUTINES
6240          ;; *****
6241          :POWER DOWN ROUTINE
6242 042324 012737 042464 000024    $PWRDN: MOV      # $ILLUP, @PWRVEC ;; SET FOR FAST UP
6243 042332 012737 000340 000026    MOV      #340, @PWRVEC+2 ;; PRIO:7
6244 042340 010046                    MOV      R0, -(SP)       ;; PUSH R0 ON STACK
6245 042342 010146                    MOV      R1, -(SP)       ;; PUSH R1 ON STACK
6246 042344 010246                    MOV      R2, -(SP)       ;; PUSH R2 ON STACK
6247 042346 010346                    MOV      R3, -(SP)       ;; PUSH R3 ON STACK
6248 042350 010446                    MOV      R4, -(SP)       ;; PUSH R4 ON STACK
6249 042352 010546                    MOV      R5, -(SP)       ;; PUSH R5 ON STACK
6250 042354 017746 136560          MOV      @SWR, -(SP)     ;; PUSH @SWR ON STACK
6251 042360 010637 042470          MOV      SP, $SAVR6      ;; SAVE SP
6252 042364 012737 042376 000024    MOV      # $PWRUP, @PWRVEC ;; SET UP VECTOR
6253 042372 000000                    HALT
6254 042374 000776                    BR          -2           ;; HANG UP
6255
6256          ;; *****
6257          :POWER UP ROUTINE
6258 042376 012737 042464 000024    $PWRUP: MOV      # $ILLUP, @PWRVEC ;; SET FOR FAST DOWN
6259 042404 013706 042470          MOV      $SAVR6, SP     ;; GET SP
6260 042410 005037 042470          CLR      $SAVR6        ;; WAIT LOOP FOR THE TTY
6261 042414 005237 042470          1$: INC      $SAVR6      ;; WAIT FOR THE INC
6262 042420 001375                    BNE         1$           ;; OF WORD
6263 042422 012677 136512          MOV      (SP)+, @SWR    ;; POP STACK INTO @SWR
6264 042426 012605                    MOV      (SP)+, R5      ;; POP STACK INTO R5
6265 042430 012604                    MOV      (SP)+, R4      ;; POP STACK INTO R4

```

```

6266 042432 012603      MOV      (SP)+,R3      ;;POP STACK INTO R3
6267 042434 012602      MOV      (SP)+,R2      ;;POP STACK INTO R2
6268 042436 012601      MOV      (SP)+,R1      ;;POP STACK INTO R1
6269 042440 012600      MOV      (SP)+,R0      ;;POP STACK INTO R0
6270 042442 012737 042324 000024      MOV      #$PWRDN,@#PWRVEC ;;SET UP THE POWER DOWN VECTOR
6271 042450 012737 000340 000026      MOV      #340,@#PWRVEC+2 ;;PRIO:7
6272 042456 104401      TYPE      ;;REPORT THE POWER FAILURE
6273 042460 042472      $PWRMG: .WORD $POWER ;;POWER FAIL MESSAGE POINTER
6274 042462 000002      RTI
6275 042464 000000      $ILLUP: HALT      ;;THE POWER UP SEQUENCE WAS STARTED
6276 042466 000776      BR      .-2      ;; BEFORE THE POWER DOWN WAS COMPLETE
6277 042470 000000      $SAVR6: 0      ;;PUT THE SP HERE
6278 042472 005015 047520 042527      $POWER: .ASCIZ <15><12>'POWER''
6279 042500 000122
6280      .EVEN
6281      .SBTTL ROUTINE TO SIZE MEMORY
6282
6283      ;*****
6284      ;*CALL:
6285      ;* JSR PC,$SIZE
6286      ;* RETURN
6287      ;*$LSTAD WILL CONTAIN THE LAST AVAILABLE MEMORY LOCATION
6288
6289 042502 010046      $SIZE: MOV      R0,-(SP)      ;;SAVE R0 ON THE STACK
6290 042504 010146      MOV      R1,-(SP)      ;;SAVE R1 ON THE STACK
6291 042506 013746 000114      MOV      @#114,-(SP)      ;;SAVE MEMORY ERPOR VECTOR PS & PC
6292 042512 013746 000116      MOV      @#116,-(SP)
6293 042516 012737 000116 000114      MOV      #116,@#114      ;;IGNORE PARITY ERRORS WHILE SIZING
6294 042524 012737 000002 000116      MOV      #RTI,@#116
6295 042532 013746 000004      MOV      @#ERRVEC,-(SP) ;;SAVE PRESENT ERROR VECTOR PS & PC
6296 042536 013746 000006      MOV      @#ERRVEC+2,-(SP)
6297 042542 010600      MOV      SP,R0      ;;SAVE THE STACK POINTER
6298      ;;SET THE ERRVEC PS TO THE PRESENT PS
6299 042544 104400      TRAP      ;;PUSH OLD PSW AND PC ON STACK
6300 042546 012637 000006      MOV      (SP)+,@#ERRVEC+2 ;;SAVE THE PSW IN @#ERRVEC+2
6301 042552 012737 042572 000004      MOV      #2$,@#ERRVEC      ;;SET FOR TIMEOUT
6302 042560 012701 020000      MOV      #20000,R1      ;;FIRST ADDRESS
6303 042564 005711      1$: TST      (R1)      ;;TEST THIS ADDRESS
6304 042566 005721      TST      (R1)+      ;;STEP TO NEXT ADDRESS
6305 042570 000775      BR      1$      ;;TRY ANOTHER
6306 042572 162701 000002      2$: SUB      #2,R1      ;;DROP BACK
6307 042576 010006      MOV      R0,SP      ;;RESTORE THE STACK
6308 042600 012637 000006      MOV      (SP)+,@#ERRVEC+2 ;;RESTORE ERROR VECTOR
6309 042604 012637 000004      MOV      (SP)+,@#ERRVEC
6310 042610 012637 000116      MOV      (SP)+,@#116      ;;RESTORE MEMORY ERROR VECTOR
6311 042614 012637 000114      MOV      (SP)+,@#114
6312 042620 010137 042632      MOV      R1,$LSTAD      ;;LAST ADDRESS
6313 042624 012601      MOV      (SP)+,R1      ;;RESTORE R1
6314 042626 012600      MOV      (SP)+,R0      ;;RESTORE R0
6315 042630 000207      RTS      PC
6316 042632 000000      $LSTAD: .WORD 0      ;;CONTAINS THE LAST ADDRESS
6317      .SBTTL SAVE AND RESTORE R0-R5 ROUTINES
6318
6319      ;*****
6320      ;*SAVE R0-R5
6321      ;*CALL:
  
```

6322
 6323
 6324
 6325
 6326
 6327
 6328
 6329
 6330
 6331
 6332
 6333
 6334 042634
 6335 042634 010046
 6336 042636 010146
 6337 042640 010246
 6338 042642 010346
 6339 042644 010446
 6340 042646 010546
 6341 042650 016646 000022
 6342 042654 016646 000022
 6343 042660 016646 000022
 6344 042664 016646 000022
 6345 042670 000002
 6346
 6347
 6348
 6349
 6350 042672
 6351 042672 012666 000022
 6352 042676 012666 000022
 6353 042702 012666 000022
 6354 042706 012666 000022
 6355 042712 012605
 6356 042714 012604
 6357 042716 012603
 6358 042720 012602
 6359 042722 012601
 6360 042724 012600
 6361 042726 000002
 6362
 6363
 6364
 6365
 6366
 6367
 6368
 6369
 6370 042730 010046
 6371 042732 016600 000002
 6372 042736 005740
 6373 042740 111000
 6374 042742 006300
 6375 042744 016000 042764
 6376 042750 000200
 6377

```

;* 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 R0,-(SP)      ;;PUSH R0 ON STACK
MOV R1,-(SP)      ;;PUSH R1 ON STACK
MOV R2,-(SP)      ;;PUSH R2 ON STACK
MOV R3,-(SP)      ;;PUSH R3 ON STACK
MOV R4,-(SP)      ;;PUSH R4 ON STACK
MOV R5,-(SP)      ;;PUSH R5 ON STACK
MOV 22(SP),-(SP)  ;;SAVE PS OF MAIN FLOW
MOV 22(SP),-(SP)  ;;SAVE PC OF MAIN FLOW
MOV 22(SP),-(SP)  ;;SAVE PS OF CALL
MOV 22(SP),-(SP)  ;;SAVE PC OF CALL
RTI

;*RESTORE R0-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 R0,-(SP)      ;;SAVE R0
MOV 2(SP),R0      ;;GET TRAP ADDRESS
TST -(R0)         ;;BACKUP BY 2
MOVB (R0),R0      ;;GET RIGHT BYTE OF TRAP
ASL R0            ;;POSITION FOR INDEXING
MOV $TRPAD(R0),R0 ;;INDEX TO TABLE
RTS R0           ;;GO TO ROUTINE
  
```

```

6378
6379          ;;THIS IS USE TO HANDLE THE "GETPRI" MACRO
6380
6381 042752 011646          $TRAP2: MOV      (SP),-(SP)      ;;MOVE THE PC DOWN
6382 042754 016666 000004 000002  MOV      4(SP),2(SP)      ;;MOVE THE PSW DOWN
6383 042762 000002          RTI                          ;;RESTORE THE PSW
6384
6385          .SBTTL TRAP TABLE
6386
6387          ;*THIS TABLE CONTAINS THE STARTING ADDRESSES OF THE ROUTINES CALLED
6388          ;*BY THE "TRAP" INSTRUCTION.
6389
6390          :          ROUTINE
6391          :          -----
6392 042764 042752          $TRPAD: .WORD  $TRAP2
6393 042766 037772          $TYPE  ;;CALL=TYPE      TRAP+1(104401)  TTY TYPEOUT ROUTINE
6394 042770 040430          $TYPOC  ;;CALL=TYPOC     TRAP+2(104402)  TYPE OCTAL NUMBER (WITH LEADING ZEROS)
6395 042772 040404          $TYPOS  ;;CALL=TYPOS     TRAP+3(104403)  TYPE OCTAL NUMBER (NO LEADING ZEROS)
6396 042774 040444          $TYPON  ;;CALL=TYPON     TRAP+4(104404)  TYPE OCTAL NUMBER (AS PER LAST CALL)
6397
6398 042776 041372          $GTSWR  ;;CALL=GTSWR     TRAP+5(104405)  GET SOFT-SWR SETTING
6399
6400 043000 041302          $CKSWR  ;;CALL=CKSWR     TRAP+6(104406)  TEST FOR CHANGE IN SOFT-SWR
6401 043002 041644          $RDCHR  ;;CALL=RDCHR     TRAP+7(104407)  TTY TYPEIN CHARACTER ROUTINE
6402 043004 041734          $RDLIN  ;;CALL=RDLIN     TRAP+10(104410) TTY TYPEIN STRING ROUTINE
6403 043006 042634          $SAVREG ;;CALL=SAVREG     TRAP+11(104411) SAVE R0-R5 ROUTINE
6404 043010 042672          $RESREG ;;CALL=RESREG     TRAP+12(104412) RESTORE R0-R5 ROUTINE
6405
6406 043012 025052 020040 047125 EM1:  .ASCIZ  /** UNIBUS PARITY ERROR/
6407 043020 041111 051525 050040
6408 043026 051101 052111 020131
6409 043034 051105 047522 000122
6410 043042 025052 020040 047516 EM2:  .ASCIZ  /** NON-EXISTANT MEMORY ERROR/
6411 043050 026516 054105 051511
6412 043056 040524 052116 046440
6413 043064 046505 051117 020131
6414 043072 051105 047522 000122
6415 043100 025052 020040 047516 EM3:  .ASCIZ  /** NON-EXISTANT DRIVE ERROR/
6416 043106 026516 054105 051511
6417 043114 040524 052116 042040
6418 043122 044522 042526 042440
6419 043130 051122 051117 000
6420 043135 052 020052 052440 EM4:  .ASCIZ  /** UNIT FIELD ERROR/
6421 043142 044516 020124 044506
6422 043150 046105 020104 051105
6423 043156 047522 000122
6424 043162 025052 020040 052523 EM5:  .ASCIZ  /** SUBSYSTEM TIMEOUT/
6425 043170 051502 051531 042524
6426 043176 020115 044524 042515
6427 043204 052517 000124
6428 043210 025052 020040 051104 EM6:  .ASCIZ  /** DRIVE BUS PARITY ERROR/
6429 043216 053111 020105 052502
6430 043224 020123 040520 044522
6431 043232 054524 042440 051122
6432 043240 051117 000
6433 043243 052 020052 042040 EM7:  .ASCIZ  /** DRIVE DETECTED PARITY ERROR/
  
```

6434	043250	044522	042526	042040				
6435	043256	052105	041505	042524				
6436	043264	020104	040520	044522				
6437	043272	054524	042440	051122				
6438	043300	051117	000					
6439	043303	052	020052	040440	EM10:	.ASCIZ	/**	AC LOW/
6440	043310	020103	047514	000127				
6441	043316	025052	020040	050123	EM11:	.ASCIZ	/**	SPEED LOSS/
6442	043324	042505	020104	047514				
6443	043332	051523	000					
6444	043335	052	020052	044440	EM12:	.ASCIZ	/**	ILLEGAL FUNCTION ERROR/
6445	043342	046114	043505	046101				
6446	043350	043040	047125	052103				
6447	043356	047511	020116	051105				
6448	043364	047522	000122					
6449	043370	025052	020040	051120	EM13:	.ASCIZ	/**	PROGRAMMING ERROR/
6450	043376	043517	040522	046515				
6451	043404	047111	020107	051105				
6452	043412	047522	000122					
6453	043416	025052	020040	047516	EM14:	.ASCIZ	/**	NON-EXISTANT FUNCTION ERROR/
6454	043424	026516	054105	051511				
6455	043432	040524	052116	043040				
6456	043440	047125	052103	047511				
6457	043446	020116	051105	047522				
6458	043454	000122						
6459	043456	025052	020040	051104	EM15:	.ASCIZ	/**	DRIVE TYPE ERROR/
6460	043464	053111	020105	054524				
6461	043472	042520	042440	051122				
6462	043500	051117	000					
6463	043503	052	020052	043040	EM16:	.ASCIZ	/**	FORMAT ERROR/
6464	043510	051117	040515	020124				
6465	043516	051105	047522	000122				
6466	043524	025052	020040	051127	EM17:	.ASCIZ	/**	WRITE LOCK ERROR/
6467	043532	052111	020105	047514				
6468	043540	045503	042440	051122				
6469	043546	051117	000					
6470	043551	052	020052	042040	EM20:	.ASCIZ	/**	DRIVE UNSAFE ERROR/
6471	043556	044522	042526	052440				
6472	043564	051516	043101	020105				
6473	043572	051105	047522	000122				
6474	043600	025052	020040	042523	EM21:	.ASCIZ	/**	SEEK INCOMPLETE ERROR/
6475	043606	045505	044440	041516				
6476	043614	046517	046120	052105				
6477	043622	020105	051105	047522				
6478	043630	000122						
6479	043632	025052	020040	054503	EM22:	.ASCIZ	/**	CYLINDER OVERFLOW ERROR/
6480	043640	044514	042116	051105				
6481	043646	047440	042526	043122				
6482	043654	047514	020127	051105				
6483	043662	047522	000122					
6484	043666	025052	020040	046111	EM23:	.ASCIZ	/**	ILLEGAL CYLINDER ADDRESS ERROR/
6485	043674	042514	040507	020114				
6486	043702	054503	044514	042116				
6487	043710	051105	040440	042104				
6488	043716	042522	051523	042440				
6489	043724	051122	051117	000				

6490	043731	052	020052	042040	EM24:	.ASCIZ	/**	DRIVE OFF TRACK/
6491	043736	044522	042526	047440				
6492	043744	043106	052040	040522				
6493	043752	045503	000					
6494	043755	052	020052	042040	EM25:	.ASCIZ	/**	DRIVE TIMING ERROR/
6495	043762	044522	042526	052040				
6496	043770	046511	047111	020107				
6497	043776	051105	047522	000122				
6498	044004	025052	020040	040504	EM26:	.ASCIZ	/**	DATA LATE ERROR/
6499	044012	040524	046040	052101				
6500	044020	020105	051105	047522				
6501	044026	000122						
6502	044030	025052	020040	047503	EM27:	.ASCIZ	/**	CONTROLLER TIMEOUT ERROR/
6503	044036	052116	047522	046114				
6504	044044	051105	052040	046511				
6505	044052	047505	052125	042440				
6506	044060	051122	051117	000				
6507	044065	052	020052	047440	EM30:	.ASCIZ	/**	OPERATION INCOMPLETE ERROR/
6508	044072	042520	040522	044524				
6509	044100	047117	044440	041516				
6510	044106	046517	046120	052105				
6511	044114	020105	051105	047522				
6512	044122	000122						
6513	044124	025052	020040	042510	EM31:	.ASCIZ	/**	HEADER VRC ERROR/
6514	044132	042101	051105	053040				
6515	044140	041522	042440	051122				
6516	044146	051117	000					
6517	044151	052	020052	042040	EM32:	.ASCIZ	/**	DATA CHECK ERROR/
6518	044156	052101	020101	044103				
6519	044164	041505	020113	051105				
6520	044172	047522	000122					
6521	044176	025052	020040	051127	EM33:	.ASCIZ	/**	WRITE CHECK ERROR/
6522	044204	052111	020105	044103				
6523	044212	041505	020113	051105				
6524	044220	047522	000122					
6525	044224	025052	020040	040504	EM34:	.ASCIZ	/**	DATA MISCOMPARE/
6526	044232	040524	046440	051511				
6527	044240	047503	050115	051101				
6528	044246	000105						
6529	044250	025052	020040	047516	EM35:	.ASCIZ	/**	NO DRIVE RESPONSE-UFE & NED/
6530	044256	042040	044522	042526				
6531	044264	051040	051505	047520				
6532	044272	051516	026505	043125				
6533	044300	020105	020046	042516				
6534	044306	000104						
6535	044310	025052	020040	051104	EM36:	.ASCIZ	/**	DRIVE ERROR WILL NOT CLEAR/
6536	044316	053111	020105	051105				
6537	044324	047522	020122	044527				
6538	044332	046114	047040	052117				
6539	044340	041440	042514	051101				
6540	044346	000						
6541	044347	052	020052	042040	EM37:	.ASCIZ	/**	DRIVE STATUS CHANGE WILL NOT CLEAR/
6542	044354	044522	042526	051440				
6543	044362	040524	052524	020123				
6544	044370	044103	047101	042507				
6545	044376	053440	046111	020114				

6546	044404	047516	020124	046103	
6547	044412	040505	000122		
6548	044416	025052	020040	052101	EM40: .ASCIZ /** ATTENTION BUT NO STATUS CHANGE OR FAULT/
6549	044424	042524	052116	047511	
6550	044432	020116	052502	020124	
6551	044440	047516	051440	040524	
6552	044446	052524	020123	044103	
6553	044454	047101	042507	047440	
6554	044462	020122	040506	046125	
6555	044470	000124			
6556	044472	025052	020040	052101	EM41: .ASCIZ /** ATTENTION BUT DRIVE NOT AVAILABLE/
6557	044500	042524	052116	047511	
6558	044506	020116	052502	020124	
6559	044514	051104	053111	020105	
6560	044522	047516	020124	053101	
6561	044530	044501	040514	046102	
6562	044536	000105			
6563	044540	025052	020040	052101	EM42: .ASCIZ /** ATTENTION WHEN NOT EXPECTED/
6564	044546	042524	052116	047511	
6565	044554	020116	044127	047105	
6566	044562	047040	052117	042440	
6567	044570	050130	041505	042524	
6568	044576	000104			
6569	044600	025052	020040	051105	EM43: .ASCIZ /** ERROR WHILE GATHERING DRIVE STATUS/
6570	044606	047522	020122	044127	
6571	044614	046111	020105	040507	
6572	044622	044124	051105	047111	
6573	044630	020107	051104	053111	
6574	044636	020105	052123	052101	
6575	044644	051525	000		
6576	044647	052	020052	046440	EM52: .ASCIZ /** MULTIPLE DRIVE SELECT/
6577	044654	046125	044524	046120	
6578	044662	020105	051104	053111	
6579	044670	020105	042523	042514	
6580	044676	052103	000		
6581	044701	052	020052	044040	EM53: .ASCIZ /** HEADER COMPARE ERROR/
6582	044706	040505	042504	020122	
6583	044714	047503	050115	051101	
6584	044722	020105	051105	047522	
6585	044730	000122			
6586	044732	025052	020040	052523	EM56: .ASCIZ /** SUBSYSTEM TIMEOUT/
6587	044740	051502	051531	042524	
6588	044746	020115	044524	042515	
6589	044754	052517	000124		
6590	044760	025052	020040	051105	EM60: .ASCIZ /** ERROR IN RECALIBRATE FOR RECOVERY/
6591	044766	047522	020122	047111	
6592	044774	051040	041505	046101	
6593	045002	041111	040522	042524	
6594	045010	043040	051117	051040	
6595	045016	041505	053117	051105	
6596	045024	000131			
6597	045026	025052	020040	051120	EM61: .ASCIZ /** PROGRAM ABORTING FATAL ERROR IN RETRY/
6598	045034	043517	040522	020115	
6599	045042	041101	051117	044524	
6600	045050	043516	043040	052101	
6601	045056	046101	042440	051122	

6602	045064	051117	044440	020116		
6603	045072	042522	051124	000131		
6604	045100	025052	020040	042510	EM62:	.ASCIZ /** HEADER MISCOMPARE/
6605	045106	042101	051105	046440		
6606	045114	051511	047503	050115		
6607	045122	051101	000105			
6608	045126	025052	020040	046103	EM63:	.ASCIZ /** CLEAR CONTROLLER DID NOT CLEAR ERROR/
6609	045134	040505	020122	047503		
6610	045142	052116	047522	046114		
6611	045150	051105	042040	042111		
6612	045156	047040	052117	041440		
6613	045164	042514	051101	042440		
6614	045172	051122	051117	000		
6615	045177	052	020052	047040	EM64:	.ASCIZ /** NO ATTENTION IN ATTENTION SUMMARY REGISTER/
6616	045204	020117	052101	042524		
6617	045212	052116	047511	020116		
6618	045220	047111	040440	052124		
6619	045226	047105	044524	047117		
6620	045234	051440	046525	040515		
6621	045242	054522	051040	043505		
6622	045250	051511	042524	000122		
6623	045256	025052	020040	047125	EM65:	.ASCIZ /** UNSOLICITED ATTENTION/
6624	045264	047523	044514	044503		
6625	045272	042524	020104	052101		
6626	045300	042524	052116	047511		
6627	045306	000116				
6628	045310	025052	020040	047125	EM66:	.ASCIZ /** UNEXPECTED DATA TYPE ERROR/
6629	045316	054105	042520	052103		
6630	045324	042105	042040	052101		
6631	045332	020101	054524	042520		
6632	045340	042440	051122	051117		
6633	045346	000				
6634	045347	052	020052	040440	EM67:	.ASCIZ /** ATTENTION DID NOT RESET WITH CLEAR/
6635	045354	052124	047105	044524		
6636	045362	047117	042040	042111		
6637	045370	047040	052117	051040		
6638	045376	051505	052105	053440		
6639	045404	052111	020110	046103		
6640	045412	040505	000122			
6641	045416	025052	020040	052523	EM70:	.ASCIZ /** SUBSYSTEM CLEAR DID NOT CLEAR DRIVE ATTENTION/
6642	045424	051502	051531	042524		
6643	045432	020115	046103	040505		
6644	045440	020122	044504	020104		
6645	045446	047516	020124	046103		
6646	045454	040505	020122	051104		
6647	045462	053111	020105	052101		
6648	045470	042524	052116	047511		
6649	045476	000116				
6650	045500	025052	020040	040504	EM71:	.ASCIZ /** DATA LATE WHEN UNLOADING HEADER/
6651	045506	040524	046040	052101		
6652	045514	020105	044127	047105		
6653	045522	052440	046116	040517		
6654	045530	044504	043516	044040		
6655	045536	040505	042504	000122		
6656	045544	025052	020040	047503	EM72:	.ASCIZ /** CONTROLLER ERROR WHEN DRIVER SERVICING/
6657	045552	052116	047522	046114		

6658	045560	051105	042440	051122	
6659	045566	051117	053440	042510	
6660	045574	020116	051104	053111	
6661	045602	051105	051440	051105	
6662	045610	044526	044503	043516	
6663	045616	000			
6664	045617	052	020052	042040	EM73: .ASCIZ /** DRIVE DETECTED PARITY ERROR/
6665	045624	044522	042526	042040	
6666	045632	052105	041505	042524	
6667	045640	020104	040520	044522	
6668	045646	054524	042440	051122	
6669	045654	051117	000		
6670	045657	052	020052	052440	EM74: .ASCIZ /** UNDEFINED ERROR/
6671	045664	042116	043105	047111	
6672	045672	042105	042440	051122	
6673	045700	051117	000		
6674	045703	052	020052	046440	EM75: .ASCIZ /** MARKING THIS SECTOR BAD/
6675	045710	051101	044513	043516	
6676	045716	052040	044510	020123	
6677	045724	042523	052103	051117	
6678	045732	041040	042101	000	
6679	045737	052	020052	041040	EM76: .ASCIZ /** BAD DATA VERIFICATION WITH READ. ECC OF LAST RETRY IS:/
6680	045744	042101	042040	052101	
6681	045752	020101	042526	044522	
6682	045760	044506	040503	044524	
6683	045766	047117	053440	052111	
6684	045774	020110	042522	042101	
6685	046002	020056	041505	020103	
6686	046010	043117	046040	051501	
6687	046016	020124	042522	051124	
6688	046024	020131	051511	000072	
6689	046032	025052	020040	042522	EM77: .ASCIZ /** RETRY SUCCESSFUL/
6690	046040	051124	020131	052523	
6691	046046	041503	051505	043123	
6692	046054	046125	000		
6693	046057	052	020052	051040	EM100: .ASCIZ /** RETRY UNSUCCESSFUL/
6694	046064	052105	054522	052440	
6695	046072	051516	041525	042503	
6696	046100	051523	052506	000114	
6697	046106	025052	020040	040503	EM101: .ASCIZ /** CANNOT FIND A VALID HEADER IN TRACK JUST READ/
6698	046114	047116	052117	043040	
6699	046122	047111	020104	020101	
6700	046130	040526	044514	020104	
6701	046136	042510	042101	051105	
6702	046144	044440	020116	051124	
6703	046152	041501	020113	052512	
6704	046160	052123	051040	040505	
6705	046166	000104			
6706	046170	025052	020040	040502	EM102: .ASCIZ /** BAD SECTOR ERROR ON SECTOR NOT LISTED BAD/
6707	046176	020104	042523	052103	
6708	046204	051117	042440	051122	
6709	046212	051117	047440	020116	
6710	046220	042523	052103	051117	
6711	046226	047040	052117	046040	
6712	046234	051511	042524	020104	
6713	046242	040502	000104		

6714	046246	025052	053440	051117	EM103: .ASCIZ /** WORD COUNT INCORRECT TO CONTINUE/
6715	046254	020104	047503	047125	
6716	046262	020124	047111	047503	
6717	046270	051122	041505	020124	
6718	046276	047524	041440	047117	
6719	046304	044524	052516	000105	
6720	046312	025052	040503	052125	EM104: .ASCIZ /**CAUTION** DRIVE STATUS REPORTED MAY NOT BE CORRECT/
6721	046320	047511	025116	020052	
6722	046326	051104	053111	020105	
6723	046334	052123	052101	051525	
6724	046342	051040	050105	051117	
6725	046350	042524	020104	040515	
6726	046356	020131	047516	020124	
6727	046364	042502	041440	051117	
6728	046372	042522	052103	000	
6729	046377	105	051122	051117	EM105: .ASCII /ERROR IN SECTOR ALREADY MARKED BAD/<15><12>
6730	046404	044440	020116	042523	
6731	046412	052103	051117	040440	
6732	046420	051114	040505	054504	
6733	046426	046440	051101	042513	
6734	046434	020104	040502	006504	
6735	046442	012			
6736	046443	123	051525	042520	.ASCII /SUSPECT PHYSICAL DAMAGE IN HEADER AREA/<15><12>
6737	046450	052103	050040	054510	
6738	046456	044523	040503	020114	
6739	046464	040504	040515	042507	
6740	046472	044440	020116	042510	
6741	046500	042101	051105	040440	
6742	046506	042522	006501	012	
6743	046513	106	051117	040515	.ASCIZ /FORMAT ABORTING/
6744	046520	020124	041101	051117	
6745	046526	044524	043516	000	
6746	046533	103	046517	040515	DH100: .ASCIZ /COMMAND START VALUES/
6747	046540	042116	051440	040524	
6748	046546	052122	053040	046101	
6749	046554	042525	000123		
6750	046560	045522	030466	020061	DH200: .ASCIZ /RK611 REGISTERS/
6751	046566	042522	044507	052123	
6752	046574	051105	000123		
6753	046600	042522	040515	047111	DH500: .ASCIZ /REMAINING REGISTERS NOT VALID/
6754	046606	047111	020107	042522	
6755	046614	044507	052123	051105	
6756	046622	020123	047516	020124	
6757	046630	040526	044514	000104	
6758	046636	044124	020105	047506	DH501: .ASCIZ /THE FOLLOWING REGISTER DATA MAY BE INCORRECT/
6759	046644	046114	053517	047111	
6760	046652	020107	042522	044507	
6761	046660	052123	051105	042040	
6762	046666	052101	020101	040515	
6763	046674	020131	042502	044440	
6764	046702	041516	051117	042522	
6765	046710	052103	000		
6766	046713	105	051122	050040	DH101: .ASCIZ /ERR PC DRIVE CMND CYLNDR TRACK SECTOR WD CNT BUFFER/
6767	046720	020103	042040	044522	
6768	046726	042526	020040	041440	
6769	046734	047115	020104	020040	

CZR6LDO RK06L CTG FMTR MACY11 30(1046)
CZR6LD.P11 27-AUG-81 11:08

28-AUG-81 13:57 PAGE 132
TRAP TABLE

SEQ 0130

6770	046742	041440	046131	042116
6771	046750	020122	052040	040522
6772	046756	045503	020040	051440
6773	046764	041505	047524	020122
6774	046772	020040	042127	041440
6775	047000	052116	020040	052502
6776	047006	043106	051105	000
6777	047013	122	041513	030523
6778	047020	020040	051040	041513
6779	047026	031123	020040	051040
6780	047034	042113	020103	020040
6781	047042	051040	042113	020101
6782	047050	020040	051040	053513
6783	047056	051103	020040	045522
6784	047064	040502	020040	020040
6785	047072	045522	051501	043117
6786	047100	000		
6787	047101	122	042113	020123
6788	047106	020040	051040	042513
6789	047114	000122		
6790	047116	030101	020060	020040
6791	047124	020040	030102	020060
6792	047132	020040	020040	030101
6793	047140	020061	020040	020040
6794	047146	030102	020061	020040
6795	047154	020040	030101	020062
6796	047162	020040	020040	030102
6797	047170	020062	020040	020040
6798	047176	030101	020063	020040
6799	047204	020040	030102	000063
6800	047212	042510	042101	051105
6801	047220	051440	047510	046125
6802	047226	020104	042502	000072
6803	047234	042510	042101	051105
6804	047242	047440	020106	042523
6805	047250	052103	051117	030040
6806	047256	052040	044510	020123
6807	047264	054503	044514	042116
6808	047272	051105	044440	035123
6809	047300	000		
6810	047301	123	040505	041522
6811	047306	044510	043516	043040
6812	047314	051117	044040	040505
6813	047322	042504	035122	000
6814	047327	120	041501	020113
6815	047334	042101	051104	051505
6816	047342	020123	043117	042440
6817	047350	051122	051117	044440
6818	047356	035123	000	
6819	047361	103	046131	042116
6820	047366	020122	052040	040522
6821	047374	045503	020040	051440
6822	047402	041505	047524	000122
6823	047410	054503	047114	051104
6824	047416	020040	051124	041501
6825	047424	000113		

DH201: .ASCIZ /RKCS1 RKCS2 RKDC RKDA RKWCP RKBA RKASOF/

DH202: .ASCIZ /RKDS RKER/

DH203: .ASCIZ /A00 B00 A01 B01 A02 B02 A03 B03/

DH601: .ASCIZ /HEADER SHOULD BE:/

DH602: .ASCIZ /HEADER OF SECTOR 0 THIS CYLINDER IS:/

DH603: .ASCIZ /SEARCHING FOR HEADER:/

DH604: .ASCIZ /PACK ADDRESS OF ERROR IS:/

DH6041: .ASCIZ /CYLNDR TRACK SECTOR/

DH6042: .ASCIZ /CYLNDR TRACK/

6826	047426	041505	020103	040504	DH605:	.ASCIZ	/ECC DATA IS:/
6827	047434	040524	044440	035123			
6828	047442	000					
6829	047443	120	051517	020040	DH6051:	.ASCIZ	/POS PAT CORRECTABLE?/
6830	047450	020040	050040	052101			
6831	047456	020040	020040	041440			
6832	047464	051117	042522	052103			
6833	047472	041101	042514	000077			
6834	047500	047527	042122	020061	DH606:	.ASCIZ	/WORD1 WORD2 WORD3/
6835	047506	020040	047527	042122			
6836	047514	020062	020040	047527			
6837	047522	042122	000063				
6838	047526	047507	042117	020040	DH701:	.ASCIZ	/GOOD BAD WORD NUM/
6839	047534	020040	040502	020104			
6840	047542	020040	020040	047527			
6841	047550	042122	047040	046525			
6842	047556	000					
6843	047557	102	042101	044040	DH607:	.ASCIZ	/BAD HEADER IS:/
6844	047564	040505	042504	020122			
6845	047572	051511	000072				
6846	047576	045522	051504	020040	DH204:	.ASCIZ	/RKDS RKER RKMR2 RKMR3/
6847	047604	020040	045522	051105			
6848	047612	020040	020040	045522			
6849	047620	051115	020062	020040			
6850	047626	045522	051115	000063			
6851	047634	051105	047522	020122	DH502:	.ASCIZ	/ERROR TRYING TO GET FAILURE INFORMATION/
6852	047642	051124	044531	043516			
6853	047650	052040	020117	042507			
6854	047656	020124	040506	046111			
6855	047664	051125	020105	047111			
6856	047672	047506	046522	052101			
6857	047700	047511	000116				
6858	047704	042523	047503	042116	DH503:	.ASCIZ	/SECOND TIMEOUT OCCURRED GETTING DRIVE STATUS/
6859	047712	052040	046511	047505			
6860	047720	052125	047440	041503			
6861	047726	051125	042522	020104			
6862	047734	042507	052124	047111			
6863	047742	020107	051104	053111			
6864	047750	020105	052123	052101			
6865	047756	051525	000				
6866	047761	116	046525	042502	DH800:	.ASCIZ	/NUMBER OF RETRIES:/
6867	047766	020122	043117	051040			
6868	047774	052105	044522	051505			
6869	050002	000072					
6870							
6871	050004	042523	052103	051117	DH900:	.ASCIZ	/SECTOR WC SB WC IS/
6872	050012	020040	041527	051440			
6873	050020	020102	020040	041527			
6874	050026	044440	000123				
6875	050032	001116	006340	006372	DT100:	.WORD	\$ERRPC,DRINUS,ERRCOM,\$REG0,\$REG1,\$REG2,\$REG3,\$REG4
6876	050040	001162	001164	001166			
6877	050046	001170	001172				
6878	050052	001174	001176	001200	DT201:	.WORD	\$REG5,\$REG6,\$REG7,\$REG10,\$REG11,\$REG12,\$REG13
6879	050060	001202	001204	001206			
6880	050066	001210					
6881	050070	001212	001214		DT202:	.WORD	\$REG14,\$REG15

6882	050074	001216	001220	001222	DT203:	.WORD	\$REG16,\$REG17,\$REG20,\$REG21,\$REG22,\$REG23,\$REG24,\$REG25	
6883	050102	001224	001226	001230				
6884	050110	001232	001234					
6885	050114	001174	001176	001200	DT601:	.WORD	\$REG5,\$REG6,\$REG7	
6886	050122	001202	001204	001206	DT602:	.WORD	\$REG10,\$REG11,\$REG12	
6887								
6888	050130	000005			DF01:	.WORD	5	;NUMBER OF HEADER LINES
6889	050132	000				.BYTE	0	;NUMBER OF COL FOR FIPST HDR
6890	050133	000				.BYTE	0	;ALL CHARACTERS OCTAL
6891	050134	046713				.WORD	DH101	;SECOND HDR LINE
6892	050136	010	000			.BYTE	10,0	;NUM OF COL-ALL OCTAL
6893	050140	046560				.WORD	DH200	
6894	050142	000	000			.BYTE	0,0	
6895	050144	047013				.WORD	DH201	
6896	050146	007	000			.BYTE	7,0	
6897	050150	046600				.WORD	DH500	
6898	050152	000	000			.BYTE	0,0	
6899								
6900	050154	000006			DF02:	.WORD	6	
6901	050156	000	000			.BYTE	0,0	
6902	050160	046713				.WORD	DH101	
6903	050162	010	000			.BYTE	10,0	
6904	050164	046560				.WORD	DH200	
6905	050166	000	000			.BYTE	0,0	
6906	050170	047013				.WORD	DH201	
6907	050172	007	000			.BYTE	7,0	
6908	050174	047101				.WORD	DH202	
6909	050176	002	000			.BYTE	2,0	
6910	050200	047116				.WORD	DH203	
6911	050202	010	000			.BYTE	10,0	
6912								
6913	050204	000007			DF03:	.WORD	7	
6914	050206	000	000			.BYTE	0,0	
6915	050210	046713				.WORD	DH101	
6916	050212	010	000			.BYTE	10,0	
6917	050214	046560				.WORD	DH200	
6918	050216	000	000			.BYTE	0,0	
6919	050220	047013				.WORD	DH201	
6920	050222	007	000			.BYTE	7,0	
6921	050224	046636				.WORD	DH501	
6922	050226	000	000			.BYTE	0,0	;'THE FOLLOWING REG DATA MB INCORRECT'
6923	050230	047101				.WORD	DH202	
6924	050232	002	000			.BYTE	2,0	
6925	050234	047116				.WORD	DH203	
6926	050236	010	000			.BYTE	10,0	
6927								
6928	050240	000006			DF05:	.WORD	6	;OPERATION INCOMPLETE
6929	050242	000	000			.BYTE	0,0	
6930	050244	046713				.WORD	DH101	
6931	050246	010	000			.BYTE	10,0	
6932	050250	047301				.WORD	DH603	
6933	050252	000	000			.BYTE	0,0	
6934	050254	047500				.WORD	DH606	
6935	050256	003	000			.BYTE	3,0	
6936	050260	047234				.WORD	DH602	
6937	050262	000	000			.BYTE	0,0	

6938	050264	047500		.WORD	DH606	
6939	050266	003	000	.BYTE	3,0	
6940						
6941	050270	000006		DF06: .WORD	6	:HEADER VRC ERROR
6942	050272	000	000	.BYTE	0,0	
6943	050274	046713		.WORD	DH101	
6944	050276	010	000	.BYTE	10,0	
6945	050300	047212		.WORD	DH601	
6946	050302	000	000	.BYTE	0,0	
6947	050304	047500		.WORD	DH606	
6948	050306	003	000	.BYTE	3,0	
6949				:		:THIS LINE WHEN SPECIFIC HEADER CAN BE READ
6950	050310	047234		.WORD	DH607	:THIS LINE THEN DELETES
6951	050312	000	000	.WORD	DH602	
6952	050314	047500		.BYTE	0,0	
6953	050316	003	000	.WORD	DH606	
6954				.BYTE	3,0	
6955	050320	000006		DF07: .WORD	6	
6956	050322	000	000	.BYTE	0,0	
6957	050324	046713		.WORD	DH101	
6958	050326	010	000	.BYTE	10,0	
6959	050330	047327		.WORD	DH604	
6960	050332	000	000	.BYTE	0,0	
6961	050334	047361		.WORD	DH6041	
6962	050336	003	000	.BYTE	3,0	
6963	050340	047426		.WORD	DH605	
6964	050342	000	000	.BYTE	0,0	
6965	050344	047443		.WORD	DH6051	
6966	050346	003	000	.BYTE	3,0	
6967						
6968	050350	000004		DF10: .WORD	4	
6969	050352	000	000	.BYTE	0,0	
6970	050354	046713		.WORD	DH101	
6971	050356	010	000	.BYTE	10,0	
6972	050360	047327		.WORD	DH604	
6973	050362	000	000	.BYTE	0,0	
6974	050364	047361		.WORD	DH6041	
6975	050366	003	000	.BYTE	3,0	
6976						
6977	050370	000003		DF11: .WORD	3	
6978	050372	000	000	.BYTE	0,0	
6979	050374	047361		.WORD	DH6041	
6980	050376	003	000	.BYTE	3,0	
6981	050400	047526		.WORD	DH701	
6982	050402	000	000	.BYTE	0,0	
6983						
6984	050404	000005		DF12: .WORD	5	
6985	050406	000	000	.BYTE	0,0	
6986	050410	046713		.WORD	DH101	
6987	050412	010	000	.BYTE	10,0	
6988	050414	046560		.WORD	DH200	
6989	050416	000	000	.BYTE	0,0	
6990	050420	047013		.WORD	DH201	
6991	050422	007	000	.BYTE	7,0	
6992	050424	047576		.WORD	DH204	
6993	050426	004	000	.BYTE	4,0	

6994									
6995	050430	000005		DF13:	.WORD	5			:FORMAT FOR 2ND LEVEL ERROR
6996	050432	000	000		.BYTE	0,0			:IN HEADER COMPARE ERROR
6997	050434	046713			.WORD	DH101			:AND 2ND LEVEL HEADER
6998	050436	010	000		.BYTE	10,0			:VRC ERROR
6999	050440	047212			.WORD	DH601			
7000	050442	000	000		.BYTE	0,0			
7001	050444	047500			.WORD	DH606			
7002	050446	003	000		.BYTE	3,0			
7003	050450	047634			.WORD	DH502			
7004	050452	000	000		.BYTE	0,0			
7005									
7006	050454	000005		DF14:	.WORD	5			:FORMAT FOR 2ND LEVEL ERROR
7007	050456	000	000		.BYTE	0,0			:IN OPERATION INCOMPLETE ERROR
7008	050460	046713			.WORD	DH101			
7009	050462	010	000		.BYTE	10,0			
7010	050464	047301			.WORD	DH603			
7011	050466	000	000		.BYTE	0,0			
7012	050470	047500			.WORD	DH606			
7013	050472	003	000		.BYTE	3,0			
7014	050474	047634			.WORD	DH502			
7015	050476	000	000		.BYTE	0,0			
7016									
7017	050500	000010		DF15:	.WORD	10			
7018	050502	000	000		.BYTE	0,0			
7019	050504	046713			.WORD	DH101			
7020	050506	010	000		.BYTE	10,0			
7021	050510	046560			.WORD	DH200			
7022	050512	000	000		.BYTE	0,0			
7023	050514	047013			.WORD	DH201			
7024	050516	007	000		.BYTE	7,0			
7025	050520	047101			.WORD	DH202			
7026	050522	002	000		.BYTE	2,0			
7027	050524	047704			.WORD	DH503			
7028	050526	000	000		.BYTE	0,0			
7029	050530	046636			.WORD	DH501			
7030	050532	000	000		.BYTE	0,0			
7031	050534	047116			.WORD	DH203			
7032	050536	010	000		.BYTE	10,0			
7033									
7034	050540	000010		DF16:	.WORD	10			
7035	050542	000	000		.BYTE	0,0			
7036	050544	046713			.WORD	DH101			
7037	050546	010	000		.BYTE	10,0			
7038	050550	046560			.WORD	DH200			
7039	050552	000	000		.BYTE	0,0			
7040	050554	047013			.WORD	DH201			
7041	050556	007	000		.BYTE	7,0			
7042	050560	047101			.WORD	DH202			
7043	050562	002	000		.BYTE	2,0			
7044	050564	047634			.WORD	DH502			
7045	050566	000	000		.BYTE	0,0			
7046	050570	046636			.WORD	DH501			
7047	050572	000	000		.BYTE	0,0			
7048	050574	047116			.WORD	DH203			
7049	050576	010	000		.BYTE	10,0			

7050						
7051	050600	000004		DF17:	.WORD	4
7052	050602	000	000		.BYTE	0.0
7053	050604	047500			.WORD	DH606
7054	050606	003	000		.BYTE	3.0
7055	050610	047557			.WORD	DH607
7056	050612	000	000		.BYTE	0.0
7057	050614	047500			.WORD	DH606
7058	050616	003	000		.BYTE	3.0
7059						
7060	050620	000002		DF21:	.WORD	2
7061	050622	000	000		.BYTE	0.0
7062	050624	047443			.WORD	DH6051
7063	050626	003	000		.BYTE	3.0
7064						
7065	050630	000001		DF23:	.WORD	1
7066	050632	001	000		.BYTE	1.0
7067						
7068	050634	000001		DF24:	.WORD	1
7069	050636	002	000		.BYTE	2.0
7070						
7071	050640	000001		DF25:	.WORD	1
7072	050642	003	000		.BYTE	3.0
7073						
7074	050644	000002		DF26:	.WORD	2
7075	050646	000	000		.BYTE	0.0
7076	050650	047361			.WORD	DH6041
7077	050652	003	000		.BYTE	3.0
7078		000001		.END		

DF06	050270	1019	6941#											
DF07	050320	1025	6955#											
DF10	050350	1031	6968#											
DF11	050370	1037	6977#											
DF12	050404	1079	1085	1091	1097	1103	1109	1115	1121	6984#				
DF13	050430	1127	1139	6995#										
DF14	050454	1133	7006#											
DF15	050500	1145	7017#											
DF16	050540	1151	7034#											
DF17	050600	1159	7051#											
DF21	050620	1247	7060#											
DF23	050630	1253	1259	1265	7065#									
DF24	050634	1271	7068#											
DF25	050640	1175	1283	7071#										
DF26	050644	1295	7074#											
DH100	046533	873	879	885	891	897	903	909	915	921	927	933	939	945
		951	957	963	969	975	981	987	993	999	1005	1011	1017	1023
		1029	1041	1059	1065	1071	1077	1083	1089	1095	1101	1107	1113	1119
		1125	1131	1137	1143	1149	1179	1185	1191	1197	1203	1209	1215	1221
		1227	1233	1275	6746#									
DH101	046713	6766#	6891	6902	6915	6930	6943	6957	6970	6986	6997	7008	7019	7036
DH200	046560	6750#	6893	6904	6917	6988	7021	7038						
DH201	047013	6777#	6895	6906	6919	6990	7023	7040						
DH202	047101	6787#	6908	6923	7025	7042								
DH203	047116	6790#	6910	6925	7031	7048								
DH204	047576	6846#	6992											
DH500	046600	6753#	6897											
DH501	046636	6758#	6921	7029	7046									
DH502	047634	6851#	7003	7014	7044									
DH503	047704	6858#	7027											
DH601	047212	1167	6800#	6945	6999									
DH602	047234	6803#	6936	6950										
DH603	047301	6810#	6932	7010										
DH604	047327	1035	1293	6814#	6959	6972								
DH6041	047361	6819#	6961	6974	6979	7076								
DH6042	047410	1269	6823#											
DH605	047426	1245	6826#	6963										
DH6051	047443	6829#	6965	7062										
DH606	047500	6834#	6934	6938	6947	6952	7001	7012	7053	7057				
DH607	047557	6843#	7055											
DH701	047526	6838#	6981											
DH800	047761	1251	1257	1263	6866#									
DH900	050004	1281	6871#											
DI =	040000	1361#	4978	5650										
DISPLA	001142	807#	2580*	2588*	5952*									
DISPRE	000174	773#	2588											
DLT =	100000	1390#	4158	4786	4916	4952	5192							
DMD =	000040	1437#												
DOL1	030452	4318	4324#											
DOL2	030526	4313	4316	4339#										
DOL3	030406	4311	4314#											
DONE	002652	1776#	3803*	3819	3829*	3924*	4021*	4341*	4344*					
DONEPR	016630	2650#	2894											
DRA =	000001	1416#	4781	5564										
DRDY =	000200	1425#	4277											
DRINIT	020154	778	2756	2826	2847	2925#	2927	2958	2965	3074				

S.ACLO=	000100	1465#						
S.BRHM=	000100	1480#						
S.BRKE=	040000	1502#						
S.CART=	000400	1482#						
S.DCLO=	010000	1472#						
S.DIB =	002000	1498#						
S.DOOR=	000200	1481#						
S.DRA =	000040	1451#						
S.DROT=	020000	1473#						
S.DRY =	000200	1453#						
S.DSC =	040000	1460#	4864	5012	5029	5092		
S.FLT =	000200	1466#	5007					
S.FORM=	001000	1455#						
S.FWD =	002000	1484#						
S.HDFL=	000200	1495#						
S.HDHM=	000040	1479#						
S.ICYL=	000040	1464#						
S.ILF =	000400	1467#						
S.LIMD=	020000	1501#						
S.LOAD=	010000	1486#						
S.MHD =	000400	1496#						
S.NMOV=	010000	1500#						
S.OFF =	002000	1456#						
S.PAR =	001000	1468#	4867	5344				
S.PIP =	020000	1459#	4875	5070				
S.PLO =	004000	1499#						
S.REV =	004000	1485#						
S.RTZ =	020000	1487#						
S.SECT=	000020	1492#						
S.SKI =	002000	1469#						
S.SPIN=	010000	1458#						
S.SPLS=	010000	1471#						
S.SPOK=	001000	1483#						
S.TYPE=	000400	1454#						
S.UNLD=	040000	1488#						
S.UNS =	040000	1474#						
S.VV =	000100	1452#						
S.WCLK=	000040	1493#						
S.WGAT=	000100	1494#						
S.WLE =	004000	1470#						
S.WRL =	004000	1457#	2952	2959				
S.XDOK=	000020	1478#						
S.XERR=	001000	1497#						
TBITVE=	000014	756#						
TDMFAC	014662	2424#	3048					
TDMSOF	014511	2406#	3023	3034				
TDMSUF	014576	2415#	3024	3035	3049			
TKVEC =	000060	763#	5999*	6000*				
TKWDCT	006306	1796#	2702*	2704*	2705*	3299	3537	3707
TOPROC	030560	3930	4076	4351#				
TOTMES	016057	2541#	4627					
TPINUS	006336	1808#	2626	2629	2650	2652	2660	
TPQ	013122	2265#	2627					
TPVEC =	000064	764#						
TRAPVE=	000034	762#	2569*	2570*				
TRKINC	021240	3081#	3267	3428				

\$GDDAT	001124	799#																			
\$GTSWR	041372	6088#	6398																		
\$HD =	000001	643	644																		
\$HIOCT	037770	5717*	5730#																		
\$ICNT	001104	790#																			
\$ILLUP	042464	6242	6258	6275#																	
\$INTAG	001135	804#	6085*	6104	6124	6237															
\$ITEMB	001114	794#	3855	5959*	5977																
\$LF	001270	853#	5818	5977	6220	6230															
\$LPADR	001106	791#																			
\$LPERR	001110	792#	2927*	3250*	3403*	5971															
\$LSTAD	042632	6312*	6316#																		
\$MAIL =	***** U	2591	5754	5965																	
\$MNEW	042312	6091	6235#																		
\$MSWR	042301	6088	6233#																		
\$NULL	001154	812#	5772	5818																	
\$OCNT	040626	5889*	5918*	5931#																	
\$OCTVL	040366	5831	5856#																		
\$OMODE	040630	5884*	5888*	5893	5896*	5907*	5933#														
\$PASS	001100	787#																			
\$POWER	042472	6273	6278#																		
\$PWRDN	042324	2571	6242#	6270																	
\$PWRMG	042460	6273#																			
\$PWRUP	042376	6252	6258#																		
\$QUES	001266	851#	5818	5977	6142	6213	6230														
\$RDCHR	041644	6155#	6401																		
\$RDDEC =	***** U	6403																			
\$RDLIN	041734	6178#	6402																		
\$RDOCT =	***** U	6403																			
\$RDSZ =	000072	6171#																			
\$REGAD	001160	816#																			
\$REG0	001162	818#	3982	4354	6875																
\$REG1	001164	819#	6875																		
\$REG10	001202	826#	3384*	3474*	3600*	3687*	3773*	4178*	4203*	4235*	6878	6886									
\$REG11	001204	827#	3382*	3475*	3598*	3688*	3771*	4179*	4204*	4236*	6878	6886									
\$REG12	001206	828#	3383*	3599*	3689*	3772*	4180*	4205*	4229*	4231*	6878	6886									
\$REG13	001210	829#	6878																		
\$REG14	001212	830#	6881																		
\$REG15	001214	831#	6881																		
\$REG16	001216	832#	6882																		
\$REG17	001220	833#	6882																		
\$REG2	001166	820#	6875																		
\$REG20	001222	834#	6882																		
\$REG21	001224	835#	6882																		
\$REG22	001226	836#	6882																		
\$REG23	001230	837#	6882																		
\$REG24	001232	838#	6882																		
\$REG25	001234	839#	6882																		
\$REG3	001170	821#	6875																		
\$REG4	001172	822#	6875																		
\$REG5	001174	823#	3339*	3340*	3345*	3355*	3356*	3521	3644*	3645*	3655*	3656*	3672*	3745*							
		3746*	3750*	3780*	3781*	3836*	3837*	4319*	4320*	4487*	4507*	4516*	4530	4533							
		6878	6885																		
\$REG6	001176	824#	3346*	3673*	3751*	4488*	4491*	4502	4504*	4506*	4523*	4524*	4525*	4526*							
		4529*	4531	4532	6878	6885															
\$REG7	001200	825#	3347*	3674*	3752*	3755*	4489*	4490*	4492	4495*	4499*	4501*	4530*	4532*							

\$RESRE 042672	4534*	6878	6885															
\$R2A = ***** U	6350#	6404																
\$SAVRE 042634	6405																	
\$SAVR6 042470	6334#	6403																
\$SETUP= 000116	6251*	6259	6260*	6261*	6277#													
	2558#	2566	2567	2569	2571	2573	2575	5949	5968	5976	6025	6030	6031					
	6061	6237																
\$SIZE 042502	6289#																	
\$STUP = 177777	2558#																	
\$SWR = 163000	632#	643	648	649	650	651	652	653	849	850	2573	5940	5941					
	5942	5943	5944	5953	5960	5965	5969	5977	6274	6097								
\$TKB 001146	809#	5791	5798	5818	5980	6001	6012	6042	6070									
\$TKCNT 040772	5981#	5996*	6031	6048*	6162	6164*												
\$TKINT 041002	2591	5996#	6022	6083														
\$TKQEN= 041001	5985#	6056	6167															
\$TKQIN 040774	5982#	5997*	5998	6054*	6055*	6056	6058*											
\$TKQOU 040776	5983#	5998*	6165	6166*	6167	6169*												
\$TKQSR 041000	5984#	5997	6058	6169														
\$TKS 001144	808#	3444*	3447*	5789	5796	5818	5980	6002*	6038*	6040	6046*	6068	6084*					
	6094	6106*	6126*															
\$TKSRV 041052	5999	6012#																
\$TMP0 001236	840#	3448*	3451*	3472	3495	3498												
\$TMP1 001240	841#	3171*	3172*	3180	3738													
\$TMP10 001256	848#	3184*	3185*	3186														
\$TMP2 001242	842#	3167*	3195	4583*	4612	4614*												
\$TMP3 001244	843#	3168*	3196															
\$TMP4 001246	844#	3169*	3197															
\$TMP5 001250	845#	3133*	3145	3146	3151*													
\$TMP6 001252	846#	3317*	3318*	3319	3320*	3321*	3322*	3380	3383	3387*	3391							
\$TMP7 001254	847#	3718*	3721*	3730*	3769	3772	3776*	3777										
\$TN = 000001	643#																	
\$TPB 001152	811#	5807*	5818															
\$TPFLG 001157	815#	5748	5818															
\$TPS 001150	810#	5805	5818															
\$TRAP 042730	2569	6370#																
\$TRAP2 042752	6381#	6392																
\$TRP = 000013	6385#	6394#	6395#	6396#	6397#	6398	6399#	6400	6401#	6402#	6403#	6404#	6405#					
\$TRPAD 042764	6375	6392#																
\$TSTNM 001102	788#	5952	5977															
\$TTYIN 042170	2820	2897	6180	6181	6193	6211	6225	6229#										
\$TYPBN= ***** U	6397																	
\$TYPDS= ***** U	6397																	
\$TYPE 037772	5748#	6385	6393															
\$TYPEC 040142	5769	5776	5783	5788#	6128													
\$TYPEX 040262	5811	5813	5816#															
\$TYPOC 040430	5887#	6394																
\$TYPON 040444	5886	5889#	6396															
\$TYPOS 040404	5882#	6395																
\$XOFF = 000023	5793	5818																
\$XON = 000021	5800	5818	6014															
\$.\$.SM= ***** U	1724	4703	4938	5129	5140	5178	5492	5533	5545	5599	5621	5639	5666					
	5675																	
\$OFILL 040627	5883*	5887*	5897	5932#														
\$4OCAT= ***** U	5962																	
.	768#	772#	777#	785#	854	1787#	1789#	1790#	1791#	1792#	1793#	1794#	2557#					
	2564	2989	5818	5856#	5977	5980	5984#	5985	5986#	6229#	6230	6237#	6254					

CZR6LDO RK06L CTG FMTR MACY11 30(1046) 28-AUG-81 13:57 PAGE 155 J 12
CZR6LD.P11 27-AUG-81 11:08 CROSS REFERENCE TABLE -- USER SYMBOLS

SEQ 0152

6276

.KT11	1#		
.SETUP	1#	632#	2558
.SWRHI	1#	632#	644
.SWRLO	653#	654	655
.\$ACT1	1#		
.\$APT8	1#		
.\$APTH	1#		
.\$APTY	1#		
.\$ASTA	1#		
.\$CATC	1#	632#	766
.\$CMTA	1#	632#	779
.\$DB2D	1#		
.\$DB20	1#	632#	5818
.\$DIV	1#		
.\$EOP	1#		
.\$ERRO	1#	632#	5934
.\$ERRT	1#		
.\$MULT	1#		
.\$POWE	1#	632#	6238
.\$RAND	1#	632#	
.\$RDDE	1#		
.\$RDOC	1#		
.\$READ	1#	632#	5977
.\$R2AZ	1#		
.\$SAVE	1#	632#	6317
.\$SB2D	1#		
.\$SB20	1#		
.\$SCOP	1#		
.\$SIZE	1#	632#	6281
.\$SUPR	1#		
.\$TRAP	1#	632#	6362
.\$TYPB	1#		
.\$TYPD	1#	632#	
.\$TYPE	1#	632#	5731
.\$TYPO	1#	632#	5857
.\$4OCA	1#		
.1170	1#		

. ABS. 050654 000

ERRORS DETECTED: 0

CZR6LD,CZR6LD.LST/SOL/CRF/NL:FOC/EQ:QNEWSW=SYSMAC.SML,DRIV10.P11,CZR6LD.P11
RUN-TIME: 22 28 2 SECONDS
RUN-TIME RATIO: 135/53=2.5
CORE USED: 56K (112 PAGES)