

RK06L

CARTRIDGE FORMATTER
CZR6LC0

AH-9133C-MC

COPYRIGHT © 76-78

FICHE 1 OF 1

MAR 1978

digital

MADE IN USA

This image shows a microfiche card with a grid of frames. The frames contain data, likely in a tabular or list format, but the text is too small to be legible. The card is oriented vertically and contains approximately 12 columns and 15 rows of frames. The data appears to be organized in a structured manner, possibly representing a directory or a list of records.

48
49
50
51
52
53
54
55
56
57
58
59
60
61
62
63
64
65
66
67
68
69
70
71
72
73
74
75
76
77
78
79
80
81
82
83
84
85
86
87
88
89
90
91
92
93
94
95

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 PROCEDURES
 - 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

96
 97
 98
 99
 100
 101
 102
 103
 104
 105
 106
 107
 108
 109
 110
 111
 112
 113
 114
 115
 116
 117
 118
 119
 120
 121
 122
 123
 124
 125
 126
 127
 128
 129
 130
 131
 132
 133
 134
 135
 136
 137
 138
 139
 140
 141
 142
 143
 144
 145
 146
 147
 148
 149
 150
 151

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

152
153
154
155
156
157
158
159
160
161
162
163
164
165
166
167
168
169
170
171
172
173
174
175
176
177
178
179
180
181
182
183
184
185
186
187
188
189
190
191
192
193
194
195
196
197
198
199
200
201
202
203
204
205
206
207

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 FDBSF 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

208
209
210
211
212
213
214
215
216
217
218
219
220
221
222
223
224
225
226
227
228
229
230
231
232
233
234
235
236
237
238
239
240
241
242
243
244
245
246
247
248
249
250
251
252
253
254
255
256
257
258
259
260
261
262
263

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.

264
265
266
267
268
269
270
271
272
273
274
275
276
277
278
279
280
281
282
283
284
285
286
287
288
289
290
291
292
293
294
295
296
297
298
299
300
301
302
303
304
305
306
307
308
309
310
311
312
313
314
315
316
317
318
319

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.

320
321
322
323
324
325
326
327
328
329
330
331
332
333
334
335
336
337
338
339
340
341
342
343
344
345
346
347
348
349
350
351
352
353
354
355
356
357
358
359
360
361
362
363
364
365
366
367
368
369
370
371
372
373
374
375

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 0-1456, 0-1456	0, 632 FOR RK06 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

376
377
378
379
380
381
382
383
384
385
386
387
388
389
390
391
392
393
394
395
396
397
398
399
400
401
402
403
404
405
406
407
408
409
410
411
412
413
414
415
416
417
418
419
420
421
422
423
424
425
426
427
428
429
430
431

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.

432
433
434
435
436
437
438
439
440
441
442
443
444
445
446
447
448
449
450
451
452
453
454
455
456
457
458
459
460
461
462
463
464
465
466
467
468
469
470
471
472
473
474
475
476
477
478
479
480
481
482
483
484
485
486
487

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:

"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 SDBSF) 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

488
489
490
491
492
493
494
495
496
497
498
499
500
501
502
503
504
505
506
507
508
509
510
511
512
513
514
515
516
517
518
519
520
521
522
523
524
525
526
527
528
529
530
531
532
533
534
535
536
537
538
539
540
541
542
543

544
545
546
547
548
549
550
551
552
553
554
555
556
557
558
559
560
561
562
563
564
565
566
567
568
569
570
571
572
573
574
575
576
577
578
579
580
581
582
583
584
585
586
587
588
589
590
591
592
593
594
595
596
597
598
599

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) (CYLNR) (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:

600
601
602
603
604
605
606
607
608
609
610
611
612
613
614
615
616
617
618
619
620
621
622
623
624
625
626
627
628
629
630
631
632
633
634
635
636

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:
CYLNR 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:
CYLNR TRACK SECTOR

ECC DATA IS:

PAT POS CORRECTABLE?

IF CORRECTABLE? = 1, ERROR IS ECC CORRECTABLE.

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

%

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
683
684
685
686
687
688
689
690
691
692

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
.SSWR= 163000
.TITLE CZR6LCO RK06L CTG FMTR
*COPYRIGHT (C) 1976,1977
*DIGITAL EQUIPMENT CORP.
*MAYNARD, MASS. 01754
*
*PROGRAM BY MARV TEGROTENHUIS
*
*THIS PROGRAM WAS ASSEMBLED USING THE PDP-11 MAINDEC SYSMAC
*PACKAGE (MAINDEC-11-DZGAC-C3), JAN 19, 1977.
*
$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
```

```

693      000007      R7=      %7      ;; GENERAL REGISTER
694      000006      SP=      %6      ;; STACK POINTER
695      000007      PC=      %7      ;; PROGRAM COUNTER
696
697      ;;PRIORITY LEVEL DEFINITIONS
698      000000      PR0=      0      ;; PRIORITY LEVEL 0
699      000040      PR1=      40     ;; PRIORITY LEVEL 1
700      000100      PR2=      100    ;; PRIORITY LEVEL 2
701      000140      PR3=      140    ;; PRIORITY LEVEL 3
702      000200      PR4=      200    ;; PRIORITY LEVEL 4
703      000240      PR5=      240    ;; PRIORITY LEVEL 5
704      000300      PR6=      300    ;; PRIORITY LEVEL 6
705      000340      PR7=      340    ;; PRIORITY LEVEL 7
706
707      ;; "SWITCH REGISTER" SWITCH DEFINITIONS
708      100000      SW15=     100000
709      040000      SW14=     40000
710      020000      SW13=     20000
711      010000      SW12=     10000
712      004000      SW11=     4000
713      002000      SW10=     2000
714      001000      SW09=     1000
715      000400      SW08=     400
716      000200      SW07=     200
717      000100      SW06=     100
718      000040      SW05=     40
719      000020      SW04=     20
720      000010      SW03=     10
721      000004      SW02=     4
722      000002      SW01=     2
723      000001      SW00=     1
724      .EQUIV     SW09, SW9
725      .EQUIV     SW08, SW8
726      .EQUIV     SW07, SW7
727      .EQUIV     SW06, SW6
728      .EQUIV     SW05, SW5
729      .EQUIV     SW04, SW4
730      .EQUIV     SW03, SW3
731      .EQUIV     SW02, SW2
732      .EQUIV     SW01, SW1
733      .EQUIV     SW00, SW0
734
735      ;;DATA BIT DEFINITIONS (BIT00 TO BIT15)
736      100000      BIT15=    100000
737      040000      BIT14=    40000
738      020000      BIT13=    20000
739      010000      BIT12=    10000
740      004000      BIT11=    4000
741      002000      BIT10=    2000
742      001000      BIT09=    1000
743      000400      BIT08=    400
744      000200      BIT07=    200
745      000100      BIT06=    100
746      000040      BIT05=    40
747      000020      BIT04=    20
748      000010      BIT03=    10

```

```

749      000004
750      000002
751      000001
752
753
754
755
756
757
758
759
760
761
762
763
764      000004
765      000010
766      000014
767      000014
768      000014
769      000020
770      000024
771      000030
772      000034
773      000060
774      000064
775      000240
776
777
778      000000
779
780
781
782      000174
783      000174 000000
784      000176 000000
785
786      000200 000137 016202
787      000204 000204
788      000204 000137 020154

```

```

BIT02= 4
BIT01= 2
BIT00= 1
.EQUIV BIT09,BIT9
.EQUIV BIT08,BIT8
.EQUIV BIT07,BIT7
.EQUIV BIT06,BIT6
.EQUIV BIT05,BIT5
.EQUIV BIT04,BIT4
.EQUIV BIT03,BIT3
.EQUIV BIT02,BIT2
.EQUIV BIT01,BIT1
.EQUIV BIT00,BIT0

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

.=0
;#ALL UNUSED LOCATIONS FROM 4 - 776 CONTAIN A ".+2,HALT"
;#SEQUENCE TO CATCH ILLEGAL TRAPS AND INTERRUPTS
;#LOCATION 0 CONTAINS 0 TO CATCH IMPROPERLY LOADED VECTORS
.=174
DISPREG: .WORD 0 ;: SOFTWARE DISPLAY REGISTER
SWREG: .WORD 0 ;: SOFTWARE SWITCH REGISTER
.SBTTL STARTING ADDRESS(ES)
JMP @#START ;: JUMP TO STARTING ADDRESS OF PROGRAM
.=204
JMP @#DRINIT

```



```

789
790
791
792
793
794
795
796 001100 001100
797 001100 000000
798 001102 000
799 001103 000
800 001104 000000
801 001106 000000
802 001110 000000
803 001112 000000
804 001114 000
805 001115 001
806 001116 000000
807 001120 000000
808 001122 000000
809 001124 000000
810 001126 000000
811 001130 000000
812 001132 000000
813 001134 000
814 001135 000
815 001136 000000
816 001140 177570
817 001142 177570
818 001144 177560
819 001146 177562
820 001150 177564
821 001152 177566
822 001154 000
823 001155 002
824 001156 012
825 001157 000
826 001160 000000
827
828 001162 000000
829 001164 000000
830 001166 000000
831 001170 000000
832 001172 000000
833 001174 000000
834 001176 000000
835 001200 000000
836 001202 000000
837 001204 000000
838 001206 000000
839 001210 000000
840 001212 000000
841 001214 000000
842 001216 000000
843 001220 000000
844 001222 000000

```

.SBTTL COMMON TAGS

```

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

```

SCMTAG: . =1100

```

SPASS: .WORD 0
STSTNM: .BYTE 0
SERFLG: .BYTE 0
SICNT: .WORD 0
SLPADR: .WORD 0
SLPERR: .WORD 0
SERTTL: .WORD 0
SITEMB: .BYTE 0
SERMAX: .BYTE 1
SERRPC: .WORD 0
SGDADR: .WORD 0
SBDADR: .WORD 0
SGDDAT: .WORD 0
SBDDAT: .WORD 0
SAUTOB: .BYTE 0
SINTAG: .BYTE 0
SWR: .WORD DSWR
DISPLAY: .WORD DDISP
$TKS: 177560
$TKB: 177562
$TPS: 177564
$TPB: 177566
$NULL: .BYTE 0
$FILLS: .BYTE 2
$FILLC: .BYTE 12
$TPFLG: .BYTE 0
$REGAD: .WORD 0
$REG0: .WORD 0
$REG1: .WORD 0
$REG2: .WORD 0
$REG3: .WORD 0
$REG4: .WORD 0
$REG5: .WORD 0
$REG6: .WORD 0
$REG7: .WORD 0
$REG10: .WORD 0
$REG11: .WORD 0
$REG12: .WORD 0
$REG13: .WORD 0
$REG14: .WORD 0
$REG15: .WORD 0
$REG16: .WORD 0
$REG17: .WORD 0
$REG20: .WORD 0

```

```

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

; AUTOMATIC MODE INDICATOR
; INTERRUPT MODE INDICATOR

; ADDRESS OF SWITCH REGISTER
; ADDRESS OF DISPLAY REGISTER
; TTY KBD STATUS
; TTY KBD BUFFER
; TTY PRINTER STATUS REG. ADDRESS
; TTY PRINTER BUFFER REG. ADDRESS
; CONTAINS NULL CHARACTER FOR FILLS
; CONTAINS # OF FILLER CHARACTERS REQUIRED
; INSERT FILL CHARS. AFTER A "LINE FEED"
; "TERMINAL AVAILABLE" FLAG (BIT<07>=0=YES)
; CONTAINS THE ADDRESS FROM WHICH ($REG0) WAS OBTAINED
; CONTAINS (($REGAD)+0)
; CONTAINS (($REGAD)+2)
; CONTAINS (($REGAD)+4)
; CONTAINS (($REGAD)+6)
; CONTAINS (($REGAD)+10)
; CONTAINS (($REGAD)+12)
; CONTAINS (($REGAD)+14)
; CONTAINS (($REGAD)+16)
; CONTAINS (($REGAD)+20)
; CONTAINS (($REGAD)+22)
; CONTAINS (($REGAD)+24)
; CONTAINS (($REGAD)+26)
; CONTAINS (($REGAD)+30)
; CONTAINS (($REGAD)+32)
; CONTAINS (($REGAD)+34)
; CONTAINS (($REGAD)+36)
; CONTAINS (($REGAD)+40)

```

845	001224	000000		\$REG21: .WORD	0	:: CONTAINS ((SREGAD)+42)
846	001226	000000		\$REG22: .WORD	0	:: CONTAINS ((SREGAD)+44)
847	001230	000000		\$REG23: .WORD	0	:: CONTAINS ((SREGAD)+46)
848	001232	000000		\$REG24: .WORD	0	:: CONTAINS ((SREGAD)+50)
849	001234	000000		\$REG25: .WORD	0	:: CONTAINS ((SREGAD)+52)
850	001236	000000		STMP0: .WORD	0	:: USER DEFINED
851	001240	000000		STMP1: .WORD	0	:: USER DEFINED
852	001242	000000		STMP2: .WORD	0	:: USER DEFINED
853	001244	000000		STMP3: .WORD	0	:: USER DEFINED
854	001246	000000		STMP4: .WORD	0	:: USER DEFINED
855	001250	000000		STMP5: .WORD	0	:: USER DEFINED
856	001252	000000		STMP6: .WORD	0	:: USER DEFINED
857	001254	000000		STMP7: .WORD	0	:: USER DEFINED
858	001256	000000		STMP10: .WORD	0	:: USER DEFINED
859	001260	000000		\$ESCAPE: 0		:: ESCAPE ON ERROR ADDRESS
860	001262	177607	000377	\$BELL: .ASCIZ	<207><377><377>	:: CODE FOR BELL
861	001266	077		\$QUES: .ASCII	/?/	:: QUESTION MARK
862	001267	015		\$CRLF: .ASCII	<15>	:: CARRIAGE RETURN
863	001270	000012		\$LF: .ASCIZ	<12>	:: LINE FEED
864				;*****		

.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 SITEMB. THIS NUMBER INDICATES WHICH ITEM IN THE TABLE IS PERTINENT.
;*NOTE1: IF SITEMB IS 0 THE ONLY PERTINENT DATA IS (SERRPC).
;*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:

865					
866					
867					
868					
869					
870					
871					
872					
873					
874					
875					
876					
877					
878					
879	001272				
880					
881					
882	001272	042672			
883	001274	046413			
884	001276	047712			
885	001300	050010			
886					
887					
888	001302	042722			
889	001304	046413			
890	001306	047712			
891	001310	050010			
892					
893					
894	001312	042760			
895	001314	046413			
896	001316	047712			
897	001320	050010			
898					
899					
900	001322	043015			
901	001324	046413			
902	001326	047712			
903	001330	050010			
904					
905					
906	001332	043042			
907	001334	046413			
908	001336	047712			
909	001340	050034			
910					
911					
912	001342	043070			
913	001344	046413			
914	001346	047712			
915	001350	050064			
916					
917					
918	001352	043123			
919	001354	046413			
920	001356	047712			

```

: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

```

921	001360	050064	DF03	
922				
923			.ERROR 10	
924	001362	043163	EM10	;AC LOW
925	001364	046413	DH100	
926	001366	047712	DT100	
927	001370	050034	DF02	
928				
929			.ERROR 11	
930	001372	043176	EM11	;SPEED LOSS
931	001374	046413	DH100	
932	001376	047712	DT100	
933	001400	050034	DF02	
934				
935			.ERROR 12	
936	001402	043215	EM12	;ILLEGAL FUNCTION
937	001404	046413	DH100	
938	001406	047712	DT100	
939	001410	050034	DF02	
940				
941			.ERROR 13	
942	001412	043250	EM13	;PROGRAMMING ERROR
943	001414	046413	DH100	
944	001416	047712	DT100	
945	001420	050010	DF01	
946				
947			.ERROR 14	;NON-EXISTANT FUNCTION
948	001422	043276	EM14	
949	001424	046413	DH100	
950	001426	047712	DT100	
951	001430	050034	DF02	
952				
953			.ERROR 15	
954	001432	043336	EM15	;DRIVE TYPE ERROR
955	001434	046413	DH100	
956	001436	047712	DT100	
957	001440	050034	DF02	
958				
959			.ERROR 16	
960	001442	043363	EM16	;FORMAT ERROR
961	001444	046413	DH100	
962	001446	047712	DT100	
963	001450	050034	DF02	
964				
965			.ERROR 17	
966	001452	043404	EM17	;WRITE LOCK ERROR
967	001454	046413	DH100	
968	001456	047712	DT100	
969	001460	050034	DF02	
970				
971			.ERROR 20	
972	001462	043431	EM20	;DRIVE UNSAFE
973	001464	046413	DH100	
974	001466	047712	DT100	
975	001470	050034	DF02	
976				

977			:ERROR 21	
978	001472	043460	EM21	;SEEK INCOMPLETE
979	001474	046413	DH100	
980	001476	047712	DT100	
981	001500	050034	DF02	
982				
983			:ERROR 22	
984	001502	043512	EM22	;CYLINDER OVERFLOW
985	001504	046413	DH100	
986	001506	047712	DT100	
987	001510	050034	DF02	
988				
989			:ERROR 23	
990	001512	043546	EM23	;ILLEGAL CYLINDER
991	001514	046413	DH100	
992	001516	047712	DT100	
993	001520	050034	DF02	
994				
995			:ERROR 24	
996	001522	043611	EM24	;DRIVE OFF TRACK
997	001524	046413	DH100	
998	001526	047712	DT100	
999	001530	050034	DF02	
1000				
1001			:ERROR 25	
1002	001532	043635	EM25	;DRIVE TIMING ERROR
1003	001534	046413	DH100	
1004	001536	047712	DT100	
1005	001540	050034	DF02	
1006				
1007			:ERROR 26	
1008	001542	043664	EM26	;DATA LATE
1009	001544	046413	DH100	
1010	001546	047712	DT100	
1011	001550	050034	DF02	
1012				
1013			:ERROR 27	
1014	001552	043710	EM27	;CONTROLLER TIMEOUT
1015	001554	046413	DH100	
1016	001556	047712	DT100	
1017	001560	050034	DF02	
1018				
1019			:ERROR 30	
1020	001562	043745	EM30	;OPERATION INCOMPLETE
1021	001564	046413	DH100	
1022	001566	047712	DT100	
1023	001570	050120	DF05	
1024				
1025			:ERROR 31	
1026	001572	044004	EM31	;HEADER CRC ERROR
1027	001574	046413	DH100	
1028	001576	047712	DT100	
1029	001600	050150	DF06	
1030				
1031			:ERROR 32	
1032	001602	044031	EM32	;DATA CHECK ERROR

1033	001604	046413	DH100	
1034	001606	047712	DT100	
1035	001610	050200	DF07	
1036				
1037			:ERROR 33	
1038	001612	044056	EM33	;WRITE CHECK ERROR
1039	001614	046413	DH100	
1040	001616	047712	DT100	
1041	001620	050230	DF10	
1042				
1043			:ERROR 34	
1044	001622	044104	EM34	;DATA MISCOMPARE
1045	001624	047207	DH604	
1046	001626	047774	DT601	
1047	001630	050250	DF11	
1048				
1049			:ERROR 35	
1050	001632	044130	EM35	;NO DRIVE RESPONSE-UFE & NED
1051	001634	046413	DH100	
1052	001636	047712	DT100	
1053	001640	050010	DF01	
1054				
1055			:ERROR 36	
1056	001642	044170	EM36	;DRIVE ERROR WILL NOT CLEAR
1057	001644	000000	0	
1058	001646	000000	0	
1059	001650	000000	0	
1060				
1061			:ERROR 37	
1062	001652	044227	EM37	;DRIVE STATUS CHANGE WILL NOT CLEAR
1063	001654	000000	0	
1064	001656	000000	0	
1065	001660	000000	0	
1066				
1067			:ERROR 40	
1068	001662	044276	EM40	;ATTENTION BUT NO STATUS CHANGE OR FAULT
1069	001664	046413	DH100	
1070	001666	047712	DT100	
1071	001670	050034	DF02	
1072				
1073			:ERROR 41	
1074	001672	044352	EM41	;ATTENTION BUT DRIVE NOT AVAILABLE
1075	001674	046413	DH100	
1076	001676	047712	DT100	
1077	001700	050034	DF02	
1078				
1079			:ERROR 42	
1080	001702	044420	EM42	;ATTENTION WHEN NOT EXPECTED
1081	001704	046413	DH100	
1082	001706	047712	DT100	
1083	001710	050034	DF02	
1084				
1085			:ERROR 43	
1086	001712	044460	EM43	;ERROR WHILE GATHERING DRIVE STATUS
1087	001714	046413	DH100	
1088	001716	047712	DT100	

1089	001720	050264	DF12	
1090				
1091			: ERROR 44	
1092	001722	045006	EM63	; CLEAR CONTROLLER DID NOT CLEAR ERROR
1093	001724	046413	DH100	
1094	001726	047712	DT100	
1095	001730	050264	DF12	
1096				
1097			: ERROR 45	
1098	001732	045057	EM64	; NO ATTENTION IN ATTENTION SUMMARY REG
1099	001734	046413	DH100	
1100	001736	047712	DT100	
1101	001740	050264	DF12	
1102				
1103			: ERROR 46	
1104	001742	045136	EM65	; UNSOLICITED ATTENTION
1105	001744	046413	DH100	
1106	001746	047712	DT100	
1107	001750	050264	DF12	
1108				
1109			: ERROR 47	
1110	001752	045170	EM66	; UNEXPECTED DATA TYPE ERROR
1111	001754	046413	DH100	
1112	001756	047712	DT100	
1113	001760	050264	DF12	
1114				
1115			: ERROR 50	
1116	001762	045227	EM67	; ATTENTION DID NOT RESET WITH CLEAR
1117	001764	046413	DH100	
1118	001766	047712	DT100	
1119	001770	050264	DF12	
1120				
1121			: ERROR 51	
1122	001772	045276	EM70	; SUBSYSTEM CLEAR DID NOT CLEAR DRIVE ATTENTION
1123	001774	046413	DH100	
1124	001776	047712	DT100	
1125	002000	050264	DF12	
1126				
1127			: ERROR 52	
1128	002002	044527	EM52	; MULTIPLE DRIVE SELECT
1129	002004	046413	DH100	
1130	002006	047712	DT100	
1131	002010	050264	DF12	
1132				
1133			: ERROR 53	
1134	002012	044561	EM53	; ABBREVIATED HCE ERROR
1135	002014	046413	DH100	
1136	002016	047712	DT100	
1137	002020	050310	DF13	
1138				
1139			: ERROR 54	
1140	002022	043745	EM30	; OPERATION INCOMPLETE ERROR
1141	002024	046413	DH100	
1142	002026	047712	DT100	
1143	002030	050334	DF14	
1144				

1145			:ERROR 55	
1146	002032	044004	EM31	;ABREVIATED HCRC ERROR
1147	002034	046413	DH100	
1148	002036	047712	DT100	
1149	002040	050310	DF13	
1150				
1151			:ERROR 56	
1152	002042	044612	EM56	;2 TIMEOUT ERROR
1153	002044	046413	DH100	
1154	002046	047712	DT100	
1155	002050	050360	DF15	
1156				
1157			:ERROR 57	
1158	002052	044612	EM56	;2ND LEVEL IN SUBSYSTEM TIMEOUT
1159	002054	046413	DH100	
1160	002056	047712	DT100	
1161	002060	050420	DF16	
1162				
1163			:ERROR 60	
1164	002062	044640	EM60	;ERROR IN RECAL FOR RECOVERY
1165	002064	000000	0	
1166	002066	000000	0	
1167	002070	000000	0	
1168				
1169			:ERROR 61	
1170	002072	044706	EM61	;ABORT MESSAGE
1171	002074	000000	0	
1172	002076	000000	0	
1173	002100	000000	0	
1174				
1175			:ERROR 62	
1176	002102	044760	EM62	;HEADER MISCOMPARE
1177	002104	047072	DH601	
1178	002106	047774	DT601	
1179	002110	050460	DF17	
1180				
1181			:ERROR 63	
1182	002112	000000	0	;DATA ERROR WORDS
1183	002114	000000	0	
1184	002116	050002	DT602	
1185	002120	050520	DF25	
1186				
1187			:ERROR 64	
1188	002122	045006	EM63	;CLEAR CONTROLLER DID NOT CLEAR ERROR
1189	002124	046413	DH100	
1190	002126	047712	DT100	
1191	002130	050034	DF02	
1192				
1193			:ERROR 65	
1194	002132	045057	EM64	;NO ATTENTION IN ATTENTION SUMMARY REG
1195	002134	046413	DH100	
1196	002136	047712	DT100	
1197	002140	050034	DF02	
1198				
1199			:ERROR 66	
1200	002142	045136	EM65	;UNSOLICITED ATTENTION

1201	002144	046413	DH100	
1202	002146	047712	DT100	
1203	002150	050034	DF02	
1204				
1205			.ERROR 67	
1206	002152	045170	EM66	;UNEXPECTED DATA TYPE ERROR
1207	002154	046413	DH100	
1208	002156	047712	DT100	
1209	002160	050034	DF02	
1210				
1211			.ERROR 70	
1212	002162	045227	EM67	;ATTENTION DID NOT RESET WITH CLEAR
1213	002164	046413	DH100	
1214	002166	047712	DT100	
1215	002170	050034	DF02	
1216				
1217			.ERROR 71	
1218	002172	045276	EM70	;SUBSYSTEM CLEAR DID NOT CLEAR ATT
1219	002174	046413	DH100	
1220	002176	047712	DT100	
1221	002200	050034	DF02	
1222				
1223			.ERROR 72	
1224	002202	045360	EM71	;DATA LATE WHEN UNLOADING HEADER
1225	002204	046413	DH100	
1226	002206	047712	DT100	
1227	002210	050034	DF02	
1228				
1229			.ERROR 73	
1230	002212	045424	EM72	;CONTROLLER ERROR DURING DRIVER SERVICE
1231	002214	046413	DH100	
1232	002216	047712	DT100	
1233	002220	050034	DF02	
1234				
1235			.ERROR 74	
1236	002222	045477	EM73	;DRIVE DETECTED PARITY ERROR
1237	002224	046413	DH100	
1238	002226	047712	DT100	
1239	002230	050034	DF02	
1240				
1241			.ERROR 75	
1242	002232	045537	EM74	;UNDEFINED ERROR
1243	002234	046413	DH100	
1244	002236	047712	DT100	
1245	002240	050034	DF02	
1246				
1247			.ERROR 76	
1248	002242	045563	EM75	;MARKING SECTOR BAD MESSAGE
1249	002244	000000	0	
1250	002246	000000	0	
1251	002250	000000	0	
1252				
1253			.ERROR 77	
1254	002252	045617	EM76	;BAD DATA VERIFICATION WITH READ
1255	002254	047306	DH605	
1256	002256	047774	DT601	

1257	002260	050500	DF21	
1258			.ERROR 100	
1259			EM77	;RETRY SUCCESSFUL MESSAGE
1260	002262	045712	DH800	
1261	002264	047641	DT601	
1262	002266	047774	DF23	
1263	002270	050510		
1264			.ERROR 101	
1265			EM77	;ANOTHER RETRY SUCCESSFUL MESSAGE
1266	002272	045712	DH800	
1267	002274	047641	DT601	
1268	002276	047774	DF23	
1269	002300	050510		
1270			.ERROR 102	
1271			EM100	;RETRY UNSUCCESSFUL MESSAGE
1272	002302	045737	DH800	
1273	002304	047641	DT601	
1274	002306	047774	DF23	
1275	002310	050510		
1276			.ERROR 103	
1277			EM101	;NO VALID HEADERS IN TRACK JUST READ
1278	002312	045766	DH6042	
1279	002314	047270	DT601	
1280	002316	047774	DF24	
1281	002320	050514		
1282			.ERROR 104	
1283			EM102	;BSE ERROR ON SECTOR NOT LISTED AS BAD
1284	002322	046050	DH100	
1285	002324	046413	DT100	
1286	002326	047712	DF02	
1287	002330	050034		
1288			.ERROR 105	
1289			EM103	;WORD COUNT NOT CORRECT TO CONTINUE
1290	002332	046126	DH900	
1291	002334	047664	DT602	
1292	002336	050002	DF25	
1293	002340	050520		
1294			.ERROR 106	
1295			EM104	;DRIVE STATUS NOT VALID
1296	002342	046172	0	
1297	002344	000000	0	
1298	002346	000000	0	
1299	002350	000000		
1300			.ERROR 107	
1301			EM105	;BAD HEADER AREA
1302	002352	046257	DH604	
1303	002354	047207	DT601	
1304	002356	047774	DF26	
1305	002360	050524		

```

1306
1307
1308
1309      000000      RKCS1= 0      ;CONTROL AND STATUS REGISTER 1
1310      000002      RKWC= 2      ;WORD COUNT REGISTER
1311      000004      RKBA= 4      ;BUS ADDRESS REGISTER
1312      000006      RKDA= 6      ;DESIRED TRACK SECTOR REGISTER
1313      000010      RKCS2= 10     ;CONTROL AND STATUS REGISTER 2
1314      000012      RKDS= 12     ;DRIVE STATUS REGISTER
1315      000014      RKER= 14     ;ERROR REGISTER
1316      000016      RKASOF= 16    ;ATTENTION SUMMARY AND OFFSET REGISTER
1317      000020      RKDC= 20     ;DESIRED CYLINDER REGISTER
1318      000020      RKDCYL= 20    ;DESIRED CYLINDER REGISTER
1319      000024      RKDB= 24     ;DATA BUFFER
1320      000026      RKMR1= 26    ;MAINTENANCE REGISTER 1
1321      000034      RKMR2= 34    ;MAINTENANCE REGISTER 2
1322      000036      RKMR3= 36    ;MAINTENANCE REGISTER 3
1323      000030      RKPOS= 30    ;ECC POSITION INFORMATION
1324      000030      RKECPS= 30    ;ECC POSITION INFORMATION
1325      000032      RKPAT= 32    ;ECC PATTERN INFORMATION
1326      000032      RKECPT= 32    ;ECC PATTERN INFORMATION
1327
1328      .SBTTL DRIVE COMMANDS
1329
1330      000101      SELDRV= 101    ;SELECT DRIVE
1331      000103      PACK= 103    ;PACK ACKNOWLEDGE
1332      000105      CLEAR= 105    ;DRIVE CLEAR
1333      000107      UNLOAD= 107    ;UNLOAD
1334      000111      SRTSPL= 111    ;START SPINDLE
1335      000113      RECAL= 113    ;RECALIBRATE
1336      000115      OFFSET= 115    ;OFFSET
1337      000117      SEEK= 117     ;SEEK
1338      000121      RDDATA= 121    ;READ DATA
1339      000123      WRDATA= 123    ;WRITE DATA
1340      000125      RDHEAD= 125    ;READ HEADER
1341      000127      WRHEAD= 127    ;WRITE HEADER AND DATA
1342      000131      WRTCHK= 131    ;WRITE CHECK
1343
1344      ; THE FOLLOWING ARE NOT DRIVE COMMANDS BUT ARE USED BY THE DRIVER
1345      ; TO SIMULATE A SPECIFIC DESIRED OPERATION
1346
1347      000140      RELEAS= 140    ;RELEASE DRIVE
1348      000141      RDSTAT= 141    ;GET ALL STATUS FROM DRIVE
1349      000164      RDALHD= 164    ;READ ALL HEADERS
1350      000176      CONCLR= 176    ;CONTROLLER CLEAR (BIT 15 OF CS1)
1351      000177      SUBCLR= 177    ;SUBSYSTEM CLEAR (BIT 5 OF CS2)
1352      000300      INTR= 300     ;GENERATE INTERRUPT TO CPU
1353
1354      ; DRIVER ISSUED SERVICE COMMANDS
1355
1356      000001      DR.SEL= 001    ;DRIVE SELECT
1357      000005      DR.CLR= 005    ;DRIVE CLEAR
1358
1359      .SBTTL CONTROL AND STATUS REGISTER 1 BITS
1360
1361      000001      GO= BIT0      ;GO BIT
    
```

```

1362      000100      IE=      BIT6      ; INTERRUPT ENABLE
1363      000200      RDY=     BIT7      ; CONTROLLER READY
1364      000400      BA16=    BIT8      ; BUS ADDRESS BIT 16
1365      001000      BA17=    BIT9      ; BUS ADDRESS BIT 17
1366      002000      CDT=     BIT10     ; CONTROLLER DRIVE TYPE (0=RK06,1=RK07)
1367      004000      CTO=     BIT11     ; CONTROLLER TIMED OUT WAITING FOR
1368                                     ; DRIVE RESPONSE
1369      010000      CFMT=    BIT12     ; CONTROLLER DRIVE FORMAT (0=22 SECTOR, 1=20 SECTOR)
1370      020000      SPAR=    BIT13     ; DRIVE BUS PARITY ERROR DETECTED BY CONTROLLER
1371      040000      DI=      BIT14     ; DRIVE INTERRUPT
1372      100000      CERR=    BIT15     ; CONTROLLER ERROR
1373      100000      CCLR=    BIT15     ; CONTROLLER CLEAR
1374
1375      ;           ; THESE BIT DEFINITIONS ARE USED FOR ADDRESS
1376      ;           ; THE HIGH BYTE OF RKCS1
1377
1378      000001      B.BA16=  BIT0      ; BUS ADDRESS BIT 16
1379      000002      B.BA17=  BIT1      ; BUS ADDRESS BIT 17
1380      000004      B.CDT=   BIT2      ; CONTROLLER DRIVE TYPE (0=RK06,1=RK07)
1381      000020      B.CFMT=  BIT4      ; CONTROLLER DRIVE FORMAT (0=22 SECTOR, 1=20 SECTOR)
1382
1383      .SBTTL CONTROL AND STATUS REGISTER 2 BITS
1384
1385      000007      DRVMSK=  7      ; MASK FOR DRIVE SELECTION CODE
1386      000010      DESL=    BIT3      ; DESELECT OR RELEASE DRIVE IN BITS 0-2
1387      000010      RLS=    BIT3      ; DESELECT OR RELEASE DRIVE IN BITS 0-2
1388      000020      BAI=    BIT4      ; BUS ADDRESS INCREMENT INHIBIT
1389      000040      CLR=    BIT5      ; CLEAR CONTROLLER AND ALL DRIVES
1390      000040      SCLR=   BIT5      ; CLEAR CONTROLLER AND ALL DRIVES
1391      000100      IR=     BIT6      ; INPUT READY
1392      000200      OR=     BIT7      ; OUTPUT READY
1393      000400      UFE=    BIT8      ; UNIT FIELD ERROR
1394      001000      MDS=    BIT9      ; MULTIPLE DRIVE SELECT
1395      002000      PGE=    BIT10     ; PROGRAMMING ERROR
1396      004000      NEM=    BIT11     ; NON-EXISTENT MEMORY
1397      010000      NED=    BIT12     ; NON-EXISTENT DRIVE
1398      020000      UPE=    BIT13     ; UNIBUS PARITY ERROR
1399      040000      WCE=    BIT14     ; WRITE CHECK ERROR
1400      100000      DLT=    BIT15     ; DATA LATE ERROR
1401
1402      .SBTTL ERROR REGISTER BIT DEFINITION
1403
1404      000001      ILC=    BIT0      ; ILLEGAL FUNCTION CODE
1405      ;*ILF=    BIT0      ; ILLEGAL FUNCTION CODE
1406      000002      SKI=    BIT1      ; SEEK INCOMPLETE
1407      000004      ILF=    BIT2      ; ILLEGAL DRIVE FUNCTION
1408      000004      NXF=    BIT2      ; ILLEGAL DRIVE FUNCTION
1409      000010      DRPAR=   BIT3      ; DRIVE DETECTED DRIVE BUS PARITY ERROR
1410      000020      FMTE=   BIT4      ; FORMAT ERROR
1411      000040      DTYE=   BIT5      ; DRIVE TYPE ERROR
1412      000100      ECH=    BIT6      ; ECC HARD
1413      000200      BSE=    BIT7      ; BAD SECTOR ERROR
1414      000400      HCRC=   BIT8      ; HEADER CRC ERROR
1415      000400      HVRC=   BIT8      ; HEADER VRC ERROR
1416      001000      COE=    BIT9      ; CYLINDER ADDRESS OVERFLOW ERROR
1417      002000      IDAE=   BIT10     ; INVALID DISK ADDRESS ERROR

```

ERROR REGISTER BIT DEFINITION

1418	004000	WLE=	BIT11	;WRITE LOCK ERROR
1419	010000	DTE=	BIT12	;DRIVE TIMING ERROR
1420	020000	OPI=	BIT13	;OPERATION (SEARCH) INCOMPLETE
1421	040000	UNS=	BIT14	;DRIVE UNSAFE
1422	100000	DCK=	BIT15	;DATA CHECK

.SBTTL STATUS REGISTER BIT DEFINITION

1423				
1424				
1425				
1426	000001	DRA=	BIT0	;DRIVE AVAILABLE (CONTROLLER IS SET IF THIS BIT IS RESET)
1427				
1428	000004	OFST=	BIT2	;DRIVE OFFSET
1429	000010	ACLO=	BIT3	;AC LOW
1430	000020	SPDLSS=	BIT4	;SPEED LOSS
1431	000020	DCLO=	BIT4	;DC LOW
1432	000040	DROT=	BIT5	;DRIVE OFF TRACK
1433	000100	VV=	BIT6	;VOLUME VALID
1434	000200	DRY=	BIT7	;DRIVE READY
1435	000200	DRDY=	BIT7	;DRIVE READY
1436	000400	DDT=	BIT8	;DRIVE TYPE (0=RK06,1=RK07)
1437	004000	WRL=	BIT11	;WRITE LOCK
1438	020000	PIP=	BIT13	;POSITIONING IN PROGRESS
1439	040000	DSC=	BIT14	;DRIVE STATUS CHANGE
1440	100000	SVAL=	BIT15	;STATUS VALID

.SBTTL MAINTENANCE REGISTER 1 BIT DEFINITION

1441				
1442				
1443				
1444	000017	MESMSK=	17	;MESSAGE MASK
1445				
1446	000020	PAT=	BIT4	;FORCE EVEN PARITY ON DRIVE BUS MESSAGE LINES
1447	000040	DMD=	BIT5	;DIAGNOSTIC MODE
1448	000100	MSP=	BIT6	;MAINTENANCE SECTOR PULSE
1449	000200	MIND=	BIT7	;MAINTENANCE INDEX
1450	000400	MCLK=	BIT8	;MAINTENANCE CLOCK
1451	001000	MERD=	BIT9	;MAINTENANCE ENCODED READ DATA
1452	002000	MEWD=	BIT10	;MAINTENANCE ENCODED WRITE DATA
1453	004000	PCA=	BIT11	;PRECOMPENSATION ADVANCE
1454	010000	PCD=	BIT12	;PRECOMPENSATION DELAY
1455	020000	ECCW=	BIT13	;ECC WORD IS BEING READ OR WRITTEN
1456	040000	WRTGAT=	BIT14	;WRITE GATE
1457	100000	RDGATE=	BIT15	;READ GATE

.SBTTL DEFINITION OF DRIVE STATUS BYTE 00 MESSAGE A

1458				
1459				
1460				
1461	000040	S.DRA=	BIT5	;DRIVE AVAILIABLE
1462	000100	S.VV=	BIT6	;VOLUME VALID
1463	000200	S.DRY=	BIT7	;DRIVE READY
1464	000400	S.TYPE=	BIT8	;DRIVE TYPE
1465	001000	S.FORM=	BIT9	;DRIVE FORMAT
1466	002000	S.OFF=	BIT10	;OFFSET
1467	004000	S.WRL=	BIT11	;WRITE LOCK
1468	010000	S.SPIN=	BIT12	;SPINDLE ON
1469	020000	S.PIP=	BIT13	;POSITIONING IN PROGRESS
1470	040000	S.DSC=	BIT14	;DRIVE STATUS CHANGE

.SBTTL DEFINITION OF DRIVE STATUS BYTE 00 MESSAGE B

1471
1472
1473

DEFINITION OF DRIVE STATUS BYTE 00 MESSAGE B

1474	000040	S.ICYL= BIT5	: ILLEGAL CYLINDER ADDRESS
1475	000100	S.ACLO= BIT6	: AC LOW
1476	000200	S.FLT= BIT7	: DRIVE FAULT
1477	000400	S.ILF= BIT8	: ILLEGAL FUNCTION
1478	001000	S.PAR= BIT9	: DRIVE DETECTED DRIVE BUS PARITY ERROR
1479	002000	S.SKI= BIT10	: SEEK INCOMPLETE
1480	004000	S.WLE= BIT11	: WRITE LOCK ERROR
1481	010000	S.SPLS= BIT12	: SPEED LOSS
1482	010000	S.DCLO= BIT12	: DC LOW
1483	020000	S.DROT= BIT13	: DRIVE OFF TRACK
1484	040000	S.UNS= BIT14	: DRIVE UNSAFE

.SBTTL DEFINITION OF DRIVE STATUS BYTE 01 MESSAGE A

1488	000020	S.XDOK= BIT4	: TRANSDUCER OK
1489	000040	S.HDHM= BIT5	: HEADS HOME
1490	000100	S.BRHM= BIT6	: BRUSHES HOME
1491	000200	S.DOOR= BIT7	: DOOR INTERLOCKED
1492	000400	S.CART= BIT8	: CARTAGE INTERLOCK
1493	001000	S.SPOK= BIT9	: SPEED OK
1494	002000	S.FWD= BIT10	: FORWARD
1495	004000	S.REV= BIT11	: REVERSE
1496	010000	S.LOAD= BIT12	: HEADS LOADING
1497	020000	S.RTZ= BIT13	: RETURN TO ZERO
1498	040000	S.UNLD= BIT14	: HEADS UNLOADING

.SBTTL DEFINITION OF DRIVE STATUS BYTE 01 MESSAGE B

1502	000020	S.SECT= BIT4	: SECTOR ERROR
1503	000040	S.WCLK= BIT5	: WRITE CLOCK AND NO WRITE GATE
1504	000100	S.WGAT= BIT6	: WRITE GATE AND NO TRANSISTIONS
1505	000200	S.HDFL= BIT7	: HEAD FAULT
1506	000400	S.MHD= BIT8	: MULTIPLE HEAD SELECT
1507	001000	S.XERR= BIT9	: INDEX ERROR
1508	002000	S.DIB= BIT10	: DIBIT ERROR
1509	004000	S.PLO= BIT11	: PLO ERROR
1510	010000	S.NMOV= BIT12	: SEEK AND NO MOTION
1511	020000	S.LIMD= BIT13	: LIMIT DETECT ON SEEK
1512	040000	S.BRAKE= BIT14	: SERVO-BRAKE

.SBTTL COMMON MASKS

1516	000007	M.DRV= 7	: DRIVE CODE
1517	100000	M.PAR= BIT15	: PARITY
1518	000003	M.ID= 3	: BYTE ID
1519	017760	M.CDIF= 17760	: CYLINDER DIFFERENCE/OFFSET
1520	017760	M.CADD= 17760	: CYLINDER ADDRESS
1521	077770	M.SER= 77770	: DRIVE SERIAL NUMBER
1522	000760	M.SECT= 760	: SECTOR COUNT
1523	007000	M.HEAD= 7000	: HEAD DECODE

.SBTTL PARAMETER BLOCK ALLOCATION

```

*****
*      1  : COMMAND          ! DRIVE NO.
*      3  : CYLINDER ADDRESS
*      5  : TRACK           ! SECTOR
*      7  : BA16-17, FORMAT, DRV TYPE ! OFFSET
*     11  : BUS ADDRESS (LOW 16 BITS)
*     13  : WORD COUNT (2'S COMPLEMENT)
*     15  : PROGRAM DRIVE STATUS INFORMATION
*     17  : COMMAND AND STATUS REGISTER 1
*     21  : COMMAND AND STATUS REGISTER 2
*     23  : WORD COUNT REGISTER
*     25  : BUS ADDRESS REGISTER
*     27  : DESIRED TRACK AND SECTOR
*     31  : DESIRED CYLINDER
*     33  : ATTENTION SUMMARY AND DRIVE OFFSET
*     35  : ERROR REGISTER
*     37  : STATUS REGISTER
*     41  : MESSAGE LINE A STATUS BYTE 00
*     43  : MESSAGE LINE B STATUS BYTE 00
*     45  : MESSAGE LINE A STATUS BYTE 01
*     47  : MESSAGE LINE B STATUS BYTE 01
*     51  : MESSAGE LINE A STATUS BYTE 10
*     53  : MESSAGE LINE B STATUS BYTE 10
*     55  : MESSAGE LINE A STATUS BYTE 11
*     57  : MESSAGE LINE B STATUS BYTE 11
*     61  : ECC POSITION INFORMATION
*     63  : ECC PATTERN INFORMATION
*****

```

102NT
116
202NT
226
302NT
326
402NT
426
502NT
526
602NT
626

.SBTTL PARAMETERS PASSED TO THE DRIVER

THE FOLLOWING DEFINITIONS ARE USED TO PASS PARAMETERS TO THE RK06/RK07 DRIVER

```

000000 P.DRVN= 0 ;DRIVE NUMBER
000001 P.CMND= 1 ;COMMAND
000002 P.CYLN= 2 ;CYLINDER ADDRESS
000004 P.SECT= 4 ;SECTOR
000005 P.TRCK= 5 ;TRACK
000006 P.OFST= 6 ;OFFSET
000007 P.CS1H= 7 ;RKCS1 BITS 8-15
000007 P.BAHI= 7 ;BUS ADDRESS (BITS 16 AND 17)
000010 P.BALO= 10 ;BUS ADDRESS (BITS 0-15)
000012 P.WC= 12 ;WORD COUNT (2'S COMPLEMENT)
000014 P.PRST= 14 ;PROGRAM DRIVE STATUS INFORMATION

```

.SBTTL PROGRAM DEVICE STATUS REGISTER DEFINITION

```

000001 DRVUSE= BIT0 ;DRIVE IN USE
000002 DRVPOS= BIT1 ;DRIVE POSITIONING
000004 DRVPDT= BIT2 ;DRIVE POSITIONED FOR DATA TRANSFER
000010 UEXATT= BIT3 ;UNEXPECTED ATTENTION
000020 DRVHRD= BIT4 ;DRIVE HAS HARD ERROR
000040 DRVDSC= BITS ;DRIVE STATUS CHANGE DID NOT CLEAR

```

1524
1525
1526
1527
1528
1529
1530
1531
1532
1533
1534
1535
1536
1537
1538
1539
1540
1541
1542
1543
1544
1545
1546
1547
1548
1549
1550
1551
1552
1553
1554
1555
1556
1557
1558
1559
1560
1561
1562
1563
1564
1565
1566
1567
1568
1569
1570
1571
1572
1573
1574
1575
1576
1577
1578
1579

```

1580          000100          CMDTO= BIT6          ; NO TERMINATION TO COMMAND FOR AT
1581          ;              ; LEAST 1 SECOND
1582          000200          W.WCK= BIT7          ; WRITE FOR WRITE WRITE CHECK
1583          000400          NOCHK= BIT8          ; NO CHECK, DO NOT SET INTERRUPT ENABLE
1584          001000          PBSVAL= BIT9          ; PARAMETER STATUS WORDS VALID
1585          ;              ; (SET WHEN ERROR TERMINATION OR
1586          ;              ; READ STATUS COMMAND)
1587          002000          DRPDRV= BIT10         ; DROP DRIVE FROM TEST SEQUENCE
1588          004000          NODSC= BIT11         ; ATTENTION SET BUT DCS AND FAULT RESET
1589          010000          DRVSZD= BIT12         ; DRIVE SEIZED BY OTHER PORT
1590          020000          E.UNLD= BIT13         ; DRIVE UNLOADED DUE TO ERROR
1591          040000          Q.INIT= BIT14         ; PARAMETER BLOCK ENQUEUED IN INITIATION QUEUE
1592          100000          DTBAII= BIT15         ; INHIBIT BUS ADDRESS INCREMENT
1593
1594          .SBTTL  PARAMETERS PASSED FROM DRIVER TO PROGRAM
1595
1596          ;              ; THE FOLLOWING DEFINITIONS ARE USED FOR REGISTER RETURNS
1597          ;              ; FROM THE DRIVER TO THE CALLING PROGRAM
1598
1599          000016          P.CS1= 16             ; COMMAND AND STATUS REGISTER 1
1600          000020          P.CS2= 20             ; COMMAND AND STATUS REGISTER 2
1601          000022          P.WCR= 22             ; WORD COUNT REGISTER
1602          000024          P.BAR= 24             ; BUS ADDRESS REGISTER
1603          000026          P.DTS= 26             ; DESIRED TRACK SECTOR REGISTER
1604          000030          P.DCYL= 30            ; DESIRED CYLINDER REGISTER
1605          000032          P.ASOF= 32            ; ATTENTION SUMMARY/OFFSET REGISTER
1606          000034          P.ER= 34              ; ERROR REGISTER
1607          000036          P.OS= 36              ; STATUS REGISTER
1608          000040          P.A00= 40             ; MESSAGE A STATUS BYTE 00
1609          000042          P.B00= 42             ; MESSAGE B STATUS BYTE 00
1610          000044          P.A01= 44             ; MESSAGE A STATUS BYTE 01
1611          000046          P.B01= 46             ; MESSAGE B STATUS BYTE 01
1612          000050          P.A10= 50             ; MESSAGE A STATUS BYTE 10
1613          000052          P.B10= 52             ; MESSAGE B STATUS BYTE 10
1614          000054          P.A11= 54             ; MESSAGE A STATUS BYTE 11
1615          000056          P.B11= 56             ; MESSAGE B STATUS BYTE 11
1616          000060          P.EPOS= 60            ; ECC POSITION INFORMATION
1617          000062          P.EPAT= 62            ; ECC PATTERN INFORMATION
1618
1619          .SBTTL  PARAMETER BLOCK 0 FOR DRIVE
1620
1621          002362          000          PARM0: .BYTE 0          ; DRIVE NUMBER
1622          002363          000          ; .BYTE 0          ; COMMAND
1623          002364          000000        ; .WORD 0          ; CYLINDER ADDRESS
1624          002366          000          ; .BYTE 0          ; SECTOR ADDRESS
1625          002367          000          ; .BYTE 0          ; TRACK ADDRESS
1626          002370          000          ; .BYTE 0          ; OFFSET VALUE
1627          002371          000          ; .BYTE 0          ; BUS ADDRESS (BITS 16 AND 17)
1628          002372          000000        ; .WORD 0          ; BUS ADDRESS (BITS 0 - 15)
1629          002374          000000        ; .WORD 0          ; WORD COUNT (2'S COMPLEMENT)
1630          002376          000000        ; .WORD 0          ; PROGRAM DRIVE STATUS INFORMATION
1631          002400          000000        ; .WORD 0          ; COMMAND AND STATUS REGISTER 1
1632          002402          000000        ; .WORD 0          ; COMMAND AND STATUS REGISTER 2
1633          002404          000000        ; .WORD 0          ; WORD COUNT REGISTER
1634          002406          000000        ; .WORD 0          ; BUS ADDRESS REGISTER
1635          002410          000000        ; .WORD 0          ; DESIRED TRACK AND SECTOR REGISTER
    
```


1636	002412	000000	.WORD	0	: DESIRED CYLINDER REGISTER
1637	002414	000000	.WORD	0	: ATTENTION SUMMARY/OFFSET REGISTER
1638	002416	000000	.WORD	0	: ERROR REGISTER
1639	002420	000000	.WORD	0	: STATUS REGISTER
1640	002422	000000	.WORD	0	: MESSAGE LINE A STATUS BYTE 00
1641	002424	000000	.WORD	0	: MESSAGE LINE B STATUS BYTE 00
1642	002426	000000	.WORD	0	: MESSAGE LINE A STATUS BYTE 01
1643	002430	000000	.WORD	0	: MESSAGE LINE B STATUS BYTE 01
1644	002432	000000	.WORD	0	: MESSAGE LINE A STATUS BYTE 10
1645	002434	000000	.WORD	0	: MESSAGE LINE B STATUS BYTE 10
1646	002436	000000	.WORD	0	: MESSAGE LINE A STATUS BYTE 11
1647	002440	000000	.WORD	0	: MESSAGE LINE B STATUS BYTE 11
1648	002442	000000	.WORD	0	: ECC POSITION INFORMATION
1649	002444	000000	.WORD	0	: ECC PATTERN INFORMATION

1650
1651 .SBTTL PARAMETER BLOCK 1 FOR DRIVE

1652					
1653	002446	000	PARM1: .BYTE	0	: DRIVE NUMBER
1654	002447	000	.BYTE	0	: COMMAND
1655	002450	000000	.WORD	0	: CYLINDER ADDRESS
1656	002452	000	.BYTE	0	: SECTOR ADDRESS
1657	002453	000	.BYTE	0	: TRACK ADDRESS
1658	002454	000	.BYTE	0	: OFFSET VALUE
1659	002455	000	.BYTE	0	: BUS ADDRESS (BITS 16 AND 17)
1660	002456	000000	.WORD	0	: BUS ADDRESS (BITS 0 - 15)
1661	002460	000000	.WORD	0	: WORD COUNT (2'S COMPLEMENT)
1662	002462	000000	.WORD	0	: PROGRAM DRIVE STATUS INFORMATION
1663	002464	000000	.WORD	0	: COMMAND AND STATUS REGISTER 1
1664	002466	000000	.WORD	0	: COMMAND AND STATUS REGISTER 2
1665	002470	000000	.WORD	0	: WORD COUNT REGISTER
1666	002472	000000	.WORD	0	: BUS ADDRESS REGISTER
1667	002474	000000	.WORD	0	: DESIRED TRACK AND SECTOR REGISTER
1668	002476	000000	.WORD	0	: DESIRED CYLINDER REGISTER
1669	002500	000000	.WORD	0	: ATTENTION SUMMARY/OFFSET REGISTER
1670	002502	000000	.WORD	0	: ERROR REGISTER
1671	002504	000000	.WORD	0	: STATUS REGISTER
1672	002506	000000	.WORD	0	: MESSAGE LINE A STATUS BYTE 00
1673	002510	000000	.WORD	0	: MESSAGE LINE B STATUS BYTE 00
1674	002512	000000	.WORD	0	: MESSAGE LINE A STATUS BYTE 01
1675	002514	000000	.WORD	0	: MESSAGE LINE B STATUS BYTE 01
1676	002516	000000	.WORD	0	: MESSAGE LINE A STATUS BYTE 10
1677	002520	000000	.WORD	0	: MESSAGE LINE B STATUS BYTE 10
1678	002522	000000	.WORD	0	: MESSAGE LINE A STATUS BYTE 11
1679	002524	000000	.WORD	0	: MESSAGE LINE B STATUS BYTE 11
1680	002526	000000	.WORD	0	: ECC POSITION INFORMATION
1681	002530	000000	.WORD	0	: ECC PATTERN INFORMATION

1682
1683 .SBTTL TEMPORARY CONTROLLER REGISTER STORAGE

1684					
1685	002532	000000	T.CS1: .WORD	0	: TEMPORARY STORAGE FOR COMMAND AND STATUS REGISTER 1
1686					
1687	002534	000000	T.CS2: .WORD	0	: TEMPORARY STORAGE FOR COMMAND AND STATUS REGISTER 2
1688					
1689	002536	000000	T.WCR: .WORD	0	: TEMPORARY STORAGE FOR WORD COUNT REGISTER
1690	002540	000000	T.BA: .WORD	0	: TEMPORARY STORAGE FOR BUS ADDRESS REGISTER
1691	002542	000000	T.DA: .WORD	0	: TEMPORARY STORAGE FOR DISK TRACK AND SECTOR

TEMPORARY CONTROLLER REGISTER STORAGE

1692 002544 000000
1693 002546 000000
1694
1695 002550 000000
1696 002552 000000
1697 002554 000000
1698 002556 000000
1699 002560 000000
1700 002562 000000
1701 002564 000000
1702 002566 000000
1703
1704
1705
1706 002570 177440
1707 002572 000210
1708 002574 000240
1709 002576 025524
1710 002600 026462
1711 002602 026040
1712 002604 000000
1713
1714
1715
1716
1717
1718 000001
1719 000002
1720 000004
1721 000010
1722 000020
1723 000040
1724
1725 000100
1726 000400
1727 001000
1728 002000
1729 040000
1730 100000
1731
1732 002606 000000
1733
1734 002610 000400
1735 002612 000400
1736
1737
1738
1739
1740
1741
1742
1743
1744
1745
1746
1747

T.DC: .WORD 0
T.ASOF: .WORD 0
T.ER: .WORD 0
T.DS: .WORD 0
T.MR1: .WORD 0
T.MR2: .WORD 0
T.MR3: .WORD 0
T.POS: .WORD 0
T.PAT: .WORD 0
T.DB: .WORD 0

.SBTTL DRIVER PARAMERTERS
RKBAS: .WORD 177440
RKVEC: .WORD 210
RKPRI: .WORD PR5
A.NORM: ERRFRE
A.ABNL: ERRHDL
A.CONT: CONERR
E.CONT: .WORD 0

E.CCLR= BIT0
E.NOAT= BIT1
E.UATT= BIT2
E.UDAT= BIT3
E.CLAT= BIT4
E.SCLR= BIT5
E.ILLD= BIT6
E.DLT= BIT8
E.CERR= BIT9
E.DPAR= BIT10
E.CMTO= BIT14
E.MDS= BIT15

O.WAIT: .WORD 0
W.MTIM: .WORD 400
W.MILI: .WORD 400

: TEMPORARY STORAGE FOR DRIVE CYLINDER
: TEMPORARY STORAGE FOR ATTENTION SUMMARY
: AND OFFSET
: TEMPORARY STORAGE FOR ERROR REGISTER
: TEMPORARY STORAGE FOR DRIVE STATUS REGISTER
: TEMPORARY STORAGE FOR MAINTENANCE REGISTER 1
: TEMPORARY STORAGE FOR MAINTENANCE REGISTER 2
: TEMPORARY STORAGE FOR MAINTENANCE REGISTER 3
: TEMPORARY STORAGE FOR ECC POSITION
: TEMPORARY STORAGE FOR ECC PATTERN
: TEMPORARY STORAGE FOR DATA BUFFER REGISTER

: ADDRESS OF RK611 UNIBUS ADDRESS BLOCK
: ADDRESS OF R611 VECTOR
: RK611 INTERRUPT PRIORITY
: ADDRESS OF NORMAL RETURN FROM DRIVER
: ADDRESS OF ABNORMAL RETURN FROM DRIVER
: ADDRESS OF CONTROLLER ERROR RETURN
: CONTROLLER ERROR STATUS
: THIS LOCATION IS CLEARED WHEN EVERY COMMAND
: IS INITIATED. IF A CONTROLLER ERROR
: OCCURS THE FOLLOWING BIT ASSIGNMENT IS
: USED:

: CLEAR CONTROLLER DID NOT CLEAR ERROR
: NO ATTENTION IN ATTENTION SUMMARY REG
: UNSOLICATED ATTENTION (SEQUENTIAL ONLY)
: UNEXPECTED DATA TYPE ERROR
: ATTENTION DID NOT RESET WITH CLEAR
: SUBSYSTEM CLEAR DID NOT CLEAR DRIVE
: ATTENTION
: ILLEGAL DRIVER COMMAND
: DATA LATE WHEN UNLOADING HEADER
: CONTROLLER ERROR DURING DRIVER SERVICING
: DRIVE DETECTED PARITY ERROR
: CONTROLLER COMMAND TIME OUT (QUEUED ONLY)
: MULTIPLE DRIVE SELECT

: PARAMETER BLOCK OF THE DRIVE
: WAITING FOR COMMAND COMPLETION
: LOOP COUNTER FOR MILLISECOND SCAN OF DRIVE
: 16 MILLISECOND TIME FOR PROGRAM

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

```

1748 002614 000300      W.SEC: .WORD 300      ; SECOND COUNT COUNT FOR ALL COMMANDS
1749                                ; EXCEPT START SPINDLE
1750 002616 003000      W.BSEC: .WORD 3000    ; 8 SECOND FOR DRIVE CYCLE DOWN
1751 002620 030000      W.MIN: .WORD 30000    ; MINUTE TIME FOR START SPINDLE
1752 002622 000000      HDR.AD: .WORD 0       ; ADDRESS USED FOR READ ALL HEADERS
1753 002624 000000      HDR.CT: .WORD 0       ; NUMBER OF HEADERS LEFT TO READ FOR READ
1754                                ; ALL HEADERS
1755 002626 000          I.ISRL: .BYTE 0       ; INTERRUPT OR RELEASED COMMAND ISSUED
1756 002627 002          H.HEAD: .BYTE 2,4,10 ; HEAD DECODES
1757 002632 000          W.TIME: .BYTE 0       ; DRIVES BEING WATCH-DOG TIMED
1758                                ;
1759                                ;.SBTTL INTERRUPT MASKS
1760                                ;
1761 002633 000          INTMSK: .BYTE 0       ; INTERRUPT MASKS FOR DRIVE IN PARAMETER BLOCK
1762                                ;
1763                                ; INTERRUPT MASK TABLE
1764                                ;
1765 002634 001          I.DRV: .BYTE 1       ; INTERRUPT MASK FOR DRIVE 0
1766 002635 002          ;.BYTE 2       ; INTERRUPT MASK FOR DRIVE 1
1767 002636 004          ;.BYTE 4       ; INTERRUPT MASK FOR DRIVE 2
1768 002637 010          ;.BYTE 10      ; INTERRUPT MASK FOR DRIVE 3
1769 002640 020          ;.BYTE 20      ; INTERRUPT MASK FOR DRIVE 4
1770 002641 040          ;.BYTE 40      ; INTERRUPT MASK FOR DRIVE 5
1771 002642 100          ;.BYTE 100     ; INTERRUPT MASK FOR DRIVE 6
1772 002643 200          ;.BYTE 200     ; INTERRUPT MASK FOR DRIVE 7
1773                                ;
1774                                ;.SBTTL PARAMETER BLOCK TABLE
1775                                ;
1776 002644 002362      PBLKT: PARM0       ; ADDRESS OF PARAMETER BLOCK GIVEN WITH
1777                                ; DRIVE CALL. MUST BE LOADED INTO PBLKT
1778                                ;
1779                                ;.SBTTL TIME FOR WATCH-DOG TIMER
1780                                ;
1781                                ;
1782 002646 000000      W.DRV: .WORD 0       ; TIME FOR INSTRUCTION IN PARAMETER BLOCK
1783                                ;.SBTTL PROGRAM SPECIFIC RESERVED LOCATIONS
1784 002650 000          DERCNT: .BYTE 0       ; DATA ERROR COUNT
1785 002651 000          OPCOMP: .BYTE 0       ; OPERATION COMPLETE FLAG
1786 002652 000          DONE: .BYTE 0        ; DONE SWITCH
1787 002653 000          TYPFMT: .BYTE 0       ; DRIVE TYPE & FORMAT CONTROL
1788 002654 000          ERRCNT: .BYTE 0       ; ERROR COUNT
1789 002655 004          ERRLMT: .BYTE 4       ; ERROR LIMIT
1790 002656 000          OPCONT: .BYTE 0       ; OPERATION CONTROL SWITCHES
1791                                ; WHDSW=BIT0 ; WRITE HEADER & DATA SWITCH
1792                                ; VFHDSW=BIT1 ; VERIFY HEADERS SWITCH
1793                                ; WCDASW=BIT2 ; WRITE CHECK DATA SWITCH
1794                                ; RCDASW=BIT3 ; READ CHECK DATA SWITCH
1795                                ; OREQSW=BIT4 ; OFFSET REQUIRED SWITCH
1796                                ; PSDBSF= BITS ; SAVE SOFTWARE DETECTED BAD SECTOR FILES SWITCH
1797                                ; .EVEN
1798                                ;
1799 002660 000400      RDBUF: .BLKW †D256    ; READ BUFFER
1800 003660 000400      BSSOFT: .BLKW †D256    ; RECORD OF BAD SECTORS FROM SOFTWARE
1801 004660 000400      BSFACT: .BLKW †D256    ; RECORD OF BAD SECTORS FROM FACTORY
1802 005660 000006      COMSTR: .BLKW 6       ; COMMAND STORAGE
1803 005674 000102      BUFFO: .BLKW †D66     ; OUTPUT BUFFER 1

```

1804	006100	000102		
1805	006304	000000		
1806	006306	000000		
1807	006310	000006		
1808	006312	000000		
1809	006314	000026		
1810	006316	000002		
1811	006320	135143		
1812	006322	072307		
1813	006324	000000		
1814	006326	000002		
1815	006330	000000		
1816	006332	000000		
1817	006334	000000		
1818	006336	000000		
1819	006340	000000		
1820	006342	000000		
1821	006344	000000		
1822	006346	000000		
1823	006350	000000		
1824	006352	000000		
1825	006354	000000		
1826	006356	000000		
1827	006360	000000		
1828	006362	000000		
1829	006364	000000		
1830	006366	000000		
1831	006370	000000		
1832	006372	000000		
1833		000002		
1834		000004		
1835		000010		
1836		000020		
1837		000040		
1838		000100		
1839		000200		
1840		000400		
1841		001000		
1842		002000		
1843		004000		
1844		100000		
1845				
1846	006374	037477	000	
1847	006377	015	044012	046105
1848	006404	020120	044506	042514
1849	006412	005015	012	
1850	006415	115	052517	052116
1851	006422	040440	020116	045522
1852	006430	033060	026513	045522
1853	006436	033460	020113	044504
1854	006444	045523	041440	051101
1855	006452	051124	042111	042507
1856	006460	005015		
1857	006462	042522	042523	020124
1858	006470	051127	052111	020105
1859	006476	047514	045503	044440

```

BUFF1: .BLKW      ↑D66
BUFPR: .WORD      0
TKWDCT: .WORD     0
INITTP: .WORD     6
INITDR: .WORD     0
INITSE: .WORD    26
INITMD: .WORD     2
INITWE: .WORD    135143
INITWO: .WORD    072307
INITST: .WORD     0
INITET: .WORD     2
INITSC: .WORD     0
INITEC: .WORD     0
INITOF: .WORD     0
TPINUS: .WORD     0
DRINUS: .WORD     0
SEINUS: .WORD     0
MDINUS: .WORD     0
WEINUS: .WORD     0
WDINUS: .WORD     0
STINUS: .WORD     0
ETINUS: .WORD     0
SCINUS: .WORD     0
ECINUS: .WORD     0
OFINUS: .WORD     0
RECODE: .WORD     0
CTINUS: .WORD     0
CCINUS: .WORD     0
ERRCOM: .WORD     0
BSERR=BIT1
HVCER=BIT2
OPIERR=BIT3
DCKERR=BIT4
ECCNC=BIT5
WCERR=BIT6
ABORT=BIT7
LEV2ER=BIT8
BADSEC=BIT9
TWOLOS=BIT10
RCLREQ=BIT11
ANYDER=BIT15

```

```

;OUTPUT BUFFER 2
;BUFFER POINTER
;FULL TRACK WORD COUNT STORAGE
;DEFAULT DRIVE TYPE TO RK06
;DEFAULT DRIVE
;DEFAULT SECTOR/TRACK
;DEFAULT MODE
;DEFAULT DATA FOR EVEN CYL
;DEFAULT DATA FOR ODD CYL
;DEFAULT STARTING TRACK
;DEFAULT ENDING TRACK
;DEFAULT STARTING CYLINDER
;DEFAULT END CYLINDER, LOADED IN PARM2
;DEFAULT OFFSET (+0)
;DRIVE TYPE IN USE
;DRIVE IN USE
;SECTOR/TRACK IN USE
;MODE IN USE
;DATA IN USE FOR EVEN CYL
;DATA IN USE FOR ODD CYL
;STARTING TRACK IN USE
;ENDING TRACK IN USE
;STARTING CYLINDER IN USE
;END CYLINDER IN USE
;OFFSET VALUE IN USE
;RECOVERY CODE WORD
;CURRENT TRACK IN USE
;CURRENT CYLINDER IN USE
;ERROR COMMAND
;BSE ERROR
;HVC ERROR
;OPI ERROR
;DATA CHECK ERROR
;ECC NON-CORRECTABLE
;WRITE CHECK ERROR
;ABORT
;LEVEL TWO ERROR
;BAD SECTOR FLAG
;TWO TIME OUTS
;RECALIBRATE REQUIRED
;ANY ERROR DETECTED FLAG

```

```

QUES: .ASCIZ  /??/
HELPFL: .ASCII <15><12>/HELP FILE/<15><12><12>

.ASCII /MOUNT AN RK06K-RK07K DISK CARTRIDGE/<15><12>

.ASCII /RESET WRITE LOCK IF WRITE HEADERS OR DATA OPERATION/<15><12>

```

1860	006504	020106	051127	052111
1861	006512	020105	042510	042101
1862	006520	051105	020123	051117
1863	006526	042040	052101	020101
1864	006534	050117	051105	052101
1865	006542	047511	006516	012
1866	006547	122	051505	047520
1867	006554	042116	052040	020117
1868	006562	044124	020105	047506
1869	006570	046114	053517	047111
1870	006576	020107	040520	040522
1871	006604	042515	042524	020122
1872	006612	042522	052521	051505
1873	006620	051524	041040	035131
1874	006626	005015	047503	052116
1875	006634	047522	020114	020132
1876	006642	057050	024532	024040
1877	006650	051103	020051	047524
1878	006656	040440	046114	053517
1879	006664	040440	046114	047440
1880	006672	020122		
1881	006674	042522	040515	047111
1882	006702	047111	020107	040520
1883	006710	040522	042515	042524
1884	006716	051522	052040	020117
1885	006724	042504	040506	046125
1886	006732	020124	043101	042524
1887	006740	020122	051104	053111
1888	006746	020105	054524	042520
1889	006754	005015		
1890	006756	040503	051122	040511
1891	006764	042507	051040	052105
1892	006772	051125	020116	047524
1893	007000	042040	043105	052501
1894	007006	052114	052040	040510
1895	007014	020124	050123	041505
1896	007022	043111	041511	050040
1897	007030	052101	046501	052105
1898	007036	051105	005015	
1899	007042	047503	052116	047522
1900	007050	020114	020103	057050
1901	007056	024503	052040	020117
1902	007064	042522	052524	047122
1903	007072	052040	020117	044506
1904	007100	051522	020124	040520
1905	007106	040522	042515	042524
1906	007114	020122	042522	052521
1907	007122	051505	006524	012
1908	007127	103	047117	051124
1909	007134	046117	041440	024040
1910	007142	041536	020051	046101
1911	007150	047523	040440	047502
1912	007156	052122	020123	044124
1913	007164	020105	051120	043517
1914	007172	040522	020115	047117
1915	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 /

1916	007203	106	051117	040515	.ASCII /FORMATTING HAS STARTED/<15><12><12>
1917	007210	052124	047111	020107	
1918	007216	040510	020123	052123	
1919	007224	051101	042524	006504	
1920	007232	005012			
1921	007234	051117	005015	012	.ASCII /OR/<15><12><12>
1922	007241	105	052116	051105	.ASCII /ENTER THE DESIRED PARAMETER OPTION SELECTED /
1923	007246	052040	042510	042040	
1924	007254	051505	051111	042105	
1925	007262	050040	051101	046501	
1926	007270	052105	051105	047440	
1927	007276	052120	047511	020116	
1928	007304	042523	042514	052103	
1929	007312	042105	040		
1930	007315	106	047522	020115	.ASCII /FROM THE LIST BELOW./<15><12>
1931	007322	044124	020105	044514	
1932	007330	052123	041040	046105	
1933	007336	053517	006456	012	
1934	007343	101	046114	053040	.ASCII /ALL VALUES TO BE ENTERED ARE OCTAL/<15><12><12>
1935	007350	046101	042525	020123	
1936	007356	047524	041040	020105	
1937	007364	047105	042524	042522	
1938	007372	020104	051101	020105	
1939	007400	041517	040524	006514	
1940	007406	005012			
1941	007410	051104	053111	020105	.ASCII /DRIVE TYPE= 6 FOR RK06, 7 FOR RK07/<15><12>
1942	007416	054524	042520	020075	
1943	007424	020066	047506	020122	
1944	007432	045522	033060	020054	
1945	007440	020067	047506	020122	
1946	007446	045522	033460	005015	
1947	007454	051104	053111	020105	.ASCII /DRIVE NUM=(0-7) DEFAULTS TO 0/<15><12><12>
1948	007462	052516	036515	030050	
1949	007470	033455	020051	042504	
1950	007476	040506	046125	051524	
1951	007504	052040	020117	006460	
1952	007512	005012			
1953	007514	042523	052103	051117	.ASCII &SECTORS/TRACK=(24,26) DEFAULTS TO 26&<15><12><12>
1954	007522	027523	051124	041501	
1955	007530	036513	031050	026064	
1956	007536	033062	020051	042504	
1957	007544	040506	046125	051524	
1958	007552	052040	020117	033062	
1959	007560	005015	012		
1960	007563	115	042117	036505	.ASCII /MODE=(0-4) DEFAULTS TO 2/<15><12>
1961	007570	030050	032055	020051	
1962	007576	042504	040506	046125	
1963	007604	051524	052040	020117	
1964	007612	006462	012		
1965	007615	040	020040	030040	.ASCII / 0=WRITE HEADERS & DATA/<15><12>
1966	007622	053475	044522	042524	
1967	007630	044040	040505	042504	
1968	007636	051522	023040	042040	
1969	007644	052101	006501	012	
1970	007651	040	020040	030440	.ASCII / 1=WRITE HEADERS & DATA; VERIFY HEADERS/<15><12>
1971	007656	053475	044522	042524	

M03

CZR6LCO RK06L CTG FMTR MACY11 30(1046)
 CZR6LC.P11 01-DEC-77 14:20

01-DEC-77 14:28 PAGE 39
 PROGRAM SPECIFIC RESERVED LOCATIONS

SEQ 0038

1972	007664	044040	040505	042504
1973	007672	051522	023040	042040
1974	007700	052101	035501	053040
1975	007706	051105	043111	020131
1976	007714	042510	042101	051105
1977	007722	006523	012	
1978	007725	040	020040	031040
1979	007732	053475	044522	042524
1980	007740	023040	053040	051105
1981	007746	043111	020131	042510
1982	007754	042101	051105	020123
1983	007762	020046	040504	040524
1984	007770	005015		
1985	007772	020040	020040	036463
1986	010000	042526	044522	054506
1987	010006	044040	040505	042504
1988	010014	051522	005015	
1989	010020	020040	020040	036464
1990	010026	042526	044522	054506
1991	010034	044040	040505	042504
1992	010042	051522	040440	042116
1993	010050	053040	051105	043111
1994	010056	020131	040504	040524
1995	010064	005015	012	
1996	010067	105	042526	020116
1997	010074	054503	044514	042116
1998	010102	051105	050040	052101
1999	010110	042524	047122	024075
2000	010116	040504	040524	053440
2001	010124	051117	024504	042040
2002	010132	043105	052501	052114
2003	010140	020123	047524	030440
2004	010146	032463	032061	006463
2005	010154	012		
2006	010155	117	042104	041440
2007	010162	046131	047111	042504
2008	010170	020122	040520	052124
2009	010176	051105	036516	042050
2010	010204	052101	020101	047527
2011	010212	042122	020051	042504
2012	010220	040506	046125	051524
2013	010226	052040	020117	033460
2014	010234	031462	033460	005015
2015	010242	051124	041501	020113
2016	010250	044514	044515	051524
2017	010256	036440	024040	026460
2018	010264	026062	026460	024462
2019	010272	042040	043105	052501
2020	010300	052114	020123	047524
2021	010306	030040	031054	005015
2022	010314	054503	044514	042116
2023	010322	051105	046040	046511
2024	010330	052111	020123	020075
2025	010336	030050	033055	031063
2026	010344	030054	033055	031063
2027	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>

N03

CZR6LCO RK06L CTG FMTR MACY11 30(1046)
CZR6LC.P11 01-DEC-77 14:20

01-DEC-77 14:28 PAGE 40
PROGRAM SPECIFIC RESERVED LOCATIONS

SEQ 0039

2028	010360	046125	051524	052040	
2029	010366	020117	026060	031466	
2030	010374	006462	005012		
2031	010400	043117	051506	052105	.ASCII /OFFSET=(OCTAL OFFSET VALUE) DEFAULTS TO 0 (0 OFFSET)/<15><12>
2032	010406	024075	041517	040524	
2033	010414	020114	043117	051506	
2034	010422	052105	053040	046101	
2035	010430	042525	020051	042504	
2036	010436	040506	046125	051524	
2037	010444	052040	020117	020060	
2038	010452	030050	047440	043106	
2039	010460	042523	024524	005015	
2040	010466	020040	020040	020040	.ASCII / 000 AND 200 = BOTH ARE ZERO OFFSET/<15><12>
2041	010474	030060	020060	047101	
2042	010502	020104	030062	020060	
2043	010510	020075	047502	044124	
2044	010516	040440	042522	055040	
2045	010524	051105	020117	043117	
2046	010532	051506	052105	005015	
2047	010540	020040	020040	020040	.ASCII / 260 = -1200 MICRO INCHES/<15><12>
2048	010546	033062	020060	020075	
2049	010554	030455	030062	020060	
2050	010562	044515	051103	020117	
2051	010570	047111	044103	051505	
2052	010576	005015			
2053	010600	020040	020040	020040	.ASCII / 060 = +1200 MICRO INCHES/<15><12>
2054	010606	033060	020060	020075	
2055	010614	030453	030062	020060	
2056	010622	044515	051103	020117	
2057	010630	047111	044103	051505	
2058	010636	005015			
2059	010640	020040	020040	020040	.ASCII / EACH COUNT IS AN OFFSET INCREMENT OF 25 MICRO INCHES/<15><12><12>
2060	010646	040505	044103	041440	
2061	010654	052517	052116	044440	
2062	010662	020123	047101	047440	
2063	010670	043106	042523	020124	
2064	010676	047111	051103	046505	
2065	010704	047105	020124	043117	
2066	010712	031040	020065	044515	
2067	010720	051103	020117	047111	
2068	010726	044103	051505	005015	
2069	010734	012			
2070	010735	101	054516	051440	.ASCII /ANY SECTORS TO BE FLAGGED BAD? (TYPE Y OR N)/<15><12>
2071	010742	041505	047524	051522	
2072	010750	052040	020117	042502	
2073	010756	043040	040514	043507	
2074	010764	042105	041040	042101	
2075	010772	020077	052050	050131	
2076	011000	020105	020131	051117	
2077	011006	047040	006451	012	
2078	011013	040	020040	020040	.ASCII / IF N IS TYPED, THE FORMAT OPERATION IS INITIALIZED/<15><12>
2079	011020	044440	020106	020116	
2080	011026	051511	052040	050131	
2081	011034	042105	020054	044124	
2082	011042	020105	047506	046522	
2083	011050	052101	047440	042520	

2084	011056	040522	044524	047117		
2085	011064	044440	020123	047111		
2086	011072	052111	040511	044514		
2087	011100	042532	006504	012		
2088	011105	040	020040	020040	.ASCII /	AND ONLY THOSE SECTORS FOUND BAD BY THE FORMAT/<15><12>
2089	011112	040440	042116	047440		
2090	011120	046116	020131	044124		
2091	011126	051517	020105	042523		
2092	011134	052103	051117	020123		
2093	011142	047506	047125	020104		
2094	011150	040502	020104	054502		
2095	011156	052040	042510	043040		
2096	011164	051117	040515	006524		
2097	011172	012				
2098	011173	040	020040	020040	.ASCII /	PROGRAM AND THOSE LISTED IN THE FACTORY TABLE/<15><12>
2099	011200	050040	047522	051107		
2100	011206	046501	040440	042116		
2101	011214	052040	047510	042523		
2102	011222	046040	051511	042524		
2103	011230	020104	047111	052040		
2104	011236	042510	043040	041501		
2105	011244	047524	054522	052040		
2106	011252	041101	042514	005015		
2107	011260	020040	020040	020040	.ASCII /	ARE FLAGED BAD. IF Y IS TYPED, THE USER HAS THE/<15><12>
2108	011266	051101	020105	046106		
2109	011274	043501	042105	041040		
2110	011302	042101	020056	043111		
2111	011310	054440	044440	020123		
2112	011316	054524	042520	026104		
2113	011324	052040	042510	052440		
2114	011332	042523	020122	040510		
2115	011340	020123	044124	006505		
2116	011346	012				
2117	011347	040	020040	020040	.ASCII /	CAPABILITY OF INSTRUCTING THE FORMATTER TO FLAG/<15><12>
2118	011354	041440	050101	041101		
2119	011362	046111	052111	020131		
2120	011370	043117	044440	051516		
2121	011376	051124	041525	044524		
2122	011404	043516	052040	042510		
2123	011412	043040	051117	040515		
2124	011420	052124	051105	052040		
2125	011426	020117	046106	043501		
2126	011434	005015				
2127	011436	020040	020040	020040	.ASCII /	SPECIFIED SECTORS AS BAD. THE PROGRAM WILL/<15><12>
2128	011444	050123	041505	043111		
2129	011452	042511	020104	042523		
2130	011460	052103	051117	020123		
2131	011466	051501	041040	042101		
2132	011474	020056	044124	020105		
2133	011502	051120	043517	040522		
2134	011510	020115	044527	046114		
2135	011516	005015				
2136	011520	020040	020040	020040	.ASCII /	DIRECT THE USER IN HOW THIS IS DONE./<15><12><12>
2137	011526	044504	042522	052103		
2138	011534	052040	042510	052440		
2139	011542	042523	020122	047111		

2140	011550	044040	053517	052040	
2141	011556	044510	020123	051511	
2142	011564	042040	047117	027105	
2143	011572	005015	012		
2144	011575	0120	042522	042523	.ASCII /PRESERVE SOFTWARE BAD SECTOR FILE?(TYPE Y OR N)(CR)/<15><12>
2145	011602	053122	020105	047523	
2146	011610	052106	040527	042522	
2147	011616	041040	042101	051440	
2148	011624	041505	047524	020122	
2149	011632	044506	042514	024077	
2150	011640	054524	042520	054440	
2151	011646	047440	020122	024516	
2152	011654	041450	024522	005015	
2153	011662	044411	020106	020131	.ASCII / IF Y IS TYPED, THE FORMATTER WILL PRESERVE THE/<15><12>
2154	011670	051511	052040	050131	
2155	011676	042105	020054	044124	
2156	011704	020105	047506	046522	
2157	011712	052101	042524	020122	
2158	011720	044527	046114	050040	
2159	011726	042522	042523	053122	
2160	011734	020105	044124	006505	
2161	011742	012			
2162	011743	011	050101	046120	.ASCII / APPLICABLE BAD SECTOR FILE, MARKING THE SECTORS/<15><12>
2163	011750	041511	041101	042514	
2164	011756	041040	042101	051440	
2165	011764	041505	047524	020122	
2166	011772	044506	042514	020054	
2167	012000	040515	045522	047111	
2168	012006	020107	044124	020105	
2169	012014	042523	052103	051117	
2170	012022	006523	012		
2171	012025	011	040502	020104	.ASCII / BAD THAT ARE LISTED IN THE FILE. ANY ADDITIONAL/<15><12>
2172	012032	044124	052101	040440	
2173	012040	042522	046040	051511	
2174	012046	042524	020104	047111	
2175	012054	052040	042510	043040	
2176	012062	046111	027105	040440	
2177	012070	054516	040440	042104	
2178	012076	052111	047511	040516	
2179	012104	006514	012		
2180	012107	011	042523	052103	.ASCII / SECTORS FOUND BAD BY THE PROGRAM ARE APPENDED/<15><12>
2181	012114	051117	020123	047506	
2182	012122	047125	020104	040502	
2183	012130	020104	054502	052040	
2184	012136	042510	050040	047522	
2185	012144	051107	046501	040440	
2186	012152	042522	040440	050120	
2187	012160	047105	042504	006504	
2188	012166	012			
2189	012167	011	047524	052040	.ASCII / TO THE FILE. IF N IS TYPED THE BAD SECTOR FILE/<15><12>
2190	012174	042510	043040	046111	
2191	012202	027105	044440	020106	
2192	012210	020116	051511	052040	
2193	012216	050131	042105	052040	
2194	012224	042510	041040	042101	
2195	012232	051440	041505	047524	

2196	012240	020122	044506	042514	
2197	012246	005015			
2198	012250	044411	020123	046103	.ASCII / IS CLEARED BEFORE THE PROGRAM BEGINS./<15><12><12>
2199	012256	040505	042522	020104	
2200	012264	042502	047506	042522	
2201	012272	052040	042510	050040	
2202	012300	047522	051107	046501	
2203	012306	041040	043505	047111	
2204	012314	027123	005015	012	
2205	012321	123	052105	051440	.ASCII /SET SWITCH 7 TO PRINT A LIST OF ALL BAD SECTORS./<15><12><12>
2206	012326	044527	041524	020110	
2207	012334	020067	047524	050040	
2208	012342	044522	052116	040440	
2209	012350	046040	051511	020124	
2210	012356	043117	040440	046114	
2211	012364	041040	042101	051440	
2212	012372	041505	047524	051522	
2213	012400	006456	005012		
2214	012404	042523	020124	053523	.ASCII /SET SWITCH 1 TO FORCE REPORTING OF ALL DATA IN ERROR/<15><12><12>
2215	012412	052111	044103	030440	
2216	012420	052040	020117	047506	
2217	012426	041522	020105	042522	
2218	012434	047520	052122	047111	
2219	012442	020107	043117	040440	
2220	012450	046114	042040	052101	
2221	012456	020101	047111	042440	
2222	012464	051122	051117	005015	
2223	012472	012			
2224	012479	123	052105	051440	.ASCII /SET SWITCH 0 TO REPEAT THE ENTIRE FORMAT OPERATION/<15><12><12>
2225	012487	044527	041524	020110	
2226	012495	020060	047524	051040	
2227	012503	050105	040505	020124	
2228	012511	044124	020105	047105	
2229	012519	044524	042522	043040	
2230	012527	051117	040515	020124	
2231	012535	050117	051105	052101	
2232	012543	047511	006516	005012	
2233	012551	043111	052040	042510	.ASCII /IF THE SOFTWARE SWITCH REGISTER IS IN USE, /<15><12>
2234	012559	051440	043117	053524	
2235	012567	051101	020105	053523	
2236	012575	052111	044103	051040	
2237	012583	043505	051511	042524	
2238	012591	020122	051511	044440	
2239	012599	020116	051525	026105	
2240	012607	006440	012		
2241	012615	124	050131	047111	.ASCII /TYPING CONTROL G (↑G) CAUSES THE PROGRAM/<15><12>
2242	012623	020107	047503	052116	
2243	012631	047522	020114	020107	
2244	012639	057050	024507	041440	
2245	012647	052501	042523	020123	
2246	012655	044124	020105	051120	
2247	012663	043517	040522	006515	
2248	012671	012			
2249	012679	124	020117	042522	.ASCII /TO REQUEST NEW SWR SETTING/<15><12><12>
2250	012687	052521	051505	020124	
2251	012695	042516	020127	053523	

2252	012730	020122	042523	052124	
2253	012736	047111	006507	005012	
2254	012744	047105	020104	042510	.ASCIZ /END HELP FILE/<15><12><12>
2255	012752	050114	043040	046111	
2256	012760	006505	005012	000	
2257	012765	040	000040		SPACE2: .ASCIZ / /
2258					
2259	012770	005015	025052	051040	PROGID: .ASCII <15><12>/** RK06K-RK07K CARTRIDGE FORMATER **/<15><12>
2260	012776	030113	045466	051055	
2261	013004	030113	045467	041440	
2262	013012	051101	051124	042111	
2263	013020	042507	043040	051117	
2264	013026	040515	042524	020122	
2265	013034	025052	005015		
2266	013040	055103	033122	041514	.ASCIZ /CZR6LCO/<15><12>
2267	013046	006460	000012		
2268	013052	054524	042520	044040	HELPQ: .ASCIZ /TYPE HELP FOR OPERATING INFO, ELSE CR/<15><12>
2269	013060	046105	020120	047506	
2270	013066	020122	050117	051105	
2271	013074	052101	047111	020107	
2272	013102	047111	047506	020054	
2273	013110	046105	042523	041440	
2274	013116	006522	000012		
2275	013122	051104	053111	020105	TPQ: .ASCIZ /DRIVE TYPE(6 OR 7)/
2276	013130	054524	042520	033050	
2277	013136	047440	020122	024467	
2278	013144	000			
2279	013145	104	044522	042526	DRVQ: .ASCIZ /DRIVE NUM=/ SECTQ: .ASCIZ &SECTOR/TRACK=&
2280	013152	047040	046525	000075	
2281	013160	042523	052103	051117	
2282	013166	052057	040522	045503	
2283	013174	000075			
2284	013176	047515	042504	000075	MODEQ: .ASCIZ /MODE=/ WDEQ: .ASCIZ /EVEN CYLINDER PATTERN=/ WDDQ: .ASCIZ /ODD CYLINDER PATTERN=/ TRKLIM: .ASCIZ /TRACK LIMITS=/ CYLLIM: .ASCIZ /CYLINDER LIMITS=/ OFSETQ: .ASCIZ /OFFSET=/ PSDQ: .ASCIZ /PRESERVE SOFTWARE BAD SECTOR FILES?(TYPE Y OR N)(CR)/
2285	013204	053105	047105	041440	
2286	013212	046131	047111	042504	
2287	013220	020122	040520	052124	
2288	013226	051105	036516	000	
2289	013233	117	042104	041440	
2290	013240	046131	047111	042504	
2291	013246	020122	040520	052124	
2292	013254	051105	036516	000	
2293	013261	124	040522	045503	
2294	013266	046040	046511	052111	
2295	013274	036523	000		
2296	013277	103	046131	047111	
2297	013304	042504	020122	044514	
2298	013312	044515	051524	000075	
2299	013320	043117	051506	052105	
2300	013326	000075			
2301	013330	051120	051505	051105	
2302	013336	042526	051440	043117	
2303	013344	053524	051101	020105	
2304	013352	040502	020104	042523	
2305	013360	052103	051117	043040	
2306	013366	046111	051505	024077	
2307	013374	054524	042520	054440	

2308	013402	047440	020122	024516	
2309	013410	041450	024522	000	
2310	013415	123	043117	053524	SDBSFP: .ASCIZ /SOFTWARE DETECTED BAD SECTOR FILES /
2311	013422	051101	020105	042504	
2312	013430	042524	052103	042105	
2313	013436	041040	042101	051440	
2314	013444	041505	047524	020122	
2315	013452	044506	042514	020123	
2316	013460	000			
2317	013461	120	042522	042523	PRSD: .ASCIZ /PRESERVED/<15><12>
2318	013466	053122	042105	005015	
2319	013474	000			
2320	013475	116	052117	050040	NPRSD: .ASCIZ /NOT PRESERVED/<15><12>
2321	013502	042522	042523	053122	
2322	013510	042105	005015	000	
2323	013515	101	054516	051440	UBSQ: .ASCIZ /ANY SECTORS TO BE FLAGED BAD? (TYPE Y OR N)(CR)/
2324	013522	041505	047524	051522	
2325	013530	052040	020117	042502	
2326	013536	043040	040514	042507	
2327	013544	020104	040502	037504	
2328	013552	024040	054524	042520	
2329	013560	054440	047440	020122	
2330	013566	024516	041450	024522	
2331	013574	000			
2332	013575	105	052116	051105	UBSFMT: .ASCIZ /ENTER OCTAL ADDRESS TO BE FLAGED BAD IN FORMAT CCC,T,SS/<15><12>
2333	013602	047440	052103	046101	
2334	013610	040440	042104	042522	
2335	013616	051523	052040	020117	
2336	013624	042502	043040	040514	
2337	013632	042507	020104	040502	
2338	013640	020104	047111	043040	
2339	013646	051117	040515	020124	
2340	013654	041503	026103	026124	
2341	013662	051523	005015	000	
2342	013667	105	052116	051105	NEXTQ: .ASCIZ /ENTER ADDRESS OR (CR) TO TERMINATE/<15><12>
2343	013674	040440	042104	042522	
2344	013702	051523	047440	020122	
2345	013710	041450	024522	052040	
2346	013716	020117	042524	046522	
2347	013724	047111	052101	006505	
2348	013732	000012			
2349	013734	047111	040526	044514	BADENT: .ASCIZ /INVALID ENTRY/<15><12>
2350	013742	020104	047105	051124	
2351	013750	006531	000012		
2352	013754	047111	042126	051440	ILLSEC: .ASCIZ /INVD SECTOR NUMBER/<15><12>
2353	013762	041505	047524	020122	
2354	013770	052516	041115	051105	
2355	013776	005015	000		
2356	014001	111	053116	020104	INVDSE: .ASCIZ /INVD STARTING OR ENDING PACK ADDRESS/<15><12>
2357	014006	052123	051101	044524	
2358	014014	043516	047440	020122	
2359	014022	047105	044504	043516	
2360	014030	050040	041501	020113	
2361	014036	042101	051104	051505	
2362	014044	006523	000012		
2363	014050	047111	042126	041440	INVDY: .ASCIZ /INVD CYLINDER NUMBER /<15><12>

2364	014056	046131	047111	042504	
2365	014064	020122	052516	041115	
2366	014072	051105	006440	000012	
2367	014100	047111	042126	052040	INVDTK: .ASCIZ /INVD TRACK NUMBER /<15><12>
2368	014106	040522	045503	047040	
2369	014114	046525	042502	020122	
2370	014122	005015	000		
2371	014125	111	053116	020104	INVDDR: .ASCIZ /INVD DRIVE NUMBER/<15><12>
2372	014132	051104	053111	020105	
2373	014140	052516	041115	051105	
2374	014146	005015	000		
2375	014151	111	053116	020104	INVDTP: .ASCIZ /INVD DRIVE TYPE/<15><12>
2376	014156	051104	053111	020105	
2377	014164	054524	042520	005015	
2378	014172	000			
2379	014173	111	053116	020104	INVDMD: .ASCIZ /INVD MODE SELECTION (MAX OF 4)/<15><12>
2380	014200	047515	042504	051440	
2381	014206	046105	041505	044524	
2382	014214	047117	024040	040515	
2383	014222	020130	043117	032040	
2384	014230	006451	000012		
2385	014234	047111	042126	047440	INVD0F: .ASCIZ /INVD OFFSET VALUE/<15><12>
2386	014242	043106	042523	020124	
2387	014250	040526	052514	006505	
2388	014256	000012			
2389	014260	046120	040505	042523	SWLREQ: .ASCIZ /PLEASE SET WRITE LOCK FOR VERIFY OPERATIONS/<15><12>
2390	014266	051440	052105	053440	
2391	014274	044522	042524	046040	
2392	014302	041517	020113	047506	
2393	014310	020122	042526	044522	
2394	014316	054506	047440	042520	
2395	014324	040522	044524	047117	
2396	014332	006523	000012		
2397	014336	046120	040505	042523	RWLREQ: .ASCIZ /PLEASE RESET WRITE LOCK FOR FORMAT OPERATIONS/<15><12>
2398	014344	051040	051505	052105	
2399	014352	053440	044522	042524	
2400	014360	046040	041517	020113	
2401	014366	047506	020122	047506	
2402	014374	046522	052101	047440	
2403	014402	042520	040522	044524	
2404	014410	047117	006523	000012	
2405	014416	051120	051505	020123	CONREQ: .ASCIZ /PRESS CONTINUE WHEN READY/<15><12>
2406	014424	047503	052116	047111	
2407	014432	042525	053440	042510	
2408	014440	020116	042522	042101	
2409	014446	006531	000012		
2410	014452	040503	047116	052117	BAD632: .ASCIZ /CANNOT READ BAD SECTOR TRACK/<15><12>
2411	014460	051040	040505	020104	
2412	014466	040502	020104	042523	
2413	014474	052103	051117	052040	
2414	014502	040522	045503	005015	
2415	014510	000			
2416	014511	123	043117	053524	TDMSOF: .ASCIZ /SOFTWARE BAD SECTOR FILE NOT FORMATTED PROPERLY OR/<15><12>
2417	014516	051101	020105	040502	
2418	014524	020104	042523	052103	
2419	014532	051117	043040	046111	

2420	014540	020105	047516	020124
2421	014546	047506	046522	052101
2422	014554	042524	020104	051120
2423	014562	050117	051105	054514
2424	014570	047440	006522	000012
2425	014576	044506	042514	044440
2426	014604	020123	047524	020117
2427	014612	040514	043522	020105
2428	014620	047506	020122	044124
2429	014626	020105	051120	043517
2430	014634	040522	020115	020050
2431	014642	030476	030060	042440
2432	014650	052116	044522	051505
2433	014656	006451	000012	
2434	014662	040506	052103	051117
2435	014670	020131	040502	020104
2436	014676	042523	052103	051117
2437	014704	043040	046111	020105
2438	014712	047516	020124	047506
2439	014720	046522	052101	042524
2440	014726	020104	051120	050117
2441	014734	051105	054514	047440
2442	014742	006522	000012	
2443	014746	040503	052122	044522
2444	014754	043504	020105	042523
2445	014762	044522	046101	047040
2446	014770	046522	042502	020122
2447	014776	025052	025052	020052
2448	015004	000040		
2449	015006	040503	052122	044522
2450	015014	043504	020105	047515
2451	015022	047125	042524	020104
2452	015030	051511	040440	020116
2453	015036	046101	043511	046516
2454	015044	047105	020124	040503
2455	015052	052122	044522	043504
2456	015060	027105	005015	
2457	015064	047506	046522	052101
2458	015072	044440	020123	041101
2459	015106	051117	042524	006504
2460	015106	005012	000	
2461	015111	116	020117	040504
2462	015116	040524	041440	042510
2463	015124	045503	020040	047117
2464	015132	051440	041505	047524
2465	015140	020122	044124	052101
2466	015146	053440	046111	020114
2467	015154	047516	020124	040520
2468	015162	051523	053440	044522
2469	015170	042524	041440	042510
2470	015176	045503	005015	000
2471	015203	104	052101	020101
2472	015210	044103	041505	020113
2473	015216	051105	047522	020122
2474	015224	047111	053440	044522
2475	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>

2476	015240	047520	051523	041111	
2477	015246	042514	020441	006441	
2478	015254	000012			
2479	015256	041505	020103	047503	CORABL: .ASCIZ /ECC CORRECTABLE/<15><12>
2480	015264	051122	041505	040524	
2481	015272	046102	006505	000012	
2482	015300	025012	020052	053440	PRGRS1: .ASCIZ <12>/** WRITE OPERATION COMPLETE **/<15><12><12>
2483	015306	044522	042524	047440	
2484	015314	042520	040522	044524	
2485	015322	047117	041440	046517	
2486	015330	046120	052105	020105	
2487	015336	025040	006452	005012	
2488	015344	000			
2489	015345	012	025052	020040	PRGRS2: .ASCIZ <12>/** VERIFY OPERATION COMPLETE **/<15><12><12>
2490	015352	042526	044522	054506	
2491	015360	047440	042520	040522	
2492	015366	044524	047117	041440	
2493	015374	046517	046120	052105	
2494	015402	020105	025040	006452	
2495	015410	005012	000		
2496	015413	040	025040	025052	STARS: .ASCIZ / *****/<15><12>
2497	015420	025052	005015	000	
2498	015425	012	051105	047522	BSWERR: .ASCIZ <12>/ERROR WRITING SOFTWARE DETECTED BAD SECTOR TABLES./<15><12>
2499	015432	020122	051127	052111	
2500	015440	047111	020107	047523	
2501	015446	052106	040527	042522	
2502	015454	042040	052105	041505	
2503	015462	042524	020104	040502	
2504	015470	020104	042523	052103	
2505	015476	051117	052040	041101	
2506	015504	042514	027123	005015	
2507	015512	000			
2508	015513	123	041505	047524	SECINE: .ASCIZ /SECTOR IN ERROR =/
2509	015520	020122	047111	042440	
2510	015526	051122	051117	036440	
2511	015534	000			
2512	015535	012	040502	020104	BSWTFI: .ASCIZ <12>/BAD SECTOR TABLE WRITTEN./<15><12>
2513	015542	042523	052103	051117	
2514	015550	052040	041101	042514	
2515	015556	053440	044522	052124	
2516	015564	047105	006456	000012	
2517	015572	052012	042510	041040	BSHEAD: .ASCIZ <12>/THE BAD SECTORS FOR THIS CARTRIDGE IN /
2518	015600	042101	051440	041505	
2519	015606	047524	051522	043040	
2520	015614	051117	052040	044510	
2521	015622	020123	040503	052122	
2522	015630	044522	043504	020105	
2523	015636	047111	000040		
2524	015642	051440	041505	047524	BSTAIL: .ASCIZ / SECTOR FORMAT ARE:/<15><12>
2525	015650	020122	047506	046522	
2526	015656	052101	040440	042522	
2527	015664	006472	000012		
2528	015670	041012	042101	051440	FBSLAB: .ASCIZ <12>/BAD SECTORS IN FACTORY TABLE:/<15><12>
2529	015676	041505	047524	051522	
2530	015704	044440	020116	040506	
2531	015712	052103	051117	020131	

PROGRAM SPECIFIC RESERVED LOCATIONS

2532	015720	040524	046102	035105	
2533	015726	005015	000		
2534	015731	015	041012	042101	SBSLAB: .ASCIZ <15><12>/BAD SECTORS IN SOFTWARE TABLE:/<15><12>
2535	015736	051440	041505	047524	
2536	015744	051522	044440	020116	
2537	015752	047523	052106	040527	
2538	015760	042522	052040	041101	
2539	015766	042514	006472	000012	
2540	015774	047516	042516	005015	NONE: .ASCIZ /NONE/<15><12>
2541	016002	000			
2542	016003	062	000060		TWENTY: .ASCIZ /20/
2543	016006	031062	000		TWENT2: .ASCIZ /22/
2544	016011	040	020040	041440	COLHD: .ASCIZ / CYLNDR TRACK SECTOR/<15><12>
2545	016016	046131	042116	020122	
2546	016024	052040	040522	045503	
2547	016032	020040	051440	041505	
2548	016040	047524	006522	000012	
2549	016046	020040	020040	000	SPACE4: .ASCIZ / /
2550	016053	040	020040	000	SPACE3: .ASCIZ / /
2551	016057	040	020075	047524	TOTMES: .ASCIZ / = TOTAL NUMBER OF BAD SECTORS/<15><12>
2552	016064	040524	020114	052516	
2553	016072	041115	051105	047440	
2554	016100	020106	040502	020104	
2555	016106	042523	052103	051117	
2556	016114	006523	000012		
2557	016120	005015	044103	047101	XXDPMG: .ASCII <CR><LF>/CHANGE XXDP PACK/
2558	016126	042507	054040	042130	
2559	016134	020120	040520	045503	
2560	016142	005015	046103	040505	.ASCIZ <CR><LF>/CLEAR LOC 40,RESTART PROGRAM/
2561	016150	020122	047514	020103	
2562	016156	030064	051054	051505	
2563	016164	040524	052122	050040	
2564	016172	047522	051107	046501	
2565	016200	000			
2566					
2567		016202			.EVEN
2568	016202				START:
2569					.SBTTL INITIALIZE THE COMMON TAGS
2570					;;CLEAR THE COMMON TAGS (SCMTAG) AREA
2571	016202	012706	001100		MOV #SCMTAG,R6 ;;FIRST LOCATION TO BE CLEARED
2572	016206	005026			CLR (R6)+ ;;CLEAR MEMORY LOCATION
2573	016210	022706	001140		CMP #SWR,R6 ;;DONE?
2574	016214	001374			BNE -6 ;;LOOP BACK IF NO
2575	016216	012706	001100		MOV #STACK,SP ;;SETUP THE STACK POINTER
2576					;;INITIALIZE A FEW VECTORS
2577	016222	012737	040560	000030	MOV #ERROR,@EMTVEC ;;EMT VECTOR FOR ERROR ROUTINE
2578	016230	012737	000340	000032	MOV #340,@EMTVEC+2 ;;LEVEL 7
2579	016236	012737	042610	000034	MOV #STRAP,@TRAPVEC ;;TRAP VECTOR FOR TRAP CALLS
2580	016244	012737	000340	000036	MOV #340,@TRAPVEC+2;LEVEL 7
2581	016252	012737	042240	000024	MOV #SPWRDN,@PWRVEC ;;POWER FAILURE VECTOR
2582	016260	012737	000340	000026	MOV #340,@PWRVEC+2 ;;LEVEL 7
2583	016266	005037	001260		CLR \$ESCAPE ;;CLEAR THE ESCAPE ON ERROR ADDRESS
2584	016272	112737	000001	001115	MOVB #1,\$ERMAX ;;ALLOW ONE ERROR PER TEST
2585					;;SIZE FOR A HARDWARE SWITCH REGISTER. IF NOT FOUND OR IT IS
2586					;;EQUAL TO A "-1" SETUP FOR A SOFTWARE SWITCH REGISTER.
2587	016300	013746	000004		MOV @ERRVEC,-(SP) ;;SAVE ERROR VECTOR

INITIALIZE THE COMMON TAGS

```

2588 016304 012737 016340 000004      MOV      #64$,2#ERRVEC      ;;SET UP ERROR VECTOR
2589 016312 012737 177570 001140      MOV      #DSWR,SWR        ;;SETUP FOR A HARDWARE SWICH REGISTER
2590 016320 012737 177570 001142      MOV      #DDISP,DISPLAY   ;;AND A HARDWARE DISPLAY REGISTER
2591 016326 022777 177777 162604      CMP      #-1,2SWR        ;;TRY TO REFERENCE HARDWARE SWR
2592 016334 001012                BNE      66$              ;;BRANCH IF NO TIMEOUT TRAP OCCURRED
2593                                ;;AND THE HARDWARE SWR IS NOT = -1
2594 016336 000403                BR       65$              ;;BRANCH IF NO TIMEOUT
2595 016340 012716 016346          64$:     MOV      #65$, (SP)        ;;SET UP FOR TRAP RETURN
2596 016344 000002                RTI
2597 016346 012737 000176 001140 65$:     MOV      #SWREG,SWR       ;;POINT TO SOFTWARE SWR
2598 016354 012737 000174 001142      MOV      #DISPREG,DISPLAY ;;
2599 016362 012637 000004          66$:     MOV      (SP)+,2#ERRVEC   ;;RESTORE ERROR VECTOR
2600
2601 016366 004737 040730                JSR      PC,STKINT        ;;INIT KEYBOARD
2602 016372 013701 002572      MOV      RKVEC,R1        ;;GET ADDRESS OF VECTOR STORAGE
2603 016376 012721 032442      MOV      #I.INTR,(R1)+   ;;SET IT TO INTERRUPT HNDLR
2604 016402 012711 000340      MOV      #PR7,(R1)       ;;SET INTERRUPT HANDLER PR7
2605 016406 013746 000000      MOV      0,-(SP)         ;;PUT NEW PS ON STACK
2606 016412 012746 016420      MOV      #67$,-(SP)      ;;PUT NEW PC ON STACK
2607 016416 000002                RTI                       ;;POP NEW PC AND PS
2608 016420
2609 016420 104401 012770          67$:     TYPE      ,PROGID        ;;TYPE IDENTIFICATION
2610 016424 104401 013052      TYPE      ,HELPG         ;;TYPE HELP QUESTION
2611 016430 104410                RDLIN
2612 016432 012601                MOV      (SP)+,R1        ;;GET ANSWER TO HELP Q.
2613 016434 105711                TSTB    (R1)             ;;GET ADDRESS OF ANSWER
2614 016436 001402                BEQ     STPARM           ;;TEST FIRST CHAR
2615 016440 104401 006377      TYPE      ,HELPL        ;;TYPE HELP FILE
2616
2617                                ;:*****
2618                                ;:SBTTL PARAMETER REQUEST & CHECK ROUTINE
2619                                ;:ENTRY:  AUTOMATIC AT START OF PROGRAM.
2620                                ;:EXIT:  GO TO MODE DECODE.
2621                                ;:
2622                                ;:THIS ROUTINE REQUESTS EACH PARAMETER IN TURN AND CALLS
2623                                ;:THE GETANS SUBROUTINE TO HANDLE THE ANSWERS AND STORE
2624                                ;:THEM.  IT THEN CHECKS THE PARAMETERS FOR LEGALITY AND
2625                                ;:VALIDITY AND, IF AN ERROR IS FOUND, REQUESTS THE
2626                                ;:PARAMETERS AGAIN, STARTING WITH THE PARAMETER IN ERROR.
2627                                ;:*****
2628
2629 016444                STPARM:
2630 016444 122737 000013 000041      CMPB    #13,2#41        ;LOAD FROM XXDP PACK ?
2631 016452 001003                BNE     1$              ;BRANCH IF NOT
2632 016454 104401 016120                TYPE    ,XXDPMG
2633 016460 000000                HALT
2634 016462                1$:
2635 016462 012702 006310      MOV     #INITTP,R2      ;STARTING ADDRESS OF DEF. PARAM
2636 016466 012703 006336      MOV     #TPINUS,R3     ;STARTING ADD OF ACTIVE PARAMS
2637 016472 104401 013122      TYPE    TPQ            ;TYPE DRIVE TYPE QUESTION
2638 016476 004437 017766      JSR    R4,GETANS       ;
2639 016502 023727 006336 000006  PARM2:  CMP     TPINUS,#6      ;SEE IF RK06
2640 016510 001404                BEQ     PARM3           ;BR IF YES
2641 016512 012737 001456 006332      MOV     #1456,INITEC   ;ELSE LOAD DEFAULT END CYL FOR RK07
2642 016520 000403                BR     PARM4           ;
2643 016522 012737 000632 006332  PARM3:  MOV     #632,INITEC    ;LOAD DEFAULT END CYL FOR RK06

```

2644	016530	104401	013145		PARM4:	TYPE	DRVQ		:TYPE DRIVE QUESTION
2645	016534	004437	017766			JSR	R4,GETANS		:GO GET PARAM
2646	016540	104401	013160			TYPE	SECTQ		:TYPE SECTOR/TRACK QUESTION
2647	016544	004437	017766			JSR	R4,GETANS		:GO GET PARAM
2648	016550	104401	013176		PARAM3:	TYPE	MODEQ		:TYPE MODE QUES
2649	016554	004437	017766			JSR	R4,GETANS		:GO GET PARAM
2650	016560	104401	013204			TYPE	WDEQ		
2651	016564	004437	017766			JSR	R4,GETANS		
2652	016570	104401	013233			TYPE	WDD		
2653	016574	004437	017766			JSR	R4,GETANS		
2654	016600	104401	013261		PARAM5:	TYPE	TRK LIM		:TYPE TRACK LIMITS QUESTION
2655	016604	004437	017766			JSR	R4,GETANS		:GO GET PARAM
2656	016610	104401	013277			TYPE	CYLLIM		:TYPE CYLINDER LIMITS QUESTION
2657	016614	004437	017766			JSR	R4,GETANS		:GO GET PARAM
2658	016620	104401	013320		PARAM6:	TYPE	OFSETQ		:TYPE OFFSET QUESTION
2659	016624	004437	017766			JSR	R4,GETANS		
2660	016630	023727	006336	000007	DONEPR:	CMP	TPINUS,#7		:TEST IF DRIVE TYPE IS
2661	016636	001405				BEQ	4\$:6 OR 7
2662	016640	023727	006336	000006		CMP	TPINUS,#6		:IF NOT TYPE ERROR
2663	016646	001401				BEQ	4\$		
2664	016650	000475				BR	IVDTP		
2665	016652	023737	006360	006356	4\$:	CMP	ECINUS,SCINUS		:COMP START & END CYL
2666	016660	002451				BLT	RSTADD		:NG-GET PARAM AGAIN
2667	016662	023737	006354	006352		CMP	ETINUS,STINUS		:COMP START & END TRK
2668	016670	002445				BLT	RSTADD		:NG-GET PARAM AGAIN
2669	016672	105037	002653			CLRB	TYPFMT		:CLEAR TYPE-FORMAT SWITCH, ASSUME RK06 HERE
2670	016676	023727	006336	000006		CMP	TPINUS,#6		:TEST IF TRACK PARAMETERS ARE
2671	016704	001407				BEQ	2\$:VALID FOR THE DRIVE TYPE
2672	016706	152737	000004	002653		BISB	#B.CDT,TYPFMT		
2673	016714	012737	001456	017312		MOV	#1456,LCYL		:SET LAST CYL FOR RK07
2674	016722	000403				BR	3\$		
2675	016724	012737	000632	017312	2\$:	MOV	#632,LCYL		:SET LAST CYL FOR RK06
2676	016732	023727	006354	000002	3\$:	CMP	ETINUS,#2		:TEST FOR VALID TRACK
2677	016740	003027				BGT	IVDTRK		:BR IF TOO BIG
2678	016742	023737	006360	017312		CMP	ECINUS,LCYL		:TEST FOR VALID CYL
2679	016750	003020				BGT	IVDCYL		:BR IF TOO BIG
2680	016752	023727	006340	000007		CMP	DRINUS,#7		:TEST IF VALID DRIVE
2681	016760	003026				BGT	IVDDR		:TOO BIG-BRANCH
2682	016762	023727	006344	000004		CMP	MDINUS,#4		:TEST IF VALID MODE
2683	016770	003031				BGT	IVDMOD		:ERROR IN MODE SELECTION
2684	016772	023727	006362	000260		CMP	OFINUS,#260		:TEST IF VALID OFFSET
2685	017000	003034				BGT	IVDOF		:TO BIG, ERROR
2686	017002	000442				BR	GETOUT		:YES-BR TO EXIT
2687	017004	104401	014001		RSTADD:	TYPE	INVDOF		:TYPE ERROR MESSAGE
2688	017010	000405				BR	SETRS		
2689	017012	104401	014050		IVDCYL:	TYPE	INVDCY		:TYPE ERROR
2690	017016	000402				BR	SETRS		
2691	017020	104401	014100		IVDTRK:	TYPE	IVDTRK		:TYPE ERROR
2692	017024	012702	006324		SETRS:	MOV	#INITST,R2		:RESET REGISTERS TO REENTER ALL
2693	017030	012703	006352			MOV	#STINUS,R3		:START & END ADDRESSES
2694	017034	000661				BR	PARAMS		:GO GET NEW START & END ADD
2695	017036	104401	014125		IVDDR:	TYPE	IVDDR		
2696	017042	000600				BR	STPARM		:GO GET ALL PARAM AGAIN
2697	017044	104401	014151		IVDTP:	TYPE	IVDTP		
2698	017050	000137	016444			JMP	STPARM		
2699	017054	104401	014173		IVDMOD:	TYPE	IVDMOD		

PARAMETER REQUEST & CHECK ROUTINE

```

2700 017060 012702 006316      MOV      #INITMD,R2      ;RESET REGISTERS FOR RE-REQUEST
2701 017064 012703 006344      MOV      #MDINUS,R3     ;OF MODE
2702 017070 000627      BR       PARAM3        ;RETURN TO MODE REQUEST
2703 017072 104401 014234      IVDOF:  TYPE      INVD0F ;TYPE INVALID OFFSET MESSAGE
2704 017076 012702 006334      MOV      #INITOF,R2    ;SET UP R2 AND R3 TO REDO OFFSET
2705 017102 012703 006362      MOV      #OFINUS,R3    ;PARAMETER
2706 017106 000644      BR       PARM6        ;GO GET OFFSET
2707 017110 105037 002656      GETOUT: CLRB      OPCONT ;CLEAR OPERATION CONTROL SWITCHES
2708 017114 023727 006342 000026      CMP      SEINUS,#26    ;TEST IF 26 SECTOR FORMAT
2709 017122 001412      BEQ      3$           ;YES-SKIP
2710 017124 012737 000024 006342      MOV      #24,SEINUS   ;FORCE TO 24 SECTORS/TRACK
2711 017132 152737 000020 002653      BISB    #B.CFMT,TYPFMT ;SET BIT FOR FORMAT
2712 017140 012737 012000 006306      MOV      #12000,TKWDCT ;SET WORD COUNT
2713 017146 000403      BR       4$
2714 017150 012737 013000 006306 3$:      MOV      #13000,TKWDCT ;ELSE SET FOR 22 SECTOR WORD COUNT
2715 017156 005437 006306 4$:      NEG      TKWDCT       ;MAKE IT NEG FOR RK611
2716 017162 023737 006362 006334      CMP      OFINUS,INITOF ;TEST IF OFFSET REQUIRED
2717 017170 001403      BEQ      5$           ;NO-SKIP
2718 017172 152737 000020 002656      BISB    #OREQSW,OPCONT ;ELSE SET SWITCH
2719 017200 023727 006344 000004 5$:      CMP      MDINUS,#4    ;IF MODE 4,
2720 017206 001004      BNE      6$           ;NO-SKIP
2721 017210 152737 000012 002656      BISB    #RCDASW!VFHDSW,OPCONT ;ELSE SET READ CHECK & HEADER VERIFY
2722 017216 000433      BR       10$
2723 017220 023727 006344 000003 6$:      CMP      MDINUS,#3    ;IF MODE 3
2724 017226 001004      BNE      7$           ;NO-SKIP
2725 017230 152737 000002 002656      BISB    #VFHDSW,OPCONT ;SET VERIFY HEADER
2726 017236 000423      BR       10$
2727 017240 023727 006344 000002 7$:      CMP      MDINUS,#2    ;IF MODE 2
2728 017246 001004      BNE      8$           ;NO-SKIP
2729 017250 152737 000007 002656      BISB    #WHDSW!VFHDSW!WCDASW,OPCONT ;SET WRITE HEADER & DATA VERIFY HEADER,
2730                                ;AND WRITE CHECK DATA SWITCHES
2731 017256 000413      BR       10$
2732 017260 023727 006344 000001 8$:      CMP      MDINUS,#1    ;IF MODE 1
2733 017266 001004      BNE      9$           ;NO-SKIP
2734 017270 152737 000003 002656      BISB    #WHDSW!VFHDSW,OPCONT ;SET VERIFY HEADER &
2735                                ;SET WRITE HEADER & DATA SWITCH
2736 017276 000403      BR       10$
2737 017300 152737 000001 002656 9$:      BISB    #WHDSW,OPCONT ;SET WRITE HDR & DATA SWITCH
2738 017306 000137 017314 10$:      JMP      GETUBS      ;GET USER BAD SECTORS
2739
2740 017312 000000      LCYL:   0             ;LAST CYL 632 FOR RK06, 1456 FOR RK07
2741
2742      ;*****
2743      ;SBTTL USER BAD SECTORS INPUT ROUTINE
2744      ;*THIS ROUTINE PROVIDES THE CAPABILITY TO INDICATE SPECIFIC PACK
2745      ;*ADDRESSES THAT ARE TO BE FLAGED AS SOFTWARE BAD SECTORS. THE USER
2746      ;*IS DIRECTED IN HOW TO ENTER THE ADDRESSES AND HOW TO END THE
2747      ;*OPERATION.
2748      ;*****
2749 017314 132737 000001 002656      GETUBS: BITB    #WHDSW,OPCONT ;TEST IF WRITING IS TO BE DONE
2750 017322 001566      BEQ      14$         ;NO - SKIP OUT
2751 017324 012700 000400      MOV      #400,R0      ;SET UP TO CLEAR BSSOFT FILE
2752 017330 012701 003660      MOV      #BSSOFT,R1   ;TO ALL ONES
2753 017334 012721 177777      1$:      MOV      #177777,(R1)+
2754 017340 005300      DEC      R0
2755 017342 001374      BNE     1$

```

2756	017344	012701	002660		MOV	#RDBUF,R1	:GET ADDRESS OF READ BUFFER
2757	017350	012700	000400		MOV	#400,R0	:SET COUNT FOR CLEAR
2758	017354	012721	177777	21\$:	MOV	#177777,(R1)+	:CLEAR TO ALL ONES
2759	017360	005300			DEC	R0	:DECREMENT COUNT
2760	017362	001374			BNE	21\$:LOOP UNTIL 0
2761	017364	104401	013515		TYPE	,UBSQ	:ASK USER BAD SECTOR QUESTION
2762	017370	104410			RDLIN		:GET ANSWER
2763	017372	012601			MOV	(SP)+,R1	
2764	017374	121127	000032		CMPB	(R1),#032	:TEST IF ↑Z
2765	017400	001002			BNE	3\$:NO - SKIP
2766	017402	000137	020154		JMP	DRINIT	:ELSE GO START PROGRAM
2767	017406	105711		3\$:	TSTB	(R1)	:TEST IF CR (NULL)
2768	017410	001531			BEQ	15\$:YES - EXIT TEST
2769	017412	121127	000116		CMPB	(R1),#'N	:TEST IF "NO"
2770	017416	001526			BEQ	15\$:YES - EXIT TEST
2771	017420	012700	002660		MOV	#RDBUF,R0	:SET POINTER TO FIRST WORD
2772	017424	104401	013575	4\$:	TYPE	,UBSFMT	:TYPE USER BAD SECTOR FORMAT
2773	017430	104401	013667		TYPE	,NEXTQ	:TYPE NEXT ENTRY REQUEST
2774	017434	104410		12\$:	RDLIN		:GET BAD SECTOR ENTRY
2775	017436	012601			MOV	(SP)+,R1	
2776	017440	105711			TSTB	(R1)	:TEST IF CR (NULL)
2777	017442	001512			BEQ	13\$:YES - EXIT TEST
2778	017444	010146			MOV	R1,-(SP)	:SET TO CONVERT TO BINARY
2779	017446	004737	037636		JSR	PC,OCTBIN	
2780	017452	017652			30\$:ERROR RETURN
2781	017454	021637	006360		CMP	(SP),ECINUS	:TEST IF CYLINDER VALID
2782	017460	003003			BGT	16\$:YES - SKIP
2783	017462	021637	006356		CMP	(SP),SCINUS	:TEST IF CYLINDER VALID
2784	017466	002004			BGE	5\$:YES - SKIP
2785	017470	104401	014050	16\$:	TYPE	,INVDCY	:INVALID CYLINDER MESSAGE
2786	017474	005726			TST	(SP)+	:CLEAN STACK
2787	017476	000752			BR	4\$:GO TYPE FORMAT MESSAGE
2788	017500	012620		5\$:	MOV	(SP)+,(R0)+	:STORE CYLINDER
2789	017502	121127	000054	6\$:	CMPB	(R1),#'	:LOOP TO BUMP R1 PAST CYLINDER ENTRY
2790	017506	001406			BEQ	7\$:LOOK FOR COMMA BUT RESTART
2791	017510	105721			TSTB	(R1)+	:THIS ENTRY IF CR (NULL) IS
2792	017512	001373			BNE	6\$:IS FOUND
2793	017514	005740			TST	-(R0)	:RESET POINTER
2794	017516	104401	013734		TYPE	,BADENT	:TYPE BAD ENTRY MESSAGE
2795	017522	000740			BR	4\$:GO TYPE ENTRY FORMAT MESSAGE
2796	017524	105721		7\$:	TSTB	(R1)+	:BUMP PAST COMMA
2797	017526	010146			MOV	R1,-(SP)	:GO CONVERT TRACK TO BINARY
2798	017530	004737	037636		JSR	PC,OCTBIN	
2799	017534	017652			30\$:ERROR RETURN
2800	017536	012603			MOV	(SP)+,R3	:STORE TRACK
2801	017540	121127	000054	8\$:	CMPB	(R1),#'	:LOOP TO BUMP R1 PAST
2802	017544	001406			BEQ	9\$:TRACK ENTRY. LOOK FOR COMMA
2803	017546	105721			TSTB	(R1)+	:BUT RESTART THIS ENTRY IF CR
2804	017550	001373			BNE	8\$: (NULL) IS FOUND
2805	017552	005740			TST	-(R0)	:RESET POINTER
2806	017554	104401	013734		TYPE	,BADENT	:TYPE BAD ENTRY MESSAGE
2807	017560	000721			BR	4\$:AND TYPE FORMAT MESSAGE
2808	017562	105721		9\$:	TSTB	(R1)+	:BUMP PAST COMMA
2809	017564	010146			MOV	R1,-(SP)	:GO CONVERT SECTOR TO BINARY
2810	017566	004737	037636		JSR	PC,OCTBIN	
2811	017572	017652			30\$:ERROR RETURN

```

2812 017574 012604          MOV      (SP)+,R4      ;STORE SECTOR
2813 017576 020337 006354  CMP      R3,ETINUS    ;TEST IF TRACK VALID
2814 017602 003003          BGT      17$          ;YES - SKIP
2815 017604 020337 006352  CMP      R3,STINUS    ;TEST IF TRACK VALID
2816 017610 002004          BGE      10$          ;YES - SKIP
2817 017612 104401 014100  17$:    TYPE      ,INVDTK  ;TYPE INVALID TRACK MESSAGE
2818 017616 005740          TST      -(R0)        ;RESET POINTER
2819 017620 000701          BR       4$          ;RESTART ENTRY
2820 017622 020437 006342  10$:    CMP      R4,SEINUS    ;TEST IF SECTOR VALID
2821 017626 002404          BLT      11$          ;YES - SKIP
2822 017630 104401 013754  TYPE      ,ILLSEC    ;TYPE ILLEGAL SECTOR MESSAGE
2823 017634 005740          TST      -(R0)        ;RESET POINTER
2824 017636 000672          BR       4$          ;GO RESTART ENTRY
2825 017640 110420          11$:    MOVB     R4,(R0)+    ;INSERT SECTOR NUMBER
2826 017642 110320          MOVB     R3,(R0)+    ;INSERT TRACK NUMBER
2827 017644 104401 013667  TYPE      ,NEXTQ     ;TYPE NEXT QUESTION
2828 017650 000671          BR       12$          ;GO GET NEXT ENTRY
2829 017652 104401 006374  30$:    TYPE      ,QUES    ;TYPE BAD ENTRY LINE
2830 017656 104401 042104  TYPE      ,STTYIN
2831 017662 104401 001267  TYPE      ,SCRLF
2832 017666 000656          BR       4$
2833 017670 012710 177777  13$:    MOV      #177777,(R0) ;GET LINE AGAIN
2834                                ;MAKE SURE THE WORD AFTER THE LAST
2835 017674 000137 017704  15$:    JMP      CKPRES     ;VALID ENTRY IS ALL ONES.
2836 017700 000137 020154  14$:    JMP      DRINIT   ;GO CHECK IF PRESERVE SDBSF
2837                                ;GO INITIALIZE DRIVE
2838                                ;*****
2839                                ;.SBTTL PRESERVE SDBSF QUESTION ROUTINE
2840                                ;* THIS ROUTINE DETERMINES IF SOFTWARE BAD SECTOR FILES ARE TO
2841                                ;* BE PRESERVED
2842 017704 104401 013330  CKPRES: TYPE      ,PSDQ ;TYPE PRESERVE SDBSF QUESTION
2843 017710 104410          ROLIN     ;GET ANSWER
2844 017712 012601          MOV      (SP)+,R1    ;GET POINTER TO ANSWER
2845 017714 104401 013415  TYPE      ,SDBSF    ;TYPE START OF PRESERVE MESSAGE
2846 017720 121127 000032  CMPB     (R1),#032   ;TEST IF ↑Z
2847 017724 001001          BNE      2$          ;NO - SKIP
2848 017726 000405          BR       1$          ;ELSE SET BIT AND GO FORMAT
2849 017730 105711          2$:    TSTB     (R1)    ;TEST IF NULL
2850 017732 001403          BEQ      1$          ;YES - EXIT
2851 017734 121127 000116  CMPB     (R1),#'N    ;TEST IF "N"
2852 017740 001406          BEQ      3$          ;YES EXIT
2853 017742 152737 000040 002656  1$:    BISB     #PSDBSF,OPCONT ;SET FLAG TO PRESERVE SDBSF
2854 017750 104401 013461  TYPE      ,PRSD     ;REPORT PRESERVED
2855 017754 000402          BR       4$          ;SKIP
2856 017756 104401 013475  3$:    TYPE      ,NPRSD   ;REPORT NOT PRESERVED
2857 017762 000137 020154  4$:    JMP      DRINIT
2858                                ;*****
2859                                ;.SBTTL GET ANSWER (PARAMETER ENTRY) SUBROUTINE
2860                                ;*ENTRY JSR R4,GETANS WITH R2 POINTING TO THE DEFAULT PARAM
2861                                ;* R3 POINTING TO THE ACTIVE PARAM
2862                                ;*
2863                                ;*RETURN RTS R4
2864                                ;*
2865                                ;*THIS ROUTINE RECEIVES THE PARAMETER ENTRY AND ALTERS THE
2866                                ;*APPROPRIATE ACTIVE PARAMETER. IF THE ENTRY IS NULL THE DEFAULT
2867                                ;*VALUE IS PLACED IN THE ACTIVE ENTRY. THE CONTROL C & Z

```

```

2868
2869
2870
2871
2872
2873 017766
2874 017766 104410
2875 017770 012601
2876 017772 121127 000032
2877 017776 001434
2878 020000 105711
2879 020002 001002
2880 020004 011213
2881 020006 000405
2882 020010 010146
2883 020012 004737 037636
2884 020016 020106
2885 020020 012513
2886 020022 022703 006352
2887 020026 001403
2888 020030 022703 006356
2889 020034 001010
2890 020036 022223
2891 020040 105711
2892 020042 001404
2893 020044 122127 000054
2894 020050 001373
2895 020052 000756
2896 020054 011213
2897 020056 022223
2898 020060 000204
2899 020062 012704 016444
2900 020066 000204
2901 020070 012223
2902 020072 020327 006364
2903 020076 001374
2904 020100 012704 016630
2905 020104 000204
2906 020106 104401 006374
2907 020112 104401 042104
2908 020116 104401 001267
2909 020122 020327 006354
2910 020126 001403
2911 020130 020327 006360
2912 020134 001004
2913 020136 162702 000002
2914 020142 162703 000002
2915 020146 162704 000010
2916 020152 000204

```

```

;*(+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
      CMP       #STINUS,R3     ;START TRACK PARAM
      BEQ       6$           ;YES-BRANCH
      CMP       #SCINUS,R3     ;STARTING CYL PARAM
      BNE       9$           ;NO-GO TO EXIT
      CMP       (R2)+,(R3)+    ;BUMP R2 & R3
      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
      MOV        (R2),(R3)     ;STORE DEFAULT
      CMP       (R2)+,(R3)+    ;BUMP R2 & R3
      RTS       R4           ;RETURN
      MOV        #STPARM,R4    ;DUMMY R4 TO RESTART PARAM
      RTS       R4           ;RETURN
      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
      TYPE      'QUES          ;TYPE QUESTION MARK
      TYPE      '$TTYIN        ;TYPE BAD LINE
      TYPE      '$CRLF
      CMP       R3,#ETINUS     ;IS THIS ENDING TRACK?
      BEQ       31$          ;YES-BRANCH
      CMP       R3,#ECINUS     ;IS THIS ENDING CYLINDER?
      BNE       32$          ;NO-BRANCH
      SUB       #2,R2          ;BACK UP REGISTERS TO REDO
      SUB       #2,R3          ;PARAM REQUEST
      SUB       #10,R4         ;BACK UP TO REREQUEST PARAM
      RTS       R4           ;RETURN

```

```

2917
2918
2919
2920 ;*****
2921 ;SBTTL DRIVE INITIALIZE ROUTINE
2922 ;*BEFORE FORMATTING BEGINS, THIS ROUTINE INITIALIZES
2923 ;*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

```

```
;*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:

```
2924  
2925  
2926  
2927  
2928  
2929  
2930  
2931  
2932  
2933  
2934  
2935 020154  
2936 020154 012706 001100  
2937 020160 012737 020154 001110  
2938 020166 012737 026462 002600  
2939 020174 012737 025524 002576  
2940 020202 105037 002654  
2941 020206 005037 006364  
2942 020212 012703 002446  
2943 020216 113763 006340 000000  
2944 020224 105063 000007  
2945 020230 153763 002653 000007  
2946 020236 013702 002570  
2947 020242 012705 002362  
2948 020246 005065 000014  
2949 020252 105065 000007  
2950 020256 153765 002653 000007  
2951 020264 112765 000177 000001  
2952 020272 004737 025442  
2953 020276 113765 006340 000000  
2954 020304 112765 000113 000001  
2955 020312 004737 025442  
2956 020316 112765 000103 000001  
2957 020324 004737 025442  
2958 020330 112765 000141 000001  
2959 020336 004737 025442  
2960 020342 023727 006344 000003  
2961 020350 002415  
2962 020352 032765 004000 000040  
2963 020360 001026  
2964 020362 104401 014260  
2965 020366 104401 014416  
2966 020372 042762 000100 000000  
2967 020400 000000  
2968 020402 000664  
2969 020404 032765 004000 000040 1$:  
2970 020412 001411  
2971 020414 104401 014336  
2972 020420 104401 014416  
2973 020424 042762 000100 000000  
2974 020432 000000  
2975 020434 000647  
2976 020436 112765 000117 000001 2$:  
2977 020444 013765 017312 000002  
2978 020452 112765 000002 000005  
2979 020460 004737 025442
```

```
MOV #STACK,SP ;RESET THE STACK  
MOV #DRINIT,$LPERR ;SET UP TO LOOP ON ERROR  
MOV #ERRHDL,A:ABNL ;SET UP FOR ERROR HANDLING  
MOV #ERRFRE,A:NORM ;SET UP FOR NO ERROR DRIVER RETURNS  
CLRB ERRCNT ;CLEAR ERROR COUNT  
CLR RECODE ;CLEAR RECOVERY CODE  
MOV #PARI,R3 ;SET UP PARAMETER BLOCK 1 FOR  
MOVB DRINUS,P:DRVN(R3) ;ADDITIONAL INFO RECOVERY IF  
CLRB P:CSIH(R3) ;CLEAR PREV. CONTENTS  
BISB TYPFMT,P:CSIH(R3) ;NEEDED IN ERROR PROCESSING  
MOV RKBAS,R2 ;SET R2 FOR RKBASE ADDRESS  
MOV #PARM0,R5 ;GET PARAMETER BLOCK ADDRESS  
CLR P:PRST(R5) ;CLEAR PROGRAM STATUS REG  
CLRB P:CSIH(R5) ;CLR PREV CONTENTS  
BISB TYPFMT,P:CSIH(R5)  
MOVB #SUBCLA,P:CMND(R5) ;DO SUBSYSTEM CLEAR  
JSR PC,DRVCAL  
MOVB DRINUS,P:DRVN(R5) ;LOAD DRIVE NUMBER  
MOVB #RECAL,P:CMND(R5) ;DO RECALIBRATE  
JSR PC,DRVCAL  
MOVB #PACK,P:CMND(R5) ;DO PACK ACK  
JSR PC,DRVCAL  
MOVB #RDSTAT,P:CMND(R5) ;DO RD STAT  
JSR PC,DRVCAL  
CMP MDINUS,#3 ;TEST IF ANY WRITE TO BE DONE  
BLT 1$ ;NO-GO TEST WRITE LOCK ON  
BIT #S.WRL,P:ADD(R5) ;ELSE TEST IF WRITE LOCK OFF  
BNE 2$ ;IF WRONG - TYPE MESSAGE  
TYPE ,SWLREQ ;AND WAIT FOR RESPONSE  
TYPE ,CONREQ  
BIC #IE,RKCSI(R2) ;PREVENT INTERRUPT WHEN WRITE LOCK CHANGED  
HALT ;WRITE LOCK IS TESTED AGAIN  
BR ;FOR CORRECT SETTING  
1$:  
BIT #S.WRL,P:ADD(R5)  
BEQ 2$  
TYPE ,RWLREQ  
TYPE ,CONREQ  
BIC #IE,RKCSI(R2) ;PREVENT INTERRUPT WHEN WRITE LOCK CHANGE  
HALT  
BR  
2$:  
MOVB #SEEK,P:CMND(R5) ;DO SEEK  
MOV LCYL,P:CYLN(R5) ;TO LAST CYL  
MOVB #2,P:TRCK(R5) ;TRACK 2  
JSR PC,DRVCAL
```


E05

CZR6LC0 RK06L CTG FMTR MACY11 30(1046) 01-DEC-77 14:28 PAGE 57
 CZR6LC.P11 01-DEC-77 14:20 DRIVE INITIALIZE ROUTINE

SEQ 0056

2980	020464	012765	004660	000010		MOV	#BSFACT,P.BALO(R5) ;SET BUFFER ADDRESS FOR BAD SECTORS
2981	020472	005000				CLR	RO
2982	020474	012703	000011			MOV	#11,R3 ;SET UP MAX SECTOR VALUE
2983	020500	112765	000121	000001	8\$:	MOVB	#RDATA,P.CMND(R5) ;DO READ DATA FROM 0 THRU ?
2984	020506	012765	000400	000012		MOV	#400,P.WC(R5) ;READ ENTIRE SECTOR
2985	020514	005465	000012			NEG	P.WC(R5)
2986	020520	142765	000020	000007		BICB	#8.CFMT,P.CS1H(R5) ;SET TO 22 SECTOR FORMAT
2987	020526	132737	000020	002653		BITB	#8.CFMT,TYPFMT ;TEST IF PACK TO BE FORMATTED 20 SECTOR
2988	020534	001401				BEQ	3\$;NO-BR. LEAVE RO=0
2989	020536	005200				INC	RO ;SET RO=1
2990	020540	110065	000004		3\$:	MOVB	RO,P.SECT(R5) ;PUT IN BLOCK
2991	020544	004737	025442			JSR	PC,DRVCAL
2992	020550	032737	100000	006364		BIT	#ANYDER,RECODE ;TEST IF ERROR
2993	020556	001410				BEQ	4\$;BR IF NO
2994	020560	062700	000002			ADD	#2,RO ;BUMP TO NEXT SECTOR
2995	020564	020003				CMP	RO,R3 ;CHECK IF STILL IN
2996	020566	003764				BLE	3\$;BAD SECTOR. IF YES, DO IT AGAIN
2997	020570	104401	014452			TYPE	,BAD632 ;TYPE MESSAGE
2998	020574	000000				HALT	
2999	020576	000776				BR	.-2
3000	020600	132737	000001	002656	4\$:	BITB	#WHDSW,OPCONT ;WILL THIS PASS WRITE?
3001	020606	001404				BEQ	17\$;NO - SKIP
3002	020610	132737	000040	002656		BITB	#PSDBSF,OPCONT ;TEST IF PRESERVE FILE FLAG SET
3003	020616	001413				BEQ	7\$;NO SKIP
3004	020620	020027	000011		17\$:	CMP	RO,#11 ;TEST IF RO GREATER THAN 11
3005	020624	003010				BGT	7\$;IF YES, WE ALREADY HAVE A GOOD READ
3006	020626	012765	003660	000010		MOV	#BSSOFT,P.BALO(R5) ;ELSE SET UP TO READ SOFTWARE FILE
3007	020634	012703	000025			MOV	#25,R3 ;SET UP MAX SECTOR VALUE
3008	020640	012700	000012			MOV	#12,RO ;SET UP STARTING SECTOR
3009	020644	000715				BR	8\$;GO READ FILE
3010	020646	104401	014746		7\$:	TYPE	,SERNUM ;TYPE OUT SERIAL NUMBER
3011	020652	012746	004660			MOV	#BSFACT,-(SP) ;OF CARTIDGE TO BE FORMATTED
3012	020656	004737	040212			JSR	PC,#\$DB20 ;CONVERT SERIAL NUMBER TO ASCII
3013	020662	012637	020670			MOV	(SP)+,6\$;GET RESULT
3014	020666	104401				TYPE	;PRINT IT
3015	020670	000000			6\$:	.WORD	;ADDRESS OF RESULT GOES HERE
3016	020672	104401	015413			TYPE.	;STARS
3017	020676	005737	004666			TST	BSFACT+6 ;TEST IF 3RD WORD ALL ZEROS
3018	020702	001404				BEQ	10\$;IF YES - SKIP
3019	020704	104401	015006			TYPE	,ALNPAC ;ELSE TYPE ALIGNMENT PACK MESSAGE
3020							;AND ABORT
3021	020710	000137	016202			JMP	START ;JUMP TO START IF CONTINUE
3022	020714	153765	002653	000007	10\$:	BISB	TYPFMT,P.CS1H(R5) ;SET TYPE & FORMAT AGAIN
3023	020722	132737	000001	002653		BITB	#WHDSW,OPCONT ;TEST IF WE WILL BE WRITING
3024	020730	001441				BEQ	19\$;NO - SKIP BSSOFT INIT
3025	020732	012700	003670			MOV	#BSSOFT+10,RO ;ELSE SET POINTER TO BAD SECTOR DATA
3026	020736	012701	000200			MOV	#200,R1 ;SET A COUNT
3027	020742	132737	000040	002656		BITB	#PSDBSF,OPCONT ;TEST IF PRESERVE FLAG SET
3028	020750	001413				BEQ	25\$;NO - SKIP
3029	020752	005720			20\$:	TST	(RO)+ ;TEST IF ALL ONES
3030	020754	100410				BMI	21\$;YES - SKIP
3031	020756	005301				DEC	R1 ;DECREMENT COUNT TO CHECK IT DOES NOT
3032	020760	001374				BNE	20\$;OVERFLOW THE BUFFER
3033	020762	104401	014511			TYPE	,TDMSOF ;MESSAGE CAUSED BY OVERFLOW
3034	020766	104401	014576			TYPE	,TDMSUF
3035	020772	000137	016202			JMP	START ;RESTART THE PROGRAM

```

3036 020776 005740          21$: TST      -(R0)          ;REPOSITION POINTER
3037 021000 012703 002660 25$: MOV      #RDBUF,R3      ;SET POINTER TO SECTORS ENTERED AS BAD
3038 021004 012320          22$: MOV      (R3)+,(R0)+    ;MOV IN THE ADDRESS
3039 021006 100412          BMI      19$           ;IF NEGATIVE, MOVE IS DONE
3040 021010 012320          MOV      (R3)+,(R0)+    ;ELSE GET SECOND WORD
3041 021012 124141          CMPB     -(R1),-(R1)    ;DEC COUNT BY 2
3042 021014 005701          TST      R1           ;CHECK IT DIDN'T OVERFLOW
3043 021016 100372          BPL      22$           ;NOT YET - LOOP
3044 021020 104401 014511  TYPE     'TDMSOF        ;ELSE TYPE MESSAGE
3045 021024 104401 014576  TYPE     'TDMSUF
3046 021030 000137 016202  JMP      $START        ;RESTART TEST
3047 021034 012700 003660 19$: MOV      #BSSOFT,R0    ;GET ADDRESS OF BSSOFT
3048 021040 013720 004660  MOV      BSFACT,(R0)+  ;GET FIRST WORD
3049 021044 013720 004662  MOV      BSFACT+2,(R0)+;GET SECOND WORD
3050 021050 005020          CLR      (R0)+        ;CLEAR NEXT TWO WORDS
3051 021052 005020          CLR      (R0)+        ;AS REQUIRED
3052 021054 012701 000200  MOV      #200,R1       ;SET COUNT
3053 021060 012700 004670  MOV      #BSFACT+10,R0 ;GET ADDR OF START OF BAD SECTORS
3054 021064 005720          24$: TST      (R0)+        ;TEST IF NEGATIVE
3055 021066 100410          BMI      23$           ;YES - SKIP
3056 021070 005301          DEC      R1           ;ELSE DEC COUNT
3057 021072 100374          BPL      24$           ;IF NOT YET ZERO - LOOP
3058 021074 104401 014662  TYPE     'TDMFAC        ;MESSAGE REPORT
3059 021100 104401 014576  TYPE     'TDMSUF
3060 021104 000137 016202  JMP      $START        ;GO RESTART PROGRAM
3061 021110 000137 021114  JMP      DSPTCH        ;GO TO DISPATCHER
3062 *****
3063 .SBTTL DISPATCH ROUTINE
3064 *****
3065 021114          DSPTCH:
3066 021114 132737 000001 002656 BITB     #WHDSW,OPCONT ;TEST IF HDRS & DATA TO BE WRITTEN
3067 021122 001404          BEQ      1$           ;YES-SKIP
3068 021124 004737 022064  JSR      PC,WRTOP      ;GO TO WRITE HEADERS & DATA
3069 021130 104401 015300  TYPE     ,PRGRS1      ;TYPE PROGRESS 1 MESSAGE
3070
3071 021134 132737 000016 002656 1$: BITB     #VFHDSW!WCDASH!RCDASH,OPCONT ;ANY OTHER OPERATION?
3072 021142 001033          BNE      3$           ;YES-SKIP
3073 021144 104401 015345          TYPE     ,PRGRS2      ;TYPE DONE MESSAGE
3074 021150 132737 000001 002656 4$: BITB     #WHDSW,OPCONT ;TEST IF HEADERS AND DATA WERE WRITTEN
3075 021156 001402          BEQ      5$           ;NO - DON'T WRITE BAD SECTOR FILES
3076 021160 004737 031662  JSR      PC,BDSTWT     ;GO WRITE BAD SECTOR TABLES
3077 021164 032777 000200 157746 5$: BITB     #SW7,2SWR    ;TEST IF BAD SECTOR REPORT REQUESTED
3078 021172 001402          BEQ      2$           ;NO
3079 021174 004737 032062  JSR      PC,BSREPT     ;GO REPORT BAD SECTORS
3080 021200 012706 001100 2$: MOV      #STACK,SP   ;RESET STACK
3081 021204 032777 000001 157726 BITB     #SW0,2SWR    ;TEST IF LOOP ON PROGRAM
3082 021212 001405          BEQ      6$           ;NO - GO GET NEXT PARAMETERS
3083 021214 012737 177777 002660  MOV      #177777,RDBUF ;ELSE RESTART OPERATION
3084 021222 000137 020154  JMP      DRINIT        ;GO GET NEXT PARAM SET
3085 021226 000137 016444 6$: JMP      STPARM      ;GO TO VERIFY HEADERS AND/OR DATA
3086 021232 004737 023064 3$: JSR      PC,VEROP    ;GO TO VERIFY HEADERS AND/OR DATA
3087 021236 000742          BR       4$           ;NO
3088 *****
3089 .SBTTL TRACK INCREMENT ROUTINE
3090 *****
3091 021240 032777 040000 157672 TRKINC: BIT  #SW14,2SWR ;TEST IF LOOP ON OPERATION

```

```

3092 021246 001011          BNE      2$          ;YES - BYPASS INCREMENT
3093 021250 020037 017312   CMP      RD,LCYL    ;CHECK IF NEXT INCREMENT WOULD
3094 021254 001007          BNE      1$          ;SELECT LAST TRACK ON THE
3095 021256 020127 000001   CMP      R1,#1      ;PACK. IF SO-DO NOT
3096 021262 001004          BNE      1$          ;INCREMENT
3097 021264 112737 000377 002651 3$:  MOVB    #377,OPCOMP ;SET DONE
3098 021272 000207          RTS      PC          ;EXIT
3099 021274 005201          INC      R1          ;ELSE INCREMENT TRACK
3100 021276 020137 006354   CMP      R1,ETINUS  ;IF THIS BUMP DOES NOT
3101 021302 003773          BLE      2$          ;EXCEED THE SELECTED PACK
3102 021304 013701 006352   MOV      STINUS,R1  ;ADDRESS FORMAT-EXIT
3103 021310 005200          INC      RD          ;IF IT DOES, SET THE TRACK TO
3104 021312 020037 006360   CMP      RD,ECINUS  ;THE STARTING TRACK VALUES AND
3105 021316 101401          BLOS    4$          ;
3106 021320 000761          BR       3$          ;
3107 021322 020037 017312   4$:  CMP      RD,LCYL    ;SEE IF ON LAST CYL
3108 021326 001401          BEQ      5$          ;BR IF YES
3109 021330 000207          RTS      PC          ;ELSE RETURN
3110 021332 020127 000002   5$:  CMP      R1,#2      ;SEE IF ON TRACK 2
3111 021336 103774          BLO     6$          ;
3112 021340 000751          BR       3$          ;
3113
3114
3115
3116
3117
3118 021342          ;:*****
3119 021344 104411          ;SBTTL BUILD HEADERS ROUTINE
3120 021344 012703 000026          ;:*****
3121 021350 012702 005674          BLDHDR:
3122 021354 006301          SAVREG
3123 021356 006301          MOV      #26,R3      ;PRESET FOR 22 SECTOR FORMAT
3124 021360 006301          MOV      #BUFFD,R2  ;GET BUFFER ADDRESS
3125 021362 006301          ASL     R1           ;SHIFT TO ALIGN FOR HEADER
3126 021364 006301          ASL     R1
3127 021366 052701 140000          ASL     R1
3128 021372 132737 000020 002653   BIS     #140000,R1   ;SET GOOD SECTOR BITS
3129 021400 001404          BITB    #8.CFMT,TYPFMT ;TEST IF 20 SET FORMAT
3130 021402 052701 001000          BEQ     1$          ;NO-SKIP
3131 021406 012703 000024          BIS     #BIT9,R1    ;ELSE SET FORMAT BIT
3132 021412 010022          MOV     #24,R3      ;ADJUST SECTOR COUNT
3133 021414 010122          MOV     RD,(R2)+    ;INSERT CYLINDER WORD
3134 021416 010004          MOV     R1,(R2)+    ;INSERT TRK/SECT WORD
3135 021420 010105          MOV     RD,R4       ;COMPUTE VRC
3136 021422 040005          MOV     R1,R5
3137 021424 040104          BIC     R0,R5
3138 021426 050504          BIC     R1,R4
3139 021430 010422          BIS     R5,R4
3140 021432 005201          MOV     R4,(R2)+    ;INSERT VRC
3141 021434 005303          INC     R1          ;BUMP SECTOR COUNT
3142 021440 104412          DEC     R3          ;DEC COUNT
3143 021442 012737 100000 001250          BNE     1$          ;EXIT BUILD IF DONE
3144 021444 012737 004660 021462          RESREG
3145 021450 004437 021556          MOV     #BIT15,STMP5 ;SET BIT TO BE BAD SECTOR INDICATOR
3146 021462 000000          MOV     #BSFACT,2$  ;SET TABLE TO BE SEARCHED
3147 021464 012603          JSR     R4,SRHTBS   ;GO SEARCH TABLE FOR BAD SECTORS
                          .WORD
                          MOV     (SP)+,R3 ;GET BAD SECTOR COUNT

```

```

3148 021466 001417
3149 021470 011602
3150 021472 006302
3151 021474 006302
3152 021476 061602
3153 021500 062602
3154 021502 062702 000002
3155 021506 043762 001250 005674
3156 021514 043762 001250 005676
3157 021522 005303
3158 021524 001361
3159 021526 023727 021462 003660
3160 021534 001407
3161 021536 012737 040000 001250
3162 021544 012737 003660 021462
3163 021552 000741
3164 021554
3165 021554 000207
3166
3167
3168
3169
3170
3171
3172
3173
3174
3175
3176
3177 021556 010237 001242
3178 021562 010337 001244
3179 021566 012637 001246
3180 021572 011402
3181 021574 012437 001240
3182 021600 062737 001000 001240
3183 021606 005003
3184 021610 062702 000010
3185 021614 005712
3186 021616 100430
3187 021620 020012
3188 021622 001406
3189 021624 062702 000004
3190 021630 020237 001240
3191 021634 002021
3192 021636 000766
3193 021640 005722
3194 021642 011237 001256
3195 021646 042737 174377 001256
3196 021654 123701 001257
3197 021660 001402
3198 021662 005722
3199 021664 000753
3200 021666 005203
3201 021670 012246
3202 021672 042716 177700
3203 021676 000746

6$: BEQ 4$ ; IF NO BAD SECTORS GO TEST BSSOFT
MOV (SP),R2 ; GET BAD SECTOR #
ASL R2 ; MULTIPLY BAD SECTOR # BY 6
ASL R2 ; AND ADD 2 TO GET INDEX
ADD (SP),R2 ; TO TRK/SEC WORD OF HEADER OF
ADD (SP)+,R2 ; BAD SECTOR
ADD #2,R2
BIC $TMP5,BUFFD(R2) ; CLEAR BIT TO INDICATE BAD SECTOR
BIC $TMP5,BUFFD+2(R2) ; CLEAR BIT IN VRC FOR BAD SECTOR
DEC R3 ; DEC # OF BAD SECTOR COUNTER
BNE 6$ ; IF 0
4$: CMP 2$,#BSSOFT ; TEST IF BAD SECTOR SOFTWARE
BEQ 5$ ; HAS BEEN TESTED. IF NOT, GO
MOV #BIT14,$TMP5 ; SET TABLE TO BE SEARCHED
MOV #BSSOFT,2$
BR 3$ ; ELSE EXIT

5$: RTS PC

;*****
;SBTTL SEARCH BAD SECTOR TABLES ROUTINE
;THIS ROUTINE RETURNS A LIST OF BAD SECTORS FOUND IN
;THE SPECIFIC TABLE (BAD SECTOR SOFTWARE OR BAD SECTOR FACTORY)
;FOR THE SPECIFIC CYLINDER AND TRACK DESIRED. R0 AND R1 MUST
;CONTAIN THE CYLINDER AND TRACK, RESPECTIVELY.
;
;THE LIST OF BAD SECTORS IS FOUND ON THE STACK WHEN THE ROUTINE
;RETURNS TO THE CALLER. THE FIRST ENTRY ON THE STACK WILL BE THE
;NUMBER OF BAD SECTORS FOUND FOR THIS CYLINDER AND TRACK.
;*****
SRHTBS: MOV R2,$TMP2
MOV R3,$TMP3
MOV (SP)+,$TMP4 ; STORE RETURN CONTENTS OF R4
MOV (R4),R2 ; GET ADDRESS OF BAD SECTOR TABLE TO SEARCH
MOV (R4)+,$TMP1 ; SETUP FOR LENGTH OF BAD SECTOR TABLE
ADD #1000,$TMP1
CLR R3 ; CLEAR R3 FOR COUNT
ADD #10,R2 ; SET R2 FOR POINTER TO CYLINDER ENTRY
1$: TST (R2) ; TEST IF ALL ONES
BMI 5$ ; YES-DONE
CMP R0,(R2) ; TEST IF BAD SECTOR IN PRESENT CYL
BEQ 3$ ; YES-GO CHECK TRACK
ADD #4,R2 ; ELSE BUMP POINTER
CMP R2,$TMP1 ; TEST IF OUT OF TABLE
BGE 5$ ; YES-EXIT
BR 1$ ; LOOP
3$: TST (R2)+ ; TRACK CHECK - PUT POINTER AT TRK/SEC BYTE
MOV (R2),$TMP10 ; GET TRK/SEC WORD
BIC #174377,$TMP10 ; CLEAR ALL BUT TRACK
CMPB $TMP10+1,R1 ; CHECK IF BAD SECTOR IN THIS TRACK
BEQ 4$ ; YES - GO PUT SECTOR NUMBER ON STACK
TST (R2)+ ; ELSE BUMP POINTER TO NEXT CYL WORD
BR 1$ ; GO TEST NEXT CYL
4$: INC R3 ; BUMP BAD SECTOR COUNT
MOV (R2)+,-(SP) ; PUT TRK/SEC WORD ON STACK
BIC #177700,(SP) ; CLEAR ALL BUT SECTOR NUMBER
BR 1$ ; GO CHECK REST OF FILE

```

3204 021700 010346
 3205 021702 013702 001242
 3206 021706 013703 001244
 3207 021712 013746 001246
 3208 021716 000204
 3209
 3210
 3211
 3212
 3213
 3214
 3215
 3216 021720 104411
 3217 021722 012737 004660 021734
 3218 021730 004437 021556
 3219 021734 000000
 3220 021736 012603
 3221 021740 001015
 3222 021742 023727 021734 003660
 3223 021750 001404
 3224 021752 012737 003660 021734
 3225 021760 000763
 3226 021762 042737 001000 006364
 3227 021770 104412
 3228 021772 000207
 3229 021774 022602
 3230 021776 001403
 3231 022000 005303
 3232 022002 001374
 3233 022004 000756
 3234 022006 052737 001000 006364
 3235 022014 005303
 3236 022016 001764
 3237 022020 005726
 3238 022022 000774
 3239
 3240
 3241
 3242 022024 104411
 3243 022026 012704 003670
 3244 022032 005724
 3245 022034 100376
 3246 022036 005744
 3247 022040 010024
 3248 022042 012703 000010
 3249 022046 006301
 3250 022050 005303
 3251 022052 001375
 3252 022054 060201
 3253 022056 010114
 3254 022060 104412
 3255 022062 000207
 3256
 3257
 3258
 3259 022064 105037 002651

```

5$:  MOV R3, -(SP) ;EXIT - PUT NUMBER OF BAD
      MOV STMP2, R2 ;SECTORS ON STACK-RESTORE
      MOV STMP3, R3 ;REGISTERS
      MOV STMP4, -(SP) ;PUT RETURN ON STACK
      RTS R4 ;RETURN
;*****
;SBTTL BAD SECTOR CHECK ROUTINE
;*THIS ROUTINE WILL SEARCH BOTH TABLES TO DETERMINE IF A
;*SPECIFIC SECTOR IS LISTED AS BAD. IF THE SECTOR IS LISTED IN THE
;*TABLES THE ROUTINE SETS THE "BADSEC" FLAG AND RETURNS. IF THE SECTOR
;*IS NOT LISTED THE FLAG IS RESET.
;*****
BDSRCK: SAVREG
        MOV #BSFACT, 1$ ;SET TABLE TO SEARCH
        JSR R4, SRHTBS ;GO SEARCH IT
1$:     .WORD ;TABLE ADDRESS GOES HERE
        MOV (SP)+, R3 ;GET NUMBER OF BAD SECTORS, IF ANY
        BNE 6$ ;IF ANY, GO TEST WHICH ONES
        CMP 1$, #BSSOFT ;ELSE TEST IF OTHER TABLE ALREADY SEARCHED
        BEQ 3$ ;IF YES-EXIT
        MOV #BSSOFT, 1$ ;SET OTHER TABLE FOR SEARCH
        BR 2$ ;GO SEARCH IT
3$:     BIC #BADSEC, RECODE ;CLEAR BAD SECTOR BIT
4$:     RESREG
        RTS PC ;RETURN
6$:     CMP (SP)+, R2 ;THIS SECTOR IN TABLE?
        BEQ 8$ ;YES-GO SET BIT & EXIT
        DEC R3 ;DECREMENT BAD SECTOR COUNT
        BNE 6$ ;IF NOT ZERO-CHECK NEXT ENTRY
        BR 7$ ;ELSE GO SEARCH OTHER TABLE
8$:     BIS #BADSEC, RECODE ;SET BAD SECTOR BIT
9$:     DEC R3 ;CLEAR STACK OF OTHER BAD SECTOR
        BEQ 4$ ;NUMBER
        TST (SP)+
        BR 9$ ;EXIT
;*****
;SBTTL INSERT BAD SECTOR IN BAD SECTOR SOFTWARE TABLE ROUTINE
;*****
BSSINS: SAVREG
        MOV #BSSOFT+10, R4 ;GET ADDRESS OF BAD SOFTWARE
1$:     TST (R4)+ ;SECTOR TABLE. FIND FIRST
        BPL 1$ ;EMPTY ENTRY
        TST -(R4) ;BACK UP POINTER
        MOV R0, (R4)+ ;INSERT CYLINDER
        MOV #10, R3 ;SET COUNT FOR SHIFT TO ALIGN
2$:     ASL R1 ;SHIFT TRACK TO ALIGN FOR ENTRY
        DEC R3 ;DEC SHIFT COUNT
        BNE 2$ ;TEST IF SHIFT DONE
        ADD R2, R1 ;ADD IN SECTOR NUMBER
        MOV R1, (R4) ;INSERT TRACK/SECTOR
        RESREG
        RTS PC
;*****
;SBTTL WRITE OPERATION CONTROL ROUTINE
;*****
WRTOP: CLRB OPCOMP ;CLEAR DONE FLAG

```

```

3260 022070 012737 022146 001110      MOV      #WERROR,SLPERR ;SETUP FOR LOOP ON ERROR
3261 022076 013700 006356      MOV      SCINUS,RO      ;GET STARTING ADDRESS
3262 022102 013701 006352      MOV      STINUS,R1
3263 022106 010037 006370      WRTLUP: MOV      RO,CCINUS    ;STORE CURRENT CYL
3264 022112 010137 006366      MOV      R1,CTINUS     ;STORE CURRENT TRACK
3265 022116 032777 001000 157014    BIT      #SW9,@SWR      ;TEST IF LOOP ON ERROR
3266 022124 001002      BNE      1$            ;IF YES - DO NOT CLEAR ERROR FLAG
3267 022126 105037 001103      CLRB     SERFLG        ;ELSE CLEAR FLAG
3268 022132 004737 021342      1$:      JSR      PC,BLDHDR ;GO BUILD HEADERS
3269 022136 004737 022206      JSR      PC,WRTHDR    ;GO WRITE HEADERS
3270 022142 004737 022312      JSR      PC,WRTTRK   ;GO WRITE DATA
3271 022146 032777 001000 156764    WERROR: BIT      #SW9,@SWR ;TEST IF LOOP ON ERROR
3272 022154 001406      BEQ      2$            ;IF NO - SKIP ERROR FLAG TEST
3273 022156 105737 001103      TSTB    SERFLG        ;ELSE TEST ERROR FLAG
3274 022162 001403      BEQ      2$            ;NO - SKIP
3275 022164 004737 031336      JSR      PC,PREPLP   ;GO PREPARE FOR LOOP
3276 022170 000746      BR       WRTLUP       ;GO TO LOOP
3277 022172 004737 021240      2$:      JSR      PC,TRKINC   ;BUMP TO NEXT ADDRESS
3278 022176 105737 002651      TSTB    OPCOMP        ;WRITE OPERATION DONE?
3279 022202 001741      BEQ      WRTLUP       ;NO-LOOP
3280 022204 000207      RTS      PC           ;ELSE RETURN TO DISPATCH
3281
3282 ;*****
3283 ;SBTTL WRITE HEADERS ROUTINE
3284 ;*****
3284 022206      WRTHDR: SAVREG
3285 022206 104411      CLR      P.PRST(R5)   ;CLEAR PROG STATUS REGISTER
3286 022210 005065 000014      MOV      #WRHEAD,P.CMND(R5) ;LOAD PARAMETER BLOCK WITH
3287 022214 112765 000127 000001    MOV      #BUFFO,P.BALO(R5) ;VALUES TO DO THE WRITE HEADER.
3288 022222 012765 005674 000010    MOV      RO,P.CYLN(R5)
3289 022230 010065 000002      MOV      #102,P.WC(R5) ;LOAD THE WORD COUNT WITH
3290 022234 012765 000102 000012    BIT      #B.CFMT,TYFMT ;THE PROPER VALUE FOR THE FORMAT
3291 022242 132737 000020 002653    BEQ      1$            ;TYPE
3292 022250 001403      MOV      #74,P.WC(R5)
3293 022252 012765 000074 000012    1$:      NEG      P.WC(R5)
3294 022260 005465 000012      MOV      R1,P.TRCK(R5)
3295 022264 110165 000005      JSR      PC,DRVCAL   ;CALL DRIVER
3296 022270 004737 025442      BIT      #ANYDER,RECODE ;TEST FOR ANY DATA ERRORS
3297 022274 032737 100000 006364    BEQ      2$
3298 022302 001401      ;INSERT CODE FOR ANY DATA TYPE ERROR ON WRITE HEADERS
3299
3300 022304 000000      HALT
3301 022306 104412      2$:      RESREG
3302 022310 000207      RTS      PC           ;RETURN
3303 ;*****
3304 ;SBTTL WRITE ONE TRACK ROUTINE
3305 ;*****
3306 022312      WRTTRK: SAVREG
3307 022312 104411      CLR      DERCNT       ;CLEAR ERROR COUNT
3308 022314 105037 002650 8$:      MOV      TKWDCT,P.WC(R5) ;SET WORD COUNT FOR ONE TRACK
3309 022320 013765 006306 000012    CLRB     P.SECT(R5)   ;SET SECTOR ZERO
3310 022326 105065 000004 4$:      MOV      #WEINUS,P.BALO(R5) ;SET BUFFER ADDRESS TO ODD
3311 022332 012765 006346 000010    BIT      #BITO,RO     ;OR EVEN CYLINDER DATA DEPENDING
3312 022340 032700 000001      BEQ      1$            ;ON THE CYLINDER NUMBER
3313 022344 001403      MOV      #WOINUS,P.BALO(R5)
3314 022346 012765 006350 000010    1$:      MOV      RO,P.CYLN(R5) ;SET UP THE PARAMETER BLOCK
3315 022354 010065 000002

```

K05

CZR6LCO RK06L CTG FMTR MACY11 30(1046) 01-DEC-77 14:28 PAGE 63
 CZR6LC.P11 01-DEC-77 14:20 WRITE ONE TRACK ROUTINE

SEQ 0062

3316	022360	110165	000005		MOV B	R1, P. TRACK(R5)	: TO WRITE THE ENTIRE TRACK
3317	022364	112765	000123	000001	MOV B	#WRDATA, P. CMND(R5)	
3318	022372	005065	000014		CLR	P. PRST(R5)	
3319	022376	052765	100000	000014	BIS	#DTBAIL, P. PRST(R5)	
3320	022404	004737	025442		JSR	PC, DRVCAL	: CALL DRIVER
3321	022410	032737	100000	006364	BIT	#ANYDER, RECODE	: ANY DATA ERROR?
3322	022416	001002			BNE	12\$: YES - PROCESS ERROR
3323	022420	000137	023060		JMP	7\$: NO-WRITE A-OK, EXIT
3324	022424	032737	000016	006364	12\$: BIT	#HVR CER! OPIERR! BSERR, RECODE	: VRC OR OPI ERROR
3325	022432	001002			BNE	14\$: YES-SKIP
3326	022434	000137	023046		JMP	6\$: GO PRINT FUNNY MESSAGE
3327	022440	016537	000026	001252	14\$: MOV	P. DTS(R5), \$TMP6	: STORE TRK/SEC
3328	022446	042737	174377	001252	BIC	#174377, \$TMP6	: CLEAR ALL BUT TRACK
3329	022454	113701	001253		MOV B	\$TMP6+1, R1	: STORE TRACK
3330	022460	016537	000022	001252	MOV	P. WCR(R5), \$TMP6	: STORE WORD COUNT
3331	022466	042737	000377	001252	BIC	#377, \$TMP6	: CLEAR PARTIAL COUNT
3332	022474	062737	000400	001252	ADD	#400, \$TMP6	: BUMP COUNT TO NEXT SECTOR
3333	022502	016502	000026		MOV	P. DTS(R5), R2	: & GET BAD SECTOR
3334	022506	042702	177740		BIC	#177740, R2	
3335	022512	032737	000002	006364	BIT	#BSERR, RECODE	: TEST IF BAD SECTOR
3336	022520	001076			BNE	3\$: GO SET UP TO WRITE REMAINDER IF TRACK
3337	022522	105237	002650		2\$: INCB	DERCNT	: BUMP ERROR COUNT
3338	022526	112765	000123	000001	MOV B	#WRDATA, P. CMND(R5)	: SET UP PARAM BLOCK
3339	022534	012765	177400	000012	MOV	#177400, P. WC(R5)	: TO WRITE ONE SECTOR
3340	022542	010065	000002		MOV	R0, P. CYLN(R5)	: THIS IS THE RETRY
3341	022546	110165	000005		MOV B	R1, P. TRACK(R5)	
3342	022552	110265	000004		MOV B	R2, P. SECT(R5)	
3343	022556	052765	100000	000014	BIS	#DTBAIL, P. PRST(R5)	
3344	022564	004737	025442		JSR	PC, DRVCAL	
3345	022570	032737	100000	006364	BIT	#ANYDER, RECODE	: ERROR IN RETRY?
3346	022576	001441			BEQ	5\$: NO-GO PRINT RETRY SUCCESSFUL
3347	022600	122737	000007	002650	CMP B	#7, DERCNT	: TEST IF ERROR COUNT TO BIG
3348	022606	002345			BGE	2\$: IF NO - GO RETRY AGAIN
3349	022610	005037	001174		CLR	\$REG5	
3350	022614	113737	002650	001174	MOV B	DERCNT, \$REG5	
3351	022622	104102			ERROR	102	
3352	022624	004737	021720		JSR	PC, BDSRCK	: GO TEST IF THIS SECTOR ALREADY MARKED BAD
3353	022630	032737	001000	006364	BIT	#BADSEC, RECODE	: TEST RESULTS OF TEST
3354	022636	001411			BEQ	11\$: NOT BAD BEFORE - SKIP
3355	022640	010037	001174		MOV	R0, \$REG5	: SET UP TO PRINT ERROR
3356	022644	010137	001176		MOV	R1, \$REG6	
3357	022650	010237	001200		MOV	R2, \$REG7	
3358	022654	104107			ERROR	107	: ERROR- BAD HEADER AREA
3359	022656	000137	016202		JMP	START	
3360	022662	104076		11\$:	ERROR	76	: REPORT MARKING SECTOR BAD
3361	022664	004737	022024		JSR	PC, BSSINS	: INSERT BAD SECTOR IN BSSOFT
3362	022670	004737	021342		JSR	PC, BLDHDR	: BUILD NEW HEADERS
3363	022674	004737	022206		JSR	PC, WRTHDR	: WRITE NEW HEADERS
3364	022700	000605			BR	8\$: RESTART WRITE TRACK
3365	022702	005037	001174		5\$: CLR	\$REG5	: CLEAR REG 5 FOR COUNT
3366	022706	113737	002650	001174	MOV B	DERCNT, \$REG5	: INSERT RETRY COUNT
3367	022714	104101			ERROR	101	: PRINT RETRY SUCCESSFUL
3368	022716	005202		3\$:	INC	R2	: THIS CODE SETS UP THE
3369	022720	020237	006342		CMP	R2, SEINUS	: PARAMETER BLOCK TO PICK
3370	022724	002055			BGE	7\$: UP THE TRACK WRITE
3371	022726	010265	000004		MOV	R2, P. SECT(R5)	: AFTER THE BAD SECTOR

```

3372 022732 012704 000010      MOV      #10,R4
3373
3374      ; THE FOLLOWING BLOCK OF CODE CHECKS THAT THE WORD COUNT IS
3375      ; CORRECTLY ADJUSTED TO PICK UP THE OPERATION AFTER THE SECTOR
3376      ; THAT IS CAUSING THE ERROR IS PROCESSED. IF THE WORD COUNT
3377      ; SOMEHOW GETS SCREWED UP THE PROGRAM HALTS. CONTINUE CAN BE
3378      ; DEPRESSED TO CAUSE THE WORD COUNT TO BE CORRECTED AND THE
3379      ; PROGRAM ADJUSTS THE COUNT AND CONTINUES THE OPERATION.
3380 022736 006302      9$:      ASL      R2
3381 022740 005304      DEC      R4
3382 022742 001375      BNE     9$
3383 022744 012704 013000      MOV     #13000,R4
3384 022750 023727 006342 000026      CMP     SEINUS,#26      ;ARE WE IN 26 SECTOR FORMAT?
3385 022756 001402      BEQ     13$           ;YES - SKIP
3386 022760 012704 012000      MOV     #12000,R4     ;ELSE CHANGE VALUE FOR 24 SECTOR FORMAT
3387 022764 160204      13$:     SUB     R2,R4
3388 022766 005404      NEG     R4
3389 022770 016502 000004      MOV     P.SECT(R5),R2
3390 022774 020437 001252      CMP     R4,$TMP6
3391 023000 001413      BEQ     10$
3392 023002 010437 001204      MOV     R4,$REG11
3393 023006 013737 001252 001206      MOV     $TMP6,$REG12
3394 023014 010237 001202      MOV     R2,$REG10
3395 023020 104105      ERROR  10$
3396 023022 000000      HALT
3397 023024 010437 001252      MOV     R4,$TMP6
3398
3399      ;END OF BLOCK OF CODE TO CHECK WORD COUNT
3400
3401 023030 013765 001252 000012 10$:     MOV     $TMP6,P.WC(R5)
3402 023036 105037 002650      CLRB   DERCNT
3403 023042 000137 022332      JMP     4$
3404 023046 104401 015203      6$:     TYPE  ,IMPERR      ;TYPE IMPOSSIBLE ERROR
3405 023052 000000      HALT                                ;HALT
3406 023054 000137 016202      JMP     START
3407 023060 104412      7$:     RESREG
3408 023062 000207      RTS     PC
3409
3410      ;*****
3411      ;SBTTL VERIFY OPERATION CONTROL ROUTINE
3412      ;*****
3412 023064      VEROP:
3413 023064 012737 023200 001110      MOV     #VERROR,$LPERR ;SET UP TO LOOP ON ERROR
3414 023072 013700 006356      MOV     SCINUS,R0      ;GET STARTING CYLINDER
3415 023076 013701 006352      MOV     STINUS,R1      ;AND TRACK
3416 023102 105037 002651      CLRB   OPCOMP
3417 023106 010037 006370      VERLUP: MOV     R0,CCINUS      ;STORE CURRENT CYL
3418 023112 010137 006366      MOV     R1,CTINUS      ;STORE CURRENT TRACK
3419 023116 032777 001000 156014      BIT     #SW9,$SWR      ;TEST IF LOOP ON ERROR
3420 023124 001002      BNE    1$            ;IF YES - DON'T CLEAR ERROR FLAG
3421 023126 105037 001103      CLRB   $ERFLG
3422 023132 004737 023240      1$:     JSR     PC,RDHDRS
3423 023136 004737 021342      JSR     PC,BLDHDR
3424 023142 004737 023576      JSR     PC,HDRCMP
3425 023146 132737 000010 002656      BITB   #RCDASW,OPCONT
3426 023154 001403      BEQ    2$
3427 023156 004737 024740      JSR     PC,RDOP      ;GO DO READ FOR DATA VERIFICATION

```


MOS

CZR6LCO RK06L CTG FMTR MACY11 30(1046) 01-DEC-77 14:28 PAGE 65
 CZR6LC.P11 01-DEC-77 14:20 VERIFY OPERATION CONTROL ROUTINE

SEQ 0064

```

3428 023162 000406          BR      VERROR
3429 023164 132737 000004 002656 2$: BITB  #WCDASW,OPCONT ;CHECK IF WRITE CHECK DATA
3430 023172 001402          BEQ    VERROR ;IF NO - SKIP WRITE CHECK
3431 023174 004737 023714      JSR    PC,WCOF ;ELSE GO DO WRITE CHECK FOR DATA VERIF.
3432 023200 032777 001000 155732 VERROR: BIT  #SW9,JSWR ;TEST IF LOOP ON ERROR
3433 023206 001406          BEQ    4$ ;IF NO - GO DO NEXT TRACK
3434 023210 105737 001103      TSTB  SERFLG ;ELSE CHECK ERROR FLAG
3435 023214 001403          BEQ    4$ ;NO SKIP
3436 023216 004737 031336      JSR    PC,PREPLP ;GO PREPARE FOR LOOPING
3437 023222 000731          BR     VERLUP ;LOOP
3438 023224 004737 021240      4$: JSR    PC,TRKINC
3439 023230 105737 002651      TSTB  OPCOMP
3440 023234 001724          BEQ    VERLUP
3441 023236 000207          RTS    PC
3442
3443 ;*****
3444 ;SBTTL READ HEADERS ROUTINE
3445 ;*THE HEADERS FOR A GIVEN CYLINDER AND TRACK (SPECIFIED IN R0 AND R1,
3446 ;*RESPECTIVELY) ARE READ IN AND SORTED INTO THE REQUIRED SEQUENCE.
3447 ;*****
3448 R0HDS: SAVREG
3449 JSR    PC,POSOFF ;GO POSITION AND OFFSET
3450 MOV    #RDBUF,P,BALO(R5) ;LOAD ADDRESS FOR HEADERS
3451 CLR    P,PRST(R5) ;CLEAR PROGRAM STATUS
3452 2$: MOV    R0,P,CYLN(R5) ;SET UP TO READ HEADER
3453 MOV    R1,P,TRCK(R5) ;OF DESIRED TRACK & CYLINDER
3454 CLR    #STKS ;TURN OFF INTERRUPTS FROM KEYBOARD
3455 MOV    #RDALHD,P,CMND(R5)
3456 JSR    PC,DRVCAL ;DO IT
3457 MOV    #100,#STKS ;TURN ON KEYBOARD
3458 MOV    #204,$TMPD ;SET UP TEMP STORE FOR THE NUMBER OF
3459 CMP    SEINUS,#26 ;HEADER WORDS THAT MUST BE STORED FOR
3460 BEQ    9$ ;THE FORMAT IN USE.
3461 MOV    #170,$TMPD
3462 9$: CLR    R3
3463 4$: CMP    RDBUF(R3),R0 ;TEST IF FIRST HEADER WORD IS RIGHT CYL
3464 BNE    3$ ;NO - GO BUMP TO NEXT
3465 MOV    RDBUF+2(R3),R4 ;GET TRACK/SECTOR WORD
3466 BIC    #177437,R4 ;CLEAR ALL BUT TRACK BITS
3467 ASR    R4 ;SHIFT TRACK TO ALIGN FOR TESTING
3468 ASR    R4
3469 ASR    R4
3470 ASR    R4
3471 ASR    R4
3472 CMP    R4,R1 ;CHECK IF TRACK IS CORRECT
3473 BNE    3$ ;NO - GO BUMP TO NEXT HEADER
3474 MOV    RDBUF+2(R3),R4 ;GET SECOND WORD
3475 MOV    RDBUF(R3),R2 ;GET FIRST WORD
3476 BIC    RDBUF(R3),R4 ;COMPUTE VRC OF HEADER
3477 BIC    RDBUF+2(R3),R2
3478 BIS    R4,R2
3479 CMP    RDBUF+4(R3),R2 ;CHECK IF VRC IS CORRECT
3480 BEQ    5$ ;YES - SKIP BUMP TO NEXT HEADER
3481 3$: ADD    #6,R3 ;BUMP TO NEXT HEADER
3482 CMP    R3,$TMPD ;TEST IF END OF HEADERS REACHED
3483 BLE    4$ ;NO - GO TEST NEXT HEADER

```

```

3484 023440 010037 001202      MOV      R0,$REG10      ;SET UP CYLINDER AND TRACK FOR REPORT
3485 023444 010137 001204      MOV      R1,$REG11
3486 023450 104103      ERROR 103              ;REPORT ERROR
3487 023452 000137 016202      JMP      START
3488 023456 016304 002662      5$: MOV      RDBUF+2(R3),R4 ;GET TRACK/SECTOR AGAIN
3489 023462 042704 177740      BIC      #177740,R4     ;CLEAR ALL BUT SECTOR
3490 023466 020437 006342      CMP      R4,SEINUS     ;TEST IF A VALID SECTOR NUMBER
3491 023472 002355      BGE      3$            ;NO - GO BUMP TO NEXT SECTOR
3492 023474 013702 006342      MOV      SEINUS,R2     ;INITIALIZE COUNTER FOR MOV
3493 023500 010401      MOV      R4,R1         ;SET THE INDEX VALUE BY MULT SECTOR READ BY 6
3494 023502 006304      ASL      R4
3495 023504 006304      ASL      R4
3496 023506 060104      ADD      R1,R4
3497 023510 060104      ADD      R1,R4
3498 023512 016364 002660 006100 6$: MOV      RDBUF(R3),BUFF1(R4) ;MOV WORD ONE
3499 023520 016364 002662 006102      MOV      RDBUF+2(R3),BUFF1+2(R4) ;MOV WORD TWO
3500 023526 016364 002664 006104      MOV      RDBUF+4(R3),BUFF1+4(R4) ;MOV WORD THREE
3501 023534 005302      DEC      R2            ;DECREMENT COUNT
3502 023536 001415      BEQ      8$            ;COUNT ZERO MOVE DONE, EXIT
3503 023540 062704 000006      ADD      #6,R4         ;BUMP INDEX POINTERS TO NEXT HEADER
3504 023544 062703 000006      ADD      #6,R3
3505 023550 020337 001236      CMP      R3,$TMPO     ;TEST IF END OF RDBUF REACHED
3506 023554 002401      BLT      7$            ;NO - SKIP RESET TO START OF BUFFER
3507 023556 005003      CLR      R3           ;ELSE RESET R3
3508 023560 020437 001236      7$: CMP      R4,$TMPO     ;SAME TEST FOR BUFF1
3509 023564 002752      BLT      6$            ;IF NOT END OF BUFFER, GO DO NEST MOVE
3510 023566 005004      CLR      R4           ;ELSE RESET R4 TO START OF BUFFER
3511 023570 000750      BR       6$           ;THEN GO DO NEXT MOVE
3512 023572 104412      8$: RESREG
3513 023574 000207      RTS      PC
3514
3515      ;*****
3516      ;SBTTL HEADER COMPARE ROUTINE
3517      ;*****
3517 023576      HDRCMP:
3518 023576 104411      SAVREG
3519 023600 005002      CLR      R2
3520 023602 013704 006342      MOV      SEINUS,R4     ;GET NUM OF SECTORS
3521 023606 026262 005674 006100 1$: CMP      BUFF0(R2),BUFF1(R2) ;COMPARE HDR WORD 1
3522 023614 001015      BNE      2$
3523 023616 026262 005676 006102      CMP      BUFF0+2(R2),BUFF1+2(R2) ;COMPARE HDR WORD 2
3524 023624 001011      BNE      2$
3525 023626 026262 005700 006104      CMP      BUFF0+4(R2),BUFF1+4(R2) ;COMPARE HDR WORD 3
3526 023634 001005      BNE      2$
3527 023636 062702 000006      3$: ADD      #6,R2         ;BUMP INDEX
3528 023642 005304      DEC      R4           ;DEC COUNT
3529 023644 001360      BNE      1$           ;LOOP UNTIL ZERO
3530 023646 000420      BR       4$           ;EXIT
3531 023650 012703 001174      2$: MOV      #REG5,R3     ;SET UP FOR ERROR REPORT
3532 023654 016223 005674      MOV      BUFF0(R2),(R3)+
3533 023660 016223 005676      MOV      BUFF0+2(R2),(R3)+
3534 023664 016223 005700      MOV      BUFF0+4(R2),(R3)+
3535 023670 016223 006100      MOV      BUFF1(R2),(R3)+
3536 023674 016223 006102      MOV      BUFF1+2(R2),(R3)+
3537 023700 016223 006104      MOV      BUFF1+4(R2),(R3)+
3538 023704 104062      ERROR 62              ;REPORT ERROR
3539 023706 000753      BR       3$           ;GO TEST NEXT HEADER

```

```

3540 023710 104412          4$: RESREG          ;EXIT
3541 023712 000207          RTS      PC
3542                                     ;*****
3543                                     ;SBTTL DATA VERIFY WITH WRITE CHECK OPERATION ROUTINE
3544                                     ;*****
3545                                     WCOP:
3546 023714          SAVREG
3547 023716 104411 006306 000012 12$: MOV      TKWDCI,P.WC(R5) ;SET WORD COUNT FOR 1 TRACK
3548 023724 005037 002650          CLR      DERCNT      ;CLEAR ERROR COUNT
3549 023730 105065 000004          CLRB     P.SECT(R5)   ;SET TO SECTOR 0
3550 023734 004737 031262          JSR      PC,POSOFF   ;GO POSITION AND OFFSET
3551 023740 012765 006346 000010 5$: MOV      #WEINUS,P.BALO(R5) ;SET UP FOR EVEN OR ODD
3552 023746 032700 000001          BIT      #BITO,R0    ;CYLINDER DATA
3553 023752 001403
3554 023754 012765 006350 000010 1$: MOV      #WOINUS,P.BALO(R5)
3555
3556 023762 010065 000002          MOV      R0,P.CYLN(R5) ;SET UP PARAMETER BLOCK
3557 023766 110165 000005          MOVB     R1,P.TRACK(R5) ;FOR 1 TRACK WRITE CHECK
3558 023772 005065 000014          CLR      P.PRST(R5)
3559 023776 052765 100000 000014 16$: BIS      #DTBARI,P.PRST(R5)
3560 024004 112765 000131 000001  MOVB     #WRTCHK,P.CMND(R5)
3561 024012 004737 025442          JSR      PC,DRVCAL
3562 024016 032737 100000 006364  BIT      #ANYDER,RECODE ;TEST IF ANY ERROR
3563 024024 001002          BNE     16$
3564 024026 000137 024566          JMP      15$
3565 024032 016503 000022          MOV      P.WCR(R5),R3 ;STORE WORD COUNT & SECTOR
3566 024036 016502 000026          MOV      P.DTS(R5),R2 ;NUMBER AT ERROR TIME
3567 024042 042702 177740          BIC      #177740,R2   ;CLEAR ALL BUT SECTOR NUMBER
3568 024046 042703 000377          BIC      #377,R3     ;MAKE THE WORD COUNT
3569                                     ;MODULO 400
3570 024052 032737 000016 006364  BIT      #OPIERR!HVR CER!BSERR,RECODE ;TEST IF OPI OR HVRC
3571 024060 001013          BNE     4$           ;IF YES-SKIP
3572 024062 032737 000020 006364  BIT      #DCKERR,RECODE ;TEST IF DATA CHECK
3573 024070 001402          BEQ     2$           ;IF NO, WD COUNT IS NOW CORRECT-SKIP
3574 024072 162703 000400          SUB      #400,R3     ;ELSE INCREASE WORD COUNT
3575 024076 005702          TST     R2           ;TEST IF SECTOR NOW ZERO
3576 024100 001002          BNE     3$           ;IF NO-GO DECREMENT IT
3577 024102 013702 006342          MOV      SEINUS,R2   ;ELSE GET MAX SECTOR +1 VALUE
3578 024106 005302          DEC     R2           ;NOW DECREMENT IT
3579 024110 032737 000002 006364  4$: BIT      #BSERR,RECODE ;TEST IF BAD SECTOR
3580 024116 001453          BEQ     6$           ;IF YES - GO CONTINUE TRACK
3581 024120 005202          INC     R2           ;ELSE BUMP TO NEXT SECTOR
3582 024122 020237 006342          CMP      R2,SEINUS  ;IF NOT AT LAST SECTOR -
3583 024126 002402          BLT     19$         ;GO TO EXIT IF LAST SECTOR IN ERROR
3584 024130 000137 024566          JMP      15$
3585 024134 105037 002650          19$: CLRB     DERCNT    ;CLEAR ERROR COUNT
3586 024140 010265 000004          MOV      R2,P.SECT(R5) ;SET UP FOR CONTINUE OF WC
3587 024144 062703 000400          ADD     #400,R3     ;BUMP COUNT TO NEXT SECTOR
3588 024150 012704 000010          MOV      #10,R4
3589
3590                                     ;THE FOLLOWING BLOCK OF CODE CHECKS THAT THE WORD COUNT IS
3591                                     ;CORRECTLY ADJUSTED TO PICK UP THE OPERATION AFTER THE SECTOR
3592                                     ;THAT IS CAUSING THE ERROR IS PROCESSED. IF THE WORD COUNT
3593                                     ;SOMEHOW GETS SCREWED UP THE PROGRAM HALTS. CONTINUE CAN BE
3594                                     ;DEPRESSED TO CAUSE THE WORD COUNT TO BE CORRECTED AND THE
3595                                     ;PROGRAM ADJUSTS THE COUNT AND CONTINUES THE OPERATION.

```

```

3596 024154 006302          17$: ASL      R2
3597 024156 005304          DEC      R4
3598 024160 001375          BNE     17$
3599 024162 012704 013000    MOV     #13000,R4
3600 024166 023727 006342 000026  CMP     SEINUS,#26 ;TEST IF WE ARE IN 26 SECTOR FORMAT
3601 024174 001402          BEQ     20$ ;YES - SKIP
3602 024176 012704 012000    MOV     #12000,R4 ;ELSE SET COUNT FOR 24 SECTOR FORMAT
3603 024202 160204          20$: SUB     R2,R4
3604 024204 005404          NEG     R4
3605 024206 016502 000004    MOV     P.SECT(R5),R2
3606 024212 020403          CMP     R4,R3
3607 024214 001411          BEQ     18$
3608 024216 010437 001204    MOV     R4,$REG11
3609 024222 010337 001206    MOV     R3,$REG12
3610 024226 010237 001202    MOV     R2,$REG10
3611 024232 104105          ERROR  10$
3612 024234 000000          HALT
3613 024236 010403          MOV     R4,R3
3614
3615          ;END OF CODE TO CHECK WORD COUNT
3616
3617 024240 010365 000012          18$: MOV     R3,P.WC(R5) ;AT NEXT SECTOR
3618 024244 000633          BR     5$ ;GO DO IT
3619 024246 105237 002650          6$: INCB   DERCNT ;RETRY SEQUENCE-
3620 024252 004737 031262          JSR    PC,POSOFF ;GO POSITION AND OFFSET
3621 024256 112765 000131 000001    MOVB   #WATCHK,P.CMND(R5) ;COUNT THE ERROR
3622 024264 010065 000002          MOV     R0,P.CYLN(R5) ;SET UP PARAM BLOCK
3623 024270 110165 000005          MOVB   R1,P.TRCK(R5)
3624 024274 110265 000004          MOVB   R2,P.SECT(R5)
3625 024300 012765 177400 000012    MOV     #177400,P.WC(R5)
3626 024306 052765 100000 000014    BIS    #DTBAII,P.PRST(R5)
3627 024314 004737 025442          JSR    PC,DRVCAL
3628 024320 032737 100000 006364    BIT    #ANYDER,RECODE ;TEST IF ANY ERROR
3629 024326 001507          BEQ    14$ ;GO TYPE RETRY SUCCESSFUL IF NO
3630 024330 022737 000007 002650    CMP    #7,DERCNT ;ELSE TEST IF ENOUGH RETRIES
3631 024336 002343          BGE    6$ ;NO-LOOP
3632 024340 032737 000100 006364    BIT    #WCERR,RECODE ;TEST IF WCE, IF
3633 024346 001446          BEQ    10$ ;NO GO CHECK FOR DATA CHECK
3634 024350 004737 031262          JSR    PC,POSOFF ;GO POSITION AND OFFSET
3635 024354 112765 000121 000001    MOVB   #RDATA,P.CMND(R5) ;ELSE DO A READ OF SECTOR
3636 024362 012765 177400 000012    MOV     #177400,P.WC(R5) ;IN ERROR
3637 024370 010065 000002          MOV     R0,P.CYLN(R5)
3638 024374 110165 000005          MOVB   R1,P.TRCK(R5)
3639 024400 110265 000004          MOVB   R2,P.SECT(R5)
3640 024404 005065 000014          CLR    P.PRST(R5) ;CLEAR PROGRAM STATUS WORD (BAII)
3641 024410 012765 002660 000010    MOV     #RDBUF,P.BALO(R5)
3642 024416 004737 025442          JSR    PC,DRVCAL
3643 024422 032737 100000 006364    BIT    #ANYDER,RECODE ;IF NO ERROR ON READ
3644 024430 001442          BEQ    13$
3645 024432 032737 000020 006364    BIT    #DCKERR,RECODE ;OR NO DATA CHECK
3646 024440 001436          BEQ    13$ ;GO PRINT FUNNY MESSAGE
3647 024442 004737 024572          JSR    PC,DACOMP ;ELSE COMPARE DATA & PRINT ERRORS
3648 024446 032737 000040 006364  9$: BIT    #ECCNC,RECODE ;CHECK IF NON-CORRECTABLE
3649 024454 001007          BNE    11$ ;IF YES-GO TO BAD SECTOR PROCESSING
3650 024456 104401 015256          TYPE  CORABL ;ELSE TYPE CORRECTABLE MESSAGE
3651 024462 000616          BR     8$ ;& GO WC REST OF TRACK

```

```

3652 024464 032737 000020 006364 10$: BIT #DCKERR,RECODE ;THIS IS TEST IF DAK WITH NO WCE
3653 024472 001365 000000 000000 11$: BNE #95 ;IF YES-GO TEST FOR CORRECTABLE
3654 024474 005037 001174 001174 11$: CLR $REG5
3655 024500 113737 002650 001174 11$: MOV# DERCNT,$REG5
3656 024506 104102 000000 000000 11$: ERROR 102
3657 024510 104076 000000 000000 11$: ERROR 76 ;REPORT MARKING SECTOR BAD
3658 024512 004737 022024 000000 11$: JSR PC,$SSINS ;PUT BAD SECTOR IN BAD SECTOR
3659 024516 004737 021392 000000 11$: JSR PC,BLDADR ;TABLE, COMPUTE NEW HEADERS,
3660 024522 004737 022206 000000 11$: JSR PC,WRTTHR ;WRITE THEM, REWRITE THE
3661 024526 004737 022312 000000 11$: JSR PC,WRTTRK ;TRACK
3662 024532 000137 023716 000000 11$: JMP 12$ ;& REDO THE WRITE CHECK
3663 024536 104401 015111 000000 13$: TYPE IMPER2 ;TYPE FUNNY MESSAGE
3664 024542 000137 023716 000000 13$: JMP 12$ ;GO RESTART THE WRITE CHECK
3665 024546 005037 001174 001174 14$: CLR $REG5
3666 024552 113737 002650 001174 14$: MOV# DERCNT,$REG5 ;GET RETRY COUNT
3667 024560 104101 000000 000000 14$: ERROR 101 ;RETRY SUCCESSFUL MESSAGE
3668 024562 000137 024120 000000 14$: JMP 8$
3669 024566 104412 000000 000000 15$: RESREG ;EXIT
3670 024570 000207 000000 000000 15$: RTS PC
3671 *****
3672 .SBTTL DATA COMPARE ROUTINE
3673 *****
3674 DACOMP:
3675 024572 104411 SAVREG
3676 024574 005005 CLR R5 ;CLEAR R5 FOR ERROR COUNTING
3677 024576 012703 006346 MOV #WEINUS,R3 ;GET THE WORD THAT SHOULD BE
3678 024602 032700 000001 BIT #BIT0,R0 ;WRITTEN THROUGHOUT THE CYLINDER
3679 024606 001402 BEQ 1$
3680 024610 012703 006350 MOV #WOINUS,R3
3681 024614 012704 002660 1$: MOV #RDBUF,R4 ;SET POINTER TO READ DATA
3682 024620 010037 001174 MOV R0,$REG5 ;STORE PACK ADDRESS FOR REPORT
3683 024624 010137 001176 MOV R1,$REG6
3684 024630 010237 001200 MOV R2,$REG7
3685 024634 012701 000001 MOV #1,R1 ;PRESET REGISTERS
3686 024640 012702 000001 MOV #1,R2 ;FOR COUNTING
3687 024644 012700 000400 MOV #400,R0
3688 024650 021324 2$: CMP (R3),(R4)+ ;COMPARE DATA
3689 024652 001005 BNE 3$ ;GO REPORT IF NOT EQUAL
3690 024654 005202 5$: INC R2 ;ELSE BUMP COUNTERS
3691 024656 005300 DEC R0
3692 024660 001373 BNE 2$ ;LOOP UNTIL DONE
3693 024662 104412 6$: RESREG
3694 024664 000207 RTS PC
3695 024666 005205 3$: INC R5 ;BUMP ERROR COUNT
3696 024670 005301 DEC R1 ;BUMP R1 TO ZERO
3697 024672 011337 001202 MOV (R3),$REG10 ;STORE GOOD, BAD, & WORD NUM
3698 024676 014437 001204 MOV -(R4),$REG11
3699 024702 010237 001206 MOV R2,$REG12
3700 024706 005724 TST (R4)+ ;BUMP R4 TO NEXT DATA WORD
3701 024710 005701 TST R1 ;TEST R1 SWITCH
3702 024712 001001 BNE 4$ ;IF NOT ZERO, THIS IS NOT FIRST PASS
3703 024714 104034 ERROR 34 ;TYPE HEADING
3704 024716 032777 000002 154214 4$: BIT #SW1,$SWR ;TEST IF NO LIMIT ON ERRORS
3705 024724 001003 BNE 7$ ;SWITCH 1 OFF - LIMIT TO 10
3706 024726 020527 000010 CMP R5,#10 ;ELSE TEST IF 10 ERRORS REPORTED
3707 024732 001753 BEQ 6$ ;YES - EXIT

```

```

3708 024734 104063
3709 024736 000746
3710
3711
3712
3713 024740
3714 024740 104411
3715 024742 105065 000004
3716 024746 105037 002650
3717 024752 013765 006306 000012
3718 024760 004737 031262
3719 024764 010065 000002
3720 024770 110165 000005
3721 024774 005065 000014
3722 025000 052765 100000 000014
3723 025006 112765 000121 000001
3724 025014 004737 025442
3725 025020 032737 100000 006364
3726 025026 001002
3727 025030 000137 025436
3728 025034 016537 000022 001254
3729 025042 016502 000026
3730 025046 042702 177400
3731 025052 062737 000400 001254
3732 025060 032737 000002 006364
3733 025066 001106
3734 025070 032737 000020 006364
3735 025076 001410
3736 025100 005702
3737 025102 001002
3738 025104 013702 006342
3739 025110 005302
3740 025112 162737 000400 001254
3741 025120 105237 002650
3742 025124 004737 031262
3743 025130 112765 000121 000001
3744 025136 010065 000002
3745 025142 110165 000005
3746 025146 110265 000004
3747 025152 012765 177400 000012
3748 025160 012765 001240 000010
3749 025166 052765 100000 000014
3750 025174 004737 025442
3751 025200 032737 100000 006364
3752 025206 001504
3753 025210 122737 000007 002650
3754 025216 002340
3755 025220 005037 001174
3756 025224 113737 002650 001174
3757 025232 104102
3758 025234 032737 000020 006364
3759 025242 001420
3760 025244 016537 000060 001174
3761 025252 016537 000062 001176
3762 025260 005037 001200
3763 025264 032737 000040 006364

```

```

7$: ERROR 63 ;TYPE DATA
BR 5$ ;GO CONTINUE COMPARE
;*****
:SBTTL DATA VERIFY WITH READ OPERATION ROUTINE
:*****
RDOP:
SAVREG
CLRB P.SECT(R5) ;SET UP TO READ A FULL TRACK
CLRB DERCNT ;WITH BUS INCREMENT INHIBIT
MOV TKWDCI,P.WC(R5)
6$: JSR PC,POSOFF ;GO POSITION AND OFFSET
MOV RO,P.CYLN(R5)
MOVB R1,P.TRACK(R5)
CLR P.PRST(R5)
BIS #DTBAIL,P.PRST(R5)
MOVB #RDATA,P.CMND(R5)
JSR PC,DRVCAL ;DO IT
BIT #ANYDER,RECODE ;TEST IF ANY DATA ERROR
BNE 11$ ;READ A-OK, EXIT
JMP 8$
11$: MOV P.WC(R5),STMP7 ;STORE WORD COUNT AT ERROR
MOV P.DTS(R5),R2 ;STORE SECTOR & TRACK
BIC #177400,R2 ;CLEAR ALL BUT SECTOR
ADD #400,STMP7 ;BUMP STORED WORD COUNT BY 1 SECTOR
BIT #BSERR,RECODE ;CHECK IF BAD SECTOR
BNE 5$ ;YES - GO CONTINUE READ
BIT #DCKERR,RECODE ;CHECK IF DATA CHECK
BEQ 3$ ;IF YES, DECREMENT SECTOR COUNT BY
R2 ;1 TO GET SECTOR IN ERROR BUT BE
BNE 1$ ;SURE SECTOR COUNT IS NOT ZERO
MOV SEINUS,R2
1$: DEC R2
SUB #400,STMP7 ;IF DCK, ADD WORD COUNT BACK IN
3$: INCB DERCNT ;BUMP ERROR COUNT
JSR PC,POSOFF ;GO POSITION AND OFFSET
MOVB #RDATA,P.CMND(R5) ;ELSE SET UP TO RETRY
MOV RO,P.CYLN(R5)
MOVB R1,P.TRACK(R5)
MOVB R2,P.SECT(R5)
MOV #177400,P.WC(R5)
MOV #STMP1,P.BALO(R5)
BIS #DTBAIL,P.PRST(R5)
JSR PC,DRVCAL
BIT #ANYDER,RECODE ;ANY DATA ERROR IN RETRY?
BEQ 7$ ;NO-GO TYPE RETRY SUCCESSFUL
CMPB #7,DERCNT ;ELSE CHECK IF ERROR COUNT TO BIG
BGE 3$ ;IF YES-TYPE UNSUCCESSFUL
CLR $REGS
MOVB DERCNT,$REGS
ERROR 102
BIT #DCKERR,RECODE ;TEST IF THIS IS A DATA ERROR
BEQ 5$ ;IF NO - SKIP ECC PRINTOUT
MOV P.EPOS(R5),$REG5 ;STORE ECC PATTERN AND POSITION
MOV P.EPAT(R5),$REG6 ;FOR ERROR REPORT
CLR $REG7 ;SET $REG7 FOR ECC UNCORRECTABLE
BIT #ECCNC,RECODE ;WAS THAT CORRECT?

```

F06

CZR6LCO RK06L CTG FMTR MACY11 30(1046)
 CZR6LC.P11 01-DEC-77 14:20

01-DEC-77 14:28 PAGE 71
 DATA VERIFY WITH READ OPERATION ROUTINE

SEQ 0070

```

3764 025272 001003          BNE      4$          ;YES - GO REPORT
3765 025274 052737 000001 001200  BIS     #1,$REG7    ;NO - SET FOR CORRECTABLE
3766 025302 104077          4$:  ERROR     77    ;REPORT ERROR
3767 025304 005202          5$:  INC      R2    ;SET UP TO CONTINUE ON NEXT SECTOR
3768 025306 020237 006342  CMP     R2,$EINUS  ;CHECK IF ERROR SECTOR WAS LAST SECTOR
3769 025312 002051          BGE     $S        ;EXIT IF YES
3770 025314 010265 000004  MOV     R2,F.SECT(R5) ;READ OF REST OF TRACK BY
3771 025320 012704 000010  MOV     #10,R4
3772 025324 006302          9$:  ASL     R2
3773 025326 005304          DEC     R4
3774 025330 001375          BNE     9$
3775 025332 012704 013000  MOV     #13000,R4
3776 025336 160204          SUB     R2,R4
3777 025340 005404          NEG     R4
3778 025342 016502 000004  MOV     P.SECT(R5),R2
3779 025346 020437 001254  CMP     R4,$TMP7
3780 025352 001413          BEQ     10$
3781 025354 010437 001204  MOV     R4,$REG11
3782 025360 013737 001254 001206  MOV     $TMP7,$REG12
3783 025366 010237 001202  MOV     R2,$REG10
3784 025372 104105          ERROR   10$
3785 025374 000000          HALT
3786 025376 010437 001254  MOV     R4,$TMP7
3787 025402 013765 001254 000012 10$:  MOV     $TMP7,P.WC(R5) ;BUMPING TO THE NEXT SECTOR
3788 025410 105037 002650          CLRB   DEFCNT     ;INSERTING THE WORD COUNT &
3789 025414 000137 024760          JMP
3790 025420 005037 001174 7$:  CLR     $REG5    ;GO START RD DATA
3791 025424 113737 002650 001174  MOVB   DEFCNT,$REG5 ;CLEAR REG 5 FOR RETRY COUNT BYTE
3792 025432 104101          ERROR  101
3793 025434 000723          BR     5$
3794 025436 104412          8$:  RESREG
3795 025440 000207          RTS     PC
3796
3797 ;*****
3798 ;SBTTL CALL DRIVER ROUTINE
3799 ;ENTRY JSR PC,DRVCAL
3800 ;* WITH R5 POINTING TO PARAMETER BLOCK
3801 ;*RETURN RTS PC
3802 ;*
3803 ;*THIS ROUTINE IS USED TO INITIATE A SUBSYSTEM OPERATION BY
3804 ;*CALLING THE DRIVER. THE PARAMETER BLOCK MUST BE SET UP
3805 ;*BY THE CALLER AND R5 MUST POINT TO THE PARAMETER
3806 ;*BLOCK WHEN THE ROUTINE IS CALLED.
3807 ;*
3808 ;*THIS ROUTINE WAITS FOR THE OPERATION TO BE COMPLETED.
3809 ;*WHILE WAITING THE WATCHDOG TIMER IS CALLED TO PREVENT
3810 ;*SILENT DEATH IN CASE THE SUBSYSTEM DOES NOT PROVIDE AN
3811 ;*INTERRUPT. THE TERMINATION HANDLER ROUTINES WILL SET THE DONE
3812 ;*FLAG WHICH KEYS THE ROUTINE TO RETURN TO THE CALLER.
3813 ;*****
3813 025442 105037 002652  DRVCAL: CLRB   DONE          ;CLEAR DONE FLAG
3814 025446 010146          MOV     R1,-(SP)    ;STORE REGS
3815 025450 010546          MOV     R5,-(SP)
3816 025452 012701 005660  MOV     #COMSTR,R1  ;GET ADDRESS OF COMMAND STORAGE
3817 025456 012521          MOV     (R5)+,(R1)+ ;STORE PARAMETER VALUES
3818 025460 012521          MOV     (R5)+,(R1)+ ;IN COMMAND STORAGE
3819 025462 012521          MOV     (R5)+,(R1)+

```

```

3820 025464 012521          MOV      (R5)+,(R1)+
3821 025466 012521          MOV      (R5)+,(R1)+
3822 025470 012521          MOV      (R5)+,(R1)+
3823 025472 012605          MOV      (SP)+,R5
3824 025474 012601          MOV      (SP)+,R1
3825 025476 010537 025506    MOV      R5,1$          ;GET PARAM BLOCK ADDRESS
3826 025502 004737 035742    JSR      PC,C.INIT      ;CALL DRIVER
3827 025506 000000          1$:      .WORD          ;P.B. ADDRESS GOES HERE
3828 025510 004737 032312    2$:      JSR      PC,W.WTCH  ;CALL WATCH DOG
3829 025514 105737 002652    TSTB    DONE           ;DONE SET?
3830 025520 001773          BEQ      2$            ;NO-LOOP
3831 025522 000207          RTS      PC            ;YES-RETURN
3832
3833          ;*****
3834          ;SBTTL DRIVE ERROR FREE RETURN ROUTINE
3835          ;*THIS ROUTINE IS CALLED BY THE DRIVER WHEN NO ERROR
3836          ;*HAS BEEN DETECTED IN THE OPERATION. THE ROUTINE SETS THE
3837          ;*DONE FLAG THAT IS TESTED IN THE DRIVER CALLING
3838          ;*ROUTINE.
3839          ;*****
3839 025524 152737 000377 002652  ERRFRE: BISB    #377,DONE  ;SET THE DONE FLAG
3840 025532 032737 100000 006364    BIT     #ANYDER,RECODE ;TEST IF ANY DATA ERROR
3841 025540 001403          BEQ     2$            ;IF NO - DO ERROR RECOVERY PRINT TEST
3842 025542 105037 002654          CLRB   ERRCNT        ;CLEAR ERROR COUNT
3843 025546 000413          BR     1$            ;EXIT
3844 025550 105737 002654    2$:      TSTB   ERRCNT        ;CHECK IF ANY ERRORS HAVE OCCURRED
3845 025554 001410          BEQ     1$            ;NO - SKIP TO EXIT
3846 025556 005037 001174          CLR    $REGS         ;GET RETRY COUNT
3847 025562 113737 002654 001174    MOVB   ERRCNT,$REGS  ;PRINT RETRY SUCCESSFUL MESSAGE
3848 025570 104101          ERROR  101          ;CLEAR ERROR COUNT
3849 025572 105037 002654          CLRB   ERRCNT        ;CLEAR RECOVERY FLAGS
3850 025576 005037 006364    1$:      CLR    RECODE      ;RETURN
3851 025602 000207          RTS     PC
3852          ;*****
3853          ;SBTTL TYPE ERROR ROUTINE
3854          ;*ENTRY JSR PC TYP ERR
3855          ;*RETURN RTS PC
3856          ;*
3857          ;*THIS ROUTINE USES THE "ITEM CONTROL BYTE" ($ITEMB) TO DETERMINE WHICH
3858          ;*ERROR IS TO BE REPORTED. IT THEN USES THE "ERROR TABLE" ($ERRTB)
3859          ;*ENTRY TO DEFINE WHAT INFORMATION IS TO BE REPORTED CONCERNING
3860          ;*THE ERROR.
3861          ;*****
3862 025604 104411          TYPERR: SAVREG
3863 025606 032777 020000 153324    BIT     #SW13,$SWR    ;INHIBIT ERROR TYPEOUTS?
3864 025614 001107          BNE    20$          ;YES-BRANCH
3865 025616 113700 001114          MOVB   $ITEMB,RO    ;ENTER ERROR NUMBER
3866 025622 042700 177400          BIC    #177400,RO   ;CLEAR UPPER BITS
3867 025626 005300          DEC    RO           ;FORM INDEX FOR ERROR TABLE
3868 025630 006300          ASL    RO
3869 025632 006300          ASL    RO
3870 025634 006300          ASL    RO
3871 025636 062700 001272    1$:      ADD     $ERRTB,RO    ;FORM ADDRESS OF ERROR ENTRY
3872 025642 012037 025656          MOV    (RO)+,2$    ;GET EM POINTER
3873 025646 001404          BEQ    3$          ;BRANCH IF THERE ISN'T ONE
3874 025650 104401 001267          TYPE  ,$CRLF       ;TYPE CARRIAGE RETURN LINE FEED
3875 025654 104401          TYPE  TYPE         ;TYPE ERROR MESSAGE (EM)

```



```

3876 025656 000000
3877 025660 012037 025674
3878 025664 001404
3879 025666 104401 001267
3880 025672 104401
3881 025674 000000
3882 025676 012001
3883 025700 001455
3884 025702 005004
3885 025704 012000
3886 025706 012002
3887 025710 001446
3888 025712 005104
3889 025714 104401 001267
3890 025720 112003
3891 025722 105720
3892 025724 005703
3893 025726 001407
3894 025730 013146
3895 025732 104402
3896 025734 005303
3897 025736 001403
3898 025740 104401 012765
3899 025744 000771
3900 025746 005302
3901 025750 003431
3902 025752 104401 001267
3903 025756 005760 000002
3904 025762 001404
3905 025764 005104
3906 025766 001002
3907 025770 104401 012765
3908 025774 012037 026002
3909 026000 104401
3910 026002 000000
3911 026004 105710
3912 026006 001003
3913 026010 062700 000002
3914 026014 000754
3915 026016 104401 001267
3916 026022 005704
3917 026024 001335
3918 026026 104401 012765
3919 026032 000732
3920 026034 104412
3921 026036 000207
3922
3923
3924
3925
3926
3927
3928
3929
3930
3931

```

```

25:      WORD      0
35:      MOV      (R0)+,4$
        BEQ      5$
        TYPE     ,$CRLF
        TYPE     DATA HEADER
45:      .WORD     0
55:      MOV      (R0)+,R1
        BEQ      20$
        CLR      R4
        MOV      (R0)+,R0
        MOV      (R0)+,R2
        BEQ      17$
        COM      R4
        TYPE     , $CRLF
10$:     MOV      (R0)+,R3
        TST      (R0)+
        TST      R3
        BEQ      14$
11$:     MOV      2(R1)+,-(SP)
        TYPOC
        DEC      R3
        BEQ      14$
        TYPE     SPACE2
        BR       11$
14$:     DEC      R2
        BLE     20$
        TYPE     , $CRLF
        TST      2(R0)
        BEQ      15$
        COM      R4
        BNE     15$
        TYPE     SPACE2
15$:     MOV      (R0)+,16$
        TYPE     DH
16$:     .WORD     0
        TST      (R0)
        BNE     21$
        ADD     #2,R0
        BR       14$
21$:     TYPE     , $CRLF
        TST      R4
        BNE     10$
17$:     TYPE     SPACE2
        BR       10$
20$:     RESREG   PC
        RTS

```

```

*****
:EM POINTER GOES HERE
:GET DH POINTER
:BRANCH IF THERE ISN'T ONE
:TYPE CR-LF
:TYPE DATA HEADER
:DH POINTER GOES HERE
:GET DT POINTER
:BRANCH IF THERE ARE NONE
:SET INDENT SWITCH
:GET DF POINTER
:STORE NUMBER OF DH'S
:DH NUM IS 0-BRANCH
:NO INDENT
:GET & STORE NUMBER OF DATA WORDS
:BUMP PAST FORMAT WORD
:TEST IF ANY DATA FOR THIS HEADER
:NO - SKIP DATA PRINT
:PUT FIRST DATA WORD ON STACK
:TYPE IT
:MORE DATA WORDS
:NO-BRANCH
:TYPE SEPARATORS
:LOOP
:MORE DH'S?
:NO-BRANCH
:ONLY A DH IN THIS REQUEST?
:YES-BRANCH BYPASS INDENT
:INDENT?
:NO-BRANCH
:YES-TYPE SPACES
:GET NEXT DH POINTER
:TYPE DH
:DH POINTER GOES HERE
:TYPE A DT?
:YES-BRANCH
:INCREMENT DF POINTER
:SEE IF END OF DF BLOCK
:INDENT?
:NO-BRANCH
:YES-TYPE SPACES
:LOOP

```

```

*****
: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.

```

```

3932          026040 104411          :*****
3933          026042 152737 000377 002652 CONERR: SAVREG
3934          026050 105237 002654          BISB      #377,DONE      ;SET DONE FLAG
3935          026054 032765 001000 000014 INCB      ERRCNT      ;BUMP ERROR COUNTER
3936          026062 001403          BIT       #PBSVAL,P.PRST(R5) ;TEST IF STATUS VALID
3937          026064 004737 026302          BEQ      12$      ;YES - SKIP
3938          026070 000402          JSR      PC REPSUP ;CALL FOR GETTING STATUS
3939          026072 004737 030560 12$:      JSR      PC TOPROC ;LOAD RK REGS INTO $REGS
3940          026076 032737 000001 002604 13$:      BIT       #BIT0,E.CONT ;ERROR 0?
3941          026104 001402          BEQ      1$      ;NO-BRANCH
3942          026106 104064          ERROR   64      ;CLEAR CONT DID NOT CLEAR ERROR
3943          026110 000463          BR      7$
3944          026112 032737 000002 002604 1$:      BIT       #BIT1,E.CONT ;ERROR 1?
3945          026120 001402          BEQ      2$      ;NO-BRANCH
3946          026122 104065          ERROR   65      ;NO ATTENTION IN ATTENTION SUM REG
3947          026124 000455          BR      7$
3948          026126 032737 000004 002604 2$:      BIT       #BIT2,E.CONT ;ERROR 2?
3949          026134 001402          BEQ      3$      ;NO-BRANCH
3950          026136 104066          ERROR   66      ;UNSOLICITED ATTENTION
3951          026140 000447          BR      7$
3952          026142 032737 000010 002604 3$:      BIT       #BIT3,E.CONT ;ERROR 3?
3953          026150 001402          BEQ      4$      ;NO-BRANCH
3954          026152 104067          ERROR   67      ;UNEXPECTED DATA TYPE ERROR
3955          026154 000441          BR      7$
3956          026156 032737 000020 002604 4$:      BIT       #BIT4,E.CONT ;ERROR 4?
3957          026164 001402          BEQ      5$      ;NO-BRANCH
3958          026166 104070          ERROR   70      ;ATTENTION DID NOT RESET WITH CLEAR
3959          026170 000433          BR      7$
3960          026172 032737 000040 002604 5$:      BIT       #BIT5,E.CONT ;ERROR 5?
3961          026200 001402          BEQ      6$      ;NO-BRANCH
3962          026202 104071          ERROR   71      ;SUBSYS CLEAR DIDN'T CLEAR DRIVE ATTENTION
3963          026204 000425          BR      7$
3964          026206 032737 000400 002604 6$:      BIT       #BIT8,E.CONT
3965          026214 001401          BEQ      8$
3966          026216 104072          ERROR   72      ;DATA LATE WHEN UNLOADING HEADER
3967          026220 032737 001000 002604 8$:      BIT       #BIT9,E.CONT
3968          026226 001401          BEQ      9$
3969          026230 104073          ERROR   73      ;CONTROLLER ERROR DURING DRIVER SERVICE
3970          026232 032737 002000 002604 9$:      BIT       #BIT10,E.CONT
3971          026240 001401          BEQ     10$
3972          026242 104074          ERROR   74      ;DRIVE DETECTED PARITY ERROR
3973          026244 032737 100000 002604 10$:      BIT       #BIT15,E.CONT
3974          026252 001401          BEQ     11$
3975          026254 104052          ERROR   52      ;MULTIPLE DRIVE SELECT
3976          026256 104075          ERROR   75      ;UNDEFINED ERROR
3977          026260 032737 000400 006364 7$:      BIT       #LEV2ER,RECODE ;TEST IF LEVEL 2 ERROR
3978          026266 001401          BEQ     14$
3979          026270 104106          ERROR   106     ;NO - SKIP
3980          026272 005037 002604 14$:      CLR      E.CONT  ;CLEAR CONTROLLER ERROR WORD
3981          026276 000137 030416          JMP      CERTY   ;GO DO RETRY
3982          :*****
3983          :SBTTL REPORT SUPPORT ROUTINE
3984          :*THIS ROUTINE LOADS ALL THE PARAMETER BLOCK DATA TO BE REPORTED
3985          :*INTO THE PROPER TEMPORARY REGISTERS FOR REPORTING. ALL THESE MAY
3986          :*NOT BE INCLUDED IN THE REPORT, BUT THEY ARE LOADED ANYWAY.
3987

```

REPSUP:

```

3988 026302
3989 026302 104411
3990 026304 005037 006372
3991 026310 116537 000001 006372
3992 026316 012700 001162
3993 026322 016520 000002
3994 026326 116520 000005
3995 026332 105020
3996 026334 116520 000004
3997 026340 105020
3998 026342 016520 000012
3999 026346 016520 000010
4000 026352 016520 000016
4001 026356 016520 000020
4002 026362 016520 000030
4003 026366 016520 000026
4004 026372 016520 000022
4005
4006 026376 016520 000024
4007 026402 016520 000032
4008 026406 016520 000036
4009 026412 016520 000034
4010 026416 016520 000040
4011 026422 016520 000042
4012 026426 016520 000044
4013 026432 016520 000046
4014 026436 016520 000050
4015 026442 016520 000052
4016 026446 016520 000054
4017 026452 016520 000056
4018 026456 104412
4019 026460 000207
4020
4021
4022
4023
4024
4025
4026
4027
4028
4029
4030 026462 104411
4031 026464 152737 000377 002652
4032 026472 105237 002654
4033 026476 005037 006364
4034 026502 032737 000400 006364
4035 026510 001402
4036 026512 012705 002446
4037 026516 012737 030532 002600
4038 026524 012737 030550 002576
4039 026532 004737 026302
4040
4041
4042
4043

```

```

SAVREG
CLR ERRCOM ;CLEAR ERROR COMMAND STORE
MOV P.CMND(R5),ERRCOM ;STORE COMMAND START VALUES
MOV #SREG0,R0 ;FOR REPORTING
MOV P.CYLN(R5),(R0)+
MOV P.TRCK(R5),(R0)+
CLRB (R0)+
MOV P.SECT(R5),(R0)+
CLRB (R0)+
MOV P.WC(R5),(R0)+
MOV P.BALO(R5),(R0)+
MOV P.CS1(R5),(R0)+ ;GET ALL THE VALUES FROM THE
MOV P.CS2(R5),(R0)+ ;PARAMETER BLOCK AND LOAD
MOV P.DCYL(R5),(R0)+ ;THE TEMPORARY REGISTERS
MOV P.DTS(R5),(R0)+ ;FOR REPORTING. ALL THIS
MOV P.WCR(R5),(R0)+ ;DATA MAY NOT BE VALID
;FOR ALL REPORTS (TO BE
;DETERMINED LATER) BUT IT IS
;STORED ANY WAY.
MOV P.BAR(R5),(R0)+
MOV P.ASOF(R5),(R0)+
MOV P.DS(R5),(R0)+
MOV P.ER(R5),(R0)+
MOV P.A00(R5),(R0)+
MOV P.B00(R5),(R0)+
MOV P.A01(R5),(R0)+
MOV P.B01(R5),(R0)+
MOV P.A10(R5),(R0)+
MOV P.B10(R5),(R0)+
MOV P.A11(R5),(R0)+
MOV P.B11(R5),(R0)+
RESREG
RTS PC
;*****
;SBTTL REPORT ERROR ROUTINE
;* ENTRY JSR PC,ERRHDL
;*RETURN RTS PC
;*
;*THIS ROUTINE IS CALLED BY THE DRIVER WHEN AN ERROR IS DETECTED
;*IN THE OPERATION. THE ROUTINE DETERMINES WHICH COMMAND WAS
;*BEING EXECUTED AND GENERATES THE PROPER ERROR MESSAGE.
;*****
ERRHDL: SAVREG
BISB #377,DONE ;SET DONE FLAG
INCB ERRCNT ;INCREMENT ERROR COUNT
CLR RECODE ;CLEAR RECOVERY CODE WORD
ER2ENT: BIT #LEV2ER,RECODE ;TEST IF 2ND LEVEL ERROR
BEQ 52$ ;NO - SKIP PARAM BLOCK CHANGE
MOV #PARM1,R5 ;ELSE SET R5 TO PARAMETER BLOCK 1
52$: MOV #RETANL,A.ABNL ;SET NOW ABNORMAL AND NORMAL RETURN FOR
MOV #RETNML,A.NORM ;DRIVER OPERATIONS IN ERROR PROCESSING
JSR PC,REPSUP ;GO SET UP REGISTERS FOR REPORT
;NOW BEGIN TESTING THE ERROR
;BITS. THE SEQUENCE IN
;WHICH THEY ARE TESTED IS
;CONSIDERED SIGNIFICANT IN

```

```

4044
4045
4046
4047
4048
4049
4050
4051
4052
4053
4054
4055
4056
4057 026536 016504 000034          MOV      P.ER(R5),R4
4058 026542 032765 020000 000020    BIT      #UPE,P.CS2(R5)
4059 026550 001406          BEQ      1$
4060 026552 104001          ERROR   1
4061 026554 052737 000200 006364    BIS      #ABORT,RECODE
4062 026562 000137 030106          JMP      37$
4063 026566 032765 004000 000020 1$:    BIT      #NEM,P.CS2(R5) ;TEST NON-EXISTANT MEMORY
4064 026574 001406          BEQ      2$
4065 026576 104002          ERROR   2
4066 026600 052737 000200 006364    BIS      #ABORT,RECODE
4067 026606 000137 030106          JMP      37$
4068 026612 032765 010000 000020 2$:    BIT      #NED,P.CS2(R5) ;TEST NON-EXISTANT DRIVE
4069 026620 001412          BEQ      3$
4070 026622 032765 000400 000020    BIT      #UFE,P.CS2(R5) ;TEST IF NED & UFE BOTH SET
4071 026630 001403          BEQ      38$
4072 026632 104035          ERROR   35 ;NED & UFE BOTH SET ERROR
4073 026634 000137 030106          JMP      37$
4074 026640 104003          ERROR   3 ;NED ONLY
4075 026642 000137 030106          JMP      37$
4076 026646 032765 000400 000020 3$:    BIT      #UFE,P.CS2(R5) ;TEST UNIT FIELD ERROR
4077 026654 001412          BEQ      4$
4078 026656 032765 010000 000020    BIT      #NED,P.CS2(R5) ;TEST IF UFE & NED BOTH SET
4079 026664 001403          BEQ      39$ ;NO-SKIP
4080 026666 104035          ERROR   35 ;REPORT NED & UFE BOTH SET
4081 026670 000137 030106          JMP      37$
4082 026674 104004          ERROR   4 ;REPORT UFE ONLY
4083 026676 000137 030106          JMP      37$
4084 026702 032765 000100 000014 4$:    BIT      #CMDTO,P.PRST(R5) ;
4085 026710 001414          BEQ      5$
4086 026712 004737 030560          JSR      PC.TOPROC ;GO PROCESS TIMEOUT
4087 026716 032737 000400 006364    BIT      #LEV2ER,RECODE ;TEST IF LEVEL 2 ERROR
4088 026724 001003          BNE      41$ ;YES - SKIP TO ERROR 57
4089 026726 104005          ERROR   5 ;ELSE MAKE FULL TIMEOUT REPORT
4090 026730 000137 030106          JMP      37$
4091 026734 104057          ERROR   57 ;2ND LEVEL ERROR REPORT
4092 026736 000137 030106          JMP      37$
4093 026742 032765 020000 000016 5$:    BIT      #SPAR,P.CS1(R5) ;TEST DRIVE BUS PARITY ERROR
4094 026750 001406          BEQ      6$
4095 026752 104006          ERROR   6
4096 026754 052737 004000 006364    BIS      #RCLREQ,RECODE
4097 026762 000137 030106          JMP      37$
4098 026766 032704 000010 6$:    BIT      #DRPAR,R4 ;TEST DRIVE DETECTED PARITY ERROR
4099 026772 001406          BEQ      7$

```

: THAT ERRORS OF A MORE
: CATASTROPHIC NATURE ARE FIRST
: TESTED.

: IF AN ERROR IS FOUND SET,
: THAT ERROR IS REPORTED AND
: THE REPORTING IS TERMINATED.
: IF ADDITIONAL ERRORS ARE SET,
: THE RK611 REGISTER PRINTOUTS
: WILL SHOW THIS BUT THE
: REGISTER CONTENTS MUST BE
: MANUALLY DECODED TO LOCATE THE
: SECOND ERROR

: SET R4 TO ERROR REGISTER
: TEST UPE. IF UES-SET
: REPORT ERROR

;TEST NON-EXISTANT MEMORY

;TEST NON-EXISTANT DRIVE

;TEST IF NED & UFE BOTH SET

;NED & UFE BOTH SET ERROR

;NED ONLY

;TEST UNIT FIELD ERROR

;TEST IF UFE & NED BOTH SET

;NO-SKIP
;REPORT NED & UFE BOTH SET

;REPORT UFE ONLY

;

;GO PROCESS TIMEOUT

;TEST IF LEVEL 2 ERROR

;YES - SKIP TO ERROR 57

;ELSE MAKE FULL TIMEOUT REPORT

;2ND LEVEL ERROR REPORT

;TEST DRIVE BUS PARITY ERROR

#RCLREQ,RECODE

;TEST DRIVE DETECTED PARITY ERROR

4100	026774	104007				ERROR	7	
4101	026776	052737	004000	006364		BIS	#RCLREQ,RECODE	
4102	027004	000137	030106			JMP	37\$	
4103	027010	032765	000010	000036	7\$:	BIT	#ACLO,P.DS(R5)	;TEST AC LOW
4104	027016	001403				BEQ	8\$	
4105	027020	104010				ERROR	10	
4106	027022	000137	030106			JMP	37\$	
4107	027026	032765	000020	000036	8\$:	BIT	#SPDLSS,P.DS(R5)	;TEST SPEED LOSS
4108	027034	001403				BEQ	24\$	
4109	027036	104011				ERROR	11	
4110	027040	000137	030106			JMP	37\$	
4111	027044	032765	004000	000016	24\$:	BIT	#CTO,P.CS1(R5)	;TEST FOR CONTROLLER TIMEOUT
4112	027052	001401				BEQ	25\$	
4113	027054	104027				ERROR	27	
4114	027056	032704	000001		25\$:	BIT	#ILC,R4	;TEST ILLEGAL FUNCTION CODE
4115	027062	001403				BEQ	10\$	
4116	027064	104012				ERROR	12	
4117	027066	000137	030106			JMP	37\$	
4118	027072	032765	002000	000020	10\$:	BIT	#PGE,P.CS2(R5)	;TEST PROGRAMMING ERROR
4119	027100	001403				BEQ	11\$	
4120	027102	104013				ERROR	13	
4121	027104	000137	030106			JMP	37\$	
4122	027110	032704	000004		11\$:	BIT	#NXF,R4	;TEST ILLEGAL DRIVE FUNCTION
4123	027114	001403				BEQ	12\$	
4124	027116	104014				ERROR	14	
4125	027120	000137	030106			JMP	37\$	
4126	027124	032704	000040		12\$:	BIT	#DTYE,R4	;TEST DRIVE TYPE ERROR
4127	027130	001403				BEQ	13\$	
4128	027132	104015				ERROR	15	
4129	027134	000137	030106			JMP	37\$	
4130	027140	032704	000020		13\$:	BIT	#FMTE,R4	;TEST FORMAT ERROR
4131	027144	001403				BEQ	14\$	
4132	027146	104016				ERROR	16	
4133	027150	000137	030106			JMP	37\$	
4134	027154	032704	004000		14\$:	BIT	#WLE,R4	;TEST WRITE LOCK ERROR
4135	027160	001403				BEQ	15\$	
4136	027162	104017				ERROR	17	
4137	027164	000137	030106			JMP	37\$	
4138	027170	032704	040000		15\$:	BIT	#UNS,R4	;TEST DRIVE UNSAFE
4139	027174	001406				BEQ	16\$	
4140	027176	104020				ERROR	20	
4141	027200	052737	000200	006364		BIS	#ABORT,RECODE	
4142	027206	000137	030206			JMP	ALLTRM	
4143	027212	032704	000002		16\$:	BIT	#SKI,R4	;TEST SEEK INCOMPLETE
4144	027216	001406				BEQ	17\$	
4145	027220	104021				ERROR	21	
4146	027222	052737	004000	006364		BIS	#RCLREQ,RECODE	
4147	027230	000137	030106			JMP	37\$	
4148	027234	032704	001000		17\$:	BIT	#COE,R4	;TEST CYLINDER OVERFLOW
4149	027240	001406				BEQ	18\$	
4150	027242	104022				ERROR	22	
4151	027244	052737	004000	006364		BIS	#RCLREQ,RECODE	
4152	027252	000137	030106			JMP	37\$	
4153	027256	032704	002000		18\$:	BIT	#IDAE,R4	;TEST ILLEGAL CYLINDER
4154	027262	001406				BEQ	19\$	
4155	027264	104023				ERROR	23	

4156	027266	052737	004000	006364		BIS	#RCLREQ,RECODE	
4157	027274	000137	030106			JMP	37\$	
4158	027300	032765	000040	000036	19\$:	BIT	#DROT,P.DS(R5)	;TEST DRIVE OFF TRACK
4159	027306	001406				BEQ	20\$	
4160	027310	104024				ERROR	24	
4161	027312	052737	004000	006364		BIS	#RCLREQ,RECODE	
4162	027320	000137	030106			JMP	37\$	
4163	027324	032704	010000		20\$:	BIT	#DTE,R4	;TEST DRIVE TIMING ERROR
4164	027330	001406				BEQ	21\$	
4165	027332	104025				ERROR	25	
4166	027334	052737	000200	006364		BIS	#ABORT,RECODE	
4167	027342	000137	030106			JMP	37\$	
4168	027346	032765	100000	000020	21\$:	BIT	#DLT,P.CS2(R5)	;TEST DATA LATE
4169	027354	001403				BEQ	22\$	
4170	027356	104026				ERROR	26	
4171	027360	000137	030106			JMP	37\$	
4172	027364	032704	020000		22\$:	BIT	#OPI,R4	;TEST IF OPI ERROR
4173	027370	001457				BEQ	29\$	
4174	027372	052737	000010	006364		BIS	#OPIERR,RECODE	
4175	027400	105737	002650			TSTB	DERCNT	;TEST IF FIRST ERROR
4176	027404	001402				BEQ	50\$	
4177	027406	000137	030106			JMP	37\$;NO - SKIP REPORT
4178	027412	032737	000400	006364	50\$:	BIT	#LEV2ER,RECODE	;TEST IF A SECOND LEVEL 2 ERROR
4179	027420	001403				BEQ	26\$;HAS ALREADY OCCURRED
4180	027422	104054			27\$:	ERROR	54	
4181	027424	000137	030106			JMP	37\$;GET OUT OF ERROR REPORT
4182	027430	004737	031534		26\$:	JSR	PC,BLDEXH	;GO BUILD EXPECTED HEADER
4183	027434	004737	031122			JSR	PC,RDH00	;GET HEADER OF SECTOR 0
4184	027440	032737	000400	006364		BIT	#LEV2ER,RECODE	;TEST IF ERROR GETTING HDR
4185	027446	001025				BNE	28\$;IF YES-GO MAKE ABBREVIATED REPORT
4186	027450	013702	002570			MOV	RKBAS,R2	;STORE HEADER 0 INTO REGISTERS
4187	027454	042762	000100	000000		BIC	#IE,RKCS1(R2)	;RESET INTERRUPT ENABLE
4188	027462	016237	000024	001202		MOV	RKDB(R2),\$REG10	;FOR REPORTING
4189	027470	016237	000024	001204		MOV	RKDB(R2),\$REG11	
4190	027476	016237	000024	001206		MOV	RKDB(R2),\$REG12	
4191	027504	032762	100000	000000		BIT	#CERR,RKCS1(R2)	;TEST IF ERROR DURING STORAGE
4192	027512	001343				BNE	27\$	
4193	027514	104030				ERROR	30	;MAKE OPI REPORT
4194	027516	000137	030106			JMP	37\$	
4195	027522	104054			28\$:	ERROR	54	
4196	027524	000137	026502			JMP	ER2ENT	;GO MAKE 2ND LEVEL REPORT
4197	027530	032704	000400		29\$:	BIT	#HVRC,R4	;TEST IF HVRC ERROR
4198	027534	001457				BEQ	23\$	
4199	027536	052737	000004	006364		BIS	#HVR CER,RECODE	
4200	027544	105737	002650			TSTB	DERCNT	;TEST IF FIRST ERROR
4201	027550	001402				BEQ	30\$;YES - REPORT
4202	027552	000137	030106			JMP	37\$;JUMP TO RETURN
4203	027556	032737	000400	006364	30\$:	BIT	#LEV2ER,RECODE	;TEST IF A 2ND LEVEL ERROR HAS ALREADY
4204	027564	001403				BEQ	31\$;OCCURRED. NO-SKIP EXIT
4205	027566	104055			51\$:	ERROR	55	
4206	027570	000137	030106			JMP	37\$;GET OUT OF ERROR REPORT
4207	027574	004737	031534		31\$:	JSR	PC,BLDEXH	;GO BUILD EXPECTED HEADER
4208	027600	004737	031122			JSR	PC,RDH00	;GO GET HDR 0
4209	027604	032737	000400	006364		BIT	#LEV2ER,RECODE	;TEST IF ERROR IN GETTING HDR
4210	027612	001025				BNE	32\$;IF YES-GO MAKE ABBREVIATED REPORT
4211	027614	013702	002570			MOV	RKBAS,R2	;GET RK611 BASE ADDRESS

4212	027620	042762	000100	000000		BIC	#IE,RKCS1(R2)	;CLEAR INTERRUPT ENABLE
4213	027626	016237	000024	001202		MOV	RKDB(R2),SREG10	;STORE OFF HEADER
4214	027634	016237	000024	001204		MOV	RKDB(R2),SREG11	
4215	027642	016237	000024	001206		MOV	RKDB(R2),SREG12	
4216	027650	032762	100000	000000		BIT	#CERR,RKCS1(R2)	;TEST IN ANY ERROR IN UNLOAD
4217	027656	001343				BNE	51\$;IY YES - GO MAKE SHORT REPORT
4218	027660	104031				ERROR	31	;MAKE FULL REPORT
4219	027662	000137	030106			JMP	37\$	
4220	027666	104055			32\$:	ERROR	55	;ABREVIATED HVRC ERROR RPORT
4221	027670	000137	026502			JMP	ER2ENT	;GO REPORT 2ND LEVEL ERROR
4222	027674	032704	000200		23\$:	BIT	#BSE,R4	;TEST FOR BAD SECTOR ERROR
4223	027700	001422				BEQ	33\$;NO - SKIP
4224	027702	052737	000002	006364		BIS	#BSERR,RECODE	;SET ERROR FLAG
4225	027710	016502	000026			MOV	P.DTS(R5),R2	;GET SECTOR IN ERROR
4226	027714	042702	177740			BIC	#177740,R2	;CLEAR ALL BITS EXCEPT SECTOR
4227	027720	004737	021720			JSR	PC,BDSRCK	;GO SEE IF THIS SECTOR LISTED
4228	027724	032737	001000	006364		BIT	#BADSEC,RECODE	;TEST RESULT
4229	027732	001065				BNE	37\$;YES - EXIT NO ERROR OR REPORT
4230	027734	104104				ERROR	104	;ELSE REPORT BAD BSE
4231	027736	042737	000002	006364		BIC	#BSERR,RECODE	;RESET BSE ERROR FLAG
4232	027744	000460				BR	37\$;GO EXIT
4233	027746	032704	100000		33\$:	BIT	#DCK,R4	;TEST IF DATA CHECK
4234	027752	001435				BEQ	36\$	
4235	027754	052737	000020	006364		BIS	#DCKERR,RECODE	;SET DATA CHECK ERROR IN RECOVERY CODE
4236	027762	032704	000100			BIT	#ECH,R4	;TEST IF ECC IS HARD. IF
4237	027766	001406				BEQ	34\$;YES SET UNCORRECTABLE IN
4238	027770	052737	000040	006364		BIS	#ECCNC,RECODE	;RECOVERY FLAG AND A 0 IN
4239	027776	005037	001206			CLR	SREG12	;REG12 TO INDICATE UNCORRECTABLE,
4240	030002	000403				BR	35\$;A 1 IN REG 12 FOR CORRECTABLE
4241	030004	012737	000001	001206	34\$:	MOV	#1,SREG12	
4242	030012	105737	002650		35\$:	TSTB	DERCNT	;TEST IF FIRST ERROR
4243	030016	001033				BNE	37\$;NO SKIP REPORT
4244	030020	004737	031410			JSR	PC,GTPKAD	;GO GET PACK ADDRESS OF ERROR
4245	030024	016537	000060	001202		MOV	P.EPOS(R5),SREG10	;STORE ECC POSITION &
4246	030032	016537	000062	001204		MOV	P.EPAT(R5),SREG11	;PATTERN
4247	030040	104032				ERROR	32	;REPORT DCK ERROR
4248	030042	000137	030106			JMP	37\$	
4249	030046	032765	040000	000020	36\$:	BIT	#WCE,P.CS2(R5)	;TEST WRITE CHECK ERROR
4250	030054	001414				BEQ	37\$	
4251	030056	042737	000200	006364		BIC	#ABORT,RECODE	;CLEAR ABORT & SET WRITE
4252	030064	052737	000100	006364		BIS	#WCERR,RECODE	;CHECK ERROR IN RECODE
4253	030072	105737	002650			TSTB	DERCNT	;TEST IF FIRST ERROR
4254	030076	001003				BNE	37\$;NO - SKIP
4255	030100	004737	031410			JSR	PC,GTPKAD	;GO GET ADDRESS OF ERROR
4256	030104	104033				ERROR	33	;REPORT WCE
4257								
4258	030106	032765	000020	000014	37\$:	BIT	#DRVHRD,P.PRST(R5)	;TEST HARD ERROR
4259	030114	001404				BEQ	43\$	
4260	030116	104036				ERROR	36	
4261	030120	052737	000200	006364		BIS	#ABORT,RECODE	
4262								
4263	030126	032765	000040	000014	43\$:	BIT	#DRVDSC,P.PRST(R5)	;TEST STATUS CHANGE NOT CLEARED
4264	030134	001404				BEQ	44\$	
4265	030136	104037				ERROR	37	
4266	030140	052737	000200	006364		BIS	#ABORT,RECODE	
4267								

```

4268 030146 032765 004000 000014 44$: BIT #NODSC,P.PRST(R5) ;IFST ATTENTION BUT NO FAULT OR DSC
4269 030154 001404 BEQ 46$
4270 030156 104040 ERROR 40
4271 030160 052737 000200 006364 BIS #ABORT,RECODE
4272 030166 032765 000010 000014 46$: BIT #UEXATT,P.PRST(R5) ;TEST UNEXPECTED ATTENTION
4273 030174 001404 BEQ ALLTRM
4274 030176 104042 ERROR 42
4275 030200 052737 000200 006364 BIS #ABORT,RECODE
4276
4277
4278 ;ALL ERRORS MUST EXIT THROUGH THIS POINT
4279
4280 030206 012705 002362 ALLTRM: MOV #PARMO,R5 ;RESTORE PARAMETER BLOCK SELECTION
4281 030212 012737 026462 002600 MOV #ERRHDL,A.ABNL ;RESTORE INTERRUPT VECTORS FOR RETRY
4282 030220 012737 025524 002576 MOV #ERRFRE,A.NORM ;DRIVER OPERATIONS, IF ANY
4283 030226 032737 000200 006364 BIT #ABORT,RECODE ;IF ABORT IS NOT SET AND
4284 030234 001050 BNE 48$ ;THE DRIVE IS READY (HAS NOT
4285 ;CYCLED DOWN)
4286 030236 013702 002570 MOV RKBAS,R2 ;GET BASE ADDRESS
4287 030242 032762 000200 000012 BIT #DRDY,RKDS(R2) ;TEST IF DRIVE READY SET
4288 030250 001004 BNE 47$ ;RECALIBRATE REQUIRED BIT IS SET
4289 030252 052737 000200 006364 BIS #ABORT,RECODE ;ELSE ABORT WITH ABORT MESSAGE
4290 030260 000436 BR 48$
4291 030262 032737 004000 006364 47$: BIT #RCLREQ,RECODE ;IF RECALIBRATE IS REQUIRED
4292 030270 001436 BEQ BGNRTY ;FOR RETRY SET UP PARAM
4293 030272 012705 002446 MOV #PARM1,R5 ;SET TO PARAMETER BLOCK ONE
4294 030276 112765 000113 000001 MOVB #RECAL,P.CMND(R5) ;BLOCK TO DO IT.
4295 030304 012737 030532 002600 MOV #RETANL,A.ABNL
4296 030312 004737 025442 JSR PC.DRVCAL
4297 030316 012737 026462 002600 MOV #ERRHDL,A.ABNL ;RESTORE ERROR RETURN
4298 030324 032737 000400 006364 BIT #LEVZER,RECODE ;IF AN ERROR OCCURRED IN THE
4299 030332 001003 BNE 49$ ;RECAL ATTEMP SET ABORT,
4300 030334 012705 002362 MOV #PARMO,R5 ;RESET PARAM TO BLOCK 0
4301 030340 000412 BR BGNRTY
4302 030342 052737 000200 006364 49$: BIS #ABORT,RECODE ;PRINT THE RECAL ERROR MESSAGE, AND
4303 030350 104060 ERROR 60 ;GO REPORT DETAILS
4304 030352 000137 026502 JMP ER2ENT
4305 030356 104061 48$: ERROR 61 ;REPORT ABORT-RETRY FAILED
4306
4307 ;THE PROGRAM WILL HALT HERE IF THE ABORT FLAG HAS BEEN SET OR
4308 ;IF THE DRIVE READY BIT IS RESET.
4309
4310 030360 000000 HLTORG: HALT
4311 030362 000137 016202 JMP START
4312
4313
4314 ;THE FOLLOWING CODE WILL DETERMINE IF AND DATA TYPE ERROR
4315 ;HAS OCCURRED. IF YES, THE RETRY IS NOT DONE HERE BUT RETURNS TO
4316 ;THE INITIATING ROUTINE FOR RETRY. ANY OTHER ERROR IS TO BE
4317 ;RETRIED HERE. IF RETRY IS UNSUCCESSFUL AFTER 4 ATTEMPTS, THE ABORT
4318 ;FLAG IS SET AND PROGRAM HALTS.
4319
4320 030366 032737 000136 006364 BGNRTY: BIT #BSERR!HVCER!OPIERR!DCKERR!WCERR,RECODE ;TEST IF ANY DATA ERROR
4321 030374 001404 BEQ DOL3
4322 030376 052737 100000 006364 BIS #ANYDER,RECODE
4323 030404 000450 BR DOL2
  
```



```

4324 030406
4325 030406 032737 001000 006364 DOL3: BIT #BADSEC,RECODE ;TEST IF BAD SECTOR FLAG SET
4326 030414 001044 BNE DOL2 ;IF YES-EXIT TO CALLER
4327 030416 123737 002654 002655 CERTY: CMPB ERRCNT,ERRLMT ;BEGIN RETRY IF ERROR COUNT HAS
4328 030424 001012 BNE DOL1 ;NOT BEEN EXCEEDED
4329 030426 005037 001174 CLR $REG5
4330 030432 113737 002654 001174 MOVB ERRCNT,$REG5 ;GET ERROR RETRY COUNT
4331 030440 104102 ERROR 102 ;REPORT RETRY UNSUCCESSFUL
4332 030442 052737 000200 006364 BIS #ABORT,RECODE ;SET ABORT & QUIT
4333 030450 000743 BR HLTPRG
4334 030452 013702 002570 DOL1: MOV RKBAS,R2 ;GET BASE ADDRESS
4335 030456 012762 000040 000010 MOV #SCLR,RKCS2(R2) ;CLEAR SUBSYSTEM
4336 030464 013746 000000 MOV 0,-(SP) ;PUT NEW PS ON STACK
4337 030470 012746 030476 MOV #64$,-(SP) ;PUT NEW PC ON STACK
4338 030474 000002 RTI ;POP NEW PC AND PS
4339 030476
4340 030476 012700 005660 64$: MOV #COMSTR,R0 ;GO AND REESTABLISH THE COMMAND
4341 030502 012025 MOV (R0)+,(R5)+ ;AS IT WAS ENTERED INTO THE
4342 030504 012025 MOV (R0)+,(R5)+ ;PARAMETER BLOCK
4343 030506 012025 MOV (R0)+,(R5)+
4344 030510 012025 MOV (R0)+,(R5)+
4345 030512 012025 MOV (R0)+,(R5)+
4346 030514 012025 MOV (R0)+,(R5)+
4347 030516 012700 002362 MOV #PARM0,R5
4348 030522 004737 025442 JSR PC,DRVCAL ;CALL DRIVER
4349 030526 104412 DOL2: RESREG ;IF RETURN GETS HERE, NO ERROR
4350 030530 000207 RTS PC ;OCCURRED, RECOVERY WAS SUCCESSFUL
4351 030532 152737 000377 002652 RETANL: BISB #377,DONE ;SET DONE
4352 030540 052737 000400 006364 BIS #LEV2ER,RECODE ;SET LEVEL TWO ERROR
4353 030546 000207 RTS PC
4354 030550 152737 000377 002652 RETNML: BISB #377,DONE ;SET DONE
4355 030556 000207 RTS PC
4356
4357
4358 ;*****
4359 ;SBTTL TIME OUT PROCESSOR ROUTINE
4360 ;THIS ROUTINE SUPPORTS THE ERROR HANDLER BY PROCESSING TIME OUT STATUS
4361 ;GATHERING DUTIES.
4362 ;*****
4363 TOPROC: SAVREG
4364 MOV RKBAS,R2
4365 MOV #SREG0,R1 ;SET UP R1 FOR RK REGISTER STORAGE
4366 CLR ERRCOM ;CLEAR FOR COMMAND
4367 MOV P.CMND(R5),ERRCOM ;STORE COMMAND
4368 MOV P.CYLN(R5),(R1)+ ;STORE CYLINDER
4369 MOV P.TRACK(R5),(R1)+ ;STORE TRACK
4370 CLRB (R1)+
4371 MOV P.SECT(R5),(R1)+ ;STORE SECTOR
4372 CLRB (R1)+
4373 MOV P.WC(R5),(R1)+ ;STORE WORD COUNT
4374 MOV P.BALO(R5),(R1)+ ;STORE BUS ADDRESS
4375 MOV RKCS1(R2),(R1)+ ;STORE ALL PARAMERERS AND VALID RK611
4376 MOV RKCS2(R2),(R1)+ ;REGISTERS
4377 MOV RKDC(R2),(R1)+
4378 MOV RKDA(R2),(R1)+
4379 MOV RKWC(R2),(R1)+
4380 MOV RKBA(R2),(R1)+

```

```

4380 030664 016221 000016      MOV      RKASOF(R2), (R1)+
4381 030670 016221 000012      MOV      RKDS(R2), (R1)+
4382 030674 016221 000014      MOV      RKER(R2), (R1)+
4383                                     ; THIS CODE WILL ATTEMPT TO
4384                                     ; RETRIEVE THE STATUS FROM THE
4385                                     ; DRIVE.
4386 030700 004737 030710      JSR      PC, GETDRST      ; GO GET DRIVE STATUS
4387 030704 104412      RESREG
4388 030706 000207      RTS      PC
4389                                     ; *****
4390                                     ; SBTTL  RETRIEVE DRIVE STATUS ROUTINE
4391
4392 030710 104411      GETDRST:  SAVREG
4393 030712 005000      CLR      R0
4394 030714 012762 100000 000000    MOV      #CCLR, RKCS1(R2) ; CLEAR CONTROLLER
4395 030722 010146      MOV      R1, -(SP)      ; STORE R5
4396 030724 012703 000010      MOV      #10, R3      ; SET COUNT FOR STATUS CLEAR
4397 030730 012721 123456 1$:     MOV      #123456, (R1)+ ; INSERT CLEAR WORD
4398 030734 005303      DEC      R3            ; BUMP COUNTER
4399 030736 001374      BNE     1$            ; LOOP UNTIL 0
4400 030740 012601      MOV      (SP)+, R1     ; RESTORE R5
4401 030742 152737 000377 002632    BISB    #377, W.TIME   ; GIVE WATCH DOG SOMETHING TO TIME
4402 030750 013746 002600      MOV      A, ABNL, -(SP) ; STORE ERROR RETURN ADDRESS
4403 030754 012737 030532 002600    MOV      #RETANL, A, ABNL ; SET UP NEW ERROR RETURN
4404 030762 013762 006340 000010    MOV      DRINUS, RKCS2(R2) ; LOAD DRIVE NUMBER
4405 030770 032762 100000 000000    BIT     #CERR, RKCS1(R2) ; TEST IF CONT ERROR SET
4406 030776 001037      BNE     4$            ; YES - GET OUT
4407 031000 010062 000026 3$:     MOV      R0, RKMR1(R2) ; SET UP STATUS WORD TO READ
4408 031004 012762 000001 000000    MOV      #1, RKCS1(R2) ; DO DRIVE SELECT
4409 031012 016204 000000      MOV      RKCS1(R2), R4 ; GET CS1
4410 031016 032704 000200      BIT     #RDY, R4      ; TEST IF READY
4411 031022 001007      BNE     2$            ; YES - SKIP
4412 031024 004737 032312      JSR      PC, W.WTCH    ; CALL WATCH DOG
4413 031030 032765 000100 000014    BIT     #CMDTO, P.PRST(R5) ; TEST IF TIMEOUT
4414 031036 001360      BNE     3$            ; NO - LOOP
4415 031040 000416      BR      4$            ; ELSE GET OUT
4416 031042 016204 000000 2$:     MOV      RKCS1(R2), R4 ; GET CS1
4417 031046 032704 100000      BIT     #CERR, R4    ; TEST IF ERROR
4418 031052 001011      BNE     4$            ; YES - GET OUT
4419 031054 016221 000034      MOV      RKMR2(R2), (R1)+ ; ELSE STORE STATUS WORDS JUST READ
4420 031060 016221 000036      MOV      RKMR3(R2), (R1)+
4421 031064 005200      INC     R0            ; BUMP TO NEXT STATUS PAIR
4422 031066 020027 000004      CMP     R0, #4       ; TEST IF ALL HAVE BEEN READ
4423 031072 001342      BNE     3$            ; YES - GOOD EXIT
4424 031074 000406      BR      5$
4425 031076 052737 000400 006364 4$:     BIS     #LEV2ER, RECODE ; SET LEVEL 2 ERROR
4426 031104 012762 000040 000010    MOV      #SCLR, RKCS2(R2) ; CLEAR SUBSYSTEM
4427 031112 012637 002600 5$:     MOV      (SP)+, A, ABNL ; RESTORE OLD ERROR RETURN
4428 031116 104412      RESREG
4429 031120 000207      RTS      PC
4430                                     ; *****
4431                                     ; SBTTL  READ HEADER 0 ROUTINE
4432                                     ; *****
4433 031122      RDHDO:  SAVREG
4434 031122 104411      MOV      P.DTS(R5), R1 ; STORE TRACK AND SECTOR
4435 031124 016501 000026

```

```

4436 031130 016500 000052      MOV      P.B10(R5),R0      ;GET THE CYLINDER ADDRESS
4437 031134 042700 160013      BIC      #160013,R0      ;FROM THE DRIVE STATUS. CLEAR
4438 031140 006200              ASR      R0              ;OFF UNUSED BITS AND POSITION
4439 031142 006200              ASR      R0              ;FOR USE AS THE DESIRED
4440 031144 006200              ASR      R0              ;CYLINDER IN THE READ
4441 031146 006200              ASR      R0              ;HEADER COMMAND
4442 031150 012705 002446      MOV      #PARM1,R5      ;SET UP TO USE PARM 1
4443 031154 010165 000004      MOV      R1,P.SECT(R5)   ;INSERT TRK/SECT FOR READ HEADER
4444 031160 010065 000002      MOV      R0,P.CYLN(R5)
4445 031164 132737 000020 002653 BITB     #B.CFMT,TYPFMT   ;TEST PRESENT FORMAT & CHANGE
4446 031172 001404              BEQ      1$              ;TO THE OPPOSITE. THIS WILL CAUSE
4447 031174 142765 000020 000007 BICB     #B.CFMT,P.CS1H(R5) ;A READ OF SECTOR 0 HEADER ON
4448 031202 000403              BR       2$              ;THE READ HEADER COMMAND
4449 031204 152765 000020 000007 1$:     BISB     #B.CFMT,P.CS1H(R5)
4450 031212 112765 000125 000001 2$:     MOVB    #RDHEAD,P.CMND(R5) ;SET UP TO READ HEADER
4451 031220 013746 000000      MOV      0-(SP)         ;PUT NEW PS ON STACK
4452 031224 012746 031232      MOV      #64$,-(SP)     ;PUT NEW PC ON STACK
4453 031230 000002              RTI                    ;POP NEW PC AND PS
4454 031232              64$:
4455 031232 004737 025442      JSR      PC,DRVCL        ;CALL DRIVER
4456 031236 142765 000020 000007 BICB     #B.CFMT,P.CS1H(R5) ;CLEAR THE FORMAT BIT
4457 031244 153765 002653 000007 BISB     TYPFMT,P.CS1H(R5) ;RESTORE TYPE AND FORMAT IN USE
4458 031252 012705 002362      MOV      #PARM0,R5      ;RESTORE PARM 0 BLOCK
4459 031256 104412      RESREG
4460 031260 000207      RTS      PC
4461
4462 ;*****
4463 ;SBTTL POSITION AND OFFSET ROUTINE
4464 ;*THIS ROUTINE CHECKS IF OFFSET IS REQUIRED. IF IT IS, THE HEADS
4465 ;*POSITIONED TO THE CYLINDER SPECIFIED IN R0 AND THE HEADS ARE OFFSET
4466 ;*TO THE AMOUNT SPECIFIED IN "OFINUS".
4467 ;*****
4468 031262 104411      POSOFF: SAVREG
4469 031264 132737 000020 002656 BITB     #OREQSW,OPCONT   ;TEST IF OFFSET REQUIRED
4470 031272 001417              BEQ      1$              ;IF NO - SKIP TO EXIT
4471 031274 112765 000117 000001 MOVB    #SEEK,P.CMND(R5) ;SET UP TO DO THE SEEK
4472 031302 010065 000002      MOV      R0,P.CYLN(R5) ;SET CYLINDER
4473 031306 004737 025442      JSR      PC,DRVCL        ;GO DO IT
4474 031312 112765 000115 000001 MOVB    #OFFSET,P.CMND(R5) ;SET UP TO OFFSET
4475 031320 113765 006362 000006 MOVB    OFINUS,P.OFST(R5) ;SET OFFSET VALUE
4476 031326 004737 025442      JSR      PC,DRVCL        ;GO DO IT
4477 031332 104412      1$:     RESREG
4478 031334 000207      RTS      PC
4479
4480 ;*****
4481 ;*****
4482 ;SBTTL PREPARE FOR LOOP ON ERROR
4483 031336 105037 002654      PREPLP: CLRB     EARCNT   ;CLEAR ERROR COUNT
4484 031342 005037 006364      CLR      RECODE        ;CLEAR RECODE
4485 031346 013700 006370      MOV      CCINUS,R0      ;SET CURRENT CYLINDER
4486 031352 013701 006366      MOV      CTINUS,R1      ;SET CURRENT TRACK
4487 031356 012737 026462 002600 MOV      #ERRHDL,A.ABNL  ;SET UP ERROR RETURN
4488 031364 012737 025524 002576 MOV      #ERRFRE,A.NORM  ;SET UP ERROR FREE RETURN
4489 031372 012704 001074      MOV      #STACK-4,R4    ;SET R4 TO STACK BASE -4
4490 031376 011624      MOV      (SP),(R4)+     ;STORE RETURN ON STACK
4491 031400 005014      CLR      (R4)          ;CLEAR FOR 0 PRIORITY

```

```

4492 031402 012706 001074      MOV      #STACK-4,SP      ;SET STACK POINTER FOR RETURN
4493 031406 000002      RTI                      ;DO RTI TO RETURN TO LOOPING ROUTINE
4494
4495
4496      .SBTTL  GET PACK ADDRESS ROUTINE
4497 031410 016537 000030 001174  GTPKAD: MOV      P.DCYL(R5),SREG5 ;GET CYLINDER NUMBER
4498 031416 005037 001176      CLR      $REG6           ;CLEAR REGISTERS FOR
4499 031422 005037 001200      CLR      $REG7           ;TRACK & SECTOR STORAGE
4500 031426 116537 000026 001200  MOVB     P.DTS(R5),SREG7 ;STORE THE TRACK AND SECTOR
4501 031434 116537 000027 001176  MOVB     P.DTS+1(R5),SREG6
4502 031442 005737 001200      TST      $REG7           ;ADJUST THE ADDRESS CONTAINED IN
4503                                     ;THE RK REGISTERS FOR THE AUTOMATIC
4504 031446 001403      BEQ      1$              ;INCREMENT
4505 031450 005337 001200      DEC      $REG7
4506 031454 000426      BR       5$
4507 031456 032765 010000 000016  1$:     BIT      #CFMT,P.CS1(R5)
4508 031464 001404      BEQ      2$
4509 031466 012737 000023 001200  MOV      #19.,SREG7
4510 031474 000403      BR       3$
4511 031476 012737 000025 001200  2$:     MOV      #21.,SREG7
4512 031504 005737 001176  3$:     TST      $REG6
4513 031510 001403      BEQ      4$
4514 031512 005337 001176      DEC      $REG6
4515 031516 000405      BR       5$
4516 031520 012737 000002 001176  4$:     MOV      #2,SREG6
4517 031526 005337 001174      DEC      $REG5
4518 031532 000207  5$:     RTS      PC
4519
4520      ;*****
4521      .SBTTL  BUILD EXPECTED HEADER
4522      ;#USES DESIRED CYLINDER, TRACK AND SECTOR REGISTERS TO DETERMINE
4523      ;#WHICH HEADER WAS EXPECTED. LOADS EXPECTED VALUES IN SREG5, 6, AND
4524      ;#7 FOR REPORTING.
4525      ;*****
4526 031534 104411 000030 001174  BLDEXH: SAVREG
4527 031536 016537 000026      MOV      P.DCYL(R5),SREG5 ;CONSTRUCT EXPECTED HDR
4528 031544 016501 000026      MOV      P.DTS(R5),R1 ;DESIRED CYLINDER & DESIRED TRACK
4529 031550 042701 174377      BIC      #174377,R1      ;CLEAR ALL BUT TRACK BITS
4530 031554 006201      ASR      R1              ;AND SECTOR. SHIFT THE TRACK
4531 031556 006201      ASR      R1              ;OVER TO CONFORM TO HEADER FORMAT
4532 031560 006201      ASR      R1              ;CHECK THE FORMAT BIT AND
4533                                     ;IF SET, SET THE HEADER FORMAT
4534 031562 016537 000026 001176  MOV      P.DTS(R5),SREG6 ;BIT
4535 031570 042737 177740 001176  BIC      #177740,SREG6   ;CLEAR ALL BUT SECTOR
4536 031576 060137 001176      ADD      R1,SREG6        ;ADD TRACK AND SECTOR TOGETHER
4537 031602 052737 140000 001176  BIS      #140000,SREG6   ;INSERT BSE BITS
4538 031610 032765 010000 000016  BIT      #CFMT,P.CS1(R5)
4539 031616 001403      BEQ      23$
4540 031620 052737 001000 001176  23$:    BIS      #1000,SREG6
4541 031626 013737 001174 001200  MOV      $REG5,SREG7     ;COMPUTE THE HEADER VRC
4542 031634 013701 001176      MOV      $REG6,R1
4543 031640 043737 001176 001200  BIC      $REG6,$REG7
4544 031646 043701 001174      BIC      $REG5,R1
4545 031652 050137 001200      BIS      R1,$REG7
4546 031656 104412      RESREG
4547 031660 000207      RTS      PC
;*****

```

.SBTTL WRITE BAD SECTOR FILE ROUTINE
;*THIS ROUTINE WRITES THE BSSOFT (BAD SECTOR FILE DETECTED BY SOFTWARE)
;*INTO THE APPROPRIATE BAD SECTOR FILE. IT DOES NOT WRITE THE FACTORY
;*DETECTED FILES.

BOSTWT:

4548
4549
4550
4551
4552
4553 031662
4554 031662 104411
4555 031664 105037 002650
4556 031670 012703 000006
4557 031674 012702 000012
4558 031700 005065 000014
4559 031704 132737 000020 002653
4560 031712 001404
4561 031714 005202
4562 031716 142765 000020 000007
4563 031724 012765 003660 000010 1\$:
4564 031732 112765 000123 000001
4565 031740 013765 017312 000002
4566 031746 112765 000002 000005
4567 031754 110265 000004
4568 031760 012765 177400 000012
4569 031766 004737 025442
4570 031772 032737 100000 006364
4571 032000 001005
4572 032002 062702 000002 2\$:
4573 032006 005303
4574 032010 001345
4575 032012 000417
4576 032014 105237 002650 3\$:
4577 032020 122737 000007 002650
4578 032026 002736
4579 032030 104401 015425
4580 032034 104401 015513
4581 032040 010246
4582 032042 104402
4583 032044 104401 001267
4584 032050 000754
4585 032052 104401 015535 4\$:
4586 032056 104412
4587 032060 000207

SAVREG
CLRB DERCNT ;CLEAR ERROR COUNT
MOV #6,R3 ;PRESET FOR 6 SECTORS TO BE WRITTEN
MOV #12,R2 ;PRESET TO 22 SECTOR/TRACK
CLR P.PAST(R5) ;CLEAR PROG STAT REG
BITB #8.CFMT,TYPFMT ;TEST IF 20 OR 22 SEC/TRACK
BEQ 1\$;22 SECTOR - SKIP
INC R2 ;20 SECTOR - BUMP TO NEXT SECTOR (13)
BICB #8.CFMT,P.CS1H(R5) ;CLEAR FOR 22 SECTOR WRITE
MOV #BSSOFT,P.BALO(R5) ;SET UP PARAMETER BLOCK
MOVB #WRDATA,P.CMND(R5) ;TO WRITE THE SECTORS OF BAD
MOV LCYL,P.CYLN(R5) ;SECTOR TABLE (SOFTWARE DETECTED)
MOVB #2,P.TRACK(R5)
MOVB R2,P.SECT(R5) ;INSERT SECTOR NUMBER
MOV #177400,P.WC(R5)
JSR PC,DRVCAL
BIT #ANYDER,RECODE ;CHECK IF ANY ERROR IN WRITE
BNE 3\$;IF YES - SKIP
ADD #2,R2 ;ELSE BUMP SECTOR BY TWO
DEC R3 ;DECREMENT COUNTER
BNE 1\$;EXIT IF DONE
BR 4\$
INCB DERCNT ;COUNT ERROR FOR RETRY
CMPB #7,DERCNT ;IF TO MANY TYPE MESSAGE
BLT 1\$;ELSE RETRY
TYPE ,BSWERR ;ERROR MESSAGE
TYPE ,SECINE
MOV R2,-(SP)
TYPOC
TYPE ,SCRLF
BR 2\$
TYPE ,BSWTFI ;TYPE FINISHED MESSAGE
RESREG
RTS PC

.SBTTL BAD SECTOR REPORT ROUTINE

BSREPT:

4588
4589
4590
4591 032062
4592 032062 104411
4593 032064 005037 001242
4594 032070 104401 015572
4595 032074 005000
4596 032076 012737 016006 032124
4597 032104 132737 000020 002653
4598 032112 001403
4599 032114 012737 016003 032124
4600 032122 104401 1\$:
4601 032124 000000 2\$:
4602 032126 104401 015642
4603 032132 104401 015670

SAVREG
CLR STMP2 ;CLEAR TEMP 2 FOR PASS SWITCH
TYPE ,BSHEAD ;TYPE HEADER LINE
CLR R0 ;CLEAR R0 FOR COUNTER
MOV #TWENTY,2\$;CHOOSE THE MODE TO REPORT
BITB #8.CFMT,TYPFMT ;20 OR 22 SECTOR FORMAT
BEQ 1\$;GUESSED RIGHT - SKIP
MOV #TWENTY,2\$;GUESSED WRONG, SET TO 22 SECTOR
TYPE
WORD
TYPE ,BSTAIL
TYPE ,FBSLAB ;TYPE FACTORY LABEL

4604	032136	012704	004670		MOV	#BSFACT+10,R4	:SET R4 TO POINT TO FACTORY TABLE
4605	032142	104401	016011	6\$:	TYPE	,COLHD	:TYPE COLUMN HEADER
4606	032146	104401	016046	3\$:	TYPE	,SPACE4	:SET FORMAT OF REPORT
4607	032152	012446			MOV	(R4)+,-(SP)	:GET FIRST CYLINDER WORD
4608	032154	100417			BMI	4\$:IF ALL 1'S EXIT THIS LOOP
4609	032156	104402			TYPOC		:ELSE TYPE VALUE
4610	032160	104401	012765		TYPE	,SPACE2	:FORMAT
4611	032164	105724			TSTB	(R4)+	:BUMP R4 TO THE TRACK IN TRK/SEC WORD
4612	032166	111446			MOVB	(R4),-(SP)	:PUT TRACK ON THE STACK
4613	032170	104402			TYPOC		:TYPE TRACK NUMBER
4614	032172	104401	012765		TYPE	,SPACE2	:FORMAT
4615	032176	114446			MOVB	-(R4),-(SP)	:PUT SECTOR ON STACK
4616	032200	104402			TYPOC		:TYPE SECTOR NUMBER
4617	032202	104401	001267		TYPE	,SCLRF	:RETURN THE CARRIAGE
4618	032206	005724			TST	(R4)+	:PUT R4 AT NEXT FILE ENTRY
4619	032210	005200			INC	RO	:COUNT BAD SECTORS
4620	032212	000755			BR	3\$:LOOP
4621	032214	005726		4\$:	TST	(SP)+	:CLEAN OFF STACK
4622	032216	005737	001242		TST	\$TMP2	:TEST IF 2ND PASS, IF YES
4623	032222	001016			BNE	5\$:EXIT
4624	032224	052737	000001	001242	BIS	#1,\$TMP2	:ELSE SET 2ND PASS SWITCH
4625	032232	012704	003670		MOV	#BSSOFT+10,R4	:SWITCH TO OTHER TABLE
4626	032236	005700			TST	RO	
4627	032240	001004			BNE	7\$	
4628	032242	104401	001267		TYPE	,SCLRF	
4629	032246	104401	015774		TYPE	,NONE	
4630	032252	104401	015731	7\$:	TYPE	,SBSLAB	:TYPE SOFTWARE LABEL
4631	032256	000731			BR	6\$:GO BACK TO LOOP
4632	032260	104401	001267	5\$:	TYPE	,SCLRF	
4633	032264	005700			TST	RO	
4634	032266	001405			BEG	9\$	
4635	032270	010046			MOV	RO,-(SP)	:EXIT ROUTINE - TYPE THE
4636	032272	104402			TYPOC		:NUMBER OF BAD SECTORS
4637	032274	104401	016057		TYPE	,TOTMES	:AND RETURN
4638	032300	000402			BR	8\$	
4639	032302	104401	015774	9\$:	TYPE	,NONE	
4640	032306	104412		8\$:	RESREG		
4641	032310	000207			RTS	PC	

.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.
*

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
4674
4675
4676
4677
4678
4679
4680
4681
4682
4683
4684 032312 010546
4685 032314 010446
4686 032316 010346
4687 032320 010246
4688 032322 013746 177776
4689 032326 005337 002610
4690 032332 001034
4691 032334 013737 002612 002610
4692 032342 105737 002632
4693 032346 001426
4694 032350 013737 002574 177776
4695 032356 013702 002570
4696 032362 005337 002646
4697 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 Z0S ;IF NOT ZERO RETURN
MOV W.MILI,W.MTIM ;REINITIALIZE MILLISECOND TIMER
TSTB W.TIME ;CHECK IF DRIVE IS BEING TIMED
BEQ Z0S ;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 Z0S ;RETURN IF NO TIME OUT

4698	032370	105037	002632		CLRB	W.TIME	:RESET TIMING INDICATOR
4699	032374	013705	002644		MOV	PBLKT,R5	:LOAD ADDRESS OF PARAMETER BLOCK
4700							:TABLE FOR INDEXING
4701	032400	052765	000100	000014	BIS	#CMDTO P.PRST(R5)	:SET COMMAND TIME OUT
4702	032406	020537	002606		CMP	R5,O.WAIT	:CHECK IF DRIVER IS WAITING FOR
4703							:COMMAND COMPLETION
4704	032412	001002			BNE	SS	:NO DO NOT ALTER WAITING FOR
4705							:COMMAND COMPLETION
4706	032414	005037	002606		CLR	O.WAIT	:CLEAR WAIT FOR COMMAND COMPLETION
4707	032420	004737	035674	SS:	JSR	PC R.ABNL	:BRANCH TO ERROR ROUTINE
4708	032424	012637	177776	20\$:	MOV	(SP)+,PS	:RESTORE PSW
4709	032430	012602			MOV	(SP)+,R2	:RESTORE R2
4710	032432	012603			MOV	(SP)+,R3	:RESTORE R3
4711	032434	012604			MOV	(SP)+,R4	:RESTORE R4
4712	032436	012605			MOV	(SP)+,R5	:RESTORE R5
4713	032440	000207			RTS	PC	:RETURN

.SBTTL *RK06 INTERRUPT SERVICE ROUTINE

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
4748
4749
4750
4751
4752
4753
4754
4755
4756
4757

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

4758	032442	010546		
4759	032444	010446		
4760	032446	010346		
4761	032450	010246		
4762	032452	010146		
4763	032454	010046		
4764	032456	013702	002570	
4765	032462	016237	000010	002534
4766	032470	032737	001000	002534
4767	032476	001407		
4768	032500	052737	100000	002604
4769	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 IS ;NO. CONTINUE PROCESSING
        BIS #E.MDS, E.CONT ;SET MULTIPLE DRIVE SELECT
        JSR PC, R.CONT ;REPORT ERROR
  
```

```

4770 032512 000137 034672          JMP      I.RTRN          ;RETURN
4771
4772 032516 105737 002626          1$:    TSTB      I.ISRL          ;CHECK IF INTERRUPT OR RELEASE
4773 032522 001410          BEQ      6$              ;NO, CHECK IF DRIVE AVAILABLE
4774 032524 100403          BMI      5$              ;CHECK IF RELEASE COMMAND
4775 032526 105037 002626          CLRB     I.ISRL          ;YES CLEAR FLAG
4776 032532 000473          BR       I.IGO          ;CONTINUE PROCESSING INTERRUPT
4777
4778 032534 105037 002626          5$:    CLRB     I.ISRL          ;CLEAR FLAG
4779 032540 000137 033654          JMP      I.ATTN          ;GO PROCESS DRIVE ATTENTIONS
4780
4781 032544 032737 010400 002534 6$:    BIT       #NED!UFE,T.CS2 ;CHECK FOR NON-EXISTENT DRIVE OR
4782                                     UNIT FIELD ERROR
4783 032552 001413          BEQ      7$              ;NO, WAIT FOR DUAL ACCESS INTERRUPT
4784 032554 013704 002534          MOV      T.CS2,R4        ;LOAD R4 FOR DRIVE NUMBER
4785 032560 042704 177770          BIC      #I<DRVMSK>,R4   ;KEEP DRIVE BITS
4786 032564 013705 002644          MOV      PBLKT,R5        ;STORE PARAMETER BLOCK ADDRESS
4787 032570 016237 000000 002532          MOV      RKCS1(R2),T.CS1 ;LOAD TEMPORARY CS1 FOR STATUS REPORT
4788 032576 000137 033064          JMP      I.ERRC          ;REPORT ERROR
4789
4790 032602 016237 000012 002552 7$:    MOV      RKDS(R2),T.DS   ;STORE STATUS REGISTER FOR COMPARISON
4791 032610 032737 000001 002552          BIT      #DRA,T.DS      ;CHECK IF DRIVE SEIZED BY OTHER
4792                                     PORT
4793 032616 001041          BNE      I.I00          ;NO, CONTINUE PROCESSING INTERRUPT
4794
4795                                     ;CHECK IF ANY DATA TRANSFER ERROR EXISTS
4796 032620 032737 164000 002534          BIT      #DLT!WCE!UPE!NEM,T.CS2
4797
4798 032626 001007          BNE      10$             ;INDICATE ERROR
4799 032630 016237 000014 002550          MOV      RKER(R2),T.ER   ;STORE ERROR REGISTER
4800
4801                                     ;CHECK FOR DATA TRANSFER ERROR TYPE ERROR
4802 032636 032737 125700 002550          BIT      #DCK!OPI!WLE!COE!HVRC!BSE!ECH,T.ER
4803
4804 032644 001407          BEQ      11$             ;NO, WAIT FOR RELEASE OF RK06 DRIVE
4805
4806 032646 052737 000010 002604 10$:   BIS      #E.UDAT,E.CONT  ;SET UNEXPECTED DATA TYPE ERROR
4807 032654 004737 035720          JSR      PC,R.CONT       ;REPORT ERROR
4808 032660 000137 034672          JMP      I.RTRN          ;RESTORE REGISTERS
4809
4810 032664 105037 002632          11$:   CLRB     W.TIME         ;RESET TIMING ON THIS DRIVE
4811 032670 005037 002646          CLR      W.DRV           ;CLEAR TIMING COUNT FOR THIS DRIVE
4812 032674 013705 002644          MOV      PBLKT,R5        ;LOAD R5 WITH PARAMETER BLOCK
4813                                     ADDRESS
4814 032700 052765 010000 000014          BIS      #DRVSZD,P.PRST(R5) ;SET DRIVE SEIZED IN THE
4815                                     PROGRAM DRIVE STATUS REGISTER
4816 032706 005037 002606          CLR      0.WAIT          ;CLEAR WAIT FOR COMMAND COMPLETION
4817 032712 004737 035674          JSR      PC,R.ABNL       ;INDICATE ABNORMAL TERMINATION
4818 032716 000137 034672          JMP      I.RTRN          ;GO RESTORE REGISTERS
4819
4820 032722 013705 002606          I.I00: MOV      0.WAIT,R5      ;LOAD PARAMETER BLOCK ADDRESS INTO R5
4821 032726 001002          BNE      2$              ;IS COMMAND WAITING PROCESSING
4822                                     ;YES, DO PROCESSING
4823 032730 000137 033654          JMP      I.ATTN          ;NO, PROCESS ATTENTION
4824
4825 032734 013704 002534          2$:    MOV      T.CS2,R4        ;STORE RKCS2 FOR DRIVE NUMBER

```

```

4826 032740 042704 177770          BIC      #1C<DRVMSK>,R4 ;MASK OUT UNNECESSARY BITS
4827
4828
4829 032744 126504 000000          CMPB    P.DRVN(R5),R4 ;CHECK IF DRIVE NUMBER IS EXPECTED
4830 032750 001401          BEQ     3$           ;YES, CONTINUE
4831 032752 000000          HALT                    ;NO, DRIVER ERROR
4832 032754 122765 000164 000001 3$:    CMPB    #RDALHD,P.CMND(R5) ;CHECK IF READ ALL HEADERS
4833 032762 001002          BNE    10$          ;NO, EXECUTE NORMAL DATA TRANSFER
4834 032764 000137 033322          JMP     I.HDAL        ;GO EXECUTE SPECIAL HEADER SEQUENCE
4835
4836 032770 005037 002606          10$:   CLR     O.WAIT        ;CLEAR WAIT FOR COMMAND COMPLETION
4837 032774 005037 002646          CLR     W.DRV        ;CLEAR WATCH-DOG TIME
4838 033000 105037 002632          CLRB   W.TIME        ;RESET TIMING ON THIS DRIVE
4839 033004 016237 000000 002532      MOV     RKCS1(R2),T.CS1 ;STORE COMMAND AND STATUS REGISTER 1
4840 033012 032737 100000 002532      BIT     #CERR,T.CS1   ;CHECK IF CONTROLLER ERROR
4841 033020 001021          BNE    I.ERRC        ;YES, PROCESS ERROR
4842 033022 016237 000016 002546      MOV     RKASOF(R2),T.ASOF ;STORE ATTENTION SUMMARY
4843 033030 133737 002633 002547      BITB   INTMSK,T.ASOF+1 ;CHECK IF DRIVE ATTENTION SET
4844 033036 001004          BNF    15$          ;YES, REPORT ERROR
4845 033040 004737 035706          JSR    PC.R.NORM     ;INDICATE NORMAL RETURN
4846 033044 000137 034672          JMP     I.RTRN       ;RESTORE REGISTERS
4847
4848 033050 052765 000010 000014 15$:   BIS    #UEXATT,P.PRST(R5) ;SET UNEXPECTED ATTENTION
4849
4850 033056 004737 035342          I.ERRA: JSR    PC.I.CSTS   ;STORE CONTROLLER STATUS
4851 033062 000405          BR     I.ERR         ;STORE PATTERN AND POSITION INFORMATION
4852
4853 033064 013765 002532 000016  I.ERRC: MOV     T.CS1,P.CS1(R5) ;GET ERROR RKCS1
4854 033072 004737 035364          JSR    PC.I.CST1     ;GET REST OF CONTROLLER STATUS
4855 033076 016265 000032 000062  I.ERR:  MOV     RKECPT(R2),P.EPAT(R5) ;STORE ECC PATTERN
4856 033104 016265 000030 000060      MOV     RKECPS(R2),P.EPOS(R5) ;STORE ECC POSITION
4857 033112 004037 034710          JSR    RD,I.CCLR     ;CLEAR CONTROLLER
4858 033116 034672          I.RTRN              ;ERROR RETURN
4859 033120 032765 010400 000020      BIT     #NED!UFE,P.CS2(R5) ;CHECK IF IT WAS NON-EXISTENT DRIVE OR
4860                                     ;UNIT FIELD ERROR
4861                                     ;YES, REPORT ERROR
4862 033130 004037 035446          JSR    RD,I.STAT     ;GATHER DRIVE STATUS
4863 033134 034672          I.RTRN              ;ERROR RETURN
4864 033136 112737 000005 002532      MOVB   #DR.CLR,T.CS1 ;LOAD COMMAND
4865 033144 004037 034772          JSR    RD,I.ISSU     ;ISSUE DRIVE CLEAR
4866 033150 034672          I.RTRN              ;ERROR RETURN
4867 033152 133737 002633 002547      BITB   INTMSK,T.ASOF+1 ;CHECK IF ATTENTION RESET
4868 033160 001407          BEQ    2$           ;NO, INDICATE DRIVE ERROR
4869 033162 052737 000020 002604      BIS    #E.CLAT,E.CONT ;SET ATTENTION DID NOT RESET
4870                                     ;WITH CLEAR
4871                                     ;REPORT CONTROLLER ERROR
4871 033170 004737 035720          JSR    PC.R.CONT     ;REPORT CONTROLLER ERROR
4872 033174 000137 034672          JMP     I.RTRN       ;GO RESTORE REGISTERS
4873
4874 033200 032737 040000 002556 2$:    BIT     #S.DSC,T.MR2   ;CHECK IF DRIVE STATUS CHANGE CLEARED
4875 033206 001403          BEQ    3$           ;YES, CHECK FAULT
4876 033210 052765 000040 000014      BIS    #DRVDSC,P.PRST(R5) ;SET DSC DID NOT CLEAR
4877 033216 032737 001000 002560 3$:    BIT     #S.PAR,T.MR3   ;CHECK IF DRIVE PARITY ERROR
4878 033224 001407          BEQ    5$           ;NO, INDICATE ABNORMAL TERMINATION
4879 033226 052737 002000 002604      BIS    #E.DPAR,E.CONT ;SET DRIVE PARITY ERROR
4880 033234 004737 035720          JSR    PC.R.CONT     ;INDICATE CONTROLLER ERROR
4881 033240 000137 034672          JMP     I.RTRN       ;RETURN

```

```

4882
4883 033244 032765 000020 000014 5$: BIT #DRVHRD,P.PRST(R5) ;CHECK IF HARD DRIVE ERROR
4884 033252 001017 BNE 10$ ;YES, GO REPORT ERROR
4885 033254 032737 020000 002556 BIT #S.PIP,T.MR2 ;CHECK IF DRIVE IS CYCLING DOWN
4886 033262 001413 BEQ 10$ ;NO, REPORT ERROR
4887 033264 052765 020000 000014 BIS #E.UNLD,P.PRST(R5) ;SET DRIVE UNLOADING
4888 033272 113737 002633 002632 MOVB INTMSK,W.TIME ;SET UP 8 SECONDS FOR DRIVE TO CYCLE UP
4889 033300 013737 002616 002646 MOV W.BSEC,W.DRV
4890 033306 000137 034672 JMP I.RTRN ;GO RESTORE REGISTERS
4891
4892 033312 004737 035674 10$: JSR PC,R.ABNL ;GO REPORT ERROR
4893 033316 000137 034672 JMP I.RTRN ;GO RESTORE REGISTERS
4894
4895 .SBTTL *READ ALL HEADERS INTERRUPT SEQUENCE
4896
4897 033322 016237 000000 002532 I.HDAL: MOV RKCS1(R2),T.CS1 ;STORE CS1 TO CHECK CONTROLLER
4898 ;ERROR
4899 033330 032737 100000 002532 BIT #CERR,T.CS1 ;CHECK IF CONTROLLER ERROR
4900 033336 001422 BEQ 5$ ;NO, CHECK FOR ATTENTION
4901
4902 033340 005037 002606 CLR O.WAIT ;CLEAR WAITING FOR COMMAND COMPLETE
4903 033344 105037 002632 CLR W.TIME ;RESET TIMING ON DRIVE
4904 033350 005037 002646 CLR W.DRV ;CLEAR TIME OUT COUNT
4905 033354 013765 002532 000016 MOV T.CS1,P.CS1(R5) ;STORE ERROR RKCS1
4906 033362 004737 035364 JSR PC,I.CST1 ;STORE CONTROLLER REGISTERS
4907 033366 004037 034710 JSR RO,I.CCLR ;CLEAR CONTROLLER
4908 033372 034672 I.RTRN ;ERROR RETURN
4909 033374 004737 035674 JSR PC,R.ABNL ;INDICATE ERROR RETURN
4910 033400 000137 034672 JMP I.RTRN ;RESTORE REGISTERS
4911
4912 033404 016237 000016 002546 5$: MOV RKASOF(R2),T.ASOF ;STORE ATTENTION SUMMARY
4913 033412 133737 002633 002547 BITB INTMSK,T.ASOF+1 ;CHECK IF DRIVE ATTENTION IS SET
4914 033420 001410 BEQ 7$ ;NO, CHECK IF READ ALL HEADERS
4915 033422 005037 002606 CLR O.WAIT ;CLEAR WAITING FOR COMMAND COMPLETION
4916 033426 105037 002632 CLR W.TIME ;RESET TIMING ON DRIVE
4917 033432 005037 002646 CLR W.DRV ;CLEAR TIME OUT COUNT
4918 033436 000137 033056 JMP I.ERRA ;GO REPORT ERROR
4919
4920 033442 013701 002622 7$: MOV HDR.AD,R1 ;GET MAIN MEMORY ADDRESS
4921 033446 016221 000024 MOV RKDB(R2),(R1)+ ;GET FIRST WORD OF HEADER
4922 033452 016221 000024 MOV RKDB(R2),(R1)+ ;GET SECOND WORD OF HEADER
4923 033456 016221 000024 MOV RKDB(R2),(R1)+ ;GET THIRD WORD OF HEADER
4924 033462 010137 002622 MOV R1,HDR.AD ;STORE ADDRESS FOR NEXT HEADER
4925 033466 016237 000010 002534 MOV RKCS2(R2),T.CS2 ;STORE CS2 TO CHECK FOR DATA LATE
4926 033474 032737 100000 002534 BIT #DLT,T.CS2 ;CHECK FOR DATA LATE
4927 033502 001055 BNE 35$ ;YES, REPORT ERROR
4928 033504 005337 002624 DEC HDR.CT ;DECREMENT NUMBER OF HEADER YET TO READ
4929 033510 001026 BNE 25$ ;IF NON-ZERO, GO ISSUE NEXT READ HEADER
4930 033512 005037 002606 CLR O.WAIT ;CLEAR DRIVER WAITING FOR COMMAND COMPLETION
4931 033516 005037 002646 CLR W.DRV ;CLEAR TIME OUT COUNT FOR THIS DRIVE
4932 033522 105037 002632 CLR W.TIME ;CLEAR WATCH DOG TIME ON THIS DRIVE
4933 033526 012762 000003 000026 MOV #3,RKMR1(R2) ;LOAD MAINTENANCE REGISTER FOR SECTOR COUNT
4934 033534 112737 000001 002532 MOVB #DR.SEL,T.CS1 ;LOAD SELECT COMMAND
4935 033542 004037 034772 JSR RO,I.ISSU ;GET SECTOR COUNT
4936 033546 034672 I.RTRN ;ERROR RETURN
4937 033550 013765 002560 000056 MOV T.MR3,P.B11(R5) ;LOAD SECTOR COUNT

```

```

4938 033556 004737 035706 JSR PC.R.NORM ;INDICATE NORMAL TERMINATION
4939 033562 000137 034672 JMP I.RTRN ;RESTORE REGISTERS
4940
4941 033566 016562 000002 000020 25$: MOV P.CYLN(R5),RKDCYL(R2) ;LOAD CYLINDER ADDRESS REGISTER
4942 033574 016562 000004 000006 MOV P.SECT(R5),RKDA(R2) ;LOAD SECTOR AND TRACK
4943 033602 116565 000007 000017 MOV#B P.CS1H(R5),P.CS1+1(R5) ;STORE BITS 8-15 OF CS1
4944 033610 042765 165777 000016 BIC #C<CDT!CFMT>,P.CS1(R5) ;CLEAR ALL BITS EXCEPT FORMAT AND
4945 ; DRIVE TYPE
4946 033616 112765 000125 000016 MOV#B #RDHEAD,P.CS1(R5) ;STORE COMMAND ISSUED
4947 033624 016562 000016 000000 MOV P.CS1(R5),RKCS1(R2) ;ISSUE READ HEADER
4948 033632 000137 034672 JMP I.RTRN ;RESTORE REGISTERS
4949
4950 033636 052737 000400 002604 35$: BIS #E.DLT,E.CONT ;SET DATA LATE WHILE UNLOADING HEADER
4951 033644 004737 035720 JSR PC.R.CONT ;REPORT ERROR
4952 033650 000137 034672 JMP I.RTRN ;RESTORE REGISTERS
4953
4954 .SBTTL *DRIVE ATTENTION SCANNER
4955
4956 033654 016237 000000 002532 I.ATTN: MOV RKCS1(R2),T.CS1 ;STORE COMMAND AND STATUS
4957 ; REGISTER 1 FOR COMPARISON
4958 033662 032737 100000 002532 BIT #CERR,T.CS1 ;CHECK IF CONTROLLER ERROR OCCURRED
4959 033670 001441 BEQ SS ;NO, CHECK IF ATTENTION
4960
4961 ;CHECK IF ANY DATA TRANSFER TYPE ERROR EXISTS
4962 033672 032737 164000 002534 BIT #DLT!WCE!UPE!NEM,T.CS2
4963
4964 033700 001007 BNE 1$ ;INDICATE ERROR
4965 033702 016237 000014 002550 MOV RKER(R2),T.ER ;STORE ERROR REGISTER
4966
4967 ; CHECK FOR DATA TRANSFER ERROR TYPE
4968 033710 032737 125700 002550 BIT #DCK!OPI!WLE!COE!HVRC!BSE!ECH,T.ER
4969
4970 033716 001407 BEQ 2$ ;NO DATA TRANSFER ERROR
4971
4972 033720 052737 000010 002604 1$: BIS #E.UDAT,E.CONT ;SET UNEXPECTED DATA TYPE ERROR
4973 033726 004737 035720 JSR PC.R.CONT ;REPORT ERROR
4974 033732 000137 034672 JMP I.RTRN ;RESTORE REGISTERS
4975
4976 033736 013704 002534 2$: MOV T.CS2,R4 ;SAVE CS2 FOR REGISTER NUMBER
4977 033742 042704 177770 BIC #C<DRVMSK>,R4 ;STRIP OFF JUNK
4978 033746 105037 002632 CLR#B W.TIME ;CLEAR WATCH DOG TIMER
4979 033752 005037 002646 CLR W.DRV ;RESET TIMER VALUE
4980 033756 013705 002644 MOV PBLKT,R5 ;STORE PARAMETER BLOCK ADDRESS IN R5
4981
4982 ; CLEAR DRIVE POSITIONING AND DRIVE POSITIONED FOR DATA TRANSFER
4983 ; IN PROGRAM DEVICE STATUS REGISTER
4984 033762 042765 000006 000014 BIC #DRVPOS!DRVPDT,P.PRST(R5)
4985
4986 033770 000137 033064 JMP I.ERRC ;GO REPORT ERROR
4987
4988 033774 032737 040000 002532 5$: BIT #DI,T.CS1 ;CHECK IF ANY DRIVE ATTENTION
4989 034002 001002 BNE 6$ ;YES, PROCESS INTERRUPT
4990 034004 000137 034672 JMP I.RTRN ;RESTORE REGISTERS
4991
4992 034010 016237 000016 002546 6$: MOV RKASOF(R2),T.ASOF ;STORE ATTENTION SUMMARY
4993 034016 105737 002547 TSTB T.ASOF+1 ;CHECK IF ANY ATTENTIONS SET

```



```

.SBTTL *ATTENTION ERROR HANDLER
5050
5051
5052 034324 042765 000004 000014 I.AERR: BIC #DRVPTD,P.PRST(R5) ;RESET POSITIONING IN PROGRESS BECAUSE
5053 ; OF DATA TRANSFER
5054 034332 105037 002632 CLR W.TIME ;CLEAR TIMING FOR THIS DRIVE
5055 034336 005037 002646 CLR W.DRV ;RESET WATCH-DOG TIME
5056 034342 042765 177741 000016 BIC #177741,P.CS1(R5) ;KEEP COMMAND ISSUED
5057 034350 042737 000036 002532 BIC #36,T.CS1 ;KEEP CURRENT CONTROLLER STATUS
5058 034356 053765 002532 000016 BIS T.CS1,P.CS1(R5) ;MAKE GOOD MESSAGE
5059 034364 013765 002534 000020 MOV T.CS2,P.CS2(R5) ;STORE CONTROLLER REGISTERS
5060 034372 013765 002536 000022 MOV T.WCR,P.WCR(R5)
5061 034400 013765 002540 000024 MOV T.BA,P.BAR(R5)
5062 034406 013765 002542 000026 MOV T.DA,P.DTS(R5)
5063 034414 013765 002544 000030 MOV T.DC,P.DCYL(R5)
5064 034422 013765 002546 000032 MOV T.ASOF,P.ASOF(R5)
5065 034430 013765 002550 000034 MOV T.ER,P.ER(R5)
5066 034436 013765 002552 000036 MOV T.DS,P.DS(R5)
5067 034444 004037 035446 JSR RO,I.STAT ;GATHER DRIVE STATUS
5068 034450 034672 I.RTRN ;ERROR RETURN
5069 034452 112737 000005 002532 MOVB #DR.CLR,T.CS1 ;LOAD COMMAND
5070 034460 004037 034772 JSR RO,I.ISSU ;CLEAR DRIVE ERRORS
5071 034464 034672 I.RTRN ;ERROR RETURN
5072 034466 133737 002633 002547 BITB INTMSK,T.ASOF+1 ;CHECK IF ATTENTION RESET
5073 034474 001407 BEQ 2$ ;YES, FLAG DRIVE ERROR
5074 034476 052737 000020 002604 BIS #E.CLAT,E.CONT ;SET ATTENTION DID NOT RESET
5075 034504 004737 035720 JSR PC.R.CONT ;REPORT ERROR
5076 034510 000137 034672 JMP I.RTRN ;RESTORE REGISTERS
5077
5078 034514 032765 000020 000014 2$: BIT #DRVHRD,P.PRST(R5) ;CHECK IF A HARD DRIVE ERROR
5079 034522 001017 BNE 10$ ;YES, REPORT ERROR
5080 034524 032737 020000 002556 BIT #S.PIP,T.MR2 ;CHECK IF DRIVE IS UNLOADING
5081 034532 001413 BEQ 10$ ;NO, REPORT ERROR
5082 034534 052765 020000 000014 BIS #E.UNLD,P.PRST(R5) ;SET DRIVE UNLOADING DUE TO ERROR
5083 034542 113737 002633 002632 MOVB INTMSK,W.TIME ;SET TIMING ON THIS DRIVE
5084 034550 013737 002616 002646 MOV W.BSEC,W.DRV ;LOAD 8 SECONDS FOR CYCLE UP TIME
5085 034556 000137 034672 JMP I.RTRN ;RESTORE REGISTERS
5086
5087 034562 004737 035674 10$: JSR PC.R.ABNL ;REPORT ERROR
5088 034566 000137 034672 JMP I.RTRN ;RESTORE REGISTERS
5089
5090 .SBTTL *ERROR CAUSING DRIVE TO UNLOAD
5091
5092 034572 052765 020000 000014 I.UNLD: BIS #E.UNLD,P.PRST(R5) ;CLEAR DRIVE UNLOADING BECAUSE OF ERROR
5093 034600 112737 000005 002532 MOVB #DR.CLR,T.CS1 ;LOAD IN DRIVE CLEAR
5094 034606 004037 034772 JSR RO,I.ISSU ;GO ISSUE DRIVE CLEAR
5095 034612 034672 I.RTRN ;ERROR RETURN
5096 034614 136437 002633 002547 BITB INTMSK(R4),T.ASOF+1 ;CHECK IF ATTENTION CLEARED
5097 034622 001406 BEQ 15$ ;YES, CONTINUE
5098 034624 012737 000020 002604 MOV #E.CLAT,E.CONT ;SET ATTENTION DID NOT RESET
5099 034632 004737 035720 JSR PC.R.CONT ;REPORT ERROR
5100 034636 000415 BR I.RTRN ;RESTORE REGISTERS
5101
5102 034640 032737 040000 002556 15$: BIT #S.DSC,T.MR2 ;CHECK IF DRIVE STAUUS CHANGE RESET
5103 034646 001403 BEQ 20$ ;YES, CONTINUE
5104 034650 052765 000040 000014 BIS #DRVDSC,P.PRST(R5) ;SET DRIVE STAUUS CHANGE DID NCT CLEAR
5105 034656 105037 002632 20$: CLRB W.TIME ;RESET TIMING ON THIS DRIVE
    
```

5106	034662	005037	002646	CLR	W.DRV	:CLEAR TIME COUNT
5107	034666	004737	035674	JSR	PC,R.ABNL	:REPORT ERROR
5108						
5109	034672	012600		I.RTRN: MOV	(SP)+,R0	:RESTORE R0
5110	034674	012601		MOV	(SP)+,R1	:RESTORE R1
5111	034676	012602		MOV	(SP)+,R2	:RESTORE R2
5112	034700	012603		MOV	(SP)+,R3	:RESTORE R3
5113	034702	012604		MOV	(SP)+,R4	:RESTORE R4
5114	034704	012605		MOV	(SP)+,R5	:RESTORE R5
5115	034706	000002		RTI		:RETURN
5116						

.SBTTL *CONTROLLER CLEAR ROUTINE

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

S117
S118
S119
S120
S121
S122
S123
S124
S125
S126
S127
S128
S129
S130
S131
S132
S133
S134
S135
S136
S137
S138
S139
S140
S141
S142
S143
S144
S145
S146
S147
S148
S149
S150
S151

034710	012762	100000	000000	I.CCLR:	MOV	#CCLR,RKCS1(R2)	; CLEAR CONTROLLER
034716	016237	000000	002532		MOV	RKCS1(R2),T.CS1	; STORE COMMAND AND STATUS REGISTER 1
034724	032737	100000	002532		BIT	#CERR,T.CS1	; CHECK IF CONTROLLER CLEAR DID
							; CLEAR ERROR
034732	001407				BEQ	SS	; YES, RETURN TO DRIVER PROCESSING
034734	052737	000001	002604		BIS	#E.CCLR,E.CONT	; SET CLEAR CONTROLLER DID NOT CLEAR ERROR
034742	004737	035720			JSR	PC,R.CONT	; REPORT CONTROLLER ERROR
034746	011000				MOV	(R0),R0	; SET UP ERROR RETURN
034750	000200				RTS	R0	; RETURN
034752	012762	000100	000000	SS:	MOV	#IE,RKCS1(R2)	; SET INTERRUPT ENABLE
034760	112737	177777	002626		MOVB	#-1,I.ISRL	; SET INTERRUPT ENABLE ISSUED
034766	005720				TST	(R0)+	; ADJUST FOR NORMAL RETURN
034770	000200				RTS	R0	; RETURN

.SBTTL *COMMAND ISSUED BY DRIVER SERVICE ROUTINE

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 RD,I,ISSU
<ADDRESS OF ERROR RETURN>
RETURN

ROUTINES USED:

I.CCLR
I.STOR

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
5198
5199
5200
5201
5202
5203
5204
5205
5206
5207

034772 013746 002532
034776 005037 002534
035002 116537 000000 002534
035010 013762 002534 000010
035016 116537 000007 002533
035024 142737 177753 002533
035032 013762 002532 000000
035040 105762 000000
035044 100375
035046 004737 035214
035052 032737 100000 002532
035060 001437
035062 032737 001000 002534
035070 001406
035072 052737 100000 002604
035100 004737 035720
035104 000440
035106 032737 024000 002532
035114 001027
035116 032737 176400 002534
035124 001023
035126 032737 131761 002550
035134 001017
035136 122716 000005

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 #1C<B.CDT!B.CFMT>,T.CS1+1 ;CLEAR ALL BITS EXCEPT
;FORMAT AND DRIVE TYPE
1\$: MOV T.CS1,RKCS1(R2) ;ISSUE COMMAND
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
2\$: ;CHECK IF ANY CONTROLLER ERROR IS SET
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!OPI!DCK,T.ER
BNE 7\$
CMPB #DR.CLR,(SP) ;CHECK IF CLEAR DRIVE

*COMMAND ISSUED BY DRIVER SERVICE ROUTINE

5208	035142	001003				BNE	3\$:NO, DO NOT SET DRIVE HARD ERROR
5209	035144	052765	000020	000014		BIS	#DRVHRD,P.PRST(R5)	:SET HARD DRIVE ERROR
5210	035152	004037	034710		3\$:	JSR	RD,I.CCLR	:GO ISSUE A CONTROLLER CLEAR
5211	035156	035206				10\$:ERROR RETURN
5212	035160	012762	000100	000000	5\$:	MOV	#IE,RKCS1(R2)	:SET INTERRUPT ENABLE
5213	035166	005726				TST	(SP)+	:ADJUST STACK
5214	035170	005720				TST	(RD)+	:ADJUST RD FOR NORMAL RETURN
5215	035172	000200				RTS	RD	:RETURN
5216								
5217	035174	052737	001000	002604	7\$:	BIS	#E.CERR,E.CONT	:SET CONTROLLER ERROR DURING
5218								:DRIVER SERVICING
5219	035202	004737	035720			JSR	PC,R.CONT	:REPORT ERROR
5220	035206	005726			10\$:	TST	(SP)+	:ADJUST STACK
5221	035210	011000				MOV	(RD),RD	:ADJUST RD FOR ERROR RETURN
5222	035212	000200				RTS	RD	:RETURN

.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
*
*****

```

5223
5224
5225
5226
5227
5228
5229
5230
5231
5232
5233
5234
5235
5236
5237
5238
5239
5240
5241
5242
5243
5244
5245
5246
5247
5248
5249
5250
5251
5252
5253
5254

```

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

```

.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

```

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 035342 042765 177741 000016
5286
5287 035350 042737 000036 002532
5288 035356 053765 002532 000016
5289 035364 016265 000010 000020
5290 035372 016265 000002 000022
5291 035400 016265 000004 000024
5292 035406 016265 000006 000026
5293 035414 016265 000012 000036
5294 035422 016265 000014 000034
5295 035430 016265 000016 000032
5296
5297 035436 016265 000020 000030
5298 035444 000207
5299

```

```

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

```

.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 RO,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
*****

```

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	035446	012762	000001	030026	I. STAT: MOV	#1, RKMR1(R2)		: LOAD MAINTENANCE REGISTER 1	
5327								: FOR STATUS BYTE 01	
5328	035454	112737	000001	002532	MOVB	#DR. SEL. T. CS1		: LOAD COMMAND	
5329	035462	004037	034772		JSR	RO, I. ISSU		: GET STATUS BYTES 01	
5330	035466	035656			3\$: ERROR RETURN	
5331	035470	013765	002556	000044	MOV	T. MR2, P. A01(R5)		: STORE STATUS BYTE 01 MESS A	
5332	035476	013765	002560	000046	MOV	T. MR3, P. B01(R5)		: STORE STATUS BYTE 01 MESS B	
5333	035504	012762	000002	000026	MOV	#2, RKMR1(R2)		: LOAD MAINTENANCE REGISTER 1	
5334								: FOR STATUS BYTE 10	
5335	035512	112737	000001	002532	MOVB	#DR. SEL. T. CS1		: LOAD COMMAND	
5336	035520	004037	034772		JSR	RO, I. ISSU		: GET STATUS BYTES 10	
5337	035524	035656			3\$: ERROR RETURN	
5338	035526	013765	002556	000050	MOV	T. MR2, P. A10(R5)		: STORE STATUS BYTE 10 MESS A	
5339	035534	013765	002560	000052	MOV	T. MR3, P. B10(R5)		: STORE STATUS BYTE 10 MESS B	
5340	035542	012762	000003	000026	MOV	#3, RKMR1(R2)		: LOAD MAINTENANCE REGISTER	
5341								: FOR STATUS BYTE 11	
5342	035550	112737	000001	002532	MOVB	#DR. SEL. T. CS1		: LOAD COMMAND	
5343	035556	004037	034772		JSR	RO, I. ISSU		: GET STATUS BYTES 11	
5344	035562	035656			3\$: ERROR RETURN	
5345	035564	013765	002556	000054	MOV	T. MR2, P. A11(R5)		: STORE STATUS BYTE 11 MESS A	
5346	035572	013765	002560	000056	MOV	T. MR3, P. B11(R5)		: STORE STATUS BYTE 11 MESS B	
5347	035600	005062	000026		CLR	RKMR1(R2)		: LOAD MAINTENANCE REGISTER 1	
5348								: FOR STATUS BYTE 00	
5349	035604	112737	000001	002532	MOVB	#DR. SEL. T. CS1		: LOAD COMMAND	
5350	035612	004037	034772		JSR	RO, I. ISSU		: GET STATUS BYTES 00	
5351	035616	035656			3\$: ERROR RETURN	
5352	035620	013765	002556	000040	MOV	T. MR2, P. A00(R5)		: STORE STATUS BYTE 00 MESS A	
5353	035626	013765	002560	000042	MOV	T. MR3, P. B00(R5)		: STORE STATUS BYTE 00 MESS B	
5354	035634	032737	001000	002560	BIT	#S. PAR, T. MR3		: CHECK IF BAD PARITY DETECTED BY DRIVE	
5355	035642	001407			BEG	5\$: NO, RETURN NORMALLY	

5356	035644	052737	002000	002604		BIS	#E.DPAR,E.CONT	:INDICATE BAD PARITY DETECTED BY DRIVE
5357	035652	004737	035720			JSR	PC R.CONT	:REPORT ERROR
5358	035656	011000			3\$:	MOV	(R0),R0	:LOAD R0 FOR ERROR RETURN
5359	035660	000200				RTS	R0	:RETURN
5360								
5361	035662	052765	001000	000014	5\$:	BIS	#PBSVAL,P.PRST(R5)	:SET PARAMETER BLOCK STATUS VALID
5362	035670	005720				TST	(R0)+	:ADJUST R0 FOR NORMAL RETURN
5363	035672	000200				RTS	R0	:RETURN
5364								

.SBTTL *COMMON DRIVER RETURNS

5365			
5366			
5367	035674	105037	002633
5368	035700	004777	144674
5369	035704	000207	
5370			
5371	035706	105037	002633
5372	035712	004777	144660
5373	035716	000207	
5374			
5375	035720	105037	002633
5376	035724	105037	002632
5377	035730	005037	002646
5378	035734	004777	144642
5379	035740	000207	

R.ABNL:	CLRB	INTMSK	: INHIBIT FUTURE DRIVE INTERRUPT REPORTING
	ISR	PC, @A.ABNL	: INDICATE ABNORMAL RETURN
	RTS	PC	: RETURN
R.NORM:	CLRB	INTMSK	: INHIBIT FUTURE DRIVE INTERRUPT REPORTING
	JSR	PC, @A.NORM	: INDICATE NORMAL RETURN
	RTS	PC	: RETURN
R.CONT:	CLRB	INTMSK	: INHIBIT FUTURE DRIVE INTERRUPT REPORTING
	CLRB	W.TIME	: RESET WATCH DOG TIMING ON THIS DRIVE
	CLR	W.DRV	: CLEAR TIMING COUNT FOR THIS DRIVE
	JSR	PC, @A.CONT	: INDICATE CONTROLLER ERROR RETURN
	RTS	PC	: RETURN

.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

5380
5381
5382
5383
5384
5385
5386
5387
5388
5389
5390
5391
5392
5393
5394
5395
5396
5397
5398
5399
400
401
402
403
404
405
406
407
408
409
410
411
412
413
414
415
416
417
418
419
420
421
422
423
424
425
426
427
428
429
430
431
432
433
434
435

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

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 #1<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

```

5436 036044 042765 075176 000014      BIC      #1C<DRVUSE!W.WCK!NOCHK!DRPDRV!DTBAII>,P.PRST(R5)
5437
5438 036052 010500                      MOV      R5,R0      ;STORE PARAMETER BLOCK ADDRESS
5439 036054 062700 000016      ADD      #P.CS1,R0   ;CALCULATE FIRST LOCATION TO BE CLEARED
5440 036060 010501                      MOV      R5,R1      ;STORE PARAMETER BLOCK ADDRESS
5441 036062 062701 000062      ADD      #P.EPAT,R1  ;CALCULATE LAST LOCATION TO BE CLEARED
5442
5443 036066 005020                      1$:     CLR      (R0)+     ;CLEAR RETURN PARAMETER
5444 036070 020001                      CMP      R0,R1      ;CHECK IF FINISHED
5445 036072 101775                      BLOS    1$          ;NO, CLEAR NEXT RETURN PARAMETER
5446 036074 105037 002626      CLR      I.ISRL     ;CLEAR RELEASE OR INTERRUPT ISSUED
5447 036100 010465 000020      MOV      R4,P.CS2(R5) ;STORE DRIVE NUMBER
5448 036104 005062 000026      CLR      RKMRI(R2)  ;CLEAR RK06 MAINTENANCE REGISTER 1
5449 036110 132765 000040 000001      BITB    #BITS,P.CMND(R5) ;CHECK IF SPECIAL COMMAND
5450 036116 001402                      BEQ     3$          ;NO, PROCESS
5451 036120 000137 036634                      JMP     C.SPEC      ;JUMP TO SPECIAL COMMAND PROCESSOR
5452
5453 036124 122765 000107 000001 3$:     CMPB    #UNLOAD,P.CMND(R5) ;CHECK IF POSITIONING COMMAND
5454                                     ;START SPINDLE
5455                                     ;RECALIBRATE
5456                                     ;OFFSET
5457                                     ;SEEK
5458                                     ;UNLOAD
5459
5460 036132 101174                      BHI     25$         ;NO, DRIVE COMMAND
5461                                     ;SELECT DRIVE
5462                                     ;PACK ACKNOWLEDGE
5463                                     ;CLEAR
5464
5465 036134 122765 000117 000001      CMPB    #SEEK,P.CMND(R5) ;CHECK IF DATA TRANSFER
5466 036142 103540                      BLO     20$         ;YES, DATA TRANSFER COMMAND
5467                                     ;READ DATA
5468                                     ;WRITE DATA
5469                                     ;READ HEADER
5470                                     ;WRITE HEADER
5471                                     ;WRITE CHECK
5472 036144 016562 000020 000010      MOV      P.CS2(R5),RKCS2(R2) ;LOAD DRIVE NUMBER
5473 036152 052765 000002 000014      BIS      #DRVPOS,P.PRST(R5) ;SET DRIVE POSITIONING
5474 036160 005037 002606                      CLR      0.WAIT     ;CLEAR WAIT FOR COMMAND
5475 036164 122765 000117 000001      CMPB    #SEEK,P.CMND(R5) ;CHECK IF SEEK
5476 036172 001007                      BNE     5$          ;NO, CHECK FOR OFFSET
5477 036174 016562 000002 000020      MOV      P.CYLN(R5),RKDCYL(R2) ;LOAD CYLINDER ADDRESS
5478 036202 016562 000004 000006      MOV      P.SECT(R5),RKDA(R2) ;LOAD SECTOR AND TRACK
5479 036210 000431                      BR      8$          ;GO ISSUE COMMAND
5480
5481 036212 122765 000115 000001 5$:     CMPB    #OFFSET,P.CMND(R5) ;CHECK IF OFFSET
5482 036220 001007                      BNE     6$          ;NO, CHECK FOR UNLOAD
5483 036222 116565 000006 000032      MOV      P.OFST(R5),P.ASOF(R5) ;STORE OFFSET
5484 036230 016562 000032 000016      MOV      P.ASOF(R5),RKASOF(R2) ;LOAD OFFSET REGISTER
5485 036236 000416                      BR      8$          ;GO ISSUE COMMAND
5486
5487 036240 122765 000111 000001 6$:     CMPB    #SRTSPL,P.CMND(R5) ;CHECK IF START SPINDLE
5488 036246 001003                      BNE     7$          ;NO, CHECK IF RECAL
5489 036250 013737 002620 002646      MOV      W.MIN,W.DRV ;LOAD WATCH DOG TIME FOR 1 MINUTE
5490 036256 122765 000113 000001 7$:     CMPB    #RECAL,P.CMND(R5) ;CHECK IF RECAL
5491 036264 001003                      BNE     8$          ;NO, CONTINUE
    
```

```

5492 036266 013737 002616 002646      MOV      W.8SEC,W.DRV      ;LOAD RECAL TIME FOR 8 SECONDS
5493 036274 116565 000007 000017 8$:      MOVVB   P.CS1H(R5),P.CS1+1(R5);STORE BITS 8-15 OF CS1
5494 036302 042765 165777 000016      BIC     #↑C<CFMT!CDT>,P.CS1(R5);CLEAR ALL BITS EXCEPT FORMAT
5495                                     ;AND DRIVE TYPE
5496 036310 116565 000001 000016      MOVVB   P.CMND(R5),P.CS1(R5);MOVE COMMAND INTO CS1
5497 036316 042765 000200 000014      BIC     #W.WCK,P.PRST(R5);RESET WRITE FOR WRITE CHECK
5498 036324 032765 000400 000014      BIT     #NOCHK,P.PRST(R5);CHECK IN NO CHECK MODE
5499 036332 001533 000100 000016      BEQ     30$              ;NO SKIP CLEAR OF INTERRUPT ENABLE
5500 036334 042765 000100 000016      BIC     #IE,P.CS1(R5);CLEAR INTERRUPT ENABLE
5501 036342 016562 000016 000000      MOV     P.CS1(R5),RKCS1(R2);ISSUE COMMAND
5502 036350 004737 032312 000000 10$:     JSR     PC.W.WTCH;CALL WATCH DOG TIMER
5503 036354 016237 000000 002532      MOV     RKCS1(R2),T.CS1;STORE COMMAND AND STATUS REGISTER 1
5504 036362 032737 000200 002532      BIT     #RDY,T.CS1;WAIT FOR READY
5505 036370 001767 100000 002532      BEQ     10$
5506 036372 032737 100000 002532      BIT     #CERR,T.CS1;CHECK FOR ERROR
5507 036400 001011 000000 000000      BNE     15$              ;YES, GIVE NORMAL RETURN
5508 036402 004737 032312 000000 11$:     JSR     PC.W.WTCH;CALL WATCH DOG TIMER
5509 036406 016237 000016 002546      MOV     RKASOF(R2),T.ASOF;STORE ATTENTION SUMMARY
5510 036414 133737 002633 002547      BITB   INTMSK,T.ASOF+1;CHECK IF INTERRUPT HAS OCCURRED
5511 036422 001767 000000 000000      BEQ     11$              ;WAIT FOR DRIVE INTERRUPT
5512 036424 105037 002632 000000 15$:     CLRB   W.TIME;RESET TIMING ON THIS DRIVE
5513 036430 005037 002646 000000      CLR    W.DRV;CLEAR DRIVE TIMING COUNT
5514 036434 004737 035706 000000      JSR     PC.R.NORM;INDICATE COMMAND IS FINISHED
5515 036440 000137 037614 000000      JMP     C.ATRN;RESTORE REGISTERS
5516
5517 036444 016562 000010 000004 20$:     MOV     P.BALO(R5),RKBA(R2);LOAD BUS ADDRESS REGISTER
5518 036452 016562 000012 000002      MOV     P.WC(R5),RKWC(R2);LOAD WORD COUNT REGISTER
5519 036460 016562 000002 000020      MOV     P.CYLN(R5),RKDCYL(R2);LOAD CYLINDER ADDRESS REGISTER
5520 036466 016562 000004 000006      MOV     P.SECT(R5),RKDA(R2);LOAD SECTOR AND TRACK NUMBER
5521 036474 122765 000131 000001      CMPB   #WRTCHK,P.CMND(R5);CHECK IF WRITE CHECK COMMAND
5522 036502 001010 000000 000000      BNE     25$              ;NO, GO ISSUE THE COMMAND
5523 036504 032765 000200 000014      BIT     #W.WCK,P.PRST(R5);CHECK IF WRITE COMMAND SHOULD BE ISSUED
5524 036512 001404 000000 000000      BEQ     25$              ;NO, GO ISSUE THE COMMAND
5525 036514 012765 000123 000016      MOV     #WRDATA,P.CS1(R5);ISSUE WRITE COMMAND
5526 036522 000406 000000 000000      BR     26$              ;GO ISSUE COMMAND
5527
5528 036524 116565 000001 000016 25$:     MOVVB   P.CMND(R5),P.CS1(R5);MOVE COMMAND INTO CS1
5529 036532 042765 000200 000014      BIC     #W.WCK,P.PRST(R5);RESET WRITE FOR WRITE CHECK
5530 036540 116565 000007 000017 26$:     MOVVB   P.CS1H(R5),P.CS1+1(R5);STORE BITS 8-15 OF CS1
5531 036546 142765 177750 000017      BICB   #↑C<B.CFMT!B.CDT!B.BAI6!B.BAI7>,P.CS1+1(R5);CLEAR ALL BITS EXCEPT
5532                                     ;FORMAT, DRIVE TYPE, AND BUS ADDRESS
5533                                     ;BITS 16-17
5534 036554 010537 002606 000016      MOV     R5,0.WAIT;LOAD WAITING FOR COMMAND
5535 036560 032765 100000 000014      BIT     #DTBAII,P.PRST(R5);CHECK IF INHIBIT BUS ADDRESS INCREMENT
5536 036566 001403 000000 000000      BEQ     27$              ;NO, LOAD CS2
5537 036570 052765 000020 000020      BIS     #BAI,P.CS2(R5);SET INHIBIT BUS ADDRESS INCREMENT
5538 036576 016562 000020 000010 27$:     MOV     P.CS2(R5),RKCS2(R2);LOAD CS2
5539 036604 032765 000400 000014      BIT     #NOCHK,P.PRST(R5);CHECK IN NO CHECK MODE
5540 036612 001403 000100 000016      BEQ     30$              ;NO SKIP CLEAR OF INTERRUPT ENABLE
5541 036614 042765 000100 000016      BIC     #IE,P.CS1(R5);CLEAR INTERRUPT ENABLE
5542 036622 016562 000016 000000 30$:     MOV     P.CS1(R5),RKCS1(R2);ISSUE COMMAND
5543 036630 000137 037614 000000      JMP     C.ATRN;RESTORE REGISTERS
5544
5545                                     .SBTTL  #SPECIAL COMMAND PROCESSING
5546
5547 036634 122765 000141 000001  C.SPEC: CMPB   #RDSTAT,P.CMND(R5);CHECK IF READ DRIVE STATUS
    
```

5548	036642	001132				BNE	10\$:NO, PROCESS OTHER COMMANDS
5549	036644	016562	000020	000010		MOV	P.CS2(R5),RKCS2(R2)	:LOAD CS2 FOR COMMAND
5550	036652	116565	000007	000017		MOVB	P.CS1H(R5),P.CS1+1(R5)	:STORE BITS 8-15 OF CS1
5551	036660	042765	165777	000016		BIC	#1C<CFMT!CDT>,P.CS1(R5)	:CLEAR ALL BITS EXCEPT FORMAT AND DRIVE TYPE
5552								
5553	036666	112765	000001	000016		MOVB	#DR,SEL,P.CS1(R5)	:STORE COMMAND
5554	036674	016562	000016	000000		MOV	P.CS1(R5),RKCS1(R2)	:ISSUE COMMAND
5555	036702	004737	032312		2\$:	JSR	PC.W.WTCH	:CALL WATCH-DOG TIMER
5556	036706	016265	000000	000016		MOV	RKCS1(R2),P.CS1(R5)	:STORE COMMAND AND STATUS REG. 1
5557	036714	032765	000200	000016		BIT	#RDY,P.CS1(R5)	:WAIT FOR READY
5558	036722	001767				BEQ	2\$	
5559	036724	004737	035364			JSR	PC,I.CST1	:STORE CONTROLLER REGISTERS
5560	036730	016265	000034	000040		MOV	RKMR2(R2),P.A00(R5)	:STORE STATUS BYTE 00 MESSAGE A
5561	036736	016265	000036	000042		MOV	RKMR3(R2),P.B00(R5)	:STORE STATUS BYTE 00 MESSAGE B
5562	036744	032765	100000	000016		BIT	#CERR,P.CS1(R5)	:CHECK IF CONTROLLER ERROR
5563	036752	001436				BEQ	6\$:NO, GATHER DRIVE STATUS
5564	036754	105037	002632			CLRB	W.TIME	:RESET WATCH-DOG TIMING ON THIS DRIVE
5565	036760	005037	002646			CLR	W.DRV	:CLEAR WATCH-DOG COUNT
5566	036764	032765	000400	000014		BIT	#NOCHK,P.PRST(R5)	:CHECK IF NO CHECK MODE
5567	036772	001043				BNE	8\$:YES, INDICATE NORMAL RETURN
5568	036774	032765	001000	000020		BIT	#MDS,P.CS2(R5)	:CHECK IF MULTIPLE DRIVE SELECT
5569	037002	001043				BNE	9\$:YES, INDICATE CONTROLLER ERROR
5570	037004	004037	034710			JSR	RD,I.CCLR	:CLEAR ERROR
5571	037010	037614				C.RTRN		:ERROR RETURN
5572	037012	032765	010400	000020		BIT	#NED!UFE,P.CS2(R5)	:CHECK IF NON-EXISTENT DRIVE OR UNIT FIELD ERROR
5573	037020	001007				BNE	5\$:REPORT ERROR
5574	037022	032765	000001	000036		BIT	#DRA,P.DS(R5)	:CHECK IF DRIVE AVAILIABLE
5575	037030	001003				BNE	5\$:YES, REPORT ERROR
5576	037032	052765	010000	000014		BIS	#DRVSZD,P.PRST(R5)	:INDICATE DRIVE IS SEIZED BY OTHER PORT
5577	037040	004737	035674		5\$:	JSR	PC.R.ABNL	:INDICATE ABNORMAL RETURN
5578	037044	000137	037614			JMP	C.ATRN	:RESTORE REGISTERS
5579								
5580	037050	004037	035446		6\$:	JSR	RD,I.STAT	:GATHER DRIVE STATUS
5581	037054	037614				C.RTRN		:ERROR RETURN
5582	037056	105037	002632			CLRB	W.TIME	:STOP WATCH-DOG TIMING ON DRIVE
5583	037062	005037	002646			CLR	W.DRV	:RESET WATCH-DOG TIME
5584	037066	032765	000400	000014		BIT	#NOCHK,P.PRST(R5)	:CHECK IF NO CHECK MODE
5585	037074	001402				BEQ	8\$:NO, REPORT ERROR
5586	037076	005062	000000			CLP	RKCS1(R2)	:CLEAR INTERRUPT ENABLE
5587	037102	004737	035706		8\$:	JSR	PC.R.NORM	:REPORT COMMAND COMPLETE
5588	037106	000137	037614			JMP	C.ATRN	:RESTORE REGISTERS
5589								
5590	037112	052737	100000	002604	9\$:	BIS	#E.MDS,E.CONT	:SET MULTIPLE DRIVE SELECT
5591	037120	004737	035720			JSR	PC.R.CONT	:INDICATE CONTROLLER ERROR
5592	037124	000137	037614			JMP	C.ATRN	
5593								
5594	037130	122765	000140	000001	10\$:	CMPB	#RELEAS,P.CMND(R5)	:CHECK IF RELEASE COMMAND
5595	037136	001040				BNE	13\$:NO, CHECK IF READ ALL HEADERS
5596	037140	010537	002606			MOV	RS,O.WAIT	:STORE PARAMETER BLOCK ADDRESS IN
5597								:WAIT FOR COMMAND
5598	037144	052765	000010	000020		BIS	#RLS,P.CS2(R5)	:SET RELEASE BIT
5599	037152	016562	000020	000010		MOV	P.CS2(R5),RKCS2(R2)	:LOAD CS2 FOR Deselect
5600	037160	112737	000001	002626		MOVB	#I,I.ISRL	:SET FLAG FOR RELEASE COMMAND
5601	037166	116565	000007	000017		MOVB	P.CS1H(R5),P.CS1+1(R5)	:STORE BITS 8-15 OF CS1
5602	037174	042765	165777	000016		BIC	#1C<CFMT!CDT>,P.CS1(R5)	:CLEAR ALL BITS EXCEPT FORMAT AND DRIVE TYPE
5603								

5604	037202	112765	000101	000016		MOV	#SELDIV,P.CS1(R5)	:STORE COMMAND
5605	037210	032765	000400	000014		BIT	#NOCHK,P.PRST(R5)	:CHECK IF NO CHECK MODE
5606	037216	001403				BEQ	11\$:NO, DO NOT RESET INTERRUPT ENABLE
5607	037220	042765	000100	000016		BIC	#IE,P.CS1(R5)	:RESET INTERRUPT ENABLE
5608	037226	016562	000016	000000	11\$:	MOV	P.CS1(R5),RKCS1(R2)	:ISSUE COMMAND
5609	037234	000137	037614			JMP	C.RTRN	:RESTORE REGISTERS
5610								
5611	037240	122765	000164	000001	13\$:	CMPB	#RDALHD,P.CMND(R5)	:CHECK IF READ ALL HEADERS
5612	037246	001053				BNE	30\$:NO, CHECK IF CONTROLLER CLEAR
5613	037250	010537	002606			MOV	RS,O.WAIT	:SET WAITING FOR COMMAND COMPLETION
5614	037254	016537	000010	002622		MOV	P.BALO(R5),HDR.AD	:LOAD HEADER ADDRESS
5615	037262	132765	000020	000007		BITB	#B.CFMT,P.CS1H(R5)	:CHECK IF 22 SECTOR FORMANT
5616	037270	001404				BEQ	14\$:YES, LOAD 22 IN HEADER COUNT
5617	037272	012737	000024	002624		MOV	#20.,HDR.CT	:LOAD 20 IN SECTOR COUNT
5618	037300	000403				BR	22\$:GO ISSUE READ HEADER COMMAND
5619								
5620	037302	012737	000026	002624	14\$:	MOV	#22.,HDR.CT	:LOAD 22 IN SECTOR COUNT
5621	037310	016562	000002	000020	22\$:	MOV	P.CYLN(R5),RKDCYL(R2)	:LOAD CYLINDER ADDRESS
5622	037316	016562	000004	000006		MOV	P.SECT(R5),RKDA(R2)	:LOAD TRACK NUMBER
5623	037324	016562	000020	000010		MOV	P.CS2(R5),RKCS2(R2)	:LOAD DRIVE NUMBER
5624	037332	116565	000007	000017		MOV	P.CS1H(R5),P.CS1+1(R5)	:STORE BITS 8-15 OF CS1
5625	037340	042765	165777	000016		BIC	#(C<CFMT!CDT),P.CS1(R5)	:CLEAR ALL BITS EXCEPT DRIVE TYPE AND FORMAT
5626								
5627	037346	112765	000125	000016		MOV	#RDHEAD,P.CS1(R5)	:STORE READ HEADER COMMAND
5628	037354	032765	000400	000014		BIT	#NOCHK,P.PRST(R5)	:CHECK IF NO CHECK MODE
5629	037362	001027				BNE	34\$:YES, INDICATE ILLEGAL DRIVER COMMAND
5630	037364	016562	000016	000000		MOV	P.CS1(R5),RKCS1(R2)	:ISSUE READ HEADER
5631	037372	000137	037614			JMP	C.RTRN	:RESTORE REGISTERS
5632								
5633	037376	122765	000176	000001	30\$:	CMPB	#CONCLR,P.CMND(R5)	:CHECK IF CONTROLLER CLEAR
5634	037404	001012				BNE	32\$:NO, CHECK IF SUBSYSTEM CLEAR
5635	037406	004037	034710			JSR	RD,I.CCLR	:CLEAR CONTROLLER
5636	037412	037614				C.RTRN		:ERROR RETURN
5637	037414	032765	000400	000014		BIT	#NOCHK,P.PRST(R5)	:CHECK IF NO CHECK MODE
5638	037422	001472				BEQ	40\$:NO, INDICATE NORMAL RETURN
5639	037424	005062	000000			CLR	RKCS1(R2)	:RESET INTERRUPT ENABLE
5640	037430	000467				BR	40\$:INDICATE NORMAL RETURN
5641								
5642	037432	122765	000177	000001	32\$:	CMPB	#SUBCLR,P.CMND(R5)	:CHECK IF SUBSYSTEM CLEAR
5643	037440	001406				BEQ	36\$:YES, CLEAR SUBSYSTEM
5644	037442	052737	000100	002604	34\$:	BIS	#E.ILLD,E.CONT	:SET ILLEGAL DRIVER COMMAND
5645	037450	004737	035720			JSR	PC.R.CONT	:REPORT ERROR
5646	037454	000457				BR	C.RTRN	:RESTORE REGISTERS
5647								
5648	037456	012762	000040	000010	36\$:	MOV	#SCLR,RKCS2(R2)	:ISSUE SUBSYSTEM CLEAR
5649	037464	016265	000000	000016		MOV	RKCS1(R2),P.CS1(R5)	:STORE COMMAND AND STATUS REGISTER 1
5650	037472	032765	100000	000016		BIT	#CERR,P.CS1(R5)	:CLEAR IF CONTROLLER ERROR RESET
5651	037500	001406				BEQ	37\$:NO, FINISH COMMAND
5652	037502	052737	000001	002604		BIS	#BITO,E.CONT	:SET CLEAR SUBSYSTEM DID NOT CLEAR
5653								:CONTROLLER ERROR
5654	037510	004737	035720			JSR	PC.R.CONT	:REPORT ERROR
5655	037514	000437				BR	C.RTRN	:RESTORE REGISTERS
5656								
5657	037516	013746	002612		37\$:	MOV	W.MILI,-(SP)	:LOAD 16 MILI-SECOND COUNT FOR ATTENTION
5658								:TO DISAPPEAR
5659	037522	016265	000000	000016	38\$:	MOV	RKCS1(R2),P.CS1(R5)	:STORE CS1

```

5660 037530 032765 040000 000016      BIT      #DI,P.CS1(R5)  ;CHECK IF ATTENTIONS CLEARED
5661 037536 001411                      BEQ      39$          ;YES, FINISH COMMAND
5662 037540 005316                      DEC      (SP)        ;DECREMENT 16 MILLISECOND COUNT
5663 037542 001367                      BNE     38$          ;CHECK DRIVE INTERRUPT AGAIN
5664 037544 005726                      TST     (SP)+        ;ADJUST STACK
5665 037546 052737 000040 002604      BIS     #E.SCLR,E.CONT ;SET SUBSYSTEM CLEAR DID NOT CLEAR
5666                                ;DRIVE ATTENTIONS
5667 037554 004737 035720                      JSR     PC.R.CONT   ;REPORT ERROR
5668 037560 000415                      BR      C.RTRN      ;RESTORE REGISTER
5669
5670 037562 005726                      39$:    TST     (SP)+        ;ADJUST STACK
5671 037564 032765 000400 000014      BIT     #NOCHK,P.PRST(R5) ;CHECK IF NO CHECK MODE
5672 037572 001010                      BNE     C.RTRN      ;YES, RESTORE REGISTERS
5673 037574 112737 177777 002626      MOV     #-1,I.ISRL  ;SET INTERRUPT ENABLE SET
5674 037602 012762 000100 000000      MOV     #IE,RKCS1(R2) ;SET INTERRUPT ENABLE
5675 037610 004737 035706                      40$:    JSR     PC,R.NORM ;INDICATE NORMAL TERMINATION
5676
5677 037614 012637 177776      C.RTRN: MOV     (SP)+,PS   ;RESTORE PSW
5678 037620 012600                      MOV     (SP)+,R0    ;RESTORE R0
5679 037622 012601                      MOV     (SP)+,R1    ;RESTORE R1
5680 037624 012602                      MOV     (SP)+,R2    ;RESTORE R2
5681 037626 012603                      MOV     (SP)+,R3    ;RESTORE R3
5682 037630 012604                      MOV     (SP)+,R4    ;RESTORE R4
5683 037632 012605                      MOV     (SP)+,R5    ;RESTORE R5
5684 037634 000207                      RTS     PC           ;RETURN
5685
5686                                .SBTTL  OCTAL TO BINARY CONVERSION ROUTINE
5687
5688                                ;*****
5689                                ;
5690                                ; THIS ROUTINE WILL CHECK A STRING OF ASCII CHARACTERS TERMINATED
5691                                ; WITH A NULL <000> OR COMMA. IF THE CHARACTERS ARE LEGAL
5692                                ; IT WILL GENERATE TWO BINARY WORDS PLACING THE LOW 16 BITS
5693                                ; ON THE STACK AND THE HIGH 16 BITS IN LOCATION $HI0CT.
5694                                ;
5695                                ;CALL
5696                                ;
5697                                ; MOV     <ADDRESS OF ASCII STRING>,-(SP)
5698                                ; JSR     PC,OCTBIN
5699                                ; <ADDRESS OF ERROR RETURN>
5700                                ; RETURN
5701                                ;*****
5702 037636 010046      OCTBIN: MOV     R0,-(SP)   ;SAVE R0
5703 037640 010146      MOV     R1,-(SP)   ;SAVE R1
5704 037642 010246      MOV     R2,-(SP)   ;SAVE R2
5705 037644 016600 000010      MOV     10(SP),R0  ;GET ADDRESS OF ASCII STRING
5706 037650 005001      CLR     R1         ;CLEAR DATA WORDS
5707 037652 005002      CLR     R2
5708 037654 112046      2$:    MOV     (R0)+,-(SP) ;PICK THIS CHARACTER
5709 037656 001423      BEQ     3$         ;IF ZERO GET OUT
5710 037660 121627 000054      CMP     (SP),#',   ;CHECK IF COMMA
5711 037664 001420      BEQ     3$         ;IF COMMA GET OUT
5712 037666 122716 000060      CMP     #'0,(SP)   ;MAKE SURE THIS CHARACTER IS
5713 037672 003030      BGT     4$         ; AN OCTAL DIGIT
5714 037674 122716 000067      CMP     #'7,(SP)
5715 037700 002425      BLT     4$

```

```

5716 037702 006301 ASL R1 ; *2
5717 037704 006102 ROL R2 ; *4
5718 037706 006301 ASL R1 ; *4
5719 037710 006102 ROL R2 ; *8
5720 037712 006301 ASL R1 ; *8
5721 037714 006102 ROL R2
5722 037716 042716 177770 BIC #1C7,(SP) ;STRIP THE ASCII JUNK
5723 037722 062601 ADD (SP)+,R1 ;ADD THIS DIGIT
5724 037724 000753 BR 2$ ;LOOP
5725 037726 005726 3$: TST (SP)+ ;CLEAN PARTIAL FROM STACK
5726 037730 010166 000010 MOV R1,10(SP) ;SAVE RESULT
5727 037734 010237 037770 MOV R2,$HIOCT
5728 037740 012602 MOV (SP)+,R2 ;RESTORE R2
5729 037742 012601 MOV (SP)+,R1 ;RESTORE R1
5730 037744 012600 MOV (SP)+,R0 ;RESTORE R0
5731 037746 062716 000002 ADD #2,(SP) ;ADJUST RETURN
5732 037752 000207 RTS PC ;RETURN
5733
5734 037754 005726 4$: TST (SP)+ ;CLEAN UP PARTIAL FROM STACK
5735 037756 012602 MOV (SP)+,R2 ;RESTORE R2
5736 037760 012601 MOV (SP)+,R1 ;RESTORE R1
5737 037762 012600 MOV (SP)+,R0 ;RESTORE R0
5738 037764 013616 MOV #2(SP)+,(SP) ;PUT ADDRESS OF ERROR ROUTINE ON STACK
5739 037766 000207 RTS PC ;GO PROCESS ERROR
5740 037770 000000 $HIOCT: WORD 0 ;HIGH ORDER BITS GO HERE
5741 .SBTTL TYPE ROUTINE
5742
5743 *****
5744 *ROUTINE TO TYPE ASCIZ MESSAGE. MESSAGE MUST TERMINATE WITH A 0 BYTE.
5745 *THE ROUTINE WILL INSERT A NUMBER OF NULL CHARACTERS AFTER A LINE FEED.
5746 *NOTE1: $NULL CONTAINS THE CHARACTER TO BE USED AS THE FILLER CHARACTER.
5747 *NOTE2: $FILLS CONTAINS THE NUMBER OF FILLER CHARACTERS REQUIRED.
5748 *NOTE3: $FILLC CONTAINS THE CHARACTER TO FILL AFTER.
5749 *
5750 *CALL:
5751 *1) USING A TRAP INSTRUCTION
5752 * TYPE ,MESADR ;;MESADR IS FIRST ADDRESS OF AN ASCIZ STRING
5753 *OR
5754 * TYPE
5755 * MESADR
5756 *
5757
5758 037772 105737 001157 $TYPE: TSTB $TPFLG ; IS THERE A TERMINAL?
5759 037776 100002 BPL 1$ ; BR IF YES
5760 040000 000000 HALT ; HALT HERE IF NO TERMINAL
5761 040002 000407 BR 3$ ; LEAVE
5762 040004 010046 1$: MOV R0,-(SP) ; SAVE R0
5763 040006 017600 000002 MOV #2(SP),R0 ; GET ADDRESS OF ASCIZ STRING
5764 040012 112046 2$: MOVB (R0)+,-(SP) ; PUSH CHARACTER TO BE TYPED ONTO STACK
5765 040014 001005 BNE 4$ ; BR IF IT ISN'T THE TERMINATOR
5766 040016 005726 TST (SP)+ ; IF TERMINATOR POP IT OFF THE STACK
5767 040020 012600 60$: MOV (SP)+,R0 ; RESTORE R0
5768 040022 062716 3$: ADD #2,(SP) ; ADJUST RETURN PC
5769 040026 000002 RTI ; RETURN
5770 040030 122716 000011 4$: CMPB #HT,(SP) ; BRANCH IF <HT>
5771 040034 001430 BEQ 8$
    
```

```

5772 040036 122716 000200      CMPB   #CRLF,(SP)      ;;BRANCH IF NOT <CRLF>
5773 040042 001006      BNE    5$              ;;
5774 040044 005726      TST    (SP)+          ;;POP <CR><LF> EQUIV
5775 040046 104401      TYPE   ;;TYPE A CR AND LF
5776 040050 001267      $CRLF
5777 040052 105037 040206      CLRB   $CHARCNT      ;;CLEAR CHARACTER COUNT
5778 040056 000755      BR     2$              ;;GET NEXT CHARACTER
5779 040060 004737 040142      5$:   JSR   PC,$TYPEC  ;;GO TYPE THIS CHARACTER
5780 040064 123726 001156      6$:   CMPB  $FILLC,(SP)+ ;;IS IT TIME FOR FILLER CHARS.?
5781 040070 001350      BNE    2$              ;;IF NO GO GET NEXT CHAR.
5782 040072 013746 001154      MOV    $NULL,-(SP)    ;;GET # OF FILLER CHARS. NEEDED
5783                                     ;;AND THE NULL CHAR.
5784 040076 105366 000001      7$:   DECB  1(SP)      ;;DOES A NULL NEED TO BE TYPED?
5785 040102 002770      BLT    6$              ;;BR IF NO--GO POP THE NULL OFF OF STACK
5786 040104 004737 040142      JSR   PC,$TYPEC      ;;GO TYPE A NULL
5787 040110 105337 040206      DECB  $CHARCNT      ;;DO NOT COUNT AS A COUNT
5788 040114 000770      BR     7$              ;;LOOP
5789
5790                                     ;HORIZONTAL TAB PROCESSOR
5791
5792 040116 112716 000040      8$:   MOVB  #' (SP)      ;;REPLACE TAB WITH SPACE
5793 040122 004737 040142      9$:   JSR   PC,$TYPEC  ;;TYPE A SPACE
5794 040126 132737 000007 040206      BITB  #7,$CHARCNT    ;;BRANCH IF NOT AT
5795 040134 001372      BNE    9$              ;;TAB STOP
5796 040136 005726      TST    (SP)+          ;;POP SPACE OFF STACK
5797 040140 000724      BR     2$              ;;GET NEXT CHARACTER
5798 040142 105777 141002      $TYPEC: TSTB  $STPS      ;;WAIT UNTIL PRINTER IS READY
5799 040146 100375      BPL   $TYPEC
5800 040150 116677 000002 140774      MOVB  2(SP), $STPB    ;;LOAD CHAR TO BE TYPED INTO DATA REG.
5801 040156 122766 000015 000002      CMPB  #CR,2(SP)      ;;IS CHARACTER A CARRIAGE RETURN?
5802 040164 001003      BNE    1$              ;;BRANCH IF NO
5803 040166 105037 040206      CLRB  $CHARCNT      ;;YES--CLEAR CHARACTER COUNT
5804 040172 000406      BR     $TYPEX        ;;EXIT
5805 040174 122766 000012 000002 1$:   CMPB  #LF,2(SP)    ;;IS CHARACTER A LINE FEED?
5806 040202 001402      BEQ   $TYPEX        ;;BRANCH IF YES
5807 040204 105227      INCB  (PC)+          ;;COUNT THE CHARACTER
5808 040206 000000      $CHARCNT: .WORD 0    ;;CHARACTER COUNT STORAGE
5809 040210 000207      $TYPEX: RTS          PC
5810
5811                                     .SBTTL  DOUBLE LENGTH BINARY TO OCTAL ASCII CONVERT ROUTINE
5812
5813                                     ;*****
5814                                     ;*THIS ROUTINE WILL CONVERT A 32-BIT UNSIGNED BINARY NUMBER TO AN
5815                                     ;*UNSIGNED OCTAL ASCII NUMBER.
5816                                     ;*CALL
5817                                     ;*
5818                                     ;*   MOV    #PNTR,-(SP)      ;; POINTER TO LOW WORD OF BINARY NUMBER
5819                                     ;*   JSR   PC,$SDB20      ;; CALL THE ROUTINE
5820                                     ;*   RETURN                ;; THE ADDRESS OF THE FIRST ASCII CHAR. IS ON THE STACK
5821
5822 $SDB20: SAVREG      ;;SAVE ALL REGISTERS
5823        MOV    2(SP),R1    ;;PICKUP THE POINTER TO LOW WORD
5824        MOV    #SOCTVL+13.,R5 ;; POINTER TO DATA TABLE
5825        MOV    #12.,R4     ;;DO ELEVEN CHARACTERS
5826        MOV    #1C7,R3     ;;MASK
5827        MOV    (R1)+,R0    ;;LOWER WORD

```


DOUBLE LENGTH BINARY TO OCTAL ASCII CONVERT ROUTINE

```

5828 040236 012101      MOV      (R1)+,R1      ;; HIGH WORD
5829 040240 005002      CLR      R2           ;; TERMINATOR
5830 040242 110245      1$:     MOVVB     R2,-(R5)  ;; PUT CHARACTER IN DATA TABLE
5831 040244 010002      MOV      R0,R2       ;; GET THIS DIGIT
5832 040246 005304      DEC      R4          ;; COUNT THIS CHARACTER
5833 040250 003007      BGT      3$         ;; BR IF NOT THE LAST DIGIT
5834 040252 001405      BEQ      2$         ;; BR IF IT IS THE LAST DIGIT
5835 040254 005205      INC      R5          ;; ALL DIGITS DONE-ADJUST POINTER FOR FIRST
5836 040256 010566 000002  MOV      R5,2(SP)    ;; ASCII CHAR. & PUT IT ON THE STACK
5837 040262 104412      RESREG                ;; RESTORE ALL REGISTERS
5838 040264 000207      RTS      PC          ;; RETURN TO USER
5839 040266 006203      2$:     ASR      R3          ;; POSITION THE MASK FOR THE LAST DIGIT
5840 040270 006001      3$:     ROR      R1          ;; POSITION THE BINARY NUMBER FOR
5841 040272 006000      ROR      R0          ;; THE NEXT OCTAL DIGIT
5842 040274 006001      ROR      R1
5843 040276 006000      ROR      R0
5844 040300 006001      ROR      R1
5845 040302 006000      ROR      R0
5846 040304 040302      BIC      R3,R2       ;; MASK OUT ALL JUNK
5847 040306 062702 000060  ADD      #0,R2       ;; MAKE THIS CHAR. ASCII
5848 040312 000753      BR       1$         ;; GO PUT IT IN THE DATA TABLE
5849 040314 000016      $OCTVL: .BLKB     14.  ;; RESERVE DATA TABLE
5850      .SBTTL  BINARY TO OCTAL (ASCII) AND TYPE

```

```

5851
5852 *****
5853 *THIS ROUTINE IS USED TO CHANGE A 16-BIT BINARY NUMBER TO A 6-DIGIT
5854 *OCTAL (ASCII) NUMBER AND TYPE IT.
5855 *$TYPOS---ENTER HERE TO SETUP SUPPRESS ZEROS AND NUMBER OF DIGITS TO TYPE
5856 *CALL:
5857 *      MOV      NUM,-(SP)      ;; NUMBER TO BE TYPED
5858 *      TYPOS                ;; CALL FOR TYPEOUT
5859 *      .BYTE   N              ;; N=1 TO 6 FOR NUMBER OF DIGITS TO TYPE
5860 *      .BYTE   M              ;; M=1 OR 0
5861 *                               ;; 1=TYPE LEADING ZEROS
5862 *                               ;; 0=SUPPRESS LEADING ZEROS
5863
5864 *$STYPON---ENTER HERE TO TYPE OUT WITH THE SAME PARAMETERS AS THE LAST
5865 *$TYPOS OR $TYPOC
5866 *CALL:
5867 *      MOV      NUM,-(SP)      ;; NUMBER TO BE TYPED
5868 *      TYPON                ;; CALL FOR TYPEOUT
5869
5870 *$TYPOC---ENTER HERE FOR TYPEOUT OF A 16 BIT NUMBER
5871 *CALL:
5872 *      MOV      NUM,-(SP)      ;; NUMBER TO BE TYPED
5873 *      TYPOC                ;; CALL FOR TYPEOUT
5874
5875 040332 017646 000000      $TYPOS: MOV      2(SP),-(SP)  ;; PICKUP THE MODE
5876 040336 116637 000001 040555  MOVVB   1(SP),%OFILL  ;; LOAD ZERO FILL SWITCH
5877 040344 112637 040557      MOVVB   (SP)+,%SOMODE+1  ;; NUMBER OF DIGITS TO TYPE
5878 040350 062716 000002      ADD     #2,(SP)        ;; ADJUST RETURN ADDRESS
5879 040354 000406      BR      $TYPON
5880 040356 112737 000001 040555  $TYPOC: MOVVB   #1,%OFILL  ;; SET THE ZERO FILL SWITCH
5881 040364 112737 000006 040557      MOVVB   #6,%SOMODE+1  ;; SET FOR SIX(6) DIGITS
5882 040372 112737 000005 040554  $TYPON: MOVVB   #5,%SOCNT  ;; SET THE ITERATION COUNT
5883 040400 010346      MOV     R3,-(SP)      ;; SAVE R3

```

```

5884 040402 010446      MOV      R4,-(SP)      ;;SAVE R4
5885 040404 010546      MOV      R5,-(SP)      ;;SAVE R5
5886 040406 113704 040557  MOVB     $OMODE+1,R4  ;;GET THE NUMBER OF DIGITS TO TYPE
5887 040412 005404      NEG      R4
5888 040414 062704 000006  ADD      #6,R4        ;;SUBTRACT IT FOR MAX. ALLOWED
5889 040420 110437 040556  MOVB     R4,$OMODE    ;;SAVE IT FOR USE
5890 040424 113704 040555  MOVB     $OFILL,R4    ;;GET THE ZERO FILL SWITCH
5891 040430 016605 000012  MOV      12(SP),R5    ;;PICKUP THE INPUT NUMBER
5892 040434 005003      CLR      R3          ;;CLEAR THE OUTPUT WORD
5893 040436 006105      1$:     ROL      R5    ;;ROTATE MSB INTO "C"
5894 040440 000404      BR      3$          ;;GO DO MSB
5895 040442 006105      2$:     ROL      R5    ;;FORM THIS DIGIT
5896 040444 006105      ROL      R5
5897 040446 006105      ROL      R5
5898 040450 010503      MOV      R5,R3
5899 040452 006103      3$:     ROL      R3    ;;GET LSB OF THIS DIGIT
5900 040454 105337 040556  DECB     $OMODE      ;;TYPE THIS DIGIT?
5901 040460 100016      BPL     7$          ;;BR IF NO
5902 040462 042703 177770  BIC     #177770,R3   ;;GET RID OF JUNK
5903 040466 001002      BNE     4$          ;;TEST FOR 0
5904 040470 005704      TST     R4          ;;SUPPRESS THIS 0?
5905 040472 001403      BEQ     5$          ;;BR IF YES
5906 040474 005204      4$:     INC      R4    ;;DON'T SUPPRESS ANYMORE 0'S
5907 040476 052703 000060  BIS     #'0,R3      ;;MAKE THIS DIGIT ASCII
5908 040502 052703 000040  BIS     #' ,R3      ;;MAKE ASCII IF NOT ALREADY
5909 040506 110337 040552  MOVB     R3,$S      ;;SAVE FOR TYPING
5910 040512 104401 040552  TYPE     #S        ;;GO TYPE THIS DIGIT
5911 040516 105337 040554  7$:     DECB     $OCNT  ;;COUNT BY 1
5912 040522 003347      BGT     2$          ;;BR IF MORE TO DO
5913 040524 002402      BLT     6$          ;;BR IF DONE
5914 040526 005204      INC     R4          ;;INSURE LAST DIGIT ISN'T A BLANK
5915 040530 000744      BR      2$          ;;GO DO THE LAST DIGIT
5916 040532 012605      6$:     MOV      (S0)+,R5  ;;RESTORE R5
5917 040534 012604      MOV      (SP)+,R4    ;;RESTORE R4
5918 040536 012603      MOV      (SP)+,R3    ;;RESTORE R3
5919 040540 016666 000002 000004  MOV      2(SP),4(SP) ;;SET THE STACK FOR RETURNING
5920 040546 012616      MOV      (SP)+,(SP)
5921 040550 000002      RTI
5922 040552 000      8$:     .BYTE 0      ;;RETURN
5923 040553 000      .BYTE 0      ;;STORAGE FOR ASCII DIGIT
5924 040554 000      .BYTE 0      ;;TERMINATOR FOR TYPE ROUTINE
5925 040555 000      $OCNT: .BYTE 0    ;;OCTAL DIGIT COUNTER
5926 040556 000000  $OFILL: .BYTE 0    ;;ZERO FILL SWITCH
5927      $OMODE: .WORD 0   ;;NUMBER OF DIGITS TO TYPE
5928      .SBTTL ERROR HANDLER ROUTINE
5929
5930      ;;*****
5931      ;;THIS ROUTINE WILL INCREMENT THE ERROR FLAG AND THE ERROR COUNT,
5932      ;;SAVE THE ERROR ITEM NUMBER AND THE ADDRESS OF THE ERROR CALL
5933      ;;AND GO TO TYPERR ON ERROR
5934      ;;THE SWITCH OPTIONS PROVIDED BY THIS ROUTINE ARE:
5935      ;;SW15=1 HALT ON ERROR
5936      ;;SW13=1 INHIBIT ERROR TYPEOUTS
5937      ;;SW10=1 BELL ON ERROR
5938      ;;SW09=1 LOOP ON ERROR
5939      ;;CALL      ERROR      N      ;;ERROR=EMT AND N=ERROR ITEM NUMBER

```

```

5940
5941 040560
5942 040560 104406
5943 040562 105237 001103
5944 040566 001775
5945 040570 013777 001102 140344
5946 040576 032777 002000 140334
5947 040604 001402
5948 040606 104401 001262
5949 040612 005237 001112
5950 040616 011637 001116
5951 040622 162737 000002 001116
5952 040630 117737 140262 001114
5953 040636 032777 020000 140274
5954 040644 001004
5955 040646 004737 025604
5956 040652 104401 001267
5957 040656
5958 040656 005777 140256
5959 040662 100002
5960 040664 000000
5961 040666 104406
5962 040670 032777 001000 140242
5963 040676 001402
5964 040700 013716 001110
5965 040704 005737 001260
5966 040710 001402
5967 040712 013716 001260
5968 040716
5969 040716 000002
5970
5971
5972
5973
5974 040720 000000
5975 040722 000000
5976 040724 000000
5977 040726 000001
5978 040727
5979 040730
5980
5981
5982
5983
5984
5985
5986
5987
5988
5989 040730 005037 040720
5990 040734 012737 040726 040722
5991 040742 013737 040722 040724
5992 040750 012737 041000 000060
5993 040756 012737 000200 000062
5994 040764 005777 140156
5995 040770 012777 000100 140146

ERROR:
7S: CKSWR          ;; TEST FOR CHANGE IN SOFT-SWR
      INCB          SERFLG          ;; SET THE ERROR FLAG
      BEQ          7S          ;; DON'T LET THE FLAG GO TO ZERO
      MOV          $STNM, @DISPLAY  ;; DISPLAY TEST NUMBER AND ERROR FLAG
      BIT          @BIT10, @SWR    ;; BELL ON ERROR?
      BEQ          1S          ;; NO - SKIP
      TYPE        $BELL          ;; RING BELL
1S:   INC          $ERTTL         ;; COUNT THE NUMBER OF ERRORS
      MOV          (SP), $ERRPC    ;; GET ADDRESS OF ERROR INSTRUCTION
      SUB          @2, $ERRPC      ;; STRIP AND SAVE THE ERROR ITEM CODE
      MOVB        @SERPC, $ITEMB   ;; SKIP TYPEOUT IF SET
      BIT          @BIT13, @SWR    ;; SKIP TYPEOUTS
      BNE          20S           ;; GO TO USER ERROR ROUTINE
      JSR          PC, TYPERR
      TYPE        , $CRLF
20S:
2S:   TST          @SWR          ;; HALT ON ERROR
      BPL          3S          ;; SKIP IF CONTINUE
      HALT        ;; HALT ON ERROR!
3S:   BIT          @BIT09, @SWR   ;; TEST FOR CHANGE IN SOFT-SWR
      BEQ          4S          ;; LOOP ON ERROR SWITCH SET?
      MOV          $LPERR, (SP)    ;; BR IF NO
      TST          $ESCAPE        ;; FUDGE RETURN FOR LOOPING
      BEQ          5S          ;; CHECK FOR AN ESCAPE ADDRESS
      MOV          $ESCAPE, (SP)  ;; BR IF NONE
      ;; FUDGE RETURN ADDRESS FOR ESCAPE
5S:   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=.
.EVEN

; *TK INITIALIZE ROUTINE
; *THIS ROUTINE WILL INITIALIZE THE TTY KEYBOARD INPUT QUEUE
; *SETUP THE INTERRUPT VECTOR AND TURN ON THE KEYBOARD INTERRUPT
; *CALL:
; *   JSR      PC, $TKINT
; *   RETURN
$TKINT: CLR          $TKCNT          ;; CLEAR COUNT OF ITEMS IN QUEUE
        MOV          @TKQSRT, $TKQIN ;; MOVE THE STARTING ADDRESS OF THE
        MOV          $TKQIN, $TKQOUT ;; QUEUE INTO THE INPUT & OUTPUT POINTERS.
        MOV          @TKSRT, @TKVEC  ;; INITIALIZE THE KEYBOARD VECTOR
        MOV          @200, @TKVEC+2  ;; "BR" LEVEL 4
        TST          @TKB           ;; CLEAR DONE FLAG
        MOV          @100, @TKS     ;; ENABLE TTY KEYBOARD INTERRUPT

```

```

5996 040776 000207          RTS      PC          ;: RETURN TO CALLER
5997
5998          ;*TK SERVICE ROUTINE
5999          ;*THIS ROUTINE WILL SERVICE THE TTY KEYBOARD INTERRUPT
6000          ;*BY READING THE CHARACTER FROM THE INPUT BUFFER AND PUTTING
6001          ;*IT IN THE QUEUE.
6002          ;*IF THE CHARACTER IS A "CONTROL-C" (↑C) STKINT IS CALLED AND
6003          ;*UPON RETURN EXIT IS MADE TO THE "CONTROL-C" RESTART ADDRESS (START)
6004
6005 041000 117746 140142  $TKSRV: MOVB   @STKB, -(SP)      ;: PICKUP THE CHARACTER
6006 041004 042716 177600      BIC     @↑C177, (SP)      ;: STRIP THE JUNK
6007 041010 021627 000003      CMP     (SP), #3        ;: IS IT A CONTROL C?
6008 041014 001007          BNE     1$              ;: BRANCH IF NO
6009 041016 104401 042176      TYPE   %CNTLC          ;: TYPE A CONTROL-C (↑C)
6010 041022 004737 040730      JSR    PC, STKINT      ;: INIT THE KEYBOARD
6011 041026 005726          TST    (SP)+           ;: CLEAN UP STACK
6012 041030 000137 016202      JMP    START           ;: CONTROL C RESTART
6013 041034 021627 000007  1$:    CMP     (SP), #7        ;: IS IT A CONTROL G?
6014 041040 001004          BNE     2$              ;: BRANCH IF NO
6015 041042 022737 000176 001140  CMP     @SWREG, SWR     ;: IS SOFT-SWR SELECTED?
6016 041050 001500          BEQ     6$              ;: GO TO SWR CHANGE
6017
6018 041052          2$:
6019 041052 022737 000001 040720  CMP     #1, STKCNT     ;: IS THE QUEUE FULL?
6020 041060 001004          BNE     3$              ;: BRANCH IF NO
6021 041062 104401 001262      TYPE   %BELL           ;: RING THE TTY BELL
6022 041066 005726          TST    (SP)+           ;: CLEAN CHARACTER OFF OF STACK
6023 041070 000451          BR     5$              ;: EXIT
6024 041072 021627 000023  3$:    CMP     (SP), #23      ;: IS IT A CONTROL-S?
6025 041076 001021          BNE     32$             ;: BRANCH IF NO
6026 041100 005077 140040      CLR    @STKS           ;: DISABLE TTY KEYBOARD INTERRUPTS
6027 041104 005726          TST    (SP)+           ;: CLEAN CHAR OFF STACK
6028 041106 105777 140032  31$:   TSTB   @STKS           ;: WAIT FOR A CHAR
6029 041112 100375          BPL    31$             ;: LOOP UNTIL ITS THERE
6030 041114 117746 140026      MOVB  @STKB, -(SP)     ;: GET THE CHARACTER
6031 041120 042716 177600      BIC   @↑C177, (SP)     ;: MAKE IT 7-BIT ASCII
6032 041124 022627 000021      CMP   (SP)+, #21      ;: IS IT A CONTROL-Q?
6033 041130 001366          BNE   31$              ;: BRANCH IF NO
6034 041132 012777 000100 140004  MOV    #100, @STKS     ;: REENABLE TTY KEYBOARD INTERRUPTS
6035 041140 000002          RTI                    ;: RETURN
6036 041142 005237 040720  32$:   INC    STKCNT          ;: COUNT THIS CHARACTER
6037 041146 021627 000140      CMP   (SP), #140      ;: IS IT UPPER CASE?
6038 041152 002405          BLT   4$              ;: BRANCH IF YES
6039 041154 021627 000175      CMP   (SP), #175      ;: IS IT A SPECIAL CHAR?
6040 041160 003002          BGT   4$              ;: BRANCH IF YES
6041 041162 042716 000040      BIC   #40, (SP)       ;: MAKE IT UPPER CASE
6042 041166 112677 177530  4$:   MOVB  (SP)+, @STKQIN  ;: AND PUT IT IN QUEUE
6043 041172 005237 040722      INC   STKQIN           ;: UPDATE THE POINTER
6044 041176 023727 040722 040727  CMP   STKQIN, @STKQEND ;: GO OFF THE END?
6045 041204 001003          BNE   5$              ;: BRANCH IF NO
6046 041206 012737 040726 040722  MOV    @STKQSR, STKQIN ;: RESET THE POINTER
6047 041214 000002  5$:   RTI                    ;: RETURN
6048
6049          ;: *****
6050          ;: *SOFTWARE SWITCH REGISTER CHANGE ROUTINE.
6051          ;: *ROUTINE IS ENTERED FROM THE TRAP HANDLER, AND WILL

```

```

6052 ;*SERVICE THE TEST FOR CHANGE IN SOFTWARE SWITCH REGISTER TRAP
6053 ;*CALL WHEN OPERATING IN TTY INTERRUPT MODE.
6054 041216 022737 000176 001140 $CKSWR: CMP      #SWREG,SWR      ;; IS THE SOFT-SWR SELECTED
6055 041224 001124                BNE      15$          ;; EXIT IF NOT
6056 041226 105777 137712        TSTB     @STKS        ;; IS A CHAR WAITING?
6057 041232 100121                BPL      15$          ;; IF NOT, EXIT
6058 041234 117746 137706        MOVVB    @STKB,-(SP)   ;; YES
6059 041240 042716 177600        BIC      #1C177,(SP) ;; MAKE IT 7-BIT ASCII
6060 041244 021627 000007        CMP      (SP),#7     ;; IS IT A CONTROL-G?
6061 041250 001300                BNE      2$          ;; IF NOT, PUT IT IN THE TTY QUEUE
6062                                ;; AND EXIT
6063
6064 ;*****
6065 ;*CONTROL IS PASSED TO THIS POINT FROM EITHER THE TTY INTERRUPT SERVICE
6066 ;*ROUTINE OR FROM THE SOFTWARE SWITCH REGISTER TRAP CALL, AS A RESULT OF A
6067 ;*CONTROL-G BEING TYPED, AND THE SOFTWARE SWITCH REGISTER BEING SELECTED.
6068 041252 123727 001134 000001 6$:  CMPB     $AUTOB,#1    ;; ARE WE RUNNING IN AUTO-MODE?
6069 041260 001674                BEQ      2$          ;; BRANCH IF YES
6070 041262 005726                TST      (SP)+       ;; CLEAR CONTROL-G OFF STACK
6071 041264 004737 040730        JSR      PC,STKINT   ;; FLUSH THE TTY INPUT QUEUE
6072 041270 005077 137650        CLR      @STKS      ;; DISABLE TTY KEYBOARD INTERRUPTS
6073 041274 112737 000001 001135        MOVVB    #1,$INTAG   ;; SET INTERRUPT MODE INDICATOR
6074
6075 041302 104401 042210        SGTSWR: TYPE     ,SCNTLG ;; ECHO THE CONTROL-G (↑G)
6076 041306 104401 042215        TYPE     $MSWR      ;; TYPE CURRENT CONTENTS
6077 041312 013746 000176        MOV      $WREG,-(SP) ;; SAVE SWREG FOR TYPEOUT
6078 041316 104402                TYPOC     ;; GO TYPE--OCTAL ASCII(ALL DIGITS)
6079 041320 104401 042226        TYPE     ,SMNEW     ;; PROMPT FOR NEW SWR
6080 041324 005046                CLR      -(SP)      ;; CLEAR COUNTER
6081 041326 005046                CLR      -(SP)      ;; THE NEW SWR
6082 041330 105777 137610        7$:  TSTB     @STKS      ;; CHAR THERE?
6083 041334 100375                BPL      7$          ;; IF NOT TRY AGAIN
6084
6085 041336 117746 137604        MOVVB    @STKB,-(SP) ;; PICK UP CHAR
6086 041342 042716 177600        BIC      #1C177,(SP) ;; MAKE IT 7-BIT ASCII
6087
6088 041346 021627 000003        CMP      (SP),#3     ;; IS IT A CONTROL-C?
6089 041352 001015                BNE      9$          ;; BRANCH IF NOT
6090 041354 104401 042176        TYPE     ,SCNTLC    ;; YES, ECHO CONTROL-C (↑C)
6091 041360 062706 000006        ADD      #6,SP      ;; CLEAN UP STACK
6092 041364 123727 001135 000001 000001 000001        CMPB     $INTAG,#1  ;; REENABLE TTY KEYBOARD INTERRUPTS?
6093 041372 001003                BNE      8$          ;; BRANCH IF NO
6094 041374 012777 000100 137542        MOV      #100,@STKS ;; ALLOW TTY KEYBOARD INTERRUPTS
6095 041402 000137 016202        8$:  JMP      START     ;; CONTROL-C RESTART
6096
6097
6098 041406 021627 000025        9$:  CMP      (SP),#25   ;; IS IT A CONTROL-U?
6099 041412 001005                BNE      10$         ;; BRANCH IF NOT
6100 041414 104401 042203        TYPE     ,SCNTLU    ;; YES, ECHO CONTROL-U (↑U)
6101 041420 062706 000006        20$:  ADD      #6,SP    ;; IGNORE PREVIOUS INPUT
6102 041424 000737                BR       19$         ;; LET'S TRY IT AGAIN
6103
6104
6105 041426 021627 000015        10$:  CMP      (SP),#15   ;; IS IT A <CR>?
6106 041432 001022                BNE      16$         ;; BRANCH IF NO
6107 041434 005766 000004        TST     4(SP)       ;; YES, IS IT THE FIRST CHAR?

```

6108	041440	001403			BEQ	11\$:: BRANCH IF YES
6109	041442	016677	000002	137470	MOV	2(SP), 2SWR	:: SAVE NEW SWR
6110	041450	062706	000006		ADD	#6, SP	:: CLEAR UP STACK
6111	041454	104401	001267		11\$: TYPE	\$CRLF	:: ECHO <CR> AND <LF>
6112	041460	123727	001135	000001	14\$: CMPB	\$INTAG, #1	:: RE-ENABLE TTY KBD INTERRUPTS?
6113	041466	001003			BNE	15\$:: BRANCH IF NOT
6114	041470	012777	000100	137446	MOV	#10G, 2\$TKS	:: RE-ENABLE TTY KBD INTERRUPTS
6115	041476	000002			15\$: RTI		:: RETURN
6116	041500	004737	040142		16\$: JSR	PC, \$TYPEPC	:: ECHO CHAR
6117	041504	021627	000060		CMP	(SP), #60	:: CHAR < 0?
6118	041510	002420			BLT	18\$:: BRANCH IF YES
6119	041512	021627	000067		CMP	(SP), #67	:: CHAR > 7?
6120	041516	003015			BGT	18\$:: BRANCH IF YES
6121	041520	042726	000060		BIC	#60 (SP)+	:: STRIP-OFF ASCII
6122	041524	005766	000002		TST	2(SP)	:: IS THIS THE FIRST CHAR
6123	041530	001403			BEQ	17\$:: BRANCH IF YES
6124	041532	006316			ASL	(SP)	:: NO, SHIFT PRESENT
6125	041534	006316			ASL	(SP)	:: CHAR OVER TO MAKE
6126	041536	006316			ASL	(SP)	:: ROOM FOR NEW ONE.
6127	041540	005266	000002		17\$: INC	2(SP)	:: KEEP COUNT OF CHAR
6128	041544	056616	177776		BIS	-2(SP), (SP)	:: SET IN NEW CHAR
6129	041550	000667			BR	7\$:: GET THE NEXT ONE
6130	041552	104401	001266		18\$: TYPE	\$QUES	:: TYPE ?<CR><LF>
6131	041556	000720			BR	20\$:: SIMULATE CONTROL-U
6132					.DSABL	LSB	

```

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

```

6143	041560	011646			\$RDCHR: MOV	(SP), -(SP)	:: PUSH DOWN THE PC AND
6144	041562	016666	000004	000002	MOV	4(SP), 2(SP)	:: THE PS
6145	041570	005066	000004		CLR	4(SP)	:: GET READY FOR A CHARACTER
6146	041574	005046			CLR	-(SP)	:: PUT NEW PS ON STACK
6147	041576	012746	041604		MOV	#64\$, -(SP)	:: PUT NEW PC ON STACK
6148	041602	000002			RTI		:: POP NEW PC AND PS
6149	041604				64\$: TST	\$TKCNT	:: WAIT ON A CHARACTER
6150	041604	005737	040720		1\$: BEQ	1\$	
6151	041610	001775			DEC	\$TKCNT	:: DECREMENT THE COUNTER
6152	041612	005337	040720		MOVB	2\$TKQOUT, 4(SP)	:: GET ONE CHARACTER
6153	041616	117766	177102	000004	INC	\$TKQOUT	:: UPDATE THE POINTER
6154	041624	005237	040724		CMP	\$TKQOUT, #2\$TKQEND	:: DID IT GO OFF OF THE END?
6155	041630	023727	040724	040727	BNE	2\$:: BRANCH IF NO
6156	041636	001003			MOV	#2\$TKQSR, \$TKQOUT	:: RESET THE POINTER
6157	041640	012737	040726	040724	RTI		:: RETURN
6158	041646	000002			2\$: RTI		

```

*****
*THIS ROUTINE WILL INPUT A STRING FROM THE TTY
*CALL:
*      RDLIN         ;; INPUT A STRING FROM THE TTY
*      RETURN HERE  ;; ADDRESS OF FIRST CHARACTER WILL BE ON THE STACK

```

6163

```

6164 ;* ; ; TERMINATOR WILL BE A BYTE OF ALL 0'S
6165
6166 041650 010346 SRDLIN: MOV R3, -(SP) ; ; SAVE R3
6167 041652 005046 CLR -(SP) ; ; CLEAR THE RUBOUT KEY
6168 041654 012703 042104 1$: MOV #STTYIN, R3 ; ; GET ADDRESS
6169 041660 022703 042176 2$: CMP #STTYIN+72, R3 ; ; BUFFER FULL?
6170 041664 101456 BLOS 4$ ; ; BR IF YES
6171 041666 104407 RDCHR ; ; GO READ ONE CHARACTER FROM THE TTY
6172 041670 112613 MOVB (SP)+, (R3) ; ; GET CHARACTER
6173 041672 122713 000177 10$: CMPB #177, (R3) ; ; IS IT A RUBOUT
6174 041676 001022 BNE 5$ ; ; BR IF NO
6175 041700 005716 TST (SP) ; ; IS THIS THE FIRST RUBOUT?
6176 041702 001007 BNE 6$ ; ; BR IF NO
6177 041704 112737 000134 042102 MOVB #' \, 9$ ; ; TYPE A BACK SLASH
6178 041712 104401 042102 TYPE 9$
6179 041716 012716 177777 MOV #-1, (SP) ; ; SET THE RUBOUT KEY
6180 041722 005303 6$: DEC R3 ; ; BACKUP BY ONE
6181 041724 020327 042104 CMP R3, #STTYIN ; ; STACK EMPTY?
6182 041730 103434 BLO 4$ ; ; BR IF YES
6183 041732 111337 042102 MOVB (R3), 9$ ; ; SETUP TO TYPEOUT THE DELETED CHAR.
6184 041736 104401 042102 TYPE 9$ ; ; GO TYPE
6185 041742 000746 BR 2$ ; ; GO READ ANOTHER CHAR.
6186 041744 005716 5$: TST (SP) ; ; RUBOUT KEY SET?
6187 041746 001406 BEQ 7$ ; ; BR IF NO
6188 041750 112737 000134 042102 MOVB #' \, 9$ ; ; TYPE A BACK SLASH
6189 041756 104401 042102 TYPE 9$
6190 041762 005016 CLR (SP) ; ; CLEAR THE RUBOUT KEY
6191 041764 122713 000025 7$: CMPB #25, (R3) ; ; IS CHARACTER A CTRL U?
6192 041770 001003 BNE 8$ ; ; BR IF NO
6193 041772 104401 042203 TYPE ' SCNTLU ; ; TYPE A CONTROL "U"
6194 041776 000726 BR 1$ ; ; GO START OVER
6195 042000 122713 000022 8$: CMPB #22, (R3) ; ; IS CHARACTER A "↑R"?
6196 042004 001011 BNE 3$ ; ; BRANCH IF NO
6197 042006 105013 CLRB (R3) ; ; CLEAR THE CHARACTER
6198 042010 104401 001267 TYPE ' SCRLF ; ; TYPE A "CR" & "LF"
6199 042014 104401 042104 TYPE ' STTYIN ; ; TYPE THE INPUT STRING
6200 042020 000717 BR 2$ ; ; GO PICKUP ANOTHER CHARACTER
6201 042022 104401 001266 4$: TYPE ' SQUES ; ; TYPE A '?'
6202 042026 000712 BR 1$ ; ; CLEAR THE BUFFER AND LOOP
6203 042030 111337 042102 3$: MOVB (R3), 9$ ; ; ECHO THE CHARACTER
6204 042034 104401 042102 TYPE 9$
6205 042040 122723 000015 CMPB #15, (R3)+ ; ; CHECK FOR RETURN
6206 042044 001305 BNE 2$ ; ; LOOP IF NOT RETURN
6207 042046 105063 177777 CLRB -1(R3) ; ; CLEAR RETURN (THE 15)
6208 042052 104401 001270 TYPE ' SLF ; ; TYPE A LINE FEED
6209 042056 005726 TST (SP)+ ; ; CLEAR RUBOUT KEY FROM THE STACK
6210 042060 012603 MOV (SP)+, R3 ; ; RESTORE R3
6211 042062 011646 MOV (SP), -(SP) ; ; ADJUST THE STACK AND PUT ADDRESS OF THE
6212 042064 016666 000004 000002 MOV 4(SP), 2(SP) ; ; FIRST ASCII CHARACTER ON IT
6213 042072 012766 042104 000004 MOV #STTYIN, 4(SP)
6214 042100 000002 RTI ; ; RETURN
6215 042102 000 9$: .BYTE 0 ; ; STORAGE FOR ASCII CHAR. TO TYPE
6216 042103 000 .BYTE 0 ; ; TERMINATOR
6217 042104 000072 STTYIN: .BLKB 72 ; ; RESERVE 72 BYTES FOR TTY INPUT
6218 042176 041536 005015 000 SCNTLC: .ASCIZ /↑C/<15><12> ; ; CONTROL "C"
6219 042203 136 006525 000012 SCNTLU: .ASCIZ /↑U/<15><12> ; ; CONTROL "U"

```

```

6220 042210 043536 005015 000 SCNTLG: .ASCIZ /fG<15><12> ;;CONTROL "G"
6221 042215 015 051412 051127 SMSWR: .ASCIZ <15><12>/SWR = /
6222 042222 036440 000040
6223 042226 020040 042516 020127 SMNEW: .ASCIZ / NEW = /
6224 042234 020075 000
6225 042240
6226 .EVEN
6227 .SBTTL POWER DOWN AND UP ROUTINES
6228
6229 *****
6230 042240 012737 042400 000024 $PWRDN: MOV $SILLUP, @PWRVEC ;; SET FOR FAST UP
6231 042246 012737 000340 000026 MOV @340, @PWRVEC+2 ;; Prio:7
6232 042254 010046 MOV R0, -(SP) ;; PUSH R0 ON STACK
6233 042256 010146 MOV R1, -(SP) ;; PUSH R1 ON STACK
6234 042260 010246 MOV R2, -(SP) ;; PUSH R2 ON STACK
6235 042262 010346 MOV R3, -(SP) ;; PUSH R3 ON STACK
6236 042264 010446 MOV R4, -(SP) ;; PUSH R4 ON STACK
6237 042266 010546 MOV R5, -(SP) ;; PUSH R5 ON STACK
6238 042270 017746 136644 MOV @SWR, -(SP) ;; PUSH @SWR ON STACK
6239 042274 010637 042404 MOV SP, $SAVR6 ;; SAVE SP
6240 042300 012737 042312 000024 MOV @PWRUP, @PWRVEC ;; SET UP VECTOR
6241 042306 000000 HALT
6242 042310 000776 BR .-2 ;; HANG UP
6243
6244 *****
6245 $PWRUP: MOV $SILLUP, @PWRVEC ;; SET FOR FAST DOWN
6246 042312 012737 042400 000024 $PWRUP: MOV $SAVR6, SP ;; GET SP
6247 042320 013706 042404 CLR $SAVR6 ;; WAIT LOOP FOR THE TTY
6248 042324 005037 042404 1$: INC $SAVR6 ;; WAIT FOR THE INC
6249 042330 005237 042404 BNE 1$ ;; OF WORD
6250 042334 001375 MOV (SP)+, @SWR ;; POP STACK INTO @SWR
6251 042336 012677 136576 MOV (SP)+, R5 ;; POP STACK INTO R5
6252 042342 012605 MOV (SP)+, R4 ;; POP STACK INTO R4
6253 042344 012604 MOV (SP)+, R3 ;; POP STACK INTO R3
6254 042346 012603 MOV (SP)+, R2 ;; POP STACK INTO R2
6255 042350 012602 MOV (SP)+, R1 ;; POP STACK INTO R1
6256 042352 012601 MOV (SP)+, R0 ;; POP STACK INTO R0
6257 042354 012600 MOV @PWRDN, @PWRVEC ;; SET UP THE POWER DOWN VECTOR
6258 042356 012737 042240 000024 MOV @340, @PWRVEC+2 ;; Prio:7
6259 042364 012737 000340 000026 TYPE SPOWER ;; REPORT THE POWER FAILURE
6260 042372 104401 SPWRMG: .WORD SPOWER ;; POWER FAIL MESSAGE POINTER
6261 042374 042406 RTI
6262 042376 000002 $SILLUP: HALT ;; THE POWER UP SEQUENCE WAS STARTED
6263 042400 000000 BR .-2 ;; BEFORE THE POWER DOWN WAS COMPLETE
6264 042402 000776 $SAVR6: 0 ;; PUT THE SP HERE
6265 042404 000000 SPOWER: .ASCIZ <15><12>"POWER"
6266 042406 005015 047520 042527
6267 042414 000122
6268 .EVEN
6269 .SBTTL ROUTINE TO SIZE MEMORY
6270
6271 *****
6272 *CALL: JSR PC, $SIZE
6273 * RETURN
6274 *$LSTAD WILL CONTAIN THE LAST AVAILABLE MEMORY LOCATION
6275

```



```

SSIZE:  MOV    RO,-(SP)      ;;SAVE RO ON THE STACK
        MOV    R1,-(SP)      ;;SAVE R1 ON THE STACK
        MOV    @#ERRVEC,-(SP) ;;SAVE PRESENT ERROR VECTOR PS & PC
        MOV    @#ERRVEC+2,-(SP)
        MOV    SP,RO         ;;SAVE THE STACK POINTER
;;SET THE ERRVEC PS TO THE PRESENT PS
        TRAP
        MOV    (SP)+,@#ERRVEC+2 ;;PUSH OLD PSM AND PC ON STACK
        MOV    @2$,@#ERRVEC     ;;SAVE THE PSM IN @#ERRVEC+2
        MOV    @2000,R1        ;;SET FOR TIMEOUT
        MOV    @2000,R1        ;;FIRST ADDRESS
1$:     TST    (R1)           ;;TEST THIS ADDRESS
        TST    (R1)+         ;;STEP TO NEXT ADDRESS
        BR     1$            ;;TRY ANOTHER
2$:     SUB    @2,R1          ;;DROP BACK
        MOV    RO,SP         ;;RESTORE THE STACK
        MOV    (SP)+,@#ERRVEC+2 ;;RESTORE ERROR VECTOR
        MOV    (SP)+,@#ERRVEC
        MOV    R1,$LSTAD      ;;LAST ADDRESS
        MOV    (SP)+,R1       ;;RESTORE R1
        MOV    (SP)+,RO       ;;RESTORE RO
        RTS    PC
$LSTAD: .WORD    0           ;;CONTAINS THE LAST ADDRESS
.SBTTL  SAVE AND RESTORE RO-R5 ROUTINES

```

```

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

```

```

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

```

```

;#RESTORE RO-R5
;#CALL:
;# RESREG

```

```

6276
6277 042416 010046
6278 042420 010146
6279 042422 013746 000004
6280 042426 013746 000006
6281 042432 010600
6282
6283 042434 104400
6284 042436 012637 000006
6285 042442 012737 042462 000004
6286 042450 012701 020000
6287 042454 005711
6288 042456 005721
6289 042460 000775
6290 042462 162701 000002
6291 042466 010006
6292 042470 012637 000006
6293 042474 012637 000004
6294 042500 010137 042512
6295 042504 012601
6296 042506 012600
6297 042510 000207
6298 042512 000000
6299
6300
6301
6302
6303
6304
6305
6306
6307
6308
6309
6310
6311
6312
6313
6314
6315
6316 042514
6317 042514 010046
6318 042516 010146
6319 042520 010246
6320 042522 010346
6321 042524 010446
6322 042526 010546
6323 042530 016646 000022
6324 042534 016646 000022
6325 042540 016646 000022
6326 042544 016646 000022
6327 042550 000002
6328
6329
6330
6331

```

```

6332 042552
6333 042552 012666 000022
6334 042556 012666 000022
6335 042562 012666 000022
6336 042566 012666 000022
6337 042572 012605
6338 042574 012604
6339 042576 012603
6340 042600 012602
6341 042602 012601
6342 042604 012600
6343 042606 000002
6344
6345
6346
6347
6348
6349
6350
6351
6352 042610 010046
6353 042612 016600 000002
6354 042616 005740
6355 042620 111000
6356 042622 006300
6357 042624 016000 042644
6358 042630 000200
6359
6360
6361
6362
6363 042632 011646
6364 042634 016666 000004 000002
6365 042642 000002
6366
6367
6368
6369
6370
6371
6372
6373
6374 042644 042632
6375 042646 037772
6376 042650 040356
6377 042652 040332
6378 042654 040372
6379
6380 042656 041306
6381
6382 042660 041216
6383 042662 041560
6384 042664 041650
6385 042666 042514
6386 042670 042552
6387

```

```

$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

```

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

```

$TRAP2: MOV (SP),-(SP) ;;MOVE THE PC DOWN
MOV 4(SP),2(SP) ;;MOVE THE PSW DOWN
RTI ;;RESTORE THE PSW

```

.SBTTL TRAP TABLE

```

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

```

```

; ROUTINE
;-----
$TRPAD: .WORD $TRAP2 TRAP+1(104401) TTY TYPEOUT ROUTINE
$TYPE ;;CALL=TYPE TRAP+2(104402) TYPE OCTAL NUMBER (WITH LEADING ZEROS)
$TYPOC ;;CALL=TYPOC TRAP+3(104403) TYPE OCTAL NUMBER (NO LEADING ZEROS)
$TYPOS ;;CALL=TYPOS TRAP+4(104404) TYPE OCTAL NUMBER (AS PER LAST CALL)
$TYPON ;;CALL=TYPON
$GTSWR ;;CALL=GTSWR TRAP+5(104405) GET SOFT-SWR SETTING
$CKSWR ;;CALL=CKSWR TRAP+6(104406) TEST FOR CHANGE IN SOFT-SWR
$RDCHR ;;CALL=RDCHR TRAP+7(104407) TTY TYPEIN CHARACTER ROUTINE
$RDLIN ;;CALL=RDLIN TRAP+10(104410) TTY TYPEIN STRING ROUTINE
$SAVREG ;;CALL=SAVREG TRAP+11(104411) SAVE R0-R5 ROUTINE
$RESREG ;;CALL=RESREG TRAP+12(104412) RESTORE R0-R5 ROUTINE

```

6388	042672	025052	020040	047125	EM1:	.ASCIZ	/**	UNIBUS PARITY ERROR/
6389	042700	041111	051525	050040				
6390	042706	051101	052111	020131				
6391	042714	051105	047522	000122				
6392	042722	025052	020040	047516	EM2:	.ASCIZ	/**	NON-EXISTANT MEMORY ERROR/
6393	042730	026516	054105	051511				
6394	042736	040524	052116	046440				
6395	042744	046505	051117	020131				
6396	042752	051105	047522	000122				
6397	042760	025052	020040	047516	EM3:	.ASCIZ	/**	NON-EXISTANT DRIVE ERROR/
6398	042766	026516	054105	051511				
6399	042774	040524	052116	042040				
6400	043002	044522	042526	042440				
6401	043010	051122	051117	000				
6402	043015	052	020052	052440	EM4:	.ASCIZ	/**	UNIT FIELD ERROR/
6403	043022	044516	020124	044506				
6404	043030	046105	020104	051105				
6405	043036	047522	000122					
6406	043042	025052	020040	052523	EM5:	.ASCIZ	/**	SUBSYSTEM TIMEOUT/
6407	043050	051502	051531	042524				
6408	043056	020115	044524	042515				
6409	043064	052517	000124					
6410	043070	025052	020040	051104	EM6:	.ASCIZ	/**	DRIVE BUS PARITY ERROR/
6411	043076	053111	020105	052502				
6412	043104	020123	040520	044522				
6413	043112	054524	042440	051122				
6414	043120	051117	000					
6415	043123	052	020052	042040	EM7:	.ASCIZ	/**	DRIVE DETECTED PARITY ERROR/
6416	043130	044522	042526	042040				
6417	043136	052105	041505	042524				
6418	043144	020104	040520	044522				
6419	043152	054524	042440	051122				
6420	043160	051117	000					
6421	043163	052	020052	040440	EM10:	.ASCIZ	/**	AC LOW/
6422	043170	020103	047514	000127				
6423	043176	025052	020040	050123	EM11:	.ASCIZ	/**	SPEED LOSS/
6424	043204	042505	020104	047514				
6425	043212	051523	000					
6426	043215	052	020052	044440	EM12:	.ASCIZ	/**	ILLEGAL FUNCTION ERROR/
6427	043222	046114	043505	046101				
6428	043230	043040	047125	052103				
6429	043236	047511	020116	051105				
6430	043244	047522	000122					
6431	043250	025052	020040	051120	EM13:	.ASCIZ	/**	PROGRAMMING ERROR/
6432	043256	043517	040522	046515				
6433	043264	047111	020107	051105				
6434	043272	047522	000122					
6435	043276	025052	020040	047516	EM14:	.ASCIZ	/**	NON-EXISTANT FUNCTION ERROR/
6436	043304	026516	054105	051511				
6437	043312	040524	052116	043040				
6438	043320	047125	052103	047511				
6439	043326	020116	051105	047522				
6440	043334	000122						
6441	043336	025052	020040	051104	EM15:	.ASCIZ	/**	DRIVE TYPE ERROR/
6442	043344	053111	020105	054524				
6443	043352	042520	042440	051122				

6444	043360	051117	000				
6445	043363	052	020052	043040	EM16:	.ASCIZ	/** FORMAT ERROR/
6446	043370	051117	040515	020124			
6447	043376	051105	047522	000122			
6448	043404	025052	020040	051127	EM17:	.ASCIZ	/** WRITE LOCK ERROR/
6449	043412	052111	020105	047514			
6450	043420	045503	042440	051122			
6451	043426	051117	000				
6452	043431	052	020052	042040	EM20:	.ASCIZ	/** DRIVE UNSAFE ERROR/
6453	043436	044522	042526	052440			
6454	043444	051516	043101	020105			
6455	043452	051105	047522	000122			
6456	043460	025052	020040	042523	EM21:	.ASCIZ	/** SEEK INCOMPLETE ERROR/
6457	043466	045505	044440	041516			
6458	043474	046517	046120	052105			
6459	043502	020105	051105	047522			
6460	043510	000122					
6461	043512	025052	020040	054503	EM22:	.ASCIZ	/** CYLINDER OVERFLOW ERROR/
6462	043520	044514	042116	051105			
6463	043526	047440	042526	043122			
6464	043534	047514	020127	051105			
6465	043542	047522	000122				
6466	043546	025052	020040	046111	EM23:	.ASCIZ	/** ILLEGAL CYLINDER ADDRESS ERROR/
6467	043554	042514	040507	020114			
6468	043562	054503	044514	042116			
6469	043570	051105	040440	042104			
6470	043576	042522	051523	042440			
6471	043604	051122	051117	000			
6472	043611	052	020052	042040	EM24:	.ASCIZ	/** DRIVE OFF TRACK/
6473	043616	044522	042526	047440			
6474	043624	043106	052040	040522			
6475	043632	045503	000				
6476	043635	052	020052	042040	EM25:	.ASCIZ	/** DRIVE TIMING ERROR/
6477	043642	044522	042526	052040			
6478	043650	046511	047111	020107			
6479	043656	051105	047522	000122			
6480	043664	025052	020040	040504	EM26:	.ASCIZ	/** DATA LATE ERROR/
6481	043672	040524	046040	052101			
6482	043700	020105	051105	047522			
6483	043706	000122					
6484	043710	025052	020040	047503	EM27:	.ASCIZ	/** CONTROLLER TIMEOUT ERROR/
6485	043716	052116	047522	046114			
6486	043724	051105	052040	046511			
6487	043732	047505	052125	042440			
6488	043740	051122	051117	000			
6489	043745	052	020052	047440	EM30:	.ASCIZ	/** OPERATION INCOMPLETE ERROR/
6490	043752	042520	040522	044524			
6491	043760	047117	044440	041516			
6492	043766	046517	046120	052105			
6493	043774	020105	051105	047522			
6494	044002	000122					
6495	044004	025052	020040	042510	EM31:	.ASCIZ	/** HEADER VRC ERROR/
6496	044012	042101	051105	053040			
6497	044020	041522	042440	051122			
6498	044026	051117	000				
6499	044031	052	020052	042040	EM32:	.ASCIZ	/** DATA CHECK ERROR/

6500	044036	052101	020101	044103			
6501	044044	041504	020113	051105			
6502	044052	047522	000122				
6503	044056	025052	020040	051127	EM33:	.ASCIZ	/** WRITE CHECK ERROR/
6504	044064	052111	020105	044103			
6505	044072	041505	020113	051105			
6506	044100	047522	000122				
6507	044104	025052	020040	040504	EM34:	.ASCIZ	/** DATA MISCOMPARE/
6508	044112	040524	046440	051511			
6509	044120	047503	050115	051101			
6510	044126	000105					
6511	044130	025052	020040	047516	EM35:	.ASCIZ	/** NO DRIVE RESPONSE-UFE & NED/
6512	044136	042040	044522	042526			
6513	044144	051040	051505	047520			
6514	044152	051516	026505	043125			
6515	044160	020105	020046	042516			
6516	044166	000104					
6517	044170	025052	020040	051104	EM36:	.ASCIZ	/** DRIVE ERROR WILL NOT CLEAR/
6518	044176	053111	020105	051105			
6519	044204	047522	020122	044527			
6520	044212	046114	047040	052117			
6521	044220	041440	042514	051101			
6522	044226	000					
6523	044227	052	020052	042040	EM37:	.ASCIZ	/** DRIVE STATUS CHANGE WILL NOT CLEAR/
6524	044234	044522	042526	051440			
6525	044242	040524	052524	020123			
6526	044250	044103	047101	042507			
6527	044256	053440	046111	020114			
6528	044264	047516	020124	046103			
6529	044272	040505	000122				
6530	044276	025052	020040	052101	EM40:	.ASCIZ	/** ATTENTION BUT NO STATUS CHANGE OR FAULT/
6531	044304	042524	052116	047511			
6532	044312	020116	052502	020124			
6533	044320	047516	051440	040524			
6534	044326	052524	020123	044103			
6535	044334	047101	042507	047440			
6536	044342	020122	040506	046125			
6537	044350	000124					
6538	044352	025052	020040	052101	EM41:	.ASCIZ	/** ATTENTION BUT DRIVE NOT AVAILABLE/
6539	044360	042524	052116	047511			
6540	044366	020116	052502	020124			
6541	044374	051104	053111	020105			
6542	044402	047516	020124	053101			
6543	044410	044501	040514	046102			
6544	044416	000105					
6545	044420	025052	020040	052101	EM42:	.ASCIZ	/** ATTENTION WHEN NOT EXPECTED/
6546	044426	042524	052116	047511			
6547	044434	020116	044127	047105			
6548	044442	047040	052117	042440			
6549	044450	050130	041505	042524			
6550	044456	000104					
6551	044460	025052	020040	051105	EM43:	.ASCIZ	/** ERROR WHILE GATHERING DRIVE STATUS/
6552	044466	047522	020122	044127			
6553	044474	046111	020105	040507			
6554	044502	044124	051105	047111			
6555	044510	020107	051104	053111			

6556	044516	020105	052123	052101			
6557	044524	051525	000				
6558	044527	052	020052	046440	EM52:	.ASCIZ	/** MULTIPLE DRIVE SELECT/
6559	044534	046125	044524	046120			
6560	044542	020105	051104	053111			
6561	044550	020105	042523	042514			
6562	044556	052103	000				
6563	044561	052	020052	044040	EM53:	.ASCIZ	/** HEADER COMPARE ERROR/
6564	044566	040505	042504	020122			
6565	044574	047503	050115	051101			
6566	044602	020105	051105	047522			
6567	044610	000122					
6568	044612	025052	020040	052523	EM56:	.ASCIZ	/** SUBSYSTEM TIMEOUT/
6569	044620	051502	051531	042524			
6570	044626	020115	044524	042515			
6571	044634	052517	000124				
6572	044640	025052	020040	051105	EM60:	.ASCIZ	/** ERROR IN RECALIBRATE FOR RECOVERY/
6573	044646	047522	020122	047111			
6574	044654	051040	041505	046101			
6575	044662	041111	040522	042524			
6576	044670	043040	051117	051040			
6577	044676	041505	053117	051105			
6578	044704	000131					
6579	044706	025052	020040	051120	EM61:	.ASCIZ	/** PROGRAM ABORTING FATAL ERROR IN RETRY/
6580	044714	043517	040522	020115			
6581	044722	041101	051117	044524			
6582	044730	043516	043040	052101			
6583	044736	046101	042440	051122			
6584	044744	051117	044440	020116			
6585	044752	042522	051124	000131			
6586	044760	025052	020040	042510	EM62:	.ASCIZ	/** HEADER MISCOMPARE/
6587	044766	042101	051105	046440			
6588	044774	051511	047503	050115			
6589	045002	051101	000105				
6590	045006	025052	020040	046103	EM63:	.ASCIZ	/** CLEAR CONTROLLER DID NOT CLEAR ERROR/
6591	045014	040505	020122	047503			
6592	045022	052116	047522	046114			
6593	045030	051105	042040	042111			
6594	045036	047040	052117	041440			
6595	045044	042514	051101	042440			
6596	045052	051122	051117	000			
6597	045057	052	020052	047040	EM64:	.ASCIZ	/** NO ATTENTION IN ATTENTION SUMMARY REGISTER/
6598	045064	020117	052101	042524			
6599	045072	052116	047511	020116			
6600	045100	047111	040440	052124			
6601	045106	047105	044524	047117			
6602	045114	051440	046525	040515			
6603	045122	054522	051040	043505			
6604	045130	051511	042524	000122			
6605	045136	025052	020040	047125	EM65:	.ASCIZ	/** UNSOLICITED ATTENTION/
6606	045144	047523	044514	044503			
6607	045152	042524	020104	052101			
6608	045160	042524	052116	047511			
6609	045166	000116					
6610	045170	025052	020040	047125	EM66:	.ASCIZ	/** UNEXPECTED DATA TYPE ERROR/
6611	045176	054105	042520	052103			

6612	045204	042105	042040	052101		
6613	045212	020101	054524	042520		
6614	045220	042440	051122	051117		
6615	045226	000				
6616	045227	052	020052	040440	EM67:	.ASCIZ /** ATTENTION DID NOT RESET WITH CLEAR/
6617	045234	052124	047105	044524		
6618	045242	047117	042040	042111		
6619	045250	047040	052117	051040		
6620	045256	051505	052105	053440		
6621	045264	052111	020110	046103		
6622	045272	040505	000122			
6623	045276	025052	020040	052523	EM70:	.ASCIZ /** SUBSYSTEM CLEAR DID NOT CLEAR DRIVE ATTENTION/
6624	045280	051502	051531	042524		
6625	045312	020115	046103	040505		
6626	045320	020122	044504	020104		
6627	045326	047516	020124	046103		
6628	045334	040505	020122	051104		
6629	045342	053111	020105	052101		
6630	045350	042524	052116	047511		
6631	045356	000116				
6632	045360	025052	020040	040504	EM71:	.ASCIZ /** DATA LATE WHEN UNLOADING HEADER/
6633	045366	040524	046040	052101		
6634	045374	020105	044127	047105		
6635	045402	052440	046116	040517		
6636	045410	044504	043516	044040		
6637	045416	040505	042504	000122		
6638	045424	025052	020040	047503	EM72:	.ASCIZ /** CONTROLLER ERROR WHEN DRIVER SERVICING/
6639	045432	052116	047522	046114		
6640	045440	051105	042440	051122		
6641	045446	051117	053440	042510		
6642	045454	020116	051104	053111		
6643	045462	051105	051440	051105		
6644	045470	044526	044503	043516		
6645	045476	000				
6646	045477	052	020052	042040	EM73:	.ASCIZ /** DRIVE DETECTED PARITY ERROR/
6647	045504	044522	042526	042040		
6648	045512	052105	041505	042524		
6649	045520	020104	040520	044522		
6650	045526	054524	042440	051122		
6651	045534	051117	000			
6652	045537	052	020052	052440	EM74:	.ASCIZ /** UNDEFINED ERROR/
6653	045544	042116	043105	047111		
6654	045552	042105	042440	051122		
6655	045560	051117	000			
6656	045563	052	020052	046440	EM75:	.ASCIZ /** MARKING THIS SECTOR BAD/
6657	045570	051101	044513	043516		
6658	045576	052040	044510	020123		
6659	045604	042523	052103	051117		
6660	045612	041040	042101	000		
6661	045617	052	020052	041040	EM76:	.ASCIZ /** BAD DATA VERIFICATION WITH READ. ECC OF LAST RETRY IS:/
6662	045624	042101	042040	052101		
6663	045632	020101	042526	044522		
6664	045640	044506	040503	044524		
6665	045646	047117	053440	052111		
6666	045654	020110	042522	042101		
6667	045662	020056	041505	020103		

6668	045670	043117	046040	051501	
6669	045676	020124	042522	051124	
6670	045704	020131	051511	000072	
6671	045712	025052	020040	042522	EM77: .ASCIZ /** RETRY SUCCESSFUL/
6672	045720	051124	020131	052523	
6673	045726	041503	051505	043123	
6674	045734	046125	000		
6675	045737	052103	020052	051040	EM100: .ASCIZ /** RETRY UNSUCCESSFUL/
6676	045744	052103	054522	052440	
6677	045752	051516	041525	042503	
6678	045760	051523	052506	000114	
6679	045766	025052	020040	040503	EM101: .ASCIZ /** CANNOT FIND A VALID HEADER IN TRACK JUST READ/
6680	045774	047116	052117	043040	
6681	046002	047111	020104	020101	
6682	046010	040526	044514	020104	
6683	046016	042510	042101	051105	
6684	046024	044440	020116	051124	
6685	046032	041501	020113	052512	
6686	046040	052123	051040	040505	
6687	046046	000104			
6688	046050	025052	020040	040502	EM102: .ASCIZ /** BAD SECTOR ERROR ON SECTOR NOT LISTED BAD/
6689	046056	020104	042523	052103	
6690	046064	051117	042440	051122	
6691	046072	051117	047440	020116	
6692	046100	042523	052103	051117	
6693	046106	047040	052117	046040	
6694	046114	051511	042524	020104	
6695	046122	040502	000104		
6696	046126	025052	053440	051117	EM103: .ASCIZ /** WORD COUNT INCORRECT TO CONTINUE/
6697	046134	020104	047503	047125	
6698	046142	020124	047111	047503	
6699	046150	051122	041505	020124	
6700	046156	047524	041440	047117	
6701	046164	044524	052516	000105	
6702	046172	025052	040503	052125	EM104: .ASCIZ /**CAUTION** DRIVE STATUS REPORTED MAY NOT BE CORRECT/
6703	046200	047511	025116	020052	
6704	046206	051104	053111	020105	
6705	046214	052123	052101	051525	
6706	046222	051040	050105	051117	
6707	046230	042524	020104	040515	
6708	046236	020131	047516	020124	
6709	046244	042502	041440	051117	
6710	046252	042522	052103	000	
6711	046257	105	051122	051117	EM105: .ASCII /ERROR IN SECTOR ALREADY MARKED BAD/<15><12>
6712	046264	044440	020116	042523	
6713	046272	052103	051117	040440	
6714	046300	051114	040505	054504	
6715	046306	046440	051101	042513	
6716	046314	020104	040502	006504	
6717	046322	012			
6718	046323	123	051525	042520	.ASCII /SUSPECT PHYSICAL DAMAGE IN HEADER AREA/<15><12>
6719	046330	052103	050040	054510	
6720	046336	044523	040503	020114	
6721	046344	040504	040515	042507	
6722	046352	044440	020116	042510	
6723	046360	042101	051105	040440	

6780	047056	030101	020063	020040	
6781	047064	020040	030102	000063	
6782	047072	042510	042101	051105	DH601: .ASCIZ /HEADER SHOULD BE: /
6783	047100	051440	047510	046125	
6784	047106	020104	042502	000072	
6785	047114	042510	042101	051105	DH602: .ASCIZ /HEADER OF SECTOR 0 THIS CYLINDER IS: /
6786	047122	047440	020106	042523	
6787	047130	052103	051117	030040	
6788	047136	052040	044510	020123	
6789	047144	054503	044514	042116	
6790	047152	051105	044440	035123	
6791	047160	000			
6792	047161	123	040505	041522	DH603: .ASCIZ /SEARCHING FOR HEADER: /
6793	047166	044510	043516	043040	
6794	047174	051117	044040	040505	
6795	047202	042504	035122	000	
6796	047207	120	041501	020113	DH604: .ASCIZ /PACK ADDRESS OF ERROR IS: /
6797	047214	042101	051104	051505	
6798	047222	020123	043117	042440	
6799	047230	051122	051117	044440	
6800	047236	035123	000		
6801	047241	103	046131	042116	DH6041: .ASCIZ /CYLNDR TRACK SECTOR/
6802	047246	020122	052040	040522	
6803	047254	045503	020040	051440	
6804	047262	041505	047524	000122	
6805	047270	054503	047114	051104	DH6042: .ASCIZ /CYLNDR TRACK/
6806	047276	020040	051124	041501	
6807	047304	000113			
6808	047306	041505	020103	040504	DH605: .ASCIZ /ECC DATA IS: /
6809	047314	040524	044440	035123	
6810	047322	000			
6811	047323	120	051517	020040	DH6051: .ASCIZ /POS PAT CORRECTABLE? /
6812	047330	020040	050040	052101	
6813	047336	020040	020040	041440	
6814	047344	051117	042522	052103	
6815	047352	041101	042514	000077	
6816	047360	047527	042122	020061	DH606: .ASCIZ /WORD1 WORD2 WORD3/
6817	047366	020040	047527	042122	
6818	047374	020062	020040	047527	
6819	047402	042122	000063		
6820	047406	047507	042117	020040	DH701: .ASCIZ /GOOD BAD WORD NUM. /
6821	047414	020040	040502	020104	
6822	047422	020040	020040	047527	
6823	047430	042122	047040	046525	
6824	047436	000			
6825	047437	102	042101	044040	DH607: .ASCIZ /BAD HEADER IS: /
6826	047444	040505	042504	020122	
6827	047452	051511	000072		
6828	047456	045522	051504	020040	DH204: .ASCIZ /RKDS RKER RKMR2 RKMR3/
6829	047464	020040	045522	051105	
6830	047472	020040	020040	045522	
6831	047500	051115	020062	020040	
6832	047506	045522	051115	000063	
6833	047514	051105	047522	020122	DH502: .ASCIZ /ERROR TRYING TO GET FAILURE INFORMATION/
6834	047522	051124	044531	043516	
6835	047530	052040	020117	042507	

6836	047536	020124	040506	046111					
6837	047544	051125	020105	047111					
6838	047552	047506	046522	052101					
6839	047560	047511	000116						
6840	047564	042523	047503	042116	DH503:	.ASCIZ	/SECOND TIMEOUT OCCURRED GETTING DRIVE STATUS/		
6841	047572	052040	046511	047505					
6842	047600	052125	047440	041503					
6843	047606	051125	042522	020104					
6844	047614	042507	052124	047111					
6845	047622	020107	051104	053111					
6846	047630	020105	052123	052101					
6847	047636	051525	000						
6848	047641	116	046525	042502	DH800:	.ASCIZ	/NUMBER OF RETRIES:/		
6849	047646	020122	043117	051040					
6850	047654	052105	044522	051505					
6851	047662	000072							
6852									
6853	047664	042523	052103	051117	DH900:	.EVEN			
6854	047672	020040	041527	051440		.ASCIZ	/SECTOR WC SB WC IS/		
6855	047700	020102	020040	041527					
6856	047706	044440	000123						
6857	047712	001116	006340	006372	DT100:	.WORD	\$ERRPC, DRINUS, ERRCOM, \$REG0, \$REG1, \$REG2, \$REG3, \$REG4		
6858	047720	001162	001164	001166					
6859	047726	001170	001172						
6860	047732	001174	001176	001200	DT201:	.WORD	\$REG5, \$REG6, \$REG7, \$REG10, \$REG11, \$REG12, \$REG13		
6861	047740	001202	001204	001206					
6862	047746	001210							
6863	047750	001212	001214		DT202:	.WORD	\$REG14, \$REG15		
6864	047754	001216	001220	001222	DT203:	.WORD	\$REG16, \$REG17, \$REG20, \$REG21, \$REG22, \$REG23, \$REG24, \$REG25		
6865	047762	001224	001226	001230					
6866	047770	001232	001234						
6867	047774	001174	001176	001200	DT601:	.WORD	\$REG5, \$REG6, \$REG7		
6868	050002	001202	001204	001206	DT602:	.WORD	\$REG10, \$REG11, \$REG12		
6869									
6870	050010	000005			DF01:	.WORD	5	: NUMBER OF HEADER LINES	
6871	050012	000				.BYTE	0	: NUMBER OF COL FOR FIRST HDR	
6872	050013	000				.BYTE	0	: ALL CHARACTERS OCTAL	
6873	050014	046573				.WORD	DH101	: SECOND HDR LINE	
6874	050016	010	000			.BYTE	10,0	: NUM OF COL-ALL OCTAL	
6875	050020	046440				.WORD	DH200		
6876	050022	000	000			.BYTE	0,0		
6877	050024	046673				.WORD	DH201		
6878	050026	007	000			.BYTE	7,0		
6879	050030	046460				.WORD	DH500		
6880	050032	000	000			.BYTE	0,0		
6881									
6882	050034	000006			DF02:	.WORD	6		
6883	050036	000	000			.BYTE	0,0		
6884	050040	046573				.WORD	DH101		
6885	050042	010	000			.BYTE	10,0		
6886	050044	046440				.WORD	DH200		
6887	050046	000	000			.BYTE	0,0		
6888	050050	046673				.WORD	DH201		
6889	050052	007	000			.BYTE	7,0		
6890	050054	046761				.WORD	DH202		
6891	050056	002	000			.BYTE	2,0		

6892	050060	046776		.WORD	DH203
6893	050062	010	000	.BYTE	10,0
6894					
6895	050064	000007		DF03: .WORD	7
6896	050066	000	000	.BYTE	0,0
6897	050070	046573		.WORD	DH101
6898	050072	010	000	.BYTE	10,0
6899	050074	046440		.WORD	DH200
6900	050076	000	000	.BYTE	0,0
6901	050100	046673		.WORD	DH201
6902	050102	007	000	.BYTE	7,0
6903	050104	046516		.WORD	DH501
6904	050106	000	000	.BYTE	0,0
6905	050110	046761		.WORD	DH202
6906	050112	002	000	.BYTE	2,0
6907	050114	046776		.WORD	DH203
6908	050116	010	000	.BYTE	10,0
6909					
6910	050120	000006		DF05: .WORD	6
6911	050122	000	000	.BYTE	0,0
6912	050124	046573		.WORD	DH101
6913	050126	010	000	.BYTE	10,0
6914	050130	047161		.WORD	DH603
6915	050132	000	000	.BYTE	0,0
6916	050134	047360		.WORD	DH606
6917	050136	003	000	.BYTE	3,0
6918	050140	047114		.WORD	DH602
6919	050142	000	000	.BYTE	0,0
6920	050144	047360		.WORD	DH606
6921	050146	003	000	.BYTE	3,0
6922					
6923	050150	000006		DF06: .WORD	6
6924	050152	000	000	.BYTE	0,0
6925	050154	046573		.WORD	DH101
6926	050156	010	000	.BYTE	10,0
6927	050160	047072		.WORD	DH601
6928	050162	000	000	.BYTE	0,0
6929	050164	047360		.WORD	DH606
6930	050166	003	000	.BYTE	3,0
6931					
6932	050170	047114		.WORD	DH607
6933	050172	000	000	.WORD	DH602
6934	050174	047360		.BYTE	0,0
6935	050176	003	000	.WORD	DH606
6936				.BYTE	3,0
6937	050200	000006		DF07: .WORD	6
6938	050202	000	000	.BYTE	0,0
6939	050204	046573		.WORD	DH101
6940	050206	010	000	.BYTE	10,0
6941	050210	047207		.WORD	DH604
6942	050212	000	000	.BYTE	0,0
6943	050214	047241		.WORD	DH6041
6944	050216	003	000	.BYTE	3,0
6945	050220	047306		.WORD	DH605
6946	050222	000	000	.BYTE	0,0
6947	050224	047323		.WORD	DH6051

; "THE FOLLOWING REG DATA MB INCORRECT"

; OPERATION INCOMPLETE

; HEADER VRC ERROR

; THIS LINE WHEN SPECIFIC HEADER CAN BE READ
; THIS LINE THEN DELETES

6948	050226	003	000		.BYTE	3,0
6949						
6950	050230	000004		DF10:	.WORD	4
6951	050232	000	000		.BYTE	0,0
6952	050234	046573			.WORD	DH101
6953	050236	010	000		.BYTE	10,0
6954	050240	047207			.WORD	DH6C4
6955	050242	000	000		.BYTE	0,0
6956	050244	047241			.WORD	DH6041
6957	050246	003	000		.BYTE	3,0
6958						
6959	050250	000003		DF11:	.WORD	3
6960	050252	000	000		.BYTE	0,0
6961	050254	047241			.WORD	DH6041
6962	050256	003	000		.BYTE	3,0
6963	050260	047406			.WORD	DH701
6964	050262	000	000		.BYTE	0,0
6965						
6966	050264	000005		DF12:	.WORD	5
6967	050266	000	000		.BYTE	0,0
6968	050270	046573			.WORD	DH101
6969	050272	010	000		.BYTE	10,0
6970	050274	046440			.WORD	DH200
6971	050276	000	000		.BYTE	0,0
6972	050300	046673			.WORD	DH201
6973	050302	007	000		.BYTE	7,0
6974	050304	047456			.WORD	DH204
6975	050306	004	000		.BYTE	4,0
6976						
6977	050310	000005		DF13:	.WORD	5
6978	050312	000	000		.BYTE	0,0
6979	050314	046573			.WORD	DH101
6980	050316	010	000		.BYTE	10,0
6981	050320	047072			.WORD	DH601
6982	050322	000	000		.BYTE	0,0
6983	050324	047360			.WORD	DH606
6984	050326	003	000		.BYTE	3,0
6985	050330	047514			.WORD	DH502
6986	050332	000	000		.BYTE	0,0
6987						
6988	050334	000005		DF14:	.WORD	5
6989	050336	000	000		.BYTE	0,0
6990	050340	046573			.WORD	DH101
6991	050342	010	000		.BYTE	10,0
6992	050344	047161			.WORD	DH603
6993	050346	000	000		.BYTE	0,0
6994	050350	047360			.WORD	DH606
6995	050352	003	000		.BYTE	3,0
6996	050354	047514			.WORD	DH502
6997	050356	000	000		.BYTE	0,0
6998						
6999	050360	000010		DF15:	.WORD	10
7000	050362	000	000		.BYTE	0,0
7001	050364	046573			.WORD	DH101
7002	050366	010	000		.BYTE	10,0
7003	050370	046440			.WORD	DH200

:FORMAT FOR 2ND LEVEL ERROR
:IN HEADER COMPARE ERROR
:AND 2ND LEVEL HEADER
:VRC ERROR

:FORMAT FOR 2ND LEVEL ERROR
:IN OPERATION INCOMPLETE ERROR

7004	050372	000	000		.BYTE	0,0
7005	050374	046673			.WORD	04201
7006	050376	007	000		.BYTE	7,0
7007	050400	046761			.WORD	04202
7008	050402	002	000		.BYTE	2,0
7009	050404	047564			.WORD	04503
7010	050406	000	000		.BYTE	0,0
7011	050410	046516			.WORD	04501
7012	050412	000	000		.BYTE	0,0
7013	050414	046776			.WORD	04203
7014	050416	010	000		.BYTE	10,0
7015						
7016	050420	000010		DF16:	.WORD	10
7017	050422	000	000		.BYTE	0,0
7018	050424	046573			.WORD	04101
7019	050426	010	000		.BYTE	10,0
7020	050430	046440			.WORD	04200
7021	050432	000	000		.BYTE	0,0
7022	050434	046673			.WORD	04201
7023	050436	007	000		.BYTE	7,0
7024	050440	046761			.WORD	04202
7025	050442	002	000		.BYTE	2,0
7026	050444	047514			.WORD	04502
7027	050446	000	000		.BYTE	0,0
7028	050450	046516			.WORD	04501
7029	050452	000	000		.BYTE	0,0
7030	050454	046776			.WORD	04203
7031	050456	010	000		.BYTE	10,0
7032						
7033	050460	000004		DF17:	.WORD	4
7034	050462	000	000		.BYTE	0,0
7035	050464	047360			.WORD	04606
7036	050466	003	000		.BYTE	3,0
7037	050470	047437			.WORD	04607
7038	050472	000	000		.BYTE	0,0
7039	050474	047360			.WORD	04606
7040	050476	003	000		.BYTE	3,0
7041						
7042	050500	000002		DF21:	.WORD	2
7043	050502	000	000		.BYTE	0,0
7044	050504	047323			.WORD	046051
7045	050506	003	000		.BYTE	3,0
7046						
7047	050510	000001		DF23:	.WORD	1
7048	050512	001	000		.BYTE	1,0
7049						
7050	050514	000001		DF24:	.WORD	1
7051	050516	002	000		.BYTE	2,0
7052						
7053	050520	000001		DF25:	.WORD	1
7054	050522	003	000		.BYTE	3,0
7055						
7056	050524	000002		DF26:	.WORD	2
7057	050526	000	000		.BYTE	0,0
7058	050530	047241			.WORD	046041
7059	050532	003	000		.BYTE	3,0

E11

CZR6LCO RK06L CTG FMTR MACY11 30(1046) 01-DEC-77 14:28 PAGE 135
CZR6LC.P11 01-DEC-77 14:20 TRAP TABLE

SEQ 0134

7060

000001

.END

EM22	043512	984	6461#																	
EM23	043546	990	6466#																	
EM24	043611	996	6472#																	
EM25	043635	1002	6476#																	
EM26	043664	1008	6480#																	
EM27	043710	1014	6484#																	
EM3	042760	894	6397#																	
EM30	043745	1020	1140	6489#																
EM31	044004	1026	1146	6495#																
EM32	044031	1032	6499#																	
EM33	044056	1038	6503#																	
EM34	044104	1044	6507#																	
EM35	044130	1050	6511#																	
EM36	044170	1056	6517#																	
EM37	044227	1062	6523#																	
EM4	043015	900	6402#																	
EM40	044276	1068	6530#																	
EM41	044352	1074	6538#																	
EM42	044420	1080	6545#																	
EM43	044460	1086	6551#																	
EM5	043042	906	6406#																	
EM52	044527	1128	6558#																	
EM53	044561	1134	6563#																	
EM56	044612	1152	1158	6568#																
EM6	043070	912	6410#																	
EM60	044640	1164	6572#																	
EM61	044706	1170	6579#																	
EM62	044760	1176	6586#																	
EM63	045006	1092	1188	6590#																
EM64	045057	1098	1194	6597#																
EM65	045136	1104	1200	6605#																
EM66	045170	1110	1206	6610#																
EM67	045227	1116	1212	6616#																
EM7	043123	918	6415#																	
EM70	045276	1122	1218	6623#																
EM71	045360	1224	6632#																	
EM72	045424	1230	6638#																	
EM73	045477	1236	6646#																	
EM74	045537	1242	6652#																	
EM75	045563	1248	6656#																	
EM76	045617	1254	6661#																	
EM77	045712	1260	1266	6671#																
ERRCNT	002654	1788#	2940#	3842#	3844	3847	3849*	3935*	4032*	4327	4330	4483*								
ERRCOM	006372	1832#	3990#	3991#	4365#	4366*	6857													
ERRFRE	025524	1709	2939	3839#	4282	4488														
ERRHDL	026462	1710	2938	4030#	4281	4297	4487													
ERRLMT	002655	1789#	4327																	
ERRVEC=	000004	764#	2587	2588#	2599*	6279	6280	6284*	6285*	6292*	6293*									
ERZENT	026502	4034#	4196	4221	4304															
ETINUS	006354	1825#	2667	2676	2813	2909	3100													
E.CCLR=	000001	1718#	5143																	
E.CERR=	001000	1727#	5217																	
E.CLAT=	000020	1722#	4869	5033	5074	5098														
E.CMTO=	040000	1729#																		
E.CONT	002604	1712#	3941	3945	3949	3953	3957	3961	3965	3968	3971	3974	3981*	4768*						
		4806*	4869*	4879*	4950*	4972*	4995*	5001*	5033*	5074*	5098*	5143*	5195*	5217*						

INVOSE	014001	2356#	2687															
INVDTK	014100	2367#	2691	2817														
INVDTP	014151	2375#	2697															
IOTVEC=	000020	769#																
IR =	000100	1391#																
IVDCYL	017012	2679	2689#															
IVDDRV	017036	2681	2695#															
IVDMOD	017054	2683	2699#															
IVDOF	017072	2685	2703#															
IVDTP	017044	2664	2697#															
IVDTRK	017020	2677	2691#															
I.AERR	034324	5019	5025	5043	5052#													
I.ATTN	033654	4779	4823	4956#														
I.CCLR	034710	4857	4907	5138#	5210	5570	5635											
I.CSTS	035342	4850	5285#															
I.CSTI	035364	4854	4906	5289#	5559													
I.DRV	002634	1765#	5424	5425														
I.ERR	033076	4851	4855#															
I.ERAA	033056	4850#	4918															
I.ERAC	033064	4788	4841	4853#	4986													
I.HDAL	033322	4834	4897#															
I.INTR	032442	2603	4758#															
I.ISRL	002626	1755#	4772	4775*	4778*	5149*	5446*	5600*	5673*									
I.ISSU	034772	4865	4935	5014	5028	5070	5094	5180#	5329	5336	5343	5350						
I.I00	032722	4776	4793	4820#														
I.RTRN	034672	4770	4808	4818	4846	4858	4863	4866	4872	4881	4890	4893	4908	4910				
		4936	4939	4948	4952	4974	4990	4997	5003	5015	5029	5036	5048	5068				
		5071	5076	5085	5089	5095	5100	5109#										
I.STAT	035446	4862	5067	5326#	5580													
I.STOR	035214	5190	5240#															
I.UNLD	034572	5009	5092#															
KIPARO=	***** U	6275																
LCYL	017312	2673*	2675*	2678	2740#	2977	3093	3107	4565									
LEV2ER=	000400	1840#	3978	4034	4087	4178	4184	4203	4209	4298	4352	4425						
LF =	000012	675#	2557	2560	5805	5811												
MCLK =	000400	1450#																
MDINUS	006344	1821#	2682	2701	2719	2723	2727	2732	2960									
MDS =	001000	1394#	4766	5193	5568													
MERD =	001000	1451#																
MESMSK=	000017	1444#																
MEWD =	002000	1452#																
MIND =	000200	1449#																
MODEQ	013176	2284#	2648															
MSP =	000100	1448#																
M.CADD=	017760	1520#																
M.COIF=	017760	1519#																
M.DRV =	000007	1516#																
M.HEAD=	007000	1523#																
M.ID =	000003	1518#																
M.PAR =	100000	1517#																
M.SECT=	000760	1522#																
M.SER =	077770	1521#																
NED =	010000	1397#	4068	4078	4781	4859	5202	5572										
NEM =	004000	1396#	4063	4796	4962	5202												
NEXTQ	013667	2342#	2773	2827														
NOCHK =	000400	1583#	5436	5498	5539	5566	5584	5605	5628	5637	5671							

NODSC = 004000	1588#	4268	5024											
NONE 015774	2540#	4629	4639											
NPRVD 013475	2320#	2856												
NXF = 000004	1408#	4122												
OCTBIN 037636	2779	2798	2810	2883	5702#									
OFFSET= 000115	1336#	4474	5481											
OFINUS 006362	1828#	2684	2705	2716	2902	4475								
OFSET0 013320	2299#	2658												
OFST = 000004	1428#													
OPCOMP 002651	1785#	3097#	3259#	3278	3416#	3439								
OPCONT 002656	1790#	2707#	2718#	2721*	2725*	2729*	2734*	2737*	2749	2853*	3000	3002	3023	
	3027	3066	3071	3074	3425	3429	4469							
OPI = 020000	1420#	4172	4802	4968	5204									
OPIERR= 000010	1835#	3324	3570	4174	4320									
OR = 000200	1392#													
OREQSW= 000020	1795#	2718	4469											
O.WAIT 002606	1732#	4702	4706*	4816*	4820	4836*	4902*	4915*	4930*	5474*	5534*	5596*	5613*	
PACK = 000103	1331#	2956												
PARAM3 016550	2648#	2702												
PARAM5 016600	2654#	2694												
PARM0 002362	1621#	1776	2947	4280	4300	4347	4458							
PARM1 002446	1653#	2942	4036	4293	4442									
PARM2 016502	2639#													
PARM3 016522	2640	2643#												
PARM4 016530	2642	2644#												
PARM6 016620	2658#	2706												
PAT = 000020	1446#													
PBLKT 002644	1776#	4699	4786	4812	4980	5005	5423*							
PBSVAL= 001000	1584#	3936	5361											
PCA = 004000	1453#													
PCD = 010000	1454#													
PGE = 002000	1395#	4118	5202											
PIP = 020000	1438#													
PIRQ = 177772	681#													
PIRQVE= 000240	775#													
POSOFF 031262	3449	3550	3620	3634	3718	3742	4468#							
PREPLP 031336	3275	3436	4483#											
PRGRS1 015300	2482#	3069												
PRGRS2 015345	2489#	3073												
PROGID 012770	2259#	2609												
PRVD 013461	2317#	2854												
PRO = 000000	698#													
PR1 = 000040	699#													
PR2 = 000100	700#													
PR3 = 000140	701#													
PR4 = 000200	702#													
PR5 = 000240	703#	1708												
PR6 = 000300	704#													
PR7 = 000340	705#	2604												
PS = 177776	678#	679	4688	4694*	4708*	5417	5418*	5677*						
PSDBSF= 000040	1796#	2853	3002	3027										
PSDQ 013330	2301#	2842												
PSW = 177776	679#													
PWRVEC= 000024	770#	2581*	2582*	6230*	6231*	6240*	6246*	6258*	6259*					
P.ASOF= 000032	1605#	4007	5030*	5031	5064*	5295*	5483*	5484						
P.A00 = 000040	1608#	2962	2969	4010	5021*	5022	5038*	5039	5352*	5560*				

S.ACLO=	000100	1475#						
S.BRHM=	000100	1490#						
S.BRKE=	040000	1512#						
S.CART=	000400	1492#						
S.DCLO=	010000	1482#						
S.DIB =	002000	1508#						
S.DOOR=	000200	1491#						
S.DRA =	000040	1461#						
S.DROT=	020000	1483#						
S.DRY =	000200	1463#						
S.DSC =	040000	1470#	4874	5022	5039	5102		
S.FLT =	000200	1476#	5017					
S.FORM=	001000	1465#						
S.FWD =	002000	1494#						
S.HDFL=	000200	1505#						
S.HDMM=	000040	1489#						
S.ICYL=	000040	1474#						
S.ILF =	000400	1477#						
S.LIND=	020000	1511#						
S.LOAD=	010000	1496#						
S.MHD =	000400	1506#						
S.NMOV=	010000	1510#						
S.OFF =	002000	1466#						
S.PAR =	001000	1478#	4877	5354				
S.PIP =	020000	1469#	4885	5080				
S.PLO =	004000	1509#						
S.REV =	004000	1495#						
S.RTZ =	020000	1497#						
S.SECT=	000020	1502#						
S.SKI =	002000	1479#						
S.SPIN=	010000	1468#						
S.SPLS=	010000	1481#						
S.SPOK=	001000	1493#						
S.TYPE=	000400	1464#						
S.UNLD=	040000	1498#						
S.UNS =	040000	1484#						
S.VV =	000100	1462#						
S.WCLK=	000040	1503#						
S.WGAT=	000100	1504#						
S.WLE =	004000	1480#						
S.WRL =	004000	1467#	2962	2969				
S.XDOK=	000020	1488#						
S.XERR=	001000	1507#						
TBITVE=	000014	766#						
TDMFAC	014662	2434#	3058					
TDMSOF	014511	2416#	3033	3044				
TDMSUF	014576	2425#	3034	3045	3059			
TKVEC =	000060	773#	5992*	5993*				
TKWDCT	006306	1806#	2712*	2714*	2715*	3309	3547	3717
TOPROC	030560	3940	4086	4361#				
TOTMES	016057	2551#	4637					
TPINUS	006336	1818#	2636	2639	2660	2662	2670	
TPQ =	013122	2275#	2637					
TPVEC =	000064	774#						
TRAPVE=	000034	772#	2579*	2580*				
TRKINC	021240	3091#	3277	3438				

\$GDOAT	001124	809#																		
\$GTSWR	041306	6076#	6380																	
\$HD =	000001	653	654																	
\$HIOCT	037770	5727*	5740#																	
\$ICNT	001104	800#																		
\$ILLUP	042400	6230	6246	6263#																
\$INTAG	001135	814#	6073*	6092	6112	6225														
\$ITEMB	001114	804#	3865	5952*	5970															
\$LF	001270	863#	5811	5970	6208	6218														
\$LPADR	001106	801#																		
\$LPERR	001110	802#	2937*	3260*	3413*	5964														
\$LSTAD	042512	6294#	6298#																	
\$MAIL =	***** U	2601	5764	5958																
\$MNEW	042226	6079	6223#																	
\$MSWR	042215	6076	6221#																	
\$NULL	001154	822#	5782	5811																
\$OCNT	040554	5882*	5911*	5924#																
\$OCTVL	040314	5824	5849#																	
\$OMODE	040556	5877*	5881*	5886	5889*	5900*	5926#													
\$PASS	001100	797#																		
\$POWER	042406	6261	6266#																	
\$PWRDN	042240	2581	6230#	6258																
\$PWRMG	042374	6261#																		
\$PWRUP	042312	6240	6246#																	
\$QUES	001266	861#	5811	5970	6130	6201	6218													
\$RDCHR	041560	6143#	6383																	
\$RDDEC =	***** U	6385																		
\$RDLIN	041650	6166#	6384																	
\$RDOCT =	***** U	6385																		
\$RDSZ =	000072	6159#																		
\$REGAD	001160	826#																		
\$REG0	001162	828#	3992	4364	6857															
\$REG1	001164	829#	6857																	
\$REG10	001202	836#	3394*	3484*	3610*	3697*	3783*	4188*	4213*	4245*	6860	6868								
\$REG11	001204	837#	3392*	3485*	3608*	3698*	3781*	4189*	4214*	4246*	6860	6868								
\$REG12	001206	838#	3393*	3609*	3699*	3782*	4190*	4215*	4239*	4241*	6860	6868								
\$REG13	001210	839#	6860																	
\$REG14	001212	840#	6863																	
\$REG15	001214	841#	6863																	
\$REG16	001216	842#	6864																	
\$REG17	001220	843#	6864																	
\$REG2	001166	830#	6857																	
\$REG20	001222	844#	6864																	
\$REG21	001224	845#	6864																	
\$REG22	001226	846#	6864																	
\$REG23	001230	847#	6864																	
\$REG24	001232	848#	6864																	
\$REG25	001234	849#	6864																	
\$REG3	001170	831#	6857																	
\$REG4	001172	832#	6857																	
\$REG5	001174	833#	3349*	3350*	3355*	3365*	3366*	3531	3654*	3655*	3665*	3666*	3682*	3755*						
		3756*	3760*	3790*	3791*	3846*	3847*	4329*	4330*	4497*	4517*	4526*	4540	4543						
		6860	6867																	
\$REG6	001176	834#	3356*	3683*	3761*	4498*	4501*	4512	4514*	4516*	4533*	4534*	4535*	4536*						
		4539*	4541	4542	6860	6867														
\$REG7	001200	835#	3357*	3684*	3762*	3765*	4499*	4500*	4502	4505*	4509*	4511*	4540*	4542*						

. SETUP	6428	2568	
. SWRHI	6428	654	
. SWRLO	6638	664	665
. SCATC	6428	776	
. SCMTA	6428	789	
. SDB20	6428	5811	
. SERRO	6428	5927	
. SPOWE	6428	6226	
. SRAND	6428		
. SREAD	6428	5970	
. SSAVE	6428	6299	
. SSIZE	6428	6269	
. STRAP	6428	6344	
. STYPD	6428		
. STYPE	6428	5741	
. STYPO	6428	5850	

. ABS. 050534 000

ERRORS DETECTED: 0

RM03:CZR6LC, RM03:CZR6LC, SEQ/SOL/CRF/NL: TOC/DOC/EQ: QNEWSW=RM03:DRIV10.P11, RM03:CZR6LC.P11
RUN-TIME: 25 20 2 SECONDS
RUN-TIME RATIO: 1224/47=25.6
CORE USED: 42K (83 PAGES)

DOCUMENT PAGES: 152