

RM03

RM03 FORMATTER
MD-11-DZRMA-B

EP-DZRMA-B-DL-A

OCT 1977

COPYRIGHT © 1977

digital

FICHE 1 OF 1

MADE IN USA

B01

EOF12TMLBSEQ

00010000

770804

PDP10 411

DOHDR1DZRMABSEQ

00010000

770804

001

MAINDEC-11-DZMA, RM03 FORMATTER
DZMA-B.P11 21-JUL-77 15:32

MACY11 30(1046) 21-JUL-77 16:06 PAGE 2

SEQ 0001

.REM %

IDENTIFICATION

PRODUCT CODE: MAINDEC-11-DZMA-B-D

PRODUCT NAME: RM03 FORMATTER

DATE CREATED: AUGUST 1977

MAINTAINER: DIAGNOSTIC ENGINEERING

AUTHOR: C. HESS/C. CHEN

COPYRIGHT (C) 1977 DIGITAL EQUIPMENT CORP., MAYNARD, MASS.

THE INFORMATION IN THIS STATEMENT IS SUBJECT TO CHANGE
WITHOUT NOTICE AND SHOULD NOT BE CONSTRUED AS A
COMMITMENT BY DIGITAL EQUIPMENT CORPORATION.

ACTUAL DISTRIBUTION OF THE SOFTWARE DESCRIBED IN THIS
DOCUMENT WILL BE SUBJECT TO TERMS AND CONDITIONS TO
BE ANNOUNCED ON SOME FUTURE DATE BY DIGITAL EQUIPMENT
CORPORATION.

DEC ASSUMES NO RESPONSIBILITY FOR THE USE OR
RELIABILITY OF ITS SOFTWARE ON EQUIPMENT WHICH IS
NOT SUPPLIED BY DEC.

THIS SOFTWARE IS FURNISHED TO PURCHASER UNDER A LICENSE
TO USE ON A SINGLE COMPUTER SYSTEM AND CAN BE COPIED (WITH
INCLUSION OF DEC'S COPYRIGHT NOTICE) ONLY FOR USE IN SUCH
SYSTEM, EXCEPT AS MAY OTHERWISE BE PROVIDED IN WRITING
BY DEC.

1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48

CONTENTS

1. ABSTRACT
2. REQUIREMENTS
 - 2.1 EQUIPMENT
 - 2.2 PRELIMINARY PROGRAMS
3. LOADING PROCEDURES
 - 3.1 PAPER TAPE AND XXDP
 - 3.2 APT
 - 3.3 APT ETABLE DEFINITIONS
4. STARTING PROCEDURES
 - 4.1 STARTING ADDRESSES
 - 4.2 OPERATOR ACTION
 - 4.3 RH11 - RH70 UNIBUS ADDRESS
 - 4.4 OTHER UNIBUS ADDRESSES
5. SWITCH REGISTER SETTINGS
6. ERROR MESSAGES
7. MISCELLANEOUS
 - 7.1 FORMAT TIMES
 - 7.2 HALTING THE PROGRAM
 - 7.3 SURFACE VERIFICATION
8. PROGRAM DESCRIPTION
 - 8.1 FORMAT OPERATION
 - 8.2 CHECK OPERATION
 - 8.3 POSITIONER VERIFICATION
9. BAD SPOT FILE
10. RMO3 SOFTWARE DRIVER DOCUMENT

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

1. ABSTRACT

THE RMO3 FORMATTER PROGRAM PROVIDES THE FACILITIES TO FORMAT OR TO CHECK THE HEADER AND DATA FIELDS OF EACH DATA BLOCK ON THE DISC PACK. EACH HEADER FIELD SPECIFIES THE ADDRESS OF ITS ASSOCIATED DATA BLOCK ON THE DISK PACK.

IN THE FORMAT OPERATION, THE PROGRAM WRITES THE HEADER OF EACH DATA BLOCK WITH A CYLINDER NUMBER, TRACK NUMBER AND SECTOR NUMBER; WRITES THE DATA FIELD WITH SELECTED DATA PATTERN. THE PROGRAM THEN VERIFIES THE WRITTEN DATA BLOCKS VIA EXECUTING THE "WRITE CHECK HEAD AND DATA" COMMAND.

IN THE CHECK OPERATION, THE PROGRAM REPEATS THE FORMAT OPERATION THREE TIMES WITH DATA PATTERN BEING ROTATED ONE BIT AT EACH PASS.

2. REQUIREMENTS

2.1 EQUIPMENT

PDP-11 PROCESSOR
8K MEMORY
TELETYPE
PROGRAM LOAD DEVICE
KW11-L OR KW11-P CLOCK
RH11 OR RH70 WITH 1 - 8 RMO3 DISK DRIVES
DISK PACK(S)

2.2 PRELIMINARY PROGRAMS

THERE ARE NO PREREQUISITE PROGRAMS PROVIDING THE RMO3 SUBSYSTEM IS KNOWN TO BE OPERATIONAL. IF THE STATE OF THE RMO3 SUBSYSTEM IS UNKNOWN, THE FOLLOWING PROGRAMS SHOULD BE RUN PRIOR TO USING THE FORMATTER PROGRAM

RMO3	DISKLESS DIAGNOSTIC
RMO3	FUNCTIONAL TEST

3. LOADING PROCEDURES

3.1 PAPER TAPE AND XXDP

THE PROGRAM MAY BE LOADED FROM PAPER TAPE USING THE ABSOLUTE LOADER OR IT MAY BE LOADED FROM 'XXDP' MEDIA USING THE APPROPRIATE LOADER.

3.2 APT

94
95
96
97
98
99
100
101
102
103
104
105
106
107
108
109
110
111
112
113
114
115
116
117
118
119
120
121
122
123
124
125
126
127
128
129
130
131
132
133
134
135
136
137
138
139
140
141
142
143
144
145
146
147
148
149

150
151
152
153
154
155
156
157
158
159
160
161
162
163
164
165
166
167
168
169
170
171
172
173
174
175
176
177
178
179
180
181
182
183
184
185
186
187
188
189
190
191
192
193
194
195
196
197
198
199
200
201
202
203
204
205

THIS PROGRAM IS APT COMPATIBLE TO THE EXTENT THAT APT HOOKS WILL BE IN THE PROGRAM AND WILL WORK THRU THE 'OPTION INTERFACE'.

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

AUTOMATIC MODE (MONITOR)

1. THE INPUT DIALOGUE IS BYPASSED.
2. THE BUSS ADDRESS AND CONTROLLER INTERRUPT VECTOR IS DEFAULTED.

DUMP MODE: INPUT DIALOGUE AFTER PROGRAM STARTS

3.3 APT ETABLE DEFINITIONS

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

1. SOFTWARE ENVIRONMENT:

= 1 IF APT SCRIPT MODE
= 0 IF STANDLONE MODE

2. ENVIRONMENT MODE:

BIT 7 = 1 ETABLE DOES SIZING
= 0 PROGRAM DOES SIZING

BIT 6 = 1 SPOOL MESSAGES TO APT IF SCRIPT MODE
= 0 DON'T SPOOL TO APT

BIT 5 = 1 SUPPRESS CONSOLE OUTPUT
= 0 ALLOW CONSOLE OUTPUT

BIT 4 TO BIT 0 ARE NOT USED

- #### 3. SWITCH 1 (SOFTWARE SWITCH REGISTER)
- IF ENVIRONMENT MODE BIT 7 (SIZING BIT) IS SET TO 1, THE SOFTWARE SWITCH REGISTER WILL BE USED, INSTEAD OF THE HARDWARE CONSOLE SWITCH REGISTER.

- #### 4. SWITCH 2 (USER SWITCH REGISTER)
- NOT USED

- #### 5. CPU OPTIONS
- NOT USED

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

- #### 7. INTERRUPT VECTOR 1:
- USED WHEN ENVIRONMENT MODE BIT 7 = 1; DEFAULT = 254

- #### 8. BUS PRIORITY 1:
- NOT USED.

206
207
208
209
210
211
212
213
214
215
216
217
218
219
220
221
222
223
224
225
226
227
228
229
230
231
232
233
234
235
236
237
238
239
240
241
242
243
244
245
246
247
248
249
250
251
252
253
254
255
256
257
258
259
260
261

9. INTERRUPT VECTOR 2:
NOT USED

10. BUS PRIORITY 2:
NOT USED

11. BASE ADDRESS:
USED WHEN ENVIRONMENT MODE BIT 7 = 1; DEFAULT = 176700

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

13. CONTROLLER DESCRIPTOR WORDS:
USED WHEN ENVIRONMENT MODE BIT 7 = 1. ANY NONE ZERO
NUMBER SPECIFIES THAT PROGRAM RUNS IN CHECK MODE.

14. CONTROLLER DESCRIPTOR WORDS:
USED WHEN ENVIRONMENT MODE BIT 7 = 1. ANY NONE ZERO
NUMBER SPECIFIES THAT PROGRAM SELECTS 30 SECTORS
FOR A TRACK IN ALL OPERATIONS.

4. STARTING PROCEDURES

4.1 STARTING ADDRESSES

THE PROGRAM IS STARTED FROM LOCATION 200(8) IF THE ADDRESS OF THE
RH11 OR RH70 WILL NOT BE CHANGED FROM ITS DEFAULT VALUE OF 176700.

THE PROGRAM IS STARTED FROM LOCATION 204(8) IF THE ADDRESS OF
THE RH11 OR RH70 IS TO BE CHANGED FROM THE PRELOADED VALUE.
NOTE THAT STARTING ADDRESS 204 IS VALID ONLY ONCE AFTER THE
PROGRAM IS LOADED.
(SEE SECTION 4.3)

4.2 OPERATION ACTION

1. LOAD THE PROGRAM INTO MEMORY (SEE SECTION 3).
2. LOAD THE STARTING ADDRESS - 200(8) OR 204(8).
3. SET THE SWITCHES AS REQUIRED AND PRESS 'START'.

IF THIS IS THE PROGRAM'S FIRST START, THE STATUS OF THE DRIVES
ON THE SELECTED MASSBUS SUBSYSTEM WILL BE TYPED OUT. THIS TYPEOUT
MAY BE INHIBITED ON SUBSEQUENT STARTS BY SETTING SW<02>.

4. THE PROGRAM WILL THEN TYPE THE FOLLOWING MESSAGE:

'PROGRAM MODE (C OR F):'

H01

MAINDEC-11-DZMA, RMO3 FORMATTER
DZRMAB.P11 21-JUL-77 15:32

MACY11 30(1046) 21-JUL-77 16:06 PAGE 7

SEQ 0006

262
263
264
265
266
267
268
269
270
271
272
273
274
275
276
277
278
279
280
281
282
283
284
285
286
287
288
289
290
291
292
293
294
295
296
297
298
299
300
301
302
303
304
305
306
307
308
309
310
311
312
313
314
315
316
317

ENTER THE APPROPRIATE CODE: 'C' FOR 'CHECK' OPERATION OR 'F' FOR 'FORMAT & VERIFY'. IF A 'CARRIAGE RETURN' IS ENTERED IN RESPONSE TO THE REQUEST, THE PROGRAM WILL ASSUME 'FORMAT & VERIFY'.

5. THE PROGRAM WILL THEN ASK FOR THE FORMATTING MODE:

'OPERATE IN 32 SECTOR (16 BIT) MODE (Y OR N)'

ENTER THE APPROPRIATE CHARACTER: 'Y' FOR 16 BIT MODE OR 'N' FOR 18 BIT MODE. IF A 'CARRIAGE RETURN' IS ENTERED IN RESPONSE TO THIS REQUEST, THE PROGRAM WILL ASSUME 16 BIT MODE.

6. THE PROGRAM WILL THEN ASK FOR A DRIVE:

'DRIVE: '

ENTER THE ADDRESS OF THE DRIVE TO BE FORMATTED. A 'PERIOD' OR 'CARRIAGE RETURN' ENTRY WILL SELECT DRIVE 0. IF THE DRIVE SELECTED IS NOT AVAILABLE, THE PROGRAM WILL TYPE AN ERROR MESSAGE AND RETURN TO THE DRIVE ADDRESS REQUEST.

7. AFTER THE OPERATOR HAS SELECTED A DRIVE, THE PROGRAM WILL ASK FOR ADDRESS LIMITS FOR THE SELECTED DRIVE:

'ENTER ADDRESS LIMITS: '

THE PREVIOUSLY SELECTED OR DEFAULT VALUES FOR BEGINNING CYLINDER AND TRACK AND FOR ENDING CYLINDER AND TRACK WILL BE TYPED OUT. IF A 'CARRIAGE RETURN' IS TYPED AS A RESPONSE, THE PRESENT VALUE WILL BE USED. IF A 'PERIOD' IS TYPED, THE PROGRAM WILL BYPASS THE REMAINING ENTRIES AND WILL USE THEIR PRESENT VALUES. NOTE THAT THE CYLINDER AND TRACK VALUES ARE DECIMAL NUMBERS. THE ADDRESS SPECIFIED BY THE BEGINNING TRACK MUST BE LESS THAN THE ENDING TRACK ADDRESS IF THESE TWO ADDRESSES ARE ON THE SAME CYLINDER. ON THE OTHER HAND, THE ENDING CYLINDER ADDRESS MAY BE LESS THAN THE STARTING CYLINDER ADDRESS. IN THIS CASE, THE PROGRAM WILL FORMAT OR VERIFY THE DISK PACK FROM THE STARTING CYLINDER TO CYLINDER 822 AND THEN FROM CYLINDER 0 TO THE ENDING CYLINDER.

8. THE PROGRAM WILL THEN ASK FOR THE DATA PATTERN:
(IN CASE OF FORMAT MODE)

'SELECT DATA PATTERN
(0) ZERO'S
(1) ONE'S
(2) WORST CASE:'

ENTER THE CODE FOR THE REQUIRED PATTERN. 'CARRIAGE RETURN' OR 'PERIOD' ENTRIES WILL CAUSE THE PROGRAM TO USE THE 'WORST CASE'

PATTERN.

THE 'WORST CASE' PATTERN IS THE FOLLOWING OCTAL SEQUENCE REPEATED THROUGH THE DATA AREA OF THE SECTOR:

066666

NOTE: IN THE CHECK OPERATION, THE ABOVE MENTIONED MESSAGE IS BYPASSED. THE DATA PATTERN IS INITIATED TO 133331 AND IS ROTATED ONE BIT TO THE RIGHT AT EACH PASS OF THE CHECK OPERATION. AFTER THE CHECK OPERATION IS DONE, THE DATA PATTERN WILL BE RESTORED TO 066666.

9. THE PROGRAM WILL THEN TYPE:

'STARTING FORMAT ON DRIVE N'

OR

'STARTING CHECK ON DRIVE N'

IF A NEW PACK IS UNDER FORMATTED, THE PROGRAM ALSO ASKS TO ENTER TWO SERIAL NUMBERS. THESE TWO SERIAL NUMBER MUST BE NON-ZERO DECIMAL NUMBERS.

10. THE OPERATOR CAN DETERMINE WHERE THE DRIVE IS DURING THE FORMAT OR CHECK OPERATION BY TYPING A 'CONTROL 0'. THE PROGRAM WILL TYPE THE FOLLOWING MESSAGE:

'PRESENT ADDRESS IS: CXXX TXX'

IF A 'CONTROL C' IS TYPED WHILE THE PROGRAM IS TYPING THE CURRENT ADDRESS, THE PROGRAM WILL ABORT THE FORMAT (OR CHECK) OPERATION AND RETURN TO THE 'DRIVE' REQUEST.

11. IF THE OPERATOR DOES NOT SELECT A DIFFERENT DRIVE WHEN THE FORMAT OR CHECK OPERATION HAS COMPLETED, THE PROGRAM WILL NOT ALTER THE ADDRESS LIMITS SPECIFIED AT THE START OF THE PREVIOUS OPERATION. IF THE FORMAT OR CHECK OPERATION IS TERMINATED BY A 'CONTROL C' OR IF A DIFFERENT DRIVE IS SELECTED, THE PROGRAM WILL RESET THE ADDRESS LIMITS TO THE VALUES APPROPRIATE FOR THE DRIVE TYPE.

12. TO CHANGE EITHER THE ADDRESSING MODE OR THE OPERATION MODE, THE PROGRAM MUST BE STARTED FROM LOCATION 200(8) OR 204(8) AGAIN.

4.3 RH11 - RH70 UNIBUS ADDRESS

THE PROGRAM ASSUMES THAT THE RH11 OR RH70 ADDRESSES START AT 176700 AND THAT THE VECTOR ADDRESS IS 254. THESE ADDRESSES MAY BE CHANGED WHEN THE PROGRAM IS STARTED FROM LOCATION 204(8).

318
319
320
321
322
323
324
325
326
327
328
329
330
331
332
333
334
335
336
337
338
339
340
341
342
343
344
345
346
347
348
349
350
351
352
353
354
355
356
357
358
359
360
361
362
363
364
365
366
367
368
369
370
371
372
373

IF THE RH11 - RH70 IS NOT AT THE DEFAULT ADDRESS, THE PROGRAM MUST BE STARTED FROM 204(8) INITIALLY AS THE PROGRAM GOES THROUGH THE ADDRESS CHANGE ROUTINE AT INITIAL START ONLY. ENTER THE RH11 RH70 ADDRESS IN RESPONSE TO THE REQUEST FROM THE PROGRAM.

4.4 OTHER UNIBUS ADDRESSES

LOC	TAG	CONTENTS	FUNCTION
---	---	-----	-----
1160	\$TKS	177560	TTY KEYBOARD STATUS REGISTER
1162	\$TKB	177562	TTY KEYBOARD BUFFER REGISTER
1164	\$TPS	177564	TTY PRINTER STATUS REGISTER
1166	\$TPB	177566	TTY PRINTER BUFFER REGISTER
1276	\$LKCSR	172540	KW11-P CONTROL REGISTER
1300	\$LKCSB	172542	KW11-P COUNTER REGISTER
1302	\$LPVEC	104	KW11-P VECTOR ADDRESS
1304	\$LKS	177546	KW11-L CONTROL REGISTER
1306	\$LKV	100	;ADDRESS OF KW11-L VECTOR

5. SWITCH REGISTER SETTINGS

SW<15>=1...HALT ON ERROR
SW<13>=1...INHIBIT ERROR TYPEOUTS
SW<10>=1...BELL ON ERROR
SW<09>=1...LOOP ON ERROR
SW<02>=1...DON'T DISPLAY SYSTEM STATUS AFTER INITIAL START
SW<01>=1...LOOP ON THE CURRENT TRACK
SW<00>=1...LOOP THE PROGRAM ON THE SELECTED DRIVE

IF THE PROGRAM IS BEING RUN ON A SWITCHLESS PROCESSOR (I.E. AN 11/34) THE PROGRAM WILL DETERMINE THAT THE HARDWARE SWITCH REGISTER IS NOT PRESENT AND WILL USE A 'SOFTWARE' SWITCH REGISTER. THE 'SOFTWARE' SWITCH REGISTER IS LOCATED AT LOCATION 176 (8). THE SETTINGS OF THE 'SOFTWARE' SWITCHES ARE CONTROLLED THROUGH A KEYBOARD ROUTINE WHICH IS CALLED BY TYPING A 'CONTROL G'. THE PROGRAM WILL RECOGNIZE THE 'CONTROL G' AT ANY TIME EXCEPT WHEN THE PROGRAM IS AT A HIGHER PRIORITY PROCESSING AN RMO3 INTERRUPT. THE 'SOFTWARE' SWITCH VALUES ARE ENTERED AS AN OCTAL NUMBER IN RESPONSE TO THE PROMPT FROM THE SWITCH ENTRY ROUTINE:

'SWR = NNNNNN NEW ='

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

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

K01

MAINDEC-11-DZRNA, RMO3 FORMATTER
DZRNAB.P11 21-JUL-77 15:32

MACY11 30(1046) 21-JUL-77 16:06 PAGE 10

SEQ 0009

BE FOLLOWED.

6. ERROR MESSAGES

1. 'RH11 INTERRUPT OCCURRED (RMAS=0) - AN INTERRUPT OCCURRED, BUT NOTHING ON THE MASSBUS IS INDICATING AN ATTENTION.
2. 'UNEXPECTED ATTENTION OCCURRED' - THE INDICATED DRIVE INTERRUPTED, BUT NO INTERRUPT WAS EXPECTED FROM THE INDICATED DRIVE.
3. 'MASSBUS PARITY ERROR (MCPE=1)' - A CONTROL BUS PARITY ERROR WAS DETECTED BY THE RH11 WHEN THE INDICATED REGISTER WAS READ.
4. 'MASSBUS PARITY ERROR (PAR=1)' - A CONTROL BUS PARITY ERROR OCCURRED WHEN THE INDICATED REGISTER WAS WRITTEN.
5. 'ADDRESS PLUG CHANGE BIT SET' - THE PROGRAM FOUND THE 'OPE' BIT SET FOR THE INDICATED DRIVE.
6. 'RH11 DIDN'T RESPOND TO ADDRESSING' - THE PROGRAM ACCESSED THE RH11 AT THE INDICATED ADDRESS AND RECEIVED NO RESPONSE.
7. 'DRIVE OFFLINE' - THE INDICATED DRIVE HAS GONE OFFLINE
8. 'PERSISTENT DRIVE UNSAFE ERROR' - THE INDICATED DRIVE HAS BECOME UNSAFE AND THE CONDITION CANNOT BE CLEARED BY ISSUING 'DRIVE CLEAR' INSTRUCTIONS.
9. 'UNCORRECTABLE MASSBUS PARITY ERROR' - THE PROGRAM ATTEMPTED TO PERFORM AN OPERATION AND DETECTED 3 SUCESSIVE MASSBUS PARITY ERRORS OR THE PROGRAM ATTEMPTED TO CLEAR A 'PAR' ERROR IN THE DRIVE AND A PARITY ERROR OCCURRED.
10. 'SOFTWARE TIMEOUT' - THE OPERATION FAILED TO COMPLETE WITHIN 1 SECOND.
11. 'DRIVE UNSAFE ERROR' - A NON-PERSISTENT UNSAFE ERROR OCCURRED DURING THE OPERATION.
12. 'CONTROLLER/DRIVE ERROR DURING WRITE' - THE INDICATED NON-DATA ERROR WAS DETECTED DURING A FORMAT OPERATION.
13. 'CONTROLLER/DRIVE ERROR DURING WRITE CHECK' - A NON-DATA ERROR OCCURRED DURING THE WRITE CHECK.
14. 'DATA ERROR DURING WRITE CHECK' - A DATA RELATED ERROR OCCURRED DURING A WRITE CHECK OPERATION. A DATA ERROR IS CONSIDERED TO BE ONE OF THE FOLLOWING ERRORS: 'DCK', 'OPI', 'DTE', 'HCRC', 'HCE', OR 'FER'.
15. 'RETRIES NOT SUCESSFUL - SECTOR NOT ACCEPTABLE' - THE INDICATED SECTOR FAILED DURING RETRY FOLLOWING A DATA ERROR.

- 486
487
488
489
490
491
492
493
494
495
496
497
498
499
500
501
502
503
504
505
506
507
508
509
510
511
512
513
514
515
516
517
518
519
520
521
522
523
524
525
526
527
528
529
530
531
532
533
534
535
536
537
538
539
540
541
16. 'CONTROLLER/DRIVE ERROR VERIFYING HEADERS' - AN ERROR OCCURRED DURING THE VERIFICATION PASS AFTER FORMATTING.
 17. ''HCE' ERROR VERIFYING HEADERS' - A HEADER ERROR OCCURRED DURING THE VERIFICATION PASS AFTER FORMATTING.
 18. 'CYLINDER FIELD IN HEADER IS NOT CORRECT' - THE CYLINDER FIELD FROM A HEADER READ DURING THE VERIFICATION PASS IS NOT CORRECT.
 19. 'WRITE CHECK ERROR' - THE RH11 REPORTED A WRITE CHECK ERROR AND NO DRIVE ERRORS WERE SET.

7. MISCELLANEOUS

7.1 FORMAT TIME

IT TAKES APPROXIMATELY 10 MINUTES TO FORMAT AN ENTIRE RMD3 PACK.
THE 'CHECK' MODE TIME FOR AN ENTIRE RMD3 PACK IS 24 MINUTES.

7.2 HALTING THE PROGRAM

THE OPERATOR SHOULD NOT HALT THE PROGRAM DURING A FORMAT OPERATION.
HALTING THE PROGRAM MAY LEAVE A SECTOR INCORRECTLY FORMATTED. TO
TERMINATE THE FORMAT, TYPE A 'CONTROL O' AND WHILE THE PROGRAM IS
TYPING THE ADDRESS, TYPE ANOTHER 'CONTROL C'; THIS SEQUENCE RETURNS
THE PROGRAM TO THE DRIVE ADDRESS ENTRY ROUTINE.

7.3 SURFACE VERIFICATION

THE FORMATTER PROGRAM IS NOT INTENDED TO BE USED TO PERFORM DISK
PACK VERIFICATION. IF THE PROGRAM REPORTS A SECTOR AS BEING 'NOT
ACCEPTABLE', THIS MAY IN FACT INDICATE A BAD SURFACE; HOWEVER,
SECTORS WHICH 'PASSED' MAY OR MAY NOT BE GOOD.

8. PROGRAM DESCRIPTION

8.1 FORMAT OPERATION

THE PROGRAM FORMATS THE PACK ONE TRACK AT A TIME BETWEEN THE LIMITS
SPECIFIED BY THE OPERATOR.

THE FORMAT OPERATION CONSISTS OF A WRITE HEADER AND DATA COMMAND FOR
THE ENTIRE TRACK FOLLOWED BY A WRITE CHECK HEADER AND DATA COMMAND.
IF AN ERROR OCCURS DURING THE WRITE, THE PROGRAM WILL RETRY THE
WRITE BEFORE CONTINUING. IF A DATA RELATED ERROR IS DETECTED
DURING THE WRITE CHECK (A DATA ERROR IS DEFINED AS ONE OF

THE FOLLOWING ERRORS: 'DCK', 'OPI', 'DTE', 'HCRC', 'HCE' OR 'FER'), THE SECTOR IN ERROR WILL BE REWRITTEN. THE PROGRAM WILL CHECK THE SECTOR TWICE; IF A DATA ERROR IS DETECTED IN THE SECTOR DURING EITHER OF THE WRITE CHECKS, THE PROGRAM WILL DECLARE THE SECTOR AS BEING 'NONRECOVERABLE'. FOLLOWING THIS SEQUENCE, THE REMAINDER OF THE TRACK IS CHECKED. IF DATA ERRORS ARE ENCOUNTERED IN ANY OF THE REMAINING SECTORS, EACH SECTOR WILL BE HANDLED AS DESCRIBED ABOVE.

IF A NON-DATA RELATED ERROR OCCURS DURING THE WRITE CHECK, THE PROGRAM WILL RETRY THE COMMAND FOR THE ENTIRE TRACK BEFORE PROCEEDING TO THE NEXT TRACK.

IT SHOULD BE NOTED THAT THE FORMATTER PROGRAM FORMATS THE PACK ONLY AND IS NOT DIRECTLY VERIFYING THE SURFACE OF THE PACK. ERRORS THAT ARE ENCOUNTERED MAY BE THE RESULT OF BAD AREAS ON THE PACK BUT, AS THE PROGRAM IS NOT DESIGNED TO PERFORM AN EXHAUSTIVE CHECK OF THE DISK PACK SURFACE, THE PROGRAM CANNOT MAKE THIS CONCLUSION. IN GENERAL, HOWEVER, THE OPERATOR CAN ASSUME THAT SECTORS WHICH THE PROGRAM CALLS 'UNACCEPTABLE' INDICATE BAD AREAS OF THE PACK; UNFORTUNATELY, SECTORS WHICH 'PASS' CANNOT BE ASSUMED TO BE GOOD.

DURING THE FORMAT OPERATION, THE PROGRAM FILLS THE DATA FIELD WITH THE PATTERN SELECTED BY THE OPERATOR.

8.2 CHECK OPERATION

THE CHECK OPERATION IS IDENTICAL TO THE THE FORMAT OPERATION DESCRIBED IN SECTION 8.1, EXCEPT THAT IN EACH CHECK OPERATION THE FORMAT OPERATION IS REPEATED THREE TIMES WITH DATA PATTERN BEING ROTATED ONE BIT POSITION AT EACH PASS.

8.3 POSITIONER VERIFICATION

AFTER THE PROGRAM COMPLETES THE FORMAT OPERATION, THE POSITIONER IS RETURNED TO THE STARTING CYLINDER AND THE HEADER FROM SECTOR 0 ON THE STARTING TRACK IS READ. THE CYLINDER ADDRESS FIELD FROM THE HEADER IS COMPARED TO THE REQUESTED CYLINDER; IF THE CYLINDER ADDRESSES DO NOT COMPARE, AN ERROR MESSAGE IS TYPED. THE PROGRAM CHECKS THE CYLINDER ADDRESS FIELD FROM THE HEADER OF SECTOR 0 ON THE BEGINNING TRACK OF EACH CYLINDER FORMATTED. THIS CHECK IS PERFORMED TO CONFIRM THAT THE DISK'S POSITIONER ADVANCED PROPERLY DURING THE FORMAT OPERATION.

9. BAD SPOT FILE

SECTORS 0 THROUGH 31, ON CYLINDER 822, TRACK 4 ARE ALWAYS FORMATTED IN 16 BIT MODE. THEY CONTAIN THE BAD SECTOR ADDRESS ON THE PACK.

SECTORS 0, 2, 4, 6, 8 ARE IDENTICAL AND RECORD THE MANUFACTURE DEFINED BAD SECTOR ADDRESS FOR 16 BIT MODE.

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

SECTORS 1,3,5,7,9 ARE IDENTICAL AND RECORD THE MANUFACTURE DEFINED
BAD SECTOR ADDRESS FOR 18 BIT MODE.

SECTORS 10,12,14,16,18,20,22,24,26,28,30 ARE IDENTICAL AND
RECORD THE USER DEFINED
BAD SECTOR ADDRESS FOR 16 BIT MODE.

SECTORS 11,13,15,17,19,21,23,25,27,29,31 ARE IDENTICAL AND
RECORD THE USER DEFINED BAD SECTOR ADDRESS FOR 18 BIT MODE.

THE FORMAT OF EVERY SECTOR OF THE BAD SPOT FILE IS DESCRIBED BELOW:
THE FIRST TWO WORDS IN THE DATA FIELD SPECIFY THE SERIAL
NUMBER OF THE PACK. THE SERIAL NUMBER MUST BE A NON-ZERO
NUMBER AND CONSISTS OF MAXIMUM 10 OCTAL DIGITS.
THE FIRST WORD OF THE SERIAL NUMBER CONTAINS THE 5
LEST SIGNIFICANT OCTAL DIGITS, WHILE THE SECOND
WORD CONTAINS THE 5 MOST SIGNIFICANT DIGITS.

THE THIRD WORD IS ALWAYS ZERO.

THE FORTH WORD DEFINES THE PACK IS A DATA PACK ,IF IT IS 0

THE FOLLOWING 252 WORDS DEFINE THE BAD SECTOR ADDRESS; TWO
WORDS ARE USED TO DEFINE A SINGLE BAD SECTOR. THE FIRST
WORD SPECIFIES THE CYLINDER ADDRESS, THE SECOND WORD
SPECIFIES THE TRACK (HIGH BYTE) AND SECTOR (LOW BATE) ADDRESSES.
ALL UNUSED LOCATIONS ARE RECORDED WITH ONES.

FOR A UNFORMATTED PACK, ALL MANUFACTURE DEFINED SECOTRS ARE
RECORDED WITH NO BAD SECTORS. ALL BAD SECTORS
DETECTED BY THE PROGRAM ARE RECORDED IN SECTORS 10,12,... OR
11,13,... DEPENDED ON THE 16 BIT MODE OR 18 BIT MODE.

FOR A FORMATTED PACK, ALL MANUFACTURE DEFINED BAD SECTORS ARE
NOT REFORMATTED OR NOT WRITTEN ANY DATA WORDS BY THE PROGRAM.
BESIDES, FOR A FORMATTED PACK, THE PROGRAM RUNS
IN 16 BITS MODE, ALL MANUFACTURE DEFINED AND 18 BIT MODE USER DEFINED
BAD SECTORS ARE NOT UPDATED OR DESTORIED ,AND VICE VERSA.

IT IS RECOMMENDED TO FORMAT THE PACK IN 18 BIT MODE FIRST
THEN RESTART THE PROGRAM TO FORMAT THE PACK IN 16 BIT MODE
IN CASE A UNFORMATTED PACK IS SELECTED.

10.1 RH70(11)/RMO3 DRIVER

THIS DOCUMENT IS THE USER'S GUIDE FOR THE RH70/RMO3 DRIVER.

10.2 TO INITIALIZE THE DRIVER:

JSR PC,RMINIT
RETURN

UPON RETURN YOU MUST EXAMINE THE "DRVSTA" TABLE TO DETERMINE THE DRIVES THAT ARE ONLINE FOR TESTING. THE 'DRVSTA' TABLE IS EIGHT BYTES; ONE BYTE PER DRIVE. THE STATE OF EACH DRIVE WILL BE INDICATED AS FOLLOWS:

DRVSTA	DRIVE STATE
-----	-----
>0	ONLINE RMO3
=0	OFFLINE RMO3, DRIVE IS NOT AN RMO3, OR NONEXISTENT DRIVE
<0	UNSAFE RMO3

THE DRIVE TYPE IS DEFINED IN AN 8 BYTE LONG TABLE TAGGED 'DRVTYP'. THE TABLE CONTAINS ONE BYTE FOR EACH DRIVE AND IS INDEXED BY THE DRIVE NUMBER. ENTRIES ARE ENCODED AS FOLLOWS:

DRVTYP	CONDITION
-----	-----
0	NONEXISTENT DRIVE
4	RMO3
-1	NOT AN RMO3

THE 'RMINIT' ROUTINE WILL DO A READIN PRESET AND WILL SET FMT22.

10.3 AFTER THE DRIVER HAS BEEN INITIALIZED, IT IS CALLED USING THE FOLLOWING SEQUENCE.

CALL:

JSR RO,RMO3
PNTDPB
RETURN1
RETURN2;MAKE THE CALL
;ADDRESS OF DPB*
;RETURN IF QUEUE IS FULL
;RETURN IF REQUEST IS IN
;QUEUE OR THERE IS AN
;ERROR CONDITION

*DPB (DATA PARAMETER BLOCK)

PNTDPB: .BYTE	0	;(0) DRIVE NUMBER
.BYTE	0	;(1) OFFSET VALUE OR FMT22, ECT, AND HCI
.BYTE	0	;(2) COMMAND
.BYTE	0	;(3) PSEL AND A17 AND A16
.WORD	0	;(4) WORD COUNT (MUST BE NEG.)

```

702      .WORD 0      ;(6) BUFFER ADDRESS OR
703      .BYTE 0      ;REGISTER TABLE POINTER
704      .BYTE 0      ;(10) SECTOR ADDRESS OR
705      .BYTE 0      ;FIRST REG. INDEX
706      .WORD 0      ;(11) TRACK ADDRESS OR
707      .WORD 0      ;LAST REG. INDEX
708      .WORD 0      ;(12) CYLINDER ADDRESS
709      .WORD 0      ;(14) ERROR TABLE POINTER
710      .WORD 0      ;POINTS TO THE FIRST OF TWENTY
711      .WORD 0      ;LOCATIONS OF WHERE THE DRIVER
712      .WORD 0      ;IS TO STORE THE RM70/RM03
713      .WORD 0      ;REGISTERS ON AN ERROR. IF LEFT
714      .WORD 0      ;ZERO REGISTERS ARE NOT SAVED.
715      .WORD 0      ;(16) STATUS/ERROR INDICATOR
716      .WORD 0      ;BIT15=1=>ERROR OCCURRED
717      .WORD 0      ;BIT07=1=>DONE
718      .WORD 0      ;BIT14-BIT09 AND BIT06-BIT03
719      .WORD 0      ;INDICATE TYPE OF ERROR
720

```

- 10.4 THE DRIVER PROVIDES A SOFTWARE TIMEOUT CAPABILITY.
TO UTILIZE THIS CAPABILITY YOU MUST SUPPLY THE "RM TIMER" ROUTINE
WITH THE ELAPSED TIME IN THE FOLLOWING MANNER:

```

MOV      #16.,-(SP)    ;16 MILLISECONDS BETWEEN
JSR      PC,RMTMR      ;CLOCK TICKS
                        ;CALL THE TIMER ROUTINE

```

IT SHOULD BE NOTED THAT YOU MUST PROVIDE THE CODE TO DRIVE THE
CLOCK. AND THE ELAPSED TIME MUST BE IN MILLISECONDS.
THE DRIVER WILL SET THE TIMEOUT TO 1 SECOND FOR ALL POSITIONING
AND DATA TRANSFER OPERATIONS AND WILL SET THE TIMEOUT TO 30
SECONDS FOR ERROR RECOVERY OPERATIONS.

10.4.1 EXAMPLE - WRITE 1000. WORDS

```

1$:      JSR      RO,RM03      ;CALL THE DRIVER
        WRTDPB      ;DPB ADDRESS
        BR        1$          ;WAIT FOR QUEUE IF FULL
2$:      TST      WRTDPB+16    ;WAIT FOR COMMAND TO COMPLETE
        BEQ       2$
        BMI      ERROR1      ;ERROR OCCURRED
        .
        .
        .
WRTDPB:  .BYTE    5            ;DRIVE #5
        .BYTE    0
        .BYTE    161         ;WRITE COMMAND
        .BYTE    0
        .WORD     -1000.      ;WORD COUNT
        .WORD     WRTBUF      ;BUFFER ADDRESS
        .BYTE     3           ;SECTOR
        .BYTE     5           ;TRACK
        .WORD     400         ;CYLINDER
        .WORD     ERRTB5      ;ERROR TABLE

```

758	.WORD	0	;STATUS/ERROR INDICATOR
759			
760	ALTERNATE DPB SETUP		
761			
762	WRTDPB:	.WORD 5	;THIS SETUP ACHIEVED
763		.WORD WRITE	;EVERYTHING THE
764		.WORD -1000.	;ABOVE TABLE DID, BUT
765		.WORD WRTBUF	;IN A CLEANER FORMAT
766		.BYTE 3,5	
767		.WORD 400,ERRTBS,0	

10.5 RH70/RMO3 REGISTERS

	<u>MNEMONIC</u>	<u>INDEX</u>
770		
771		
772		
773		
774	RMCS1	0
775	RMWC	2
776	RMBA	4
777	RMDA	6
778	RMCS2	10
779	RMD5	12
780	RMER1	14
781	RMS	16
782	RMLA	20
783	RMD8	22
784	RMMR1	24
785	RMDT	26
786	RMSN	30
787	RMOF	32
788	RMDC	34
789	RMHR	36
790	RMMR2	40
791	RMER2	42
792	RMEC1	44
793	RMEC2	46

10.6 COMMANDS PERFORMED BY THE DRIVER

	<u>COMMAND</u>	<u>CODE</u>	<u>COMMAND TYPE</u>
794			
795			
796			
797			
798			
799			
800	NO OPERATION	101	N
801			
802	UNLOAD	103	N
803			
804	SEEK	105	P
805			
806	RECALIRATE	107	P
807			
808	DRIVE CLEAR	111	N
809			
810	RELEASE	113	N
811			
812	OFFSET	115	P
813			

E02

814	RETURN TO CENTER	117	P
815	READIN PRESET	121	N
816	PACK ACKNOWLEDGE	123	N
817	SEARCH	131	P
818	GET REGISTER(S)	141	S
819	SET FORMAT	143	S
820	SELECT DRIVE	145	S
821	WRITE CHECK DATA	151	D
822	WRITE CHECK HEADER	153	D
823	AND DATA		
824	WRITE DATA	161	D
825	WRITE HEADER & DATA	163	D
826	READ DATA	171	D
827	READ HEADER & DATA	173	D

N = HOUSEKEEPING
P = POSITIONING
D = DATA TRANSFER
S = SPECIAL PROVIDED BY THE DRIVER

10.7 DPB STATUS/ERROR INDICATOR WORD

THIS INDICATOR WILL INFORM THE USER OF THE RESULTS OF THE REQUEST.
THIS IS ACCOMPLISHED BY SETTING VARIOUS BITS OF THE INDICATOR TO A ONE.

BIT NO.	MEANING IF ON A "1"
-----	-----
15	ERROR OCCURRED DONE (BIT07=0); BITS 14-10 SPECIFIES TYPE DONE (BIT07=1); BITS 06-03 SPECIFIES TYPE
14(1)	USER MADE A REQUEST FOR A FUNCTION TO BE PERFORMED ON AN OFFLINE OR UNSAFE DRIVE
13(1)	USER MADE A REQUEST FOR A FUNCTION TO BE PERFORMED ON A DRIVE THAT HAS AN UNLOAD REQUEST IN QUEUE.
12(2)	PERSISTENT UNSAFE CONDITION EXIST.
11(2)	UNCORRECTABLE PARITY ERROR OCCURRED

814
815
816
817
818
819
820
821
822
823
824
825
826
827
828
829
830
831
832
833
834
835
836
837
838
839
840
841
842
843
844
845
846
847
848
849
850
851
852
853
854
855
856
857
858
859
860
861
862
863
864
865
866
867
868
869

870		
871	10(2)(4)	FATAL PARITY ERROR. A MASSBUS CLEAR WAS
872		PERFORMED. ALL QUEUES WERE EMPTIED, AND
873		ALL DRVACT'S SET TO THE IDLE STATE
874		
875	9(3)(4)	SOFTWARE TIMEOUT OCCURRED ON THIS DRIVE
876		
877	8(4)	SOFTWARE TIMEOUT OCCURRED ON ANOTHER DRIVE
878		
879	7	DONE
880		
881	6(2)	ERROR OCCURRED DURING AN I/O OPERATION
882		
883	5(2)	ERROR OCCURRED DURING AN OPERATION
884		OTHER THAN I/O.
885		
886	4(2)	CORRECTABLE UNSAFE CONDITION OCCURRED
887		
888	3(2)	DRIVE ERROR OCCURRED THAT CAUSED AN
889		AUTOMATIC "RECALIBRATE" SEQUENCE
890		
891	2	PORT REQUEST TIMEOUT. THE DRIVER REQUESTED
892		THE DRIVE BUT THE OPPOSITE PORT DID NOT
893		RELEASE THE DRIVE WITHIN 20 SECONDS.
894		
895	1	NON-EXISTENT DRIVE REQUESTED. USER MADE
896		A REQUEST FOR A NON-EXISTENT DRIVE.
897		
898	(1) =>	REQUEST WASN'T PUT IN QUEUE. (RH70/RMO3
899		REGISTERS WERE NOT SAVED)
900		
901	(2) =>	REQUEST QUEUE HAS BEEN EMPTIED. THE DRIVER
902		ISSUED A "DRIVE CLEAR" TO THE DRIVE.
903		NOTE: ALL RH70/RMO3 REGISTERS ARE SAVED
904		AS PER DPB+14 BEFORE THE "DRIVE CLEAR".
905		
906	(3) =>	REQUEST QUEUE HAS BEEN EMPTIED. THE
907		DRIVER ISSUED A MASSBUS INIT. ALL
908		RH70/RMO3 REGISTERS FOR THE DRIVE WERE
909		SAVED AS PER DPB+14 BEFORE THE INIT.
910		
911	(4) =>	A "RECALIBRATE" SHOULD BE ISSUED
912		BEFORE ANY OTHER COMMAND.
913		
914	10.8	ERROR CALLS MADE BY THE DRIVER.
915		
916		THERE ARE A FEW ERRORS THAT CAN OCCUR THAT CAN NOT BE INDICATED IN A DPB.
917		
918		WHEN THIS TYPE OF ERROR IS DETECTED BY THE DRIVER IT WILL MAKE
919		AN ERROR CALL OF THE FORM "ERROR N", WHERE "N" IS THE ERROR
920		NUMBER AND THE ERROR WILL BE AN EMT INSTRUCTION.
921		
922		
923	N	TYPE
924	-	-----
925		DATA AVAILABLE

G02

MAINDEC-11-DZRNA, RM03 FORMATTER
DZRNAB.P11 21-JUL-77 15:32

MACY11 30(1046) 21-JUL-77 16:06 PAGE 19

SEQ 0018

926
927
928
929
930
931
932
933
934
935
936
937
938
939
940
941
942
943
944
945
946
947
948
949
950
951
952
953
954
955
956
957

1	RH70 INTERRUPT OCCURRED (RHAS=0)	*R4= RMCS1'S ADDRESS
2	UNEXPECTED ATTENTION OCCURRED	R1= DRIVE NUMBER R3= ATA BIT *R4= RMCS1'S ADDRESS R5= (RMAS) RMERRS =RMD5 RMERRS+2=RMER1 RMERRS+4=RMER2 RMERRS+6=RMMR2
3	MASSBUS PARITY ERROR (MCPE=1)	RD.ADR= ADDRESS OF REG. READ RD.WRD= WORD READ
4	MASSBUS PARITY ERROR (PAR=1)	WRT.AD= ADDRESS OF REG. WRITTEN WRT.WD= WORD WRITTEN RD.WRD= WORD READ BACK
5	ADDRESS PLUG CHANGE BIT SET ('OPE' ERROR)	R1= DRIVE NUMBER R3= ATA BIT *R4= RMCS1'S ADDRESS R5= (RMAS) RMERRS =RMD5 RMERRS+2=RMER1 RMERRS+4=RMER2 RMERRS+6=RMMR2

* THIS IS THE ACTUAL UNIBUS ADDRESS (176700)

%

H02

```

958 .TITLE MAINDEC-11-DZRMA, RMO3 FORMATTER
959 .*COPYRIGHT (C) 1977
960 .*DIGITAL EQUIPMENT CORP.
961 .*MAYNARD, MASS. 01754
962 .*
963 .*PROGRAM BY C. HESS/C. CHEN
964 .*
965 .*THIS PROGRAM WAS ASSEMBLED USING THE PDP-11 MAINDEC SYSMAC
966 .*PACKAGE (MAINDEC-11-DZQAC-C3), JAN 19, 1977.
967 .*
968 .SBTTL BASIC DEFINITIONS
969
970 .*INITIAL ADDRESS OF THE STACK POINTER *** 1100 ***
971 001100 STACK= 1100
972 .EQUIV EMT,ERROR ;;BASIC DEFINITION OF ERROR CALL
973 .EQUIV IOT,SCOPE ;;BASIC DEFINITION OF SCOPE CALL
974
975 .*MISCELLANEOUS DEFINITIONS
976 000011 HT= 11 ;;CODE FOR HORIZONTAL TAB
977 000012 LF= 12 ;;CODE FOR LINE FEED
978 000015 CR= 15 ;;CODE FOR CARRIAGE RETURN
979 000200 CRLF= 200 ;;CODE FOR CARRIAGE RETURN-LINE FEED
980 177776 PS= 177776 ;;PROCESSOR STATUS WORD
981 .EQUIV PS,PSW
982 177774 STKLMT= 177774 ;;STACK LIMIT REGISTER
983 177772 PIRQ= 177772 ;;PROGRAM INTERRUPT REQUEST REGISTER
984 177570 DSWR= 177570 ;;HARDWARE SWITCH REGISTER
985 177570 DDISP= 177570 ;;HARDWARE DISPLAY REGISTER
986
987 .*GENERAL PURPOSE REGISTER DEFINITIONS
988 000000 R0= %0 ;;GENERAL REGISTER
989 000001 R1= %1 ;;GENERAL REGISTER
990 000002 R2= %2 ;;GENERAL REGISTER
991 000003 R3= %3 ;;GENERAL REGISTER
992 000004 R4= %4 ;;GENERAL REGISTER
993 000005 R5= %5 ;;GENERAL REGISTER
994 000006 R6= %6 ;;GENERAL REGISTER
995 000007 R7= %7 ;;GENERAL REGISTER
996 000006 SP= %6 ;;STACK POINTER
997 000007 PC= %7 ;;PROGRAM COUNTER
998
999 .*PRIORITY LEVEL DEFINITIONS
1000 000000 PR0= 0 ;;PRIORITY LEVEL 0
1001 000040 PR1= 40 ;;PRIORITY LEVEL 1
1002 000100 PR2= 100 ;;PRIORITY LEVEL 2
1003 000140 PR3= 140 ;;PRIORITY LEVEL 3
1004 000200 PR4= 200 ;;PRIORITY LEVEL 4
1005 000240 PR5= 240 ;;PRIORITY LEVEL 5
1006 000300 PR6= 300 ;;PRIORITY LEVEL 6
1007 000340 PR7= 340 ;;PRIORITY LEVEL 7
1008
1009 .*"SWITCH REGISTER" SWITCH DEFINITIONS
1010 100000 SW15= 100000
1011 040000 SW14= 40000
1012 020000 SW13= 20000
1013 010000 SW12= 10000

```

MAINDEC-11-DZRNA, RM03 FORMATTER
DZRNAB.P11 21-JUL-77 15:32

MACY11 30(1046) 21-JUL-77 16:06 PAGE 21
BASIC DEFINITIONS

SEQ 0020

1014 004000
1015 002000
1016 001000
1017 000400
1018 000200
1019 000100
1020 000040
1021 000020
1022 000010
1023 000004
1024 000002
1025 000001

SW11= 4000
SW10= 2000
SW09= 1000
SW08= 400
SW07= 200
SW06= 100
SW05= 40
SW04= 20
SW03= 10
SW02= 4
SW01= 2
SW00= 1
.EQUIV SW09,SW9
.EQUIV SW08,SW8
.EQUIV SW07,SW7
.EQUIV SW06,SW6
.EQUIV SW05,SW5
.EQUIV SW04,SW4
.EQUIV SW03,SW3
.EQUIV SW02,SW2
.EQUIV SW01,SW1
.EQUIV SW00,SW0

1037 100000
1038 040000
1039 020000
1040 010000
1041 004000
1042 002000
1043 001000
1044 000400
1045 000200
1046 000100
1047 000040
1048 000020
1049 000010
1050 000004
1051 000002
1052 000001

.*DATA BIT DEFINITIONS (BIT00 TO BIT15)

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

1065 000004
1066 000010
1067 000014
1068 000014
1069 000014

.*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

1070 000014
1071 000020
1072 000024
1073 000030
1074 000034
1075 000060
1076 000064
1077 000240
1078 176700
1079 000254
1080
1081
1082
1083
1084
1085
1086
1087
1088
1089
1090
1091
1092
1093
1094
1095 000000
1096
1097
1098
1099 000174 000174
1100 000174 000000
1101 000176 000000
1102
1103
1104
1105
1106
1107 000200
1108 000046 000046
1109 000046 015066
1110 000052 000052
1111 000052 020000
1112 000200 000200
1113 000200 000200
1114 000200 000137 006422
1115 000204 000137 006412
1116
1117 001100
1118
1119
1120
1121
1122
1123 001100
1124 000024 000024
1125 000024 000200

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
FIRQVEC=240 ; PROGRAM INTERRUPT REQUEST VECTOR
ABASE=176700
AVECT1=254

.SBTTL OPERATIONAL SWITCH SETTINGS

SWITCH	USE
15	HALT ON ERROR
13	INHIBIT ERROR TYPEOUTS
10	BELL ON ERROR
9	LOOP ON ERROR
2	DON'T DISPLAY SYSTEM STATUS AFTER INITIAL START
1	LOOP ON THE CURRENT TRACK
0	LOOP THE PROGRAM ON THE SELECTED DRIVE

.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 ACT11 HOOKS

; HOOKS REQUIRED BY ACT11
\$SVPC= . ; SAVE PC
. =46
\$ENDAD ; 1) SET LOC.46 TO ADDRESS OF \$ENDAD IN .SEOP
. =52
.WORD 20000 ; 2) SET LOC.52 TO 20000
.=\$SVPC ; RESTORE PC
. =200
JMP BEGIN1 ; NORMAL STARTING ADDRESS
JMP BEGIN ; CHANGE THE RH11 ADDRESS

. =1100
.SBTTL APT PARAMETER BLOCK

; SET LOCATIONS 24 AND 44 AS REQUIRED FOR APT

\$X= . ; SAVE CURRENT LOCATION
. =24 ; SET POWER FAIL TO POINT TO START OF PROGRAM
200 ; FOR APT START UP

K02

MAINDEC-11-DZRNA, RM03 FORMATTER
DZRNA8.P11 21-JUL-77 15:32

MACY11 30(1046) 21-JUL-77 16:06 PAGE 23
APT PARAMETER BLOCK

SEQ 0022

1126 000044 000044
1127 000044 001100
1128 001100
1129
1130
1131
1132
1133 001100
1134 001100 000000
1135 001102 001206
1136 001104 000310
1137 001106 000310
1138 001110 000310
1139 001112 000032
1140 001114
1141
1142
1143
1144
1145
1146
1147
1148
1149
1150 000100
1151 000200
1152 000400
1153 001000
1154 002000
1155 020000
1156 040000
1157
1158
1159
1160
1161
1162
1163
1164
1165
1166
1167 000001
1168 000002
1169 000004
1170 000010
1171 000020
1172 000040
1173 000100
1174 000200
1175 000400
1 6 001000
1177 002000
1178 004000
1179 010000
1180 020000
1181 040000

```

      . = 44      ;; POINT TO APT INDIRECT ADDRESS PNTR.
      $APTHDR    ;; POINT TO APT HEADER BLOCK
      . = $X      ;; RESET LOCATION COUNTER
      ;;*****
      ;; SETUP APT PARAMETER BLOCK AS DEFINED IN THE APT-POP11 DIAGNOSTIC
      ;; INTERFACE SPEC.

      $APTHD:
      $HIBTS: .WORD 0      ;; TWO HIGH BITS OF 18 BIT MAILBOX ADDR.
      $MBADR: .WORD $MAIL   ;; ADDRESS OF APT MAILBOX (BITS 0-15)
      $TSTM: .WORD 200      ;; RUN TIME OF LONGEST TEST
      $PASTM: .WORD 200      ;; RUN TIME IN SECS. OF 1ST PASS ON 1 UNIT (QUICK VERIFY)
      $UNITM: .WORD 200      ;; ADDITIONAL RUN TIME (SECS) OF A PASS FOR EACH ADDITIONAL UNIT
      .WORD $ETEND-$MAIL/2 ;; LENGTH MAILBOX-ETABLE (WORDS)
      TAB.XY = .      ;; CMTAG STARTING ADDRESS

      ;;*****

      .SBTTL RH11 REGISTERS

      ;;*****

      ; CONTROL AND STATUS REGISTER 1 (RMCS1)

      IE= 100      ; INTERRUPT ENABLE (BIT #6)
      RDY= 200     ; READY (BIT #7)
      A16= 400     ; HIGH ORDER BUS ADDRESS BIT (BIT #8)
      A17= 1000    ; HIGH ORDER BUS ADDRESS BIT (BIT #9)
      PSEL= 2000   ; PORT SELECT (BIT #10)
      MCPE= 20000  ; MASSBUS PARITY ERROR (BIT #13)
      TRE= 40000   ; TRANSFER ERROR (BIT #14)
      SC= 100000   ; SPECIAL CONDITION (BIT #15)

      ; WORD COUNT REGISTER (RMWC)
      ; (EACH BIT IS CALLED BY BIT NUMBER)

      ; BUS ADDRESS REGISTER (RMBA)
      ; (EACH BIT IS CALLED BY BIT NUMBER)

      ; CONTROL AND STATUS REGISTER 2 (RMCS2)

      US1= 1      ; UNIT SELECT (BIT #0)
      US2= 2      ; UNIT SELECT (BIT #1)
      US4= 4      ; UNIT SELECT (BIT #2)
      BAI= 10     ; BUS ADDRESS INCREMENT INHIBIT (BIT #3)
      PAT= 20     ; MASSBUS PARITY TEST (BIT #4)
      CLR= 40     ; CLEAR (BIT #5)
      IR= 100     ; INPUT READY (BIT #6)
      OR= 200     ; OUTPUT READY (BIT #7)
      MPE= 400    ; MASS BUS PARITY ERROR (BIT #8)
      MXF= 1000   ; MISSED TRANSFER ERROR (BIT #9)
      PGE= 2000   ; PROGRAM ERROR (BIT #10)
      NEM= 4000   ; NON EXISTENT MEMORY (BIT #11)
      NED= 10000  ; NON EXISTENT DRIVE (BIT #12)
      UPE= 20000  ; UNIBUS PARITY ERROR (BIT #13)
      WCE= 40000  ; WRITE CHECK ERROR (BIT #14)

```

MAINDEC-11-DZRNA, RM03 FORMATTER
DZRNA8.P11 21-JUL-77 15:32

MACY11 30(1046) 21-JUL-77 16:06 PAGE 24
RM11 REGISTERS

SEQ 0023

1182	100000	DLT=	100000	;DATA LATE (BIT #15)
1183				
1184				;DATA BUFFER REGISTER (RMD8)
1185				; (EACH BIT IS CALLED BY BIT NUMBER)
1186				
1187				
1188				;;*****
1189				
1190				.SBTTL RM03 REGISTERS
1191				
1192				;;*****
1193				
1194				;CONTROL AND STATUS 1 REGISTER. (#00)
1195				
1196	000001	GO=	1	;GO BIT (BIT #0)
1197	000002	F1=	2	;FUNCTION CODE BIT #1
1198	000004	F2=	4	;FUNCTION CODE BIT #2
1199	000010	F3=	10	;FUNCTION CODE BIT #3
1200	000020	F4=	20	;FUNCTION CODE BIT #4
1201	000040	F5=	40	;FUNCTION CODE BIT #5
1202	004000	DVA=	4000	;DEVICE AVAILABLE (BIT #11)
1203				
1204				;DRIVE STATUS REGISTER (RMDS1) (#01)
1205				
1206	000100	VV=	100	;VOLUME VALID (BIT #6)
1207	000200	DRY=	200	;DRIVE READY (BIT #7)
1208	000400	DPR=	400	;DRIVE PRESENT (BIT #8)
1209	001000	PGM=	1000	;PROGRAMABLE (BIT #9)
1210	002000	LST=	2000	;LAST SECTOR TRANSFERRED (BIT #10)
1211	004000	WRL=	4000	;WRITE LOCK (BIT #11)
1212	010000	MOL=	10000	;MEDIUM ON-LINE (BIT #12)
1213	020000	PIP=	20000	;POSITIONING OPERATION IN PROGRESS (BIT #13)
1214	040000	ERR=	40000	;COMPOSITE ERROR (BIT #14)
1215	100000	ATA=	100000	;ATTENTION ACTIVE (BIT #15)
1216				
1217				;ERROR REGISTER #01 (RMER1) (#02)
1218				
1219	000001	ILF=	1	;ILLEGAL FUNCTION (BIT #0)
1220	000002	ILR=	2	;ILLEGAL REGISTER (BIT #1)
1221	000004	RMR=	4	;REGISTER MODIFICATION REFUSED (BIT #2)
1222	000010	PAR=	10	;PARITY ERROR (BIT #3)
1223	000020	FER=	20	;FORMAT ERROR (BIT #4)
1224	000040	WCF=	40	;WRITE CLOCK FAIL (BIT #5)
1225	000100	ECH=	100	;ECC HARD ERROR (BIT #6)
1226	000200	HCE=	200	;HEADER COMPARE ERROR (BIT #7)
1227	000400	HCRC=	400	;HEADER CRC ERROR (BIT #8)
1228	001000	AOE=	1000	;ADDRESS OVERFLOW ERROR (BIT #9)
1229	002000	IAE=	2000	;INVALID ADDRESS ERROR (BIT #10)
1230	004000	WLE=	4000	;WRITE LOCK ERROR (BIT #11)
1231	010000	DTE=	10000	;DRIVE TIMING ERROR (BIT #12)
1232	020000	OPI=	20000	;OPERATION INCOMPLETE (BIT #13)
1233	040000	UNS=	40000	;DRIVE UNSAFE (BIT #14)
1234	100000	DCK=	100000	;DATA CHECK ERROR (BIT 15)
1235				
1236				;MAINTAINABILITY REGISTER (RMMR1) (#03)
1237				

MAINDEC-11-DZRNA, RM03 FORMATTER
DZRNA8.P11 21-JUL-77 15:32

MACY11 30(1046) 21-JUL-77 16:06 PAGE 25
RM03 REGISTERS

SEQ 0024

1238 000001
1239 000002
1240 000004
1241 000010
1242 000020
1243 000040
1244 000200

DMD= 1 ;DIAGNOSTIC MODE (BIT #0)
MCLK= 2 ;MAINTAINABILITY CLOCK (BIT #1)
MINX= 4 ;MAINTAINABILITY INDEX (BIT #2)
MSTCK= 10 ;MAINTAINABILITY SECTOR CLOCK (BIT #3)
MRD= 20 ;MAINTAINABILITY READ (BIT #4)
MWR= 40 ;MAINTAINABILITY WRITE (BIT #5)
DTSY= 200 ;MAINTAINABILITY SYNC DETECTED (BIT #7)

;ATTENTION SUMMARY PSEUDO-REGISTER (RMAS) (#04)

1248 000001
1249 000002
1250 000004
1251 000010
1252 000020
1253 000040
1254 000100
1255 000200

ATO= 1 ;DEVICE 0 (BIT #0)
AT1= 2 ;DEVICE 1 (BIT #1)
AT2= 4 ;DEVICE 2 (BIT #2)
AT3= 10 ;DEVICE 3 (BIT #3)
AT4= 20 ;DEVICE 4 (BIT #4)
AT5= 40 ;DEVICE 5 (BIT #5)
AT6= 100 ;DEVICE 6 (BIT #6)
AT7= 200 ;DEVICE 7 (BIT #7)

;DESIRED SECTOR/TRACK ADDRESS REGISTER (RMDA) (#05)
;(EACH BIT IS CALLED BY BIT NUMBER)

;DRIVE TYPE REGISTER (RMDT) (#06)

1262 000001
1263 000002
1264 000004
1265 000010
1266 000020
1267 000040
1268 000100
1269 000200
1270 000400
1271 000100
1272 020000
1273 040000
1274 100000

DT00= 1 ;DRIVE TYPE NUMBER BIT 1
DT01= 2 ;DRIVE TYPE NUMBER BIT 2
DT02= 4 ;DRIVE TYPE NUMBER BIT 3
DT03= 10 ;DRIVE TYPE NUMBER BIT 4
DT04= 20 ;DRIVE TYPE NUMBER BIT 5
DT05= 40 ;DRIVE TYPE NUMBER BIT 6
DT06= 100 ;DRIVE TYPE NUMBER BIT 7
DT07= 200 ;DRIVE TYPE NUMBER BIT 8
DT08= 400 ;DRIVE TYPE NUMBER BIT 9
DRQ= 4000 ;DRIVE REQUEST REQUIRED (BIT #11)
MOH= 20000 ;MOVING HEAD (BIT #13)
TAP= 40000 ;TAPE DRIVE (BIT #14)
NBA= 100000 ;NOT BLOCK ADDRESSED (BIT #15)

;LOOK-AHEAD REGISTER (RMLA) (#07)

1278 000100
1279 000200
1280 000400
1281 001000
1282 002000
1283 004000
1284 010000
1285 020000
1286 040000
1287 100000

SC01= 100 ;SECTOR COUNT FIELD 0 (BIT #6)
SC02= 200 ;SECTOR COUNT FIELD 1 (BIT #7)
SC04= 400 ;SECTOR COUNT FIELD 2 (BIT #8)
SC10= 1000 ;SECTOR COUNT FIELD 3 (BIT #9)
SC20= 2000 ;SECTOR COUNT FIELD 4 (BIT #10)
TRK1= 4000 ;TRACK FIELD 1 (BIT #11)
TRK2= 10000 ;TRACK FIELD 2 (BIT #12)
TRK4= 20000 ;TRACK FIELD 3 (BIT #13)
TRK10= 40000 ;TRACK FIELD 4 (BIT #14)
TRK20= 100000 ;TRACK FIELD 5 (BIT #15)

;OFFSET REGISTER (RMOF) (#11)

1291 000001
1292 002000
1293 004000

OFDIR= 1 ;OFFSET DIRECTION FOR RM03
HCI= 2000 ;HEADER COMPARE INHIBIT (BIT #10)
ECI= 4000 ;ERROR CORRECTION CODE INHIBIT (BIT #11)

N02

MAINDEC-11-DZMA, RMO3 FORMATTER
DZMA8.P11 21-JUL-77 15:32

MACY11 30(1046) 21-JUL-77 16:06 PAGE 26
RMO3 REGISTERS

SEQ 0025

1294 010000

FMT16= 10000 ;FORMAT BIT (BIT #12)

1295
1296 ;DESIRED CYLINDER ADDRESS (RMD0) (#12)
1297 ;(EACH BIT IS CALLED BY BIT NUMBER)

1298
1299 ;SERIAL NUMBER REGISTER (RMSN) (#14)
1300 ;(EACH IS CALLED BY BIT NUMBER)

1301
1302 ;RMO3 MAINTENANCE REGISTER #02 (RMMR2) (#15)

1303
1304 040000 SKI= 40000 ;SEEK INCOMPLETE (BIT #14)

1305
1306 ;ECC POSITION REGISTER (RMEC1) (#16)
1307 ;(EACH BIT IS CALLED BY BIT NUMBER)

1308
1309 ;ECC PATTERN REGISTER (RMEC2) (#17)
1310 ;(EACH BIT IS CALLED BY BIT NUMBER)

1311
1312 ;*****

1313
1314 .SBTTL RMO3 DRIVER COMMANDS

1315
1316 ;*****

1317			
1318	000101	RNOP =	101 ;NO OPERATION
1319	000105	SEEK =	105 ;SEEK
1320	000107	RECAL =	107 ;RECALIBRATE
1321	000111	DRVCLR =	111 ;DRIVE CLEAR
1322	000113	RELSE =	113 ;RELEASE
1323	000115	OFFSET =	115 ;OFFSET
1324	000117	RTC =	117 ;RETURN TO CENTER LINE
1325	000121	READIN =	121 ;READ IN PRESET
1326	000123	ACK =	123 ;PACK ACKNOWLEDGE
1327	000131	SEARCH =	131 ;SEARCH
1328	000141	GETREG =	141 ;GET REGISTERS
1329	000143	SETFMT =	143 ;SET FORMAT (& ECI OR HCI)
1330	000145	SELDRV =	145 ;SELECT DRIVE
1331	000151	WCKD =	151 ;WRITE CHECK DATA
1332	000153	WCKHD =	153 ;WRITE CHECK HEADER & DATA
1333	000161	WRDAT =	161 ;WRITE DATA
1334	000163	WRTHD =	163 ;WRITE HEADER & DATA
1335	000171	RDDAT =	171 ;READ DATA
1336	000173	RDHD =	173 ;READ HEADER & DATA
1337			
1338			

1339
1340
1341
1342
1343
1344
1345 001114
1346 001114 000000
1347 001114 000000
1348 001116 000
1349 001117 000
1350 001120 000000
1351 001122 000000
1352 001124 000000
1353 001126 000000
1354 001130 000
1355 001131 001
1356 001132 000000
1357 001134 000000
1358 001136 000000
1359 001140 000000
1360 001142 000000
1361 001144 000000
1362 001146 000000
1363 001150 000
1364 001151 000
1365 001152 000000
1366 001154 177570
1367 001156 177570
1368 001160 177560
1369 001162 177562
1370 001164 177564
1371 001166 177566
1372 001170 000
1373 001171 002
1374 001172 012
1375 001173 000
1376 001174 000000
1377 001176 177607 000377
1378 001202 077
1379 001203 015
1380 001204 000012
1381
1382
1383
1384
1385
1386
1387
1388
1389
1390 001206
1391 001206 000000
1392 001210 000000
1393 001212 000000
1394 001214 000000

.SBTTL COMMON TAGS

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

SCMTAG: . =TAB.XY

.WORD 0
\$TSTNM: .BYTE 0
\$ERFLG: .BYTE 0
\$ICNT: .WORD 0
\$LPADR: .WORD 0
\$LPERR: .WORD 0
\$ERTTL: .WORD 0
\$ITEMB: .BYTE 0
\$ERMAX: .BYTE 1
\$ERRPC: .WORD 0
\$GDADR: .WORD 0
\$BDADR: .WORD 0
\$GDAT: .WORD 0
\$BDAT: .WORD 0
\$AUTOB: .BYTE 0
\$INTAG: .BYTE 0
\$SWR: .WORD DSWR
\$DISPLAY: .WORD DDISP
\$TKS: 177560
\$TKB: 177562
\$TPS: 177564
\$TPB: 177566
\$NULL: .BYTE 0
\$FILLS: .BYTE 2
\$FILLC: .BYTE 12
\$TPFLG: .BYTE 0
\$ESCAPE: 0
\$BELL: .ASCIZ <207><377><377>
\$QUES: .ASCII /?
\$CRLF: .ASCII <15>
\$LF: .ASCIZ <12>

;; START OF COMMON TAGS

;; 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)
;; ESCAPE ON ERROR ADDRESS
;; CODE FOR BELL
;; QUESTION MARK
;; CARRIAGE RETURN
;; LINE FEED

.SBTTL APT MAILBOX-ETABLE

.NLIST ME

.EVEN
\$MAIL: .APT MAILBOX
\$MSGTY: .WORD AMSGTY : MESSAGE TYPE CODE
\$FATAL: .WORD AFATAL : FATAL ERROR NUMBER
\$TESTN: .WORD ATESTN : TEST NUMBER
\$PASS: .WORD APASS : PASS COUNT

C03

MAINDEC-11-DZRNA, RM03 FORMATTER
DZRNA8.P11 21-JUL-77 15:32

MACY11 30(1046) 21-JUL-77 16:06 PAGE 28
APT MAILBOX-ETABLE

SEQ 0027

1395 001216 000000
1396 001220 000000
1397 001222 000000
1398 001224 000000
1399 001226 000000
1400 001226 000
1401 001227 000
1402 001230 000000
1403 001232 000000
1404 001234 000000
1405
1406
1407
1408
1409
1410
1411 001236 000
1412 001237 000
1413
1414
1415
1416
1417 001240 000000
1418
1419 001242 000
1420 001243 000
1421 001244 000000
1422 001246 000
1423 001247 000
1424 001250 000000
1425 001252 000
1426 001253 000
1427 001254 000000
1428 001256 000254
1429 001260 000000
1430 001262 176700
1431 001264 000000
1432 001266 000000
1433 001270 000000
1434 001272
1435
1436 000015
1437 000012
1438 001272 176700
1439 001274 000254
1440 001276 172540
1441 001300 172542
1442 001302 000104 000106
1443 001306 177546
1444 001310 000100 000102
1445 001220
1446
1447 001314 000000
1448 001316 000000
1449 001320 001466
1450 001322 000000

SDEVCT: .WORD ADEVCT
SUNIT: .WORD AUNIT
\$MSGAD: .WORD AMSGAD
\$MSGLG: .WORD AMSGLG
SETABLE:
\$ENV: .BYTE AENV
\$ENVH: .BYTE AENVH
\$SWREG: .WORD ASWREG
\$USWR: .WORD AUSWR
\$CPUOP: .WORD ACPUOP
*
*
*
*
*
*
\$MAMS1: .BYTE AMAMS1
\$MTYP1: .BYTE AMTYP1
*
*
*
*
*
\$MADR1: .WORD AMADR1
*
\$MAMS2: .BYTE AMAMS2
\$MTYP2: .BYTE AMTYP2
\$MADR2: .WORD AMADR2
\$MAMS3: .BYTE AMAMS3
\$MTYP3: .BYTE AMTYP3
\$MADR3: .WORD AMADR3
\$MAMS4: .BYTE AMAMS4
\$MTYP4: .BYTE AMTYP4
\$MADR4: .WORD AMADR4
\$VECT1: .WORD AVECT1
\$VECT2: .WORD AVECT2
\$BASE: .WORD ABASE
\$DEVH: .WORD ADEVH
\$CDW1: .WORD ACDW1
\$CDW2: .WORD ACDW2
SETEND:
MEXIT
CR = 15
LF = 12
\$RMADR: .WORD 176700
\$RMVEC: .WORD 254
\$LKCSR: .WORD 172540
\$LKCSB: .WORD 172542
\$LPVEC: .WORD 104,106
\$LKS: .WORD 177546
\$LLVEC: .WORD 100,102
DRIVE =SUNIT
SOF\$W: .WORD 0
MODE: .WORD 0
ENDCYL: .WORD 822.
BEGCYL: .WORD 0

;; DEVICE COUNT
;; I/O UNIT NUMBER
;; MESSAGE ADDRESS
;; MESSAGE LENGTH
;; APT ENVIRONMENT TABLE
;; ENVIRONMENT BYTE
;; ENVIRONMENT MODE BITS
;; APT SWITCH REGISTER
;; USER SWITCHES
;; CPU TYPE, OPTIONS
BITS 15-11=CPU TYPE
11/04=01,11/05=02,11/20=03,11/40=04,11/45=05
11/70=06,PO0=07,Q=10
BIT 10=REAL TIME CLOCK
BIT 9=FLOATING POINT PROCESSOR
BIT 8=MEMORY MANAGEMENT
;; HIGH ADDRESS, M.S. BYTE
;; MEM. TYPE, BLK#1
MEM. TYPE BYTE -- (HIGH BYTE)
900 NSEC CORE=001
300 NSEC BIPOLAR=002
500 NSEC MOS=003
;; HIGH ADDRESS, BLK#1
MEM. LAST ADDR.=3 BYTES, THIS WORD AND LOW OF "TYPE" ABOVE
;; HIGH ADDRESS, M.S. BYTE
;; MEM. TYPE, BLK#2
MEM. LAST ADDRESS, BLK#2
;; HIGH ADDRESS, M.S. BYTE
;; MEM. TYPE, BLK#3
MEM. LAST ADDRESS, BLK#3
;; HIGH ADDRESS, M.S. BYTE
;; MEM. TYPE, BLK#4
MEM. LAST ADDRESS, BLK#4
;; INTERRUPT VECTOR#1, BUS PRIORITY#1
;; INTERRUPT VECTOR#2, BUS PRIORITY#2
;; BASE ADDRESS OF EQUIPMENT UNDER TEST
;; DEVICE MAP
;; CONTROLLER DESCRIPTION WORD#1
;; CONTROLLER DESCRIPTION WORD#2
;; RH11/RM03 UNIBUS ADDRESS
;; RH11 INTERRUPT VECTOR
;; ADDRESS OF KW11-P CSR
;; ADDRESS OF KW11-P COUNTER BUFFER
;; ADDRESS OF KW11-P VECTOR
;; ADDRESS OF KW11-L CONTROL REGISTER
;; ADDRESS OF KW11-L VECTOR
;; CONTAINS DRIVE NUMBER SELECTED
;; SAME PARAMETER USED IN APT
;; CONTENTS ARE FOR SOFTWARE DECISIONS
;; 'FORMAT & VERIFY' OR 'CHECK' MODE INDICATOR
;; ENDING CYLINDER
;; STARTING CYLINDER

1451	001324	000004	ENDTRK: .WORD	4.	;ENDING TRACK
1452	001326	000000	BEGTRK: .WORD	0	;STARTING TRACK
1453	001330	000000	TTRKS: .WORD	0	;TOTAL # OF TRACKS TO BE FORMATTED
1454	001332	000000	TTRKSC: .WORD	0	;TOTAL # OF TRACKS COUNTER
1455	001334	000000	TRKCNT: .WORD	0	;COUNTS TRKS FROM 0-5 PER CYL
1456	001336	000000	PATSEL: .WORD	0	;CONTAINS PATTERN SELECTED
1457	001340	000000	PATA: .WORD	0	;1ST WORD OF PATTERN
1458	001342	000000	PATB: .WORD	0	;2ND WORD OF PATTERN
1459	001344	000000	CYLCK: .WORD	0	;STORE CYLINDER ADDRESS FOR FORMAT VERIFICATION
1460	001346	000000	RETRY: .WORD	0	;MAINTAINS # OF WRITE RETRIES MADE
1461	001350	000000	SAVSEC: .WORD	0	;CONTAINS LAST BAD SECTOR ON FORMAT
1462	001352	000000	SAVMC: .WORD	0	;CONTAINS MC FOR REMAINING SECTORS ON ERROR
1463	001354	000000	MC: .WORD	0	;30 OR 32 SECTOR TRACK SIZE (IN WORDS)
1464	001356	000000	MWC: .WORD	0	;2'S COMPLEMENT OF 'MC'
1465	001360	000000	HEDERR: .WORD	0	;POSITIONING ERROR DURING FORMAT INDICATOR
1466	001362	000000	SEC30: .WORD	0	;30 OR 32 SECTOR MODE INDICATOR
1467					;0 = 30 SECTOR MODE
1468					;1'S = 32 SECTOR MODE
1469	001364	000000	MAXSEC: .WORD	0	;MAXIMUM SECTOR ADDRESS (FOR EITHER 30 OR 32 SECTOR
1470					;FORMAT)
1471	001366	000000	DS.CYL: .WORD	0	;ADDRESS OF CURRENT CYLINDER
1472	001370	000000	DS.TRK: .WORD	0	;ADDRESS OF CURRENT TRACK
1473	001372	000000	DORIVE: .WORD	0	;DRIVE NUMBER OF 'DRIVER' ERROR MESSAGES
1474	001374	000000	ATTN: .WORD	0	;ATTENTION REGISTER IMAGE FOR 'DRIVER'
1475					;ERROR MESSAGES
1476	001376	000000	CNTLC: .WORD	0	;ADDRESS OF 'IC' RETURN
1477	001400	000000	CHGADR: .WORD	0	; 'CHANGE RH11 ADDRESS' FLAG
1478	001402	000000	WRAP: .WORD	0	;WRAP AROUND FLAG,FORMAT FROM HIGH TO LOW
1479					;CYLINDER NUMBER
1480	001404	000000	DEVMP: .WORD	0	;DEVICE MAP FOR APT
1481	001406	000000	PACK: .WORD	0	;PACK>0, BAD SPOT FILE AVAILABLE FROM PACK
1482	001410	000000	EDIT: .WORD	0	;EDIT THE BAD SPOT FILE ,IF EDIT>0
1483	001412	000000	HEADX: .WORD	0	;HEAD BIT INDICATOR
1484					;BIT15:BIT14 GOOD SECTOR
1485					;BIT15 ALONE, USER DEFINED BAD SECTOR
1486					;BIT14 ALONE, MFG DEFINED BAD SECTOR
1487	001414	000400	BAD16: .BLKW	256.	;MFG 16 BIT BAD SPOT TABLE
1488	002414	177777	.WORD	-1	
1489	002416	177777	.WORD	-1	
1490	002420	000400	BAD18: .BLKW	256.	;MFG 18 BIT BAD SPOT TABLE
1491	003420	177777	.WORD	-1	
1492	003422	177777	.WORD	-1	
1493	003424	000400	USTAB: .BLKW	256.	;USER BAD SPOT TABLE
1494	004424	177777	.WORD	-1	
1495	004426	177777	.WORD	-1	
1496	004430	000400	USSAV: .BLKW	256.	;USER BAD SPOT TABLE SAVE FROM PACK
1497	005430	177777	.WORD	-1	
1498	005432	177777	.WORD	-1	
1499	005434	000040	SECCU: .BLKW	32.	;32 WORDS MAX TO STORE BAD SECTORS
1500	005534	177777	.WORD	-1	
1501	005536	000000	NEXT: .WORD	0	;FLAG, IF NEXT=-1,ALL GOOD SPOT ON
1502					;THE CURRENT TRACK ARE FORMATTED
1503					
1504					
1505					
1506	005540	000000	RM.REG: .WORD	0	;RH11/RM03 REGISTERS STORED HERE AFTER AN OPERATION
					;RMCS1

1507	005542	000000	.WORD	0	;RMWC
1508	005544	000000	.WORD	0	;RMBA
1509	005546	000000	.WORD	0	;RMOA
1510	005550	000000	.WORD	0	;RMCS2
1511	005552	000000	.WORD	0	;RMD5
1512	005554	000000	.WORD	0	;RMER1
1513	005556	000000	.WORD	0	;RMAS
1514	005560	000000	.WORD	0	;RMLA
1515	005562	000000	.WORD	0	;RMD8
1516	005564	000000	.WORD	0	;RMMR1
1517	005566	000000	.WORD	0	;RMDT
1518	005570	000000	.WORD	0	;RMSN
1519	005572	000000	.WORD	0	;RMOF
1520	005574	000000	.WORD	0	;RMCA
1521	005576	000000	.WORD	0	;RMDC
1522	005600	000000	.WORD	0	;RMER2
1523	005602	000000	.WORD	0	;RMMR2
1524	005604	000000	.WORD	0	;RMEC1
1525	005606	000000	.WORD	0	;RMEC2

;DATA/PARAMETER BLOCK - USED FOR ALL DRIVE OPERATION

1526					
1527					
1528					
1529					
1530	005610	000	FMTDPB: .BYTE	0	;DRIVE NUMBER
1531	005611	000	.BYTE	0	;OFFSET VALUE OR FMT22,ECI, AND HCI
1532	005612	000	.BYTE	0	;COMMAND
1533	005613	000	.BYTE	0	;PSEL AND A17 AND A16
1534	005614	000000	.WORD	0	;WORD COUNT (NEG)
1535	005616	000000	.WORD	0	;BUFFER ADDRESS
1536	005620	000	.BYTE	0	;SECTOR ADDRESS
1537	005621	000	.BYTE	0	;TRACK ADDRESS
1538	005622	000000	.WORD	0	;CYLINDER ADDRESS
1539	005624	005540	.WORD	RM.REG	;ERROR TABLE POINTER
1540	005626	000000	.WORD	0	;STATUS-ERROR INDICATOR
1541					;BIT 15 = 1: ERROR OCCURRED
1542					;BIT 07 = 1: DONE
1543					;BIT 14-10 AND BIT 06-03
1544					;INDICATE TYPE OF ERROR
1545					
1546					

1547	005630	000	FMTX: .BYTE	0	;SPECIAL DPB TO WRITE HEADER OF BAD SPOT
1548	005631	000	.BYTE	0	
1549	005632	000	.BYTE	0	
1550	005633	000	.BYTE	0	
1551	005634	177776	.WORD	-2	;ONLY TWO WORDS FOR HEADER
1552	005636	037544	.WORD	RBUF	;BUFFER ADDRESS
1553	005640	000	.BYTE	0	;SECTOR
1554	005641	000	.BYTE	0	;TRACK
1555	005642	000000	.WORD	0	;CYLINDER ADDRESS
1556	005644	000000	.WORD	0	;ERROR TABLE
1557					;DONT SAVE ANYTHING
1558	005646	000000	.WORD	0	;STATUS WORD

;PARAMETER POINTER TABLE

1559					
1560					
1561	005650	005702	001467	001322	TABLE: PAR1,823.,BEGCYL
1562	005656	005715	000005	001326	PAR2,5.,BEGTRK

MAINDEC-11-DZRNA, RM03 FORMATTER
DZRNA8.P11 21-JUL-77 15:32

MACY11 30(1046) 21-JUL-77 16:06 PAGE 31
APT MAILBOX-ETABLE

SEQ 0030

1563 005664 005730 001467 001320
1564 005672 005741 000005 001324
1565 005700 000000

PAR3,823,ENDCYL
PAR4,5,ENDTRK,0

;ASCII MESSAGES FOR ADDRESS PARAMETERS

1569 005702 052123 051101 020124 PAR1: .ASCIZ @START CYL @
1570 005710 054503 020114 000 PAR2: .ASCIZ @START TRK @
1571 005715 123 040524 052122
1572 005722 052040 045522 000040 PAR3: .ASCIZ @END CYL @
1573 005730 047105 020104 054503
1574 005736 020114 000 PAR4: .ASCIZ @END TRK @
1575 005741 105 042116 052040
1576 005746 045522 000040

;SECTOR BUFFER ADDRESS TABLE

1580	005752	037554	ADRTBL: .WORD	BUF	ADDRESS OF SECTOR 0
1581	005754	040560	.WORD	BUF+<516.*1.>	ADDRESS OF SECTOR 1
1582	005756	041564	.WORD	BUF+<516.*2.>	ADDRESS OF SECTOR 2
1583	005760	042570	.WORD	BUF+<516.*3.>	ADDRESS OF SECTOR 3
1584	005762	043574	.WORD	BUF+<516.*4.>	ADDRESS OF SECTOR 4
1585	005764	044600	.WORD	BUF+<516.*5.>	ADDRESS OF SECTOR 5
1586	005766	045604	.WORD	BUF+<516.*6.>	ADDRESS OF SECTOR 6
1587	005770	046610	.WORD	BUF+<516.*7.>	ADDRESS OF SECTOR 7
1588	005772	047614	.WORD	BUF+<516.*8.>	ADDRESS OF SECTOR 8
1589	005774	050620	.WORD	BUF+<516.*9.>	ADDRESS OF SECTOR 9
1590	005776	051624	.WORD	BUF+<516.*10.>	ADDRESS OF SECTOR 10
1591	006000	052630	.WORD	BUF+<516.*11.>	ADDRESS OF SECTOR 11
1592	006002	053634	.WORD	BUF+<516.*12.>	ADDRESS OF SECTOR 12
1593	006004	054640	.WORD	BUF+<516.*13.>	ADDRESS OF SECTOR 13
1594	006006	055644	.WORD	BUF+<516.*14.>	ADDRESS OF SECTOR 14
1595	006010	056650	.WORD	BUF+<516.*15.>	ADDRESS OF SECTOR 15
1596	006012	057654	.WORD	BUF+<516.*16.>	ADDRESS OF SECTOR 16
1597	006014	060660	.WORD	BUF+<516.*17.>	ADDRESS OF SECTOR 17
1598	006016	061664	.WORD	BUF+<516.*18.>	ADDRESS OF SECTOR 18
1599	006020	062670	.WORD	BUF+<516.*19.>	ADDRESS OF SECTOR 19
1600	006022	063674	.WORD	BUF+<516.*20.>	ADDRESS OF SECTOR 20
1601	006024	064700	.WORD	BUF+<516.*21.>	ADDRESS OF SECTOR 21
1602	006026	065704	.WORD	BUF+<516.*22.>	ADDRESS OF SECTOR 22
1603	006030	066710	.WORD	BUF+<516.*23.>	ADDRESS OF SECTOR 23
1604	006032	067714	.WORD	BUF+<516.*24.>	ADDRESS OF SECTOR 24
1605	006034	070720	.WORD	BUF+<516.*25.>	ADDRESS OF SECTOR 25
1606	006036	071724	.WORD	BUF+<516.*26.>	ADDRESS OF SECTOR 26
1607	006040	072730	.WORD	BUF+<516.*27.>	ADDRESS OF SECTOR 27
1608	006042	073734	.WORD	BUF+<516.*28.>	ADDRESS OF SECTOR 28
1609	006044	074740	.WORD	BUF+<516.*29.>	ADDRESS OF SECTOR 29
1610	006046	075744	.WORD	BUF+<516.*30.>	ADDRESS OF SECTOR 30
1611	006050	076750	.WORD	BUF+<516.*31.>	ADDRESS OF SECTOR 31

;REMAINING WORD COUNT TABLE

1615	006052	000402	WCTBL: .WORD	258.	REMAINING WORD COUNT AFTER SECTOR 0
1616	006054	001004	.WORD	258.+<258.*1.>	REMAINING WORD COUNT AFTER SECTOR 1
1617	006056	001406	.WORD	258.+<258.*2.>	REMAINING WORD COUNT AFTER SECTOR 2
1618	006060	002010	.WORD	258.+<258.*3.>	REMAINING WORD COUNT AFTER SECTOR 3

G03

MAINDEC-11-DZRNA, RM03 FORMATTER
DZRNAB.P11 21-JUL-77 15:32MACY11 30(1046) 21-JUL-77 16:06 PAGE 32
APT MAILBOX-ETABLE

SEQ 0031

1619	006062	002412	.WORD	258.+(258.#4.)	;REMAINING WORD COUNT AFTER SECTOR 4
1620	006064	003014	.WORD	258.+(258.#5.)	;REMAINING WORD COUNT AFTER SECTOR 5
1621	006066	003416	.WORD	258.+(258.#6.)	;REMAINING WORD COUNT AFTER SECTOR 6
1622	006070	004020	.WORD	258.+(258.#7.)	;REMAINING WORD COUNT AFTER SECTOR 7
1623	006072	004422	.WORD	258.+(258.#8.)	;REMAINING WORD COUNT AFTER SECTOR 8
1624	006074	005024	.WORD	258.+(258.#9.)	;REMAINING WORD COUNT AFTER SECTOR 9
1625	006076	005426	.WORD	258.+(258.#10.)	;REMAINING WORD COUNT AFTER SECTOR 10
1626	006100	006030	.WORD	258.+(258.#11.)	;REMAINING WORD COUNT AFTER SECTOR 11
1627	006102	006432	.WORD	258.+(258.#12.)	;REMAINING WORD COUNT AFTER SECTOR 12
1628	006104	007034	.WORD	258.+(258.#13.)	;REMAINING WORD COUNT AFTER SECTOR 13
1629	006106	007436	.WORD	258.+(258.#14.)	;REMAINING WORD COUNT AFTER SECTOR 14
1630	006110	010040	.WORD	258.+(258.#15.)	;REMAINING WORD COUNT AFTER SECTOR 15
1631	006112	010442	.WORD	258.+(258.#16.)	;REMAINING WORD COUNT AFTER SECTOR 16
1632	006114	011044	.WORD	258.+(258.#17.)	;REMAINING WORD COUNT AFTER SECTOR 17
1633	006116	011446	.WORD	258.+(258.#18.)	;REMAINING WORD COUNT AFTER SECTOR 18
1634	006120	012050	.WORD	258.+(258.#19.)	;REMAINING WORD COUNT AFTER SECTOR 19
1635	006122	012452	.WORD	258.+(258.#20.)	;REMAINING WORD COUNT AFTER SECTOR 20
1636	006124	013054	.WORD	258.+(258.#21.)	;REMAINING WORD COUNT AFTER SECTOR 21
1637	006126	013456	.WORD	258.+(258.#22.)	;REMAINING WORD COUNT AFTER SECTOR 22
1638	006130	014060	.WORD	258.+(258.#23.)	;REMAINING WORD COUNT AFTER SECTOR 23
1639	006132	014462	.WORD	258.+(258.#24.)	;REMAINING WORD COUNT AFTER SECTOR 24
1640	006134	015064	.WORD	258.+(258.#25.)	;REMAINING WORD COUNT AFTER SECTOR 25
1641	006136	015466	.WORD	258.+(258.#26.)	;REMAINING WORD COUNT AFTER SECTOR 26
1642	006140	016070	.WORD	258.+(258.#27.)	;REMAINING WORD COUNT AFTER SECTOR 27
1643	006142	016472	.WORD	258.+(258.#28.)	;REMAINING WORD COUNT AFTER SECTOR 28
1644	006144	017074	.WORD	258.+(258.#29.)	;REMAINING WORD COUNT AFTER SECTOR 29
1645	006146	017476	.WORD	258.+(258.#30.)	;REMAINING WORD COUNT AFTER SECTOR 30
1646	006150	020100	.WORD	258.+(258.#31.)	;REMAINING WORD COUNT AFTER SECTOR 31
1647					
1648					
1649					

.EVEN

H03

MAINDEC-11-DZRNA, RM03 FORMATTER
DZRMAB.P11 21-JUL-77 15:32

MACY11 30(1046) 21-JUL-77 16:06 PAGE 33
ERROR POINTER TABLE

SEQ 0032

1650
1651
1652
1653
1654
1655
1656
1657
1658
1659
1660
1661
1662
1663
1664
1665
1666
1667
1668
1669
1670
1671
1672
1673
1674
1675
1676
1677
1678
1679
1680
1681
1682
1683
1684
1685
1686
1687
1688
1689
1690
1691
1692
1693
1694
1695
1696
1697
1698
1699
1700
1701
1702
1703
1704
1705

006152

006152 034426
006154 035545
006156 037100
006160 037434

006162 034467
006164 035613
006166 037116
006170 037440

006172 034525
006174 035665
006176 037132
006200 037444

006202 034563
006204 035736
006206 037146
006210 037450

006212 000000
006214 000000
006216 000000
006220 000000

006222 034654
006224 036021
006226 037164
006230 037454

.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 SITE#B. THIS NUMBER INDICATES WHICH ITEM IN THE TABLE IS PERTINENT.
;;NOTE1: IF SITE#B IS 0 THE ONLY PERTINENT DATA IS (\$ERRPC).
;;NOTE2: EACH ITEM IN THE TABLE CONTAINS 4 POINTERS EXPLAINED AS FOLLOWS:

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

\$ERRTB:
;ERROR 1

EM1
DH1
DT1
DF1

;RH11 INTERRUPT OCCURRED (RMAS=0)

;ERROR 2

EM2
DH2
DT2
DF2

;UNEXPECTED ATTENTION OCCURRED

;ERROR 3

EM3
DH3
DT3
DF3

;MASSBUS PARITY ERROR (MCPE=1)

;ERROR 4

EM4
DH4
DT4
DF4

;MASSBUS PARITY ERROR (PAR=1)

;ERROR 5

0
0
0
0

;UNUSED

;ERROR 6

EM6
DH6
DT6
DF6

;RH11 DIDN'T RESPOND TO ADDRESSING

MAINDEC-11-DZRNA, RM03 FORMATTER
DZRNAB.P11 21-JUL-77 15:32

MACY11 30(1046) 21-JUL-77 16:06 PAGE 34
ERROR POINTER TABLE

SEQ 0033

1706				
1707				
1708				
1709	006232	000000	0	;UNUSED
1710	006234	000000	0	
1711	006236	000000	0	
1712	006240	000000	0	
1713				
1714				
1715				
1716	006242	034716	EM10	;DRIVE OFFLINE
1717	006244	036034	DH10	
1718	006246	037166	DT10	
1719	006250	037460	DF10	
1720				
1721				
1722				
1723	006252	034734	EM11	;PERSISTENT DRIVE UNSAFE ERROR
1724	006254	036034	DH10	
1725	006256	037166	DT10	
1726	006260	037460	DF10	
1727				
1728				
1729				
1730	006262	034772	EM12	;UNCORRECTABLE MASSBUS PARITY ERROR
1731	006264	036034	DH10	
1732	006266	037166	DT10	
1733	006270	037460	DF10	
1734				
1735				
1736				
1737	006272	035035	EM13	;SOFTWARE TIMEOUT
1738	006274	036034	DH10	
1739	006276	037166	DT10	
1740	006300	037460	DF10	
1741				
1742				
1743				
1744	006302	035056	EM14	;DRIVE UNSAFE ERROR
1745	006304	036034	DH10	
1746	006306	037166	DT10	
1747	006310	037460	DF10	
1748				
1749				
1750				
1751	006312	035101	EM15	;CONTROLLER/DRIVE ERROR DURING WRITE
1752	006314	036034	DH10	
1753	006316	037166	DT10	
1754	006320	037460	DF10	
1755				
1756				
1757				
1758	006322	035145	EM16	;CONTROLLER/DRIVE ERROR DURING WRITE CHECK
1759	006324	036034	DH10	
1760	006326	037166	DT10	
1761	006330	037460	DF10	

1762					
1763					
1764					
1765	006332	035217			
1766	006334	036263			
1767	006336	037240			
1768	006340	037474			
1769					
1770					
1771					
1772	006342	035275			
1773	006344	036332			
1774	006346	037252			
1775	006350	037500			
1776					
1777					
1778					
1779	006352	035333			
1780	006354	036034			
1781	006356	037166			
1782	006360	037460			
1783					
1784					
1785					
1786	006362	035404			
1787	006364	036034			
1788	006366	037166			
1789	006370	037460			
1790					
1791					
1792					
1793	006372	035453			
1794	006374	036627			
1795	006376	037334			
1796	006400	037520			
1797					
1798					
1799					
1800	006402	035523			
1801	006404	036725			
1802	006406	037346			
1803	006410	037524			
1804					
1805					
1806					
1807					
1808					
1809					
1810					
1811	006412	012737	177777	001400	BEGIN: MOV #1,CHGADR ;SET CHANGE 'RH70 BUS ADDRESS' INDICATOR
1812	006420	000406			BEGIN2 ;START THE PROGRAM
1813	006422	005037	001400		BEGIN1: CLR CHGADR ;CLEAR THE 'CHANGE RH70 ADDRESS' INDICATOR
1814					*****
1815	006426	000240			ST1: NOP
1816	006430	012737	000001	001212	MOV #1,STESTN ;SET TEST NUMBER IN APT MAIL BOX
1817	006436	000005			BEGIN2: RESET ;CLEAR THE BUS

;ERROR 17

EM17
DH17
DT17
DF17

;RETRIES NOT SUCCESSFUL - SECTOR NOT ACCEPTABLE

;ERROR 20

EM20
DH20
DT20
DF20

;DATA ERROR DURING WRITE CHECK

;ERROR 21

EM21
DH10
DT10
DF10

;CONTROLLER/DRIVE ERROR VERIFYING HEADERS

;ERROR 22

EM22
DH10
DT10
DF10

;'HCE' ERROR VERIFYING HEADERS

;ERROR 23

EM23
DH23
DT23
DF23

;CYLINDER FIELD IN HEADER IS NOT CORRECT

;ERROR 24

EM24
DH24
DT24
DF24

;WRITE CHECK ERROR

;*****

.SBTTL MAIN PROGRAM

;*****

```

BEGIN: MOV #1,CHGADR ;SET CHANGE 'RH70 BUS ADDRESS' INDICATOR
      BR BEGIN2 ;START THE PROGRAM
BEGIN1: CLR CHGADR ;CLEAR THE 'CHANGE RH70 ADDRESS' INDICATOR
*****
ST1: NOP
      MOV #1,STESTN ;SET TEST NUMBER IN APT MAIL BOX
BEGIN2: RESET ;CLEAR THE BUS

```

K03

MAINDEC-11-DZRNA, RM03 FORMATTER
DZRNA8.P11 21-JUL-77 15:32

MACY11 30(1046) 21-JUL-77 16:06 PAGE 36
INITIALIZE THE COMMON TAGS

SEQ 0035

1818					.SBTTL INITIALIZE THE COMMON TAGS	
1819					;;CLEAR THE COMMON TAGS (\$CMTAG) AREA	
1820	006440	012706	001114		MOV \$CMTAG,R6	;;FIRST LOCATION TO BE CLEARED
1821	006444	005026			CLR (R6)+	;;CLEAR MEMORY LOCATION
1822	006446	022706	001154		CMP \$SWR,R6	;;DONE?
1823	006452	001374			BNE -6	;;LOOP BACK IF NO
1824	006454	012706	001100		MOV \$STACK,SP	;;SETUP THE STACK POINTER
1825					;;INITIALIZE A FEW VECTORS	
1826	006460	012737	020066	000030	MOV \$ERROR,\$EMTVEC	;;EMT VECTOR FOR ERROR ROUTINE
1827	006466	012737	000340	000032	MOV \$340,\$EMTVEC+2	;;LEVEL 7
1828	006474	012737	023762	000034	MOV \$TRAP,\$TRAPVEC	;;TRAP VECTOR FOR TRAP CALLS
1829	006502	012737	000340	000036	MOV \$340,\$TRAPVEC+2	;;LEVEL 7
1830	006510	012737	023224	000024	MOV \$SPWRON,\$PWAVEC	;;POWER FAILURE VECTOR
1831	006516	012737	000340	000026	MOV \$340,\$PWAVEC+2	;;LEVEL 7
1832	006524	005037	001174		CLR \$ESCAPE	;;CLEAR THE ESCAPE ON ERROR ADDRESS
1833	006530	112737	000001	001131	MOVB \$1,\$SERMAX	;;ALLOW ONE ERROR PER TEST
1834					;;SIZE FOR A HARDWARE SWITCH REGISTER. IF NOT FOUND OR IT IS	
1835					;;EQUAL TO A "-1" SETUP FOR A SOFTWARE SWITCH REGISTER.	
1836	006536	013746	000004		MOV \$ERRVEC,-(SP)	;;SAVE ERROR VECTOR
1837	006542	012737	006576	000004	MOV \$64,\$ERRVEC	;;SET UP ERROR VECTOR
1838	006550	012737	177570	001154	MOV \$DSWR,\$SWR	;;SETUP FOR A HARDWARE SWICH REGISTER
1839	006556	012737	177570	001156	MOV \$DDISP,\$DISPLAY	;;AND A HARDWARE DISPLAY REGISTER
1840	006564	022777	177777	172362	CMP \$-1,\$DSWR	;;TRY TO REFERENCE HARDWARE SWR
1841	006572	001012			BNE 66\$	;;BRANCH IF NO TIMEOUT TRAP OCCURRED
1842						;;AND THE HARDWARE SWR IS NOT = -1
1843	006574	000403			BR 65\$	;;BRANCH IF NO TIMEOUT
1844	006576	012716	006604		MOV \$65\$,(SP)	;;SET UP FOR TRAP RETURN
1845	006602	000002			RTI	
1846	006604	012737	000176	001154	MOV \$SWREG,\$SWR	;;POINT TO SOFTWARE SWR
1847	006612	012737	000174	001156	MOV \$DISPREG,\$DISPLAY	
1848	006620	012637	000004		MOV (SP)+,\$ERRVEC	;;RESTORE ERROR VECTOR
1849						
1850	006624	005037	001214		CLR \$PASS	;;CLEAR PASS COUNT
1851	006630	132737	000200	001227	BITB \$APTSIZE,\$ENV	;;TEST USER SIZE UNDER APT
1852	006636	001403			BEQ 67\$	;;YES,USE NON-APT SWITCH
1853	006640	012737	001230	001154	MOV \$SSWREG,\$SWR	;;NO,USE APT SWITCH REGISTER
1854	006646				67\$:	
1855	006646	000005			RESET	;;CLEAR WORLD
1856	006650	012737	000240	000036	MOV \$PR5,\$TRAPVEC+2	;;CHANGE TRAP PRIORITY BACK TO 5
1857	006656	012737	000240	000032	MOV \$PR5,\$EMTVEC+2	;;CHANGE EMT PRIORITY BACK TO 5
1858	006664	012737	006422	001376	MOV \$BEGIN1,CNTLC	;;CONTROL C' ADDRESS
1859	006672	005227	177777		INC \$-1	;;FIRST START ?
1860	006676	001002			BNE 1\$	;;BR IF NOT
1861	006700	104401	037554		TYPE ,TITLE	;;ADRS OF 'TITLE' MESSAGE
1862						;;4 LINES HAS DELETED 3/14/77
1863	006704	004737	021772		JSR PC,\$TKINT	;;TURN ON THE KEYBOARD INTERRUPT
1864					.SBTTL GET VALUE FOR SOFTWARE SWITCH REGISTER	
1865	006710	005737	000042		TST \$42	;;ARE WE RUNNING UNDER XXDP/ACT?
1866	006714	001012			BNE 68\$	;;BRANCH IF YES
1867	006716	123727	001226	000001	CMPB \$ENV,\$1	;;ARE WE RUNNING UNDER APT?
1868	006724	001406			BEQ 68\$	;;BRANCH IF YES
1869	006726	023727	001154	000176	CMP \$SWR,\$SWREG	;;SOFTWARE SWITCH REG SELECTED?
1870	006734	001005			BNE 69\$	;;BRANCH IF NO
1871	006736	104406			GTSWR	;;GET SOFT-SWR SETTINGS
1872	006740	000403			BR 69\$	
1873	006742	112737	000001	001150	MOV \$1,\$AUTOB	;;SET AUTO-MODE INDICATOR

MAINDEC-11-DZRNA, RMO3 FORMATTER
DZRNA8.P11 21-JUL-77 15:32

MACY11 30(1046) 21-JUL-77 16:06 PAGE 37
GET VALUE FOR SOFTWARE SWITCH REGISTER

SEQ 0036

1874	006750			69\$:	INC	#-1	;CHECK FOR FIRST START
1875	006750	005227	177777		BNE	SETVEC	;BR IF NOT
1876	006754	001035			JSR	PC,BUSADR	;CHECK THE RH11 ADDRESS
1877	006756	004737	040076		MOV	\$RMADR,RMADR	;RH11 ADDRESS
1878	006762	013737	001272 024214		MOV	\$RMVEC,RMVEC	;RH11 VECTOR ADDRESS
1879	006770	013737	001274 024216		TSTB	SENV	;RUN UNDER APT MODE?
1880	006776	105737	001226		BEQ	SETVEC	;NO,DONOT LOAD FROM E-TABLE
1881	007002	001422			TSTB	\$VECT1	;UUT INTERRUPT VECTOR
1882	007004	105737	001256		BEQ	+.10	;NOT CHANGE IF = 0
1883	007010	001403			MOVB	\$VECT1,\$RMVEC	;NEWVALUE
1884	007012	113737	001256 001274		TST	\$BASE	;UUT BASE REGISTER ADDRESS
1885	007020	005737	001262		BEQ	+.10	;NOT CHANGE IF = 0
1886	007024	001403			MOV	\$BASE,\$RMADR	;NEW BASE ADDRESS
1887	007026	013737	001262 001272		MOV	\$RMADR,RMADR	;RH11/70 ADDRESS
1888	007034	013737	001272 024214		MOV	\$RMVEC,RMVEC	;RH11/70 VECTOR
1889	007042	013737	001274 024216				
1890							
1891							
1892							
1893							
1894	007050	004737	016104		SETVEC: JSR	PC,ST.CLK	;START THE CLOCK
1895	007054	004737	024232		JSR	PC,RMINIT	;INITIALIZE THE RMO3 DRIVER
1896	007060	012737	177777 024154		MOV	#-1,SAVEFG	;SET THE SAVE REGISTERS FLAG
1897	007066	005227	177777		INC	#-1	;SEE IF FIRST START
1898	007072	001404			BEQ	11\$	;BR IF YES
1899	007074	032777	000004 172052		BIT	#SW02,\$SWR	;TYPEOUT THE DRIVE STATUS TABLE ?
1900	007102	001101			BNE	10\$	;BR IF NOT
1901	007104	012737	000340 177776	11\$:	MOV	#PR7,PS	;SET PRIORITY TO 7
1902	007112	005004			CLR	R4	;DRIVE TABLE POINTER
1903	007114	104401	001203		TYPE	, \$CRLF	;CR-LF
1904	007120	104401	032436		TYPE	,SYSTAT	;TYPE STATUS HEADING
1905	007124			1\$:			
1906	007124	010446			MOV	R4,-(SP)	;SAVE R4 FOR TYPEOUT
1907							;TYPE DRIVE NUMBER
1908	007126	104403			TYPOS		;GO TYPE--OCTAL ASCII
1909	007130	002			.BYTE	2	;TYPE 2 DIGIT(S)
1910	007131	000			.BYTE	0	;SUPPRESS LEADING ZEROS
1911	007132	104401	032501		TYPE	,LIN4SP	;SPACES
1912	007136	105764	024066		TSTB	DRVSTA(R4)	;CHECK DRIVE'S STATUS
1913	007142	100416			BMI	4\$	;BR IF UNSAFE
1914	007144	001020			BNE	5\$	;BR IF ONLINE
1915	007146	105764	024076		TSTB	DRVTYP(R4)	;SEE IF OFFLINE OR NONEXISTENT
1916	007152	001404			BEQ	2\$	;BR IF NONEXISTENT
1917	007154	100006			BPL	3\$	;BR IF OFFLINE
1918	007156	104401	032355		TYPE	,NOTRM	;DRIVE NOT AN RMO3
1919	007162	000443			BR	4\$	;CHECK NEXT DRIVE
1920	007164	104401	032372	2\$:	TYPE	,NOTPRS	;DRIVE NOT PRESENT
1921	007170	000440			BR	4\$	;CHECK NEXT DRIVE
1922	007172	104401	032334	3\$:	TYPE	,UNTOFF	;DRIVE OFFLINE
1923	007176	000410			BR	6\$	;PRINT DRIVE TYPE
1924	007200	104401	032407	4\$:	TYPE	,NOTSAF	;DRIVE UNSAFE
1925	007204	000405			BR	6\$	;PRINT DRIVE TYPE
1926	007206	104401	032345	5\$:	TYPE	,UNTON	;DRIVE ONLINE
1927	007212	156437	024202 001404		BISB	ATABIT(R4),DEVMP	;SET DEVICE MAP
1928	007220	104401	032503	6\$:	TYPE	,LINSF	;SPACES
1929	007224	012737	032417 007270		MOV	\$RMO3B,8\$	;ADDRESS OF RPO4 MESSAGE

M03

MAINDEC-11-DZMA, RMO3 FORMATTER
DZMA8.P11 21-JUL-77 15:32

MACY11 30(1046) 21-JUL-77 16:06 PAGE 38
GET VALUE FOR SOFTWARE SWITCH REGISTER

SEQ 0037

1930	007232	132764	000001	024076	BITB	#BIT00,DRVTP(R4)	;RMO3 ?
1931	007240	001012			BNE	7\$	;BR IF YES
1932	007242	012737	032424	007270	MOV	#RPOS,8\$	;ADDRESS OF RPOS MESSAGE
1933	007250	132764	000002	024076	BITB	#BIT01,DRVTP(R4)	;RPOS ?
1934	007256	001003			BNE	7\$	;BR IF YES
1935	007260	012737	032431	007270	MOV	#RMO3A,8\$	;ADDRESS OF RMO3 MESSAGE
1936	007266	104401			TYPE		;TYPE THE DRIVE TYPE MESSAGE
1937	007270	000000			.WORD	0	;MESSAGE ADDRESS HERE
1938	007272	104401	001203		9\$: TYPE	\$CRLF	;CR-LF
1939	007276	005204			INC	R4	;INCREMENT DRIVE NUMBER/TABLE POINTER
1940	007300	020427	007010		CMP	R4,#8.	;FINISHED ?
1941	007304	001307			BNE	1\$	;BR IF NOT
1942	007306	104401	001203		10\$: TYPE	\$CRLF	;CR-LF
1943					.		
1944	007312	105737	001150		APT: CODING	FOR APT---LOAD PARAMETERS FROM E-TABLE	
1945	007316	001544			TSTB	\$AUTOB	;RUN UNDER APT ?
1946	007320	005037	001410		BEQ	M1	;NO DON'T BOTHER
1947	007324	005737	001264		CLR	EDIT	;CLEAR THE TEXT LAST TRACK FLAG
1948	007330	001406			TST	\$DEVN	;DEVICE MAP AVAILABLE FROM APT MAIL BOX
1949	007332	013737	001264	001404	BEQ	1\$	;BRANCH IF NONE
1950	007340	042737	177400	001404	MOV	\$DEVN,DEVMP	;LOAD IT FORM MAIL BOX
1951	007346	005737	001404		BIC	#177400,DEVMP	;ALLOW MAX 8 DRIVES
1952	007352	001002			1\$: TST	DEVMP	;CHECK DEVICE MAP
1953	007354	000137	014506		BNE	MSFX	;BRANCH IF DRIVES ASSIGNED
1954	007360	000037	001326		JMP	SEOP	;OTHERWISE, EXIT
1955	007364	000037	001322		MSFX: CLR	BEGTRK	;ALWAY STARTS FROM 0 TRK
1956	007370	012737	001466	001320	CLR	BEGCYL	;ALWAYS STARTS FROM 0 CYLINDER
1957	007376	012737	000004	001324	MOV	#822,ENDCYL	;LAST CYLINDER
1958	007404	005037	001402		MOV	#4,ENDTRK	;LAST TRACK
1959	007410	012737	177777	001316	CLR	WRAP	;CLEAR WRAP AROUND FLAG
1960	007416	005737	001266		MOV	#-1,MODE	;SET ORMAT AND VERIFY MODE
1961	007422	001402			TST	\$CDW1	;APT PARAMETER = 0
1962	007424	005037	001316		BEQ	+.6	;YES, DEFAULT TO -1
1963	007430	012737	177777	001362	CLR	MODE	;NO, SET TO CHECK MODE
1964	007436	005737	001270		MOV	#-1,SEC30	;SET 32 SECTOR
1965	007442	001012			TST	\$CDW2	;APT PARAMETER = 0
1966	007444	012737	020100	001354	BNE	1\$	;NO, 30 SECTOR
1967	007452	012737	157700	001356	MOV	#(258.*32.),WC	;COUNT OF WORDS
1968	007460	012737	000037	001364	MOV	#-(258.*32.),MWC	;WORD COUNT
1969	007466	000413			MOV	#31.,MAXSEC	;MAX 32 SECTOR
1970	007470	012737	017074	001354	BR	2\$	
1971	007476	012737	160704	001356	1\$: MOV	#(258.*30.),WC	;COUNT OF WORDS
1972	007504	005037	001362		MOV	#-(258.*30.),MWC	;WORD COUNT
1973	007510	012737	000035	001364	CLR	SEC30	;30 SECTOR
1974	007516	005737	001316		MOV	#29.,MAXSEC	;MAX 30 SECTOR
1975	007522	001006			2\$: TST	MODE	;CHECK MODE ?
1976	007524	012737	133331	001340	BNE	+.16	;NO
1977	007532	012737	133331	001342	MOV	#133331,PATA	;INITIATE PATTERN
1978	007540	012737	000002	001336	MOV	#133331,PATB	;INITIATE PATTERN
1979	007546	012700	000001		MOV	#2,PATSEL	;WORSE CASE
1980	007552	005001			MOV	#1,RO	;UUT STARTS AT DRIVE 0
1981	007554	030037	001404		CLR	R1	;DRIVE NUMBER
1982	007560	001005			3\$: BIT	RO,DEVMP	;BIT OF DEVICE MAP SET ?
1983	007562	000241			BNE	4\$	;FIND THE DRIVE # FROM DEVICE MAP
1984	007564	006100			CLC		
1985	007566	003416			ROL	RO	;NEXT DRIVE
					BLE	6\$	;BRANCH IF ALL DRIVES ARE DONE

N03

MAINDEC-11-DZRNA, RM03 FORMATTER
DZRNAB.P11 21-JUL-77 15:32

MACY11 30(1046) 21-JUL-77 16:06 PAGE 39
GET VALUE FOR SOFTWARE SWITCH REGISTER

SEQ 0038

1986 007570 005201
1987 007572 000770
1988 007574 010137 001220
1989 007600 040037 001404
1990 007604 105761 024066
1991 007610 003003
1992 007612 005237 001216
1993 007616 000756
1994 007620 000137 010542
1995 007624 000137 014506

```

INC R1 ; INCREMENT DRIVE #
BR 3$ ; LOOP
4$: MOV R1, DRIVE ; LOAD R1 INTO DRIVE
BIC R0, DEVMP ; CLEAR THE BIT FROM DEVICE MAP
TSTB DRVSTA(R1) ; DRIVE ON LINE ?
BGT 5$ ; BRANCH IF DRIVE ON LINE
INC $DEVCT ; INCREMENT DEVICE COUNT TO CHANGE MAIL BOX
BR 3$ ; LOOP BACK
5$: JMP M5 ; START TO FORMAT OR CHECK
6$: JMP $EOP ; END OF PASS FOR ALL DRIVES ON $DEVCT

```

; SEE WHICH MODE THE PROGRAM IS TO BE RUN IN.
; MODES ARE: 'FORMAT & VERIFY' OR 'CHECK FORMAT'

2001 007630 005037 001316
2002 007634 104401 032713
2003 007640 104411
2004 007642 012601
2005 007644 105711
2006 007646 001414
2007 007650 122711 000103
2008 007654 001406
2009 007656 122711 000106
2010 007662 001411
2011 007664 104401 001202
2012 007670 000757
2013 007672 104401 032766
2014 007676 000410
2015 007700 104401 032745
2016 007704 000402
2017 007706 104401 033001
2018 007712 012737 177777 001316

```

M1: CLR MODE ; SET MODE TO 'CHECK FORMAT'
TYPE ,MMODE ; TYPE 'PROGRAM MODE'
RDLIN ; READ THE KEYBOARD
MOV (SP)+, R1 ; GET ADDRESS OF INPUT
TSTB (R1) ; 'CR' ENTERED (DEFAULT)
BEQ 2$ ; BR IF YES
CMPB #'C', (R1) ; 'CHECK' ?
BEQ 1$ ; BR IF YES
CMPB #'F', (R1) ; 'FORMAT' ?
BEQ 3$ ; BR IF YES
TYPE ,SQUES ; NO CORRECT ENTRY
BR M1 ; TRY AGAIN
1$: TYPE MHECK ; TYPE REST OF 'CHECK'
BR M1A ; GET STARTING ADDRESS
2$: TYPE MFORMAT ; TYPE DEFAULT MESSAGE
BR 4$ ; SET UP MODE
3$: TYPE MORMAT ; TYPE REST OF 'FORMAT'
4$: MOV #-1, MODE ; SET MODE TO 'FORMAT & VERIFY'

```

; FIND OUT IF FORMAT IS TO BE IN 30 OR 32 SECTOR MODE

2022 007720 104401 033021
2023 007724 104411
2024 007726 012601
2025 007730 122711 000131
2026 007734 001410
2027 007736 122711 000116
2028 007742 001424
2029 007744 105711
2030 007746 001403
2031 007750 104401 001202
2032 007754 000761
2033 007756 104401 033072
2034 007762 012737 020100 001354
2035 007770 012737 157700 001356
2036 007776 012737 177777 001362
2037 010004 012737 000037 001364
2038 010012 000415
2039 010014 104401 033151
2040 010020 012737 017074 001354
2041 010026 012737 160704 001356

```

M1A: TYPE ,MSIZE ; TYPE FORMAT MODE REQUEST
RDLIN ; READ THE KEYBOARD
MOV (SP)+, R1 ; ADDRESS OF ENTRY
CMPB #'Y', (R1) ; IS ENTRY A 'Y' ?
BEQ 1$ ; BR IF IT IS
CMPB #'N', (R1) ; IS ENTRY A 'N' ?
BEQ 2$ ; BR IF IT IS
TSTB (R1) ; DEFAULT MODE ?
BEQ 1$ ; BR IF IT IS
TYPE ,SQUES ; TYPE ''
BR M1A ; TRY AGAIN
1$: TYPE MSEC22 ; TYPEOUT MODE SELECTED
MOV #-258.*32., WC ; TRACK SIZE IN 32 SECTOR MODE
MOV #-258.*32., MWC ; 2'S COMPLEMENT WORD COUNT
MOV #-1, SEC30 ; 32 SECTOR INDICATOR
MOV #31., MAXSEC ; MAX SECTOR ADDRESS IN 32 SECTOR MODE
BR M1B ; CONTINUE
2$: TYPE MSEC20 ; TYPE OUT 30 SECTOR MODE SELECTED
MOV #-258.*30., WC ; TRACK SIZE IN 30 SECTOR MODE
MOV #-258.*30., MWC ; 2'S COMPLEMENT WORD COUNT

```

2042	010034	005037	001362		CLR	SEC30	;30 SECTOR INDICATOR
2043	010040	012737	000035	001364	MOV	#29, MAXSEC	;MAX SECTOR ADDRESS IN 30 SECTOR MODE
2044	010046	005037	001326		CLR	BEGTRK	;CLEAR STARTING TRACK ADDRESS
2045	010052	005037	001322		CLR	BEGCYL	;CLEAR BEGINNING CYLINDER ADDRESS
2046	010056	005037	001402		CLR	WRAP	;CLEAR WRAP AROUND FLAG
2047	010062	012737	000004	001324	MOV	#4, ENDTRK	;SETUP END TRACK ADDRESS
2048	010070	012737	000002	001336	MOV	#2, PATSEL	;SETUP FOR WORST CASE PATTERN
2049	010076	112737	177777	005610	MOVB	#-1, FMTDPB	;SETUP INVALID DRIVE NUMBER
2050	010104	000400			BR	MO	;BRANCH TO START
2051							
2052							
2053							
2054	010106	012737	016324	001376	MO:	MOV	#0ENTER, CNTLC
2055	010114	005037	001126		CLR	SERITL	;CONTROL C ABORT ENTRANCE
2056	010120	004737	021772		JSR	PC, STKINT	;CLEAR THE ERROR ACCUMULATOR
2057	010124	104401	032455		TYPE	, MUNIT	;INITIALIZE THE TTY KEYBOARD
2058	010130	104411			RDLIN		;ASK FOR DRIVE NUMBER
2059	010132	012601			MOV	(SP)+, R1	;READ THE KEYBOARD
2060	010134	004537	017656		JSR	R5, CK.CHR	;ADDRESS OF ENTRY
2061	010140	010164			3\$		;CHECK ONE CHARACTER
2062	010142	010154			2\$		;ILLEGAL CHARACTER
2063	010144	010164			3\$		;CARRIAGE RETURN
2064	010146	010154			2\$		
2065	010150	010172			4\$		;DIGIT 0-7
2066	010152	010164			3\$		;DIGIT 8-9
2067	010154	005002		2\$:	CLR	R2	;SELECT DRIVE ZERO
2068	010156	104401	032477		TYPE	, MORVD	;GO TYPE DEFAULT DRIVE NUMBER
2069	010162	000403			BR	4\$	;GO SEE IF DRIVE ZERO IS THERE
2070	010164	104401	001202		3\$:	TYPE	, SQUES
2071	010170	000746			MO		;TYPE ''
2072	010172	010237	001220		4\$:	MOV	R2, DRIVE
2073	010176	005702			TST	R2	;ASK FOR DRIVE NUMBER AGAIN
2074	010200	001004			BNE	5\$	;SAVE DRIVE NUMBER
2075	010202	122737	000011	000041	CMPB	#11, 41	;SEE IF DRIVE 0
2076	010210	001431			BEQ	7\$	;BR IF NOT
2077	010212	004737	016104		5\$:	JSR	PC, ST.CLK
2078	010216	004737	024232		JSR	PC, RMINIT	;PROGRAM LOADED FROM AN RMO3 ?
2079	010222	012737	177777	024154	MOV	#-1, SAVEFG	;BR IF IT WAS, CAN'T FORMAT THE DRIVE
2080	010230	012737	177777	024156	MOV	#-1, SEEKFG	;START THE CLOCK
2081	010236	005037	177776		CLR	PS	;GO SEE WHAT DRIVES ARE AVAILABLE
2082	010242	105762	024066		TSTB	DRVSTA(R2)	;SAVE THE REGISTERS
2083	010246	003015			BGT	8\$	;SET 'NO OPTIMIZATION' FLAG
2084	010250	001403			BEQ	6\$	;SET PRIORITY BACK TO ZERO
2085	010252	104401	032661		TYPE	, MUSDR	;LOOK AT DRIVE STATUS
2086	010256	000713			BR	MO	;BRANCH IF ONLINE
2087	010260	105762	024076	6\$:	TSTB	DRVTP(R2)	;BR IF DRIVE NOT AVAILABLE
2088	010264	001003			BNE	7\$	;GO GET DRIVE NUMBER AGAIN
2089	010266	104401	032560		TYPE	, MORNP	;A DRIVE PRESENT?
2090	010272	000705			BR	MO	;BR IF SO
2091	010274	104401	032605	7\$:	TYPE	, MER11	;TYPE 'DRIVE NOT PRESENT'
2092	010300	000702			BR	MO	;GO GET DRIVE NUMBER AGAIN
2093	010302	123737	001220	005610	8\$:	CMPB	DRIVE, FMTDPB
2094	010310	001422			BEQ	M2	;SAME DRIVE AS LAST TIME ?
2095	010312	113737	001220	005610	MOVB	DRIVE, FMTDPB	;BR IF IT IS
2096	010320	012737	000632	001320	MOV	#410, ENDCYL	;SETUP DRIVE ADDRESS
2097	010326	132762	000003	024076	BITB	#BIT00!BIT01, DRVTP(R2)	;SETUP FOR RMO3

;GO FIND OUT WHAT DRIVE

C04

 MAINDEC-11-DZRNA, RMO3 FORMATTER
 DZRNA8.P11 21-JUL-77 15:32

 MACY11 30(1046) 21-JUL-77 16:06 PAGE 41
 GET VALUE FOR SOFTWARE SWITCH REGISTER

SEQ 0040

```

2098 010334 001003      BNE      9$      ;BR IF EITHER
2099 010336 012737 001466 001320 9$: MOV      #822, ENDCYL ;SETUP ENDING CYLINDER FOR RMO3
2100 010344 005037 001326      CLR      BEGTRK ;CLEAR STARTING TRACK ADDRESS
2101 010350 005037 001322      CLR      BEGCTL ;CLEAR STARTING CYLINDER ADDRESS
2102 010354 000450      BR      M2      ;GET ADDRESS LIMITS FROM THE OPERATOR
2103
2104      ;GET ADDRESS LIMITS
2105
2106 010356 104401 032506      M2: TYPE      ,ENTADR ;'ENTER ADDRESS LIMITS'
2107 010362 004737 016502      JSR      PC,PARENT ;GET THE ADDRESS LIMITS
2108 010366 005037 001402      CLR      WRAP ;CLEAR WRAP FLAG
2109 010372 023737 001320 001322 CMP      ENDCYL,BEGCTL ;SEE IF ENDING CYLINDER EQ TO GT THAN BEGINNING
2110 010400 001402      BEQ      2$      ;BR IF ON THE SAME CYLINDER
2111 010402 103410      BLO      4$      ;BR IF LESS
2112 010404 000413      BR      M4      ;TO CHECK DRIVE
2113 010406 023737 001324 001326 2$: CMP      ENDCYL,BEGTRK ;SEE IF ENDING TRACK EQ OR GT THAN BEGINNING
2114 010414 103007      BHS      M4      ;BR IF YES
2115 010416 104401 033250      3$: TYPE      ,MADRER ;INVALID ADDRESS ENTERED
2116 010422 000755      BR      M2      ;TRY AGAIN
2117 010424 012737 177777 001402 4$: MOV      #1,WRAP ;SET WRAP AROUND FLAG
2118 010432 000400      BR      M4
2119
2120      ;GO GET DATA PATTERN FOR FORMAT
2121
2122 010434 005737 001316      M4: TST      MODE ;VERIFY MODE ?
2123 010440 001007      BNE      M4A ;NO
2124 010442 012737 133331 001340 MOV      #133331,PATA ;VERIFY MODE USE WORST CASE DATA
2125 010450 012737 133331 001342 MOV      #133331,PATB ;INITIAL THE DATA PATTERN -SHIFT 3 BITS
2126 010456 000431      BR      M5 ;START TO FORMAT AND CHECK
2127 010460 104401 033352      M4A: TYPE      ,MSELP ;GO TYPE 'SELECT PATTERN'
2128 010464 104411      RDLIN ;READ THE KEYBOARD
2129 010466 012601      MOV      (SP)+,R1 ;ENTRY ADDRESS
2130 010470 004537 017656      JSR      R5,CK.CHR ;CHECK ONE CHARACTER
2131 010474 010530      3$ ;ILLEGAL CHARACTER
2132 010476 010510      1$ ;CARRIAGE RETURN
2133 010500 010530      3$ ;
2134 010502 010510      1$ ;
2135 010504 010522      2$ ;DIGIT 0-7
2136 010506 010530      3$ ;DIGIT 8-9
2137 010510 104401 033502      1$: TYPE      ,MPATD ;TYPE DEFAULT PATTERN
2138 010514 012702 000002      MOV      #2,R2 ;WORST CASE PATTERN
2139 010520 000406      BR      4$ ;GO SAVE PATTERN
2140 010522 020227 000002      2$: CMP      R2,#2 ;IS # LARGER THAN 2
2141 010526 101403      BLOS      4$ ;BRANCH IF NOT
2142 010530 104401 001202      3$: TYPE      ,SQUES ;TYPE '?'
2143 010534 000737      BR      M4 ;RETYPE LINE
2144 010536 010237 001336      4$: MOV      R2,PATSEL ;SAVE PATTERN SELECTED
2145
2146      ;GO TYPE 'STARTING FORMAT ON DRIVE N'
2147
2148 010542 012737 016344 000060 M5: MOV      #NEWSVC,#TKVEC ;VECTOR FOR CONTROL-0
2149 010550 005037 177776      CLR      #PSW ;LEVEL 0
2150 010554 005737 001316      TST      MODE ;'FORMAT' OR 'CHECK' MODE ?
2151 010560 001403      BEQ      1$ ;BR IF 'CHECK' MODE
2152 010562 104401 033515      TYPE      ,MSFOU ;TYPE 'STARTING FORMAT ON DRIVE N'
2153 010566 000402      BR      2$

```

MAINDEC-11-DZRNA, RM03 FORMATTER
DZRNAB.P11 21-JUL-77 15:32

MACY11 30(1046) 21-JUL-77 16:06 PAGE 42
GET VALUE FOR SOFTWARE SWITCH REGISTER

SEQ 0041

2154 010570 104401 033552
2155 010574
2156 010574 013746 001220
2157
2158 010600 104403
2159 010602 001
2160 010603 000
2161 010604 104401 001203
2162
2163
2164
2165

1\$: TYPE ,MSCHK ;TYPE 'STARTING CHECK ON DRIVE N'
2\$:
MOV DRIVE,-(SP) ;SAVE DRIVE FOR TYPEOUT
;TYPE DRIVE NUMBER
TYP0S ;GO TYPE--OCTAL ASCII
;TYPE 1 DIGIT(S)
.BYTE 1 ;SUPPRESS LEADING ZEROS
.BYTE 0 ;CR-LF
TYPE ,SCRLF

;THIS CODE RETRIEVES THE BAD SPOT FILE FROM THEPACK (IF FORMATTED PACK)
;SET UP DPB BLOCK READ CYL 822. TRK 4

2166 010610 005037 001346
2167 010614 013737 001220 005610
2168 010622 012737 157700 005614
2169 010630 012737 037554 005616
2170 010636 105037 005620
2171 010642 112737 000004 005621
2172 010650 012737 001466 005622
2173 010656 112737 000173 005612
2174 010664 012737 177777 001406
2175 010672 112737 000143 005612
2176 010700 112737 000020 005611
2177 010706 004037 024736
2178 010712 005610
2179 010714 000774
2180 010716 005737 005626
2181 010722 001775
2182 010724 112737 000173 005612
2183 010732 012737 010732 001124
2184 010740 012706 001100
2185 010744 004037 024736
2186 010750 005610
2187 010752 000774
2188 010754 005737 005626
2189 010760 001775
2190 010762 100023
2191 010764 012737 011010 001174
2192 010772 004737 015230
2193 010776 104010
2194 011000 104011
2195 011002 104012
2196 011004 104013
2197 011006 104014
2198 011010 004737 015330
2199 011014 022737 000003 001346
2200 011022 103462
2201 011024 005237 001346
2202 011030 000745
2203 011032 005037 001174
2204 011036 005037 001346
2205 011042 005737 005626
2206 011046 100450
2207 011050 032737 140000 037554
2208 011056 001444
2209 011060 032737 010000 037554

RETBAD: CLR RETRY ;CLEAR THE RETRY FLAG
MOV DRIVE,FMTDPB ;LOAD DRIVE #
MOV #-<32.*258.> FMTDPB+4 ;WORD COUNT FULL TRACK INCLUDING HEAD
MOV #BUF,FMTDPB+6 ;BUFFER ADDRESS
CLRB FMTDPB+10 ;SECTOR 0
MOVB #4,FMTDPB+11 ;TRACK 4
MOV #822,FMTDPB+12 ;CYL 822
MOVB #RDHD,FMTDPB+2 ;READ HEAD AND DATA COMMAND
MOV #-1,PACK ;RESET THE MFG AVAILABLE FLAG
MOVB #SETFMT,FMTDPB+2 ;SET 16 BIT MODE
MOVB #20,FMTDPB+1 ;FMT SET
1\$: JSR RO,RM03 ;SET THE FORMAT BIT
FMTDPB
BR 1\$
2\$: TST FMTDPB+16 ;THE COMMAND IS DONE ?
BEQ 2\$;BRANCH IF NOT
MOVB #RDHD,FMTDPB+2 ;RELOAD THE READ HEAD AND DATA COMMAND
MOV #.SLPERR ;LOOP ON ERROR ADDRESS
MOV #STACK,SP ;RESET STACK POINT
3\$: JSR RO,RM03 ;READ THE LAST TRACK
FMTDPB
BR 3\$
4\$: TST FMTDPB+16 ;BRANCH IF QUEUE FAIL
BEQ 4\$;ALL DONE
BPL 6\$;BRANCH IF NOT
MOV #5\$,SESCAPE ;BRANCH IF NO ERROR
JSR PC,ERINDX ;ESCAPE TO 5\$ ON ERROR
;SEE WHICH ERROR
ERROR 10
ERROR 11
ERROR 12
ERROR 13
ERROR 14
5\$: JSR PC,LOP.CK ;CHECK LOOP ON ERROR
CMP #3,RETRY ;DONT' TRY OVER 3 TIMES
BLO 8\$
INC RETRY
BR 3\$;TRY AGAIN
6\$: CLR SESCAPE
CLR RETRY
TST FMTDPB+16 ;ANY ERROR ?
BMI 8\$;BRANCH IF ANY
BIT #BIT15!BIT14,#BUF ;HEADER INFORMATION OK ?
BEQ 8\$;BRANCH IF NOT
BIT #BIT12,#BUF ;FMT BET MUST BE SET

E04

MAINDEC-11-DZRNA, RM03 FORMATTER
DZRNA8.P11 21-JUL-77 15:32MACY11 30(1046) 21-JUL-77 16:06 PAGE 43
GET VALUE FOR SOFTWARE SWITCH REGISTER

SEQ 0042

2210	011066	001440			BEQ	8\$	; BRANCH IF NOT SET
2211	011070	042737	150000	037554	BIC	#150000, 2#BUFP	; GET THE CYLINDER ADDRESS
2212	011076	022737	001466	037554	CMP	#822., 2#BUFP	; ON THE LAST CYLINDER ?
2213	011104	001031			BNE	8\$	; BRANCH IF NOT
2214	011106	022737	002000	037556	CMP	#2000, 2#BUFP+2	; ON TRACK 4 AND SECTOR 0
2215	011114	001025			BNE	8\$	; BRANCH IF NOT
2216	011116	005737	037560		TST	2#BUFP+4	; SERIAL NUMBER SHOULD NOT 0
2217	011122	001003			BNE	7\$	; BRANCH IF NOT 0
2218	011124	005737	037562		TST	2#BUFP+6	; SECOND WORD OF SERIAL NUMBER
2219	011130	001417			BEQ	8\$	; BRANCH IF ZERO
2220	011132	005737	037564		TST	2#BUFP+10	; THE THIRD WORD MUST BE 0
2221	011136	001014			BNE	8\$	; BRANCH IF NOT
2222	011140	022737	177777	037566	CMP	#-1, 2#BUFP+12	; AN ALIGNMENT PACK ?
2223	011146	001002			BNE	.+6	; BRANCH IF NOT
2224	011150	000137	015212		JMP	REJEC1	
2225	011154	005737	037566		TST	2#BUFP+12	; A DATA PACK ?
2226	011160	001003			BNE	8\$	; BRANCH IF NOT IDENTIFIED
2227	011162	012737	000400	001406	MOV	#400, PACK	; SET MFG BAD SPOT FILE AVAILABLE FLAG
2228	011170	000240			NOP		; DONE
2229					8\$:		; THIS CODE INITIALIZES THE BAD16, BAD18, USTAB, USSAV TABLES
2230					:		
2231					:		
2232	011172	012706	001100		TABINT: MOV	#STACK, SP	; INITIAL STACK POINT
2233	011176	005046			CLR	-(SP)	; TERMINATOR
2234	011200	012746	004430		MOV	#USSAV, -(SP)	; USSAV TABLE ADDRESS
2235	011204	012746	002420		MOV	#BAD18, -(SP)	; MFG 18 BIT FILE ADDRESS
2236	011210	012746	001414		MOV	#BAD16, -(SP)	; MFG 16 BIT FILE ADDRESS
2237	011214	012703	003424		MOV	#USTAB, R3	; USER BAD SPOT FILE ADDRESS
2238	011220	012704	000004		1\$: MOV	#4, R4	; FIRST 4 WORDS SET TO 0 TEMPORARY
2239	011224	005023			2\$: CLR	(R3)+	
2240	011226	005304			DEC	R4	
2241	011230	001375			BNE	2\$	; BRANCH IF NOT DONE
2242	011232	012704	000374		MOV	#252., R4	; SET -1 TO ALL THE REST LOCATIONS
2243	011236	012705	177777		MOV	#-1, R5	
2244	011242	010523			3\$: MOV	R5, (R3)+	
2245	011244	005304			DEC	R4	; DECREMENT WORD COUNT
2246	011246	001375			BNE	3\$	; BRANCH IF NOT DONE
2247	011250	012603			MOV	(SP)+, R3	; NEXT TABLE
2248	011252	001362			BNE	1\$	; LOOPING BACK IF NOT TERMINATOR
2249							
2250							
2251							
2252							
2253	011254	005737	001406		TABLD: TST	PACK	; BAD SPOT FILE AVAILABLE FROM PACK ?
2254	011260	003440			BLE	5\$	; BRANCH IF NOT
2255	011262	012702	037560		MOV	#BUFP+4, R2	; LOAD BAD16 TABLE FIRST
2256	011266	012703	001414		MOV	#BAD16, R3	; TABLE ADDRESS
2257	011272	012704	000400		MOV	#256., R4	; WORD COUNT
2258	011276	012223			1\$: MOV	(R2)+, (R3)+	; LOAD THE WHOLE SECTOR
2259	011300	005304			DEC	R4	; BRANCH IF NOT DONE
2260	011302	001375			BNE	1\$	
2261	011304	012702	040564		MOV	#BUFP+520., R2	; 258 X 2 + 2 X 2 + BUFP
2262	011310	012703	002420		MOV	#BAD18, R3	; TABLE ADDRESS
2263	011314	012704	000400		MOV	#256., R4	; WORD COUNT
2264	011320	012223			2\$: MOV	(R2)+, (R3)+	; LOAD INTO TABLE
2265	011322	005304			DEC	R4	; DECREMENT WORD COUNT

F04

MAINDEC-11-DZRNA, RMO3 FORMATTER
DZRNAB.P11 21-JUL-77 15:32

MACY11 30(1046) 21-JUL-77 16:06 PAGE 44
GET VALUE FOR SOFTWARE SWITCH REGISTER

SEQ 0043

2266	011324	001375		BNE	2\$	; BRANCH IF NOT DONE
2267	011326	012702	052634	MOV	#BUF+4+<11.*516.	> R2 : SECTOR 11
2268	011332	005737	001362	TST	SEC30	; FORMAT IN 16 BIT MODE
2269	011336	100402		BMI	3\$	; BRANCH IF IT IS
2270	011340	012702	051630	MOV	#BUF+4+<10.*516.	> R2 : ECTOR 10
2271	011344	012703	004430	3\$: MOV	#USSAV, R3	; LOAD THE TABLE
2272	011350	012704	000400	4\$: MOV	#256, R4	; WORD COUNT
2273	011354	012223		4\$: MOV	(R2)+, (R3)+	
2274	011356	005304		DEC	R4	
2275	011360	001375		BNE	4\$	
2276	011362	000240		5\$: NOP		; DONE
2277						
2278						; THIS CODE ASK O.P TO ENTER A SERIAL NUMBER FOR UN FORMATTED PACK
2279						
2280						
2281						
2282	011364	005737	001406	SERNL: TST	PACK	; BAD SPOT FILE AVAILABLE ?
2283	011370	003124		BGT	4\$	; BRANCH IF AVAILABLE
2284	011372	105737	001150	TSTB	\$AUTOB	; RUN UNDER APT ?
2285	011376	001144		BNE	5\$	; BRANCH IF IT IS
2286	011400	004737	021772	JSR	PC, \$TKINT	; INITIAL THE KEYBOARD
2287	011404	104401	001203	TYPE	, \$CRLF	; TYPE CRLF
2288	011410	104401	033777	TYPE	, \$MSG1	; "ENTER TWO WORDS FOR SN"
2289	011414	104401	034065	1\$: TYPE	, \$SN1	
2290	011420	012702	000012	MOV	#10, R2	; MAXIMUM 10 OCTAL DIGITS ALLOWED
2291	011424	005037	001414	CLR	BAD16	; CLEAR THE LSW OF THE SERIAL NUMBER
2292	011430	005037	001416	CLR	BAD16+2	; CLEAR THE MSW OF THE SERIAL NUMBER
2293	011434	104411		RDLIN		; READ IN THE STRING
2294	011436	012601		MOV	(SP)+, R1	; ADDRESS OF THE READ IN STRING
2295	011440	105711		2\$: TSTB	(R1)	; TERMINATOR OF THE INPUT STRING
2296	011442	001471		BEQ	3\$	; BRANCH IF IT IS
2297	011444	121127	000060	CMPB	(R1), #'0	; LESS THAN ASCII 0 ?
2298	011450	103761		BLO	1\$	; ENTER AGAIN IF IT IS
2299	011452	121127	000067	CMPB	(R1), #'7	; HIGH THAN ASCII 7 ?
2300	011456	101356		BHI	1\$	; ENTER AGAIN IF SO
2301	011460	013746	001414	MOV	BAD16, -(SP)	; SAVE THE LSW
2302	011464	042737	100000 001414	BIC	#BIT15, BAD16	; CLEAR THE SIGN BIT OF LSW
2303	011472	006137	001414	ROL	BAD16	; ROTATE THE LSW FIVE TIMES
2304	011476	006137	001414	ROL	BAD16	; TO MOVE THE MOST SIGN DIGIT
2305	011502	006137	001414	ROL	BAD16	; TO BIT 0 TO BIT 2
2306	011506	006137	001414	ROL	BAD16	
2307	011512	006137	001414	ROL	BAD16	
2308	011516	042737	177770 001414	BIC	#177770, BAD16	; LEFT ON THE MS DIGIT OF THE LSW
2309	011524	006337	001416	ASL	BAD16+2	; SHIFT THE MSW LEFT ONE OCTAL DIGIT
2310	011530	100731		BMI	1\$	; ENTER AGAIN IF SIGN BIT SET
2311	011532	006337	001416	ASL	BAD16+2	
2312	011536	100726		BMI	1\$	
2313	011540	006337	001416	ASL	BAD16+2	
2314	011544	100723		BMI	1\$	
2315	011546	063737	001414 001416	ADD	BAD16, BAD16+2	; APPENDING THE DIGITS INTO MSW
2316	011554	012637	001414	MOV	(SP)+, BAD16	; RESTORE THE LSW
2317	011560	006337	001414	ASL	BAD16	; SHIFT THE LSW LEFT ONE OCTAL DIGIT
2318	011564	006337	001414	ASL	BAD16	
2319	011570	006337	001414	ASL	BAD16	
2320	011574	042737	100000 001414	BIC	#BIT15, BAD16	; CLEAR THE SIGN BIT
2321	011602	111103		MOVB	(R1), R3	; LOAD THE CURRENT READ IN DIGITS

G04

MAINDEC-11-DZRNA, RMD3 FORMATTER
DZRNA8.P11 21-JUL-77 15:32MACY11 30(1046) 21-JUL-77 16:06 PAGE 45
GET VALUE FOR SOFTWARE SWITCH REGISTER

SEQ 0044

2322	011604	042703	177770		BIC	#177770,R3	:LEFT ONLY ONE OCTAL DIGIT	
2323	011610	047337	001414		ADD	R3,BAD16	:APPEND THE READ IN CHARACTER TO LSW	
2324	011614	04201			INC	R1	:ADJUST THE READ IN STRING ADDRESS	
2325	011616	005302			DEC	R2	:DECREMENT ONE DIGIT COUNT	
2326	011620	001307			BNE	2\$	:BRANCH IF NOT DONE	
2327	011622	105711			TSTB	(R1)	:IF 10 DIGITS HAVE BEEN ENTERED	
2328	011624	001273			BNE	1\$	:MUST BE FOLLOWED BY <CR>	
2329	011626	005737	001414	3\$:	TST	BAD16	:SERIAL NUMBER MUST BE NOT 0	
2330	011632	001003			BNE	4\$	:BRANCH IF NOT 0	
2331	011634	005737	001416		TST	BAD16+2		
2332	011640	001665			BEQ	1\$	:ENTER AGAIN IF ZERO	
2333	011642	013737	001414	002420	4\$:	MOV	BAD16,BAD18	:LOAD ALL TABLE WITH THE SERIAL NUMBER
2334	011650	013737	001414	003424		MOV	BAD16,USTAB	
2335	011656	013737	001414	004430		MOV	BAD16,USSAV	
2336	011664	013737	001416	003426		MOV	BAD16+2,USTAB+2	
2337	011672	013737	001416	004432		MOV	BAD16+2,USSAV+2	
2338	011700	013737	001416	002422		MOV	BAD16+2,BAD18+2	
2339	011706	000410			BR	6\$	:EXIT	
2340	011710	005237	004430		5\$:	INC	USSAV	:CODE FOR AUTO MODE APT,ACT,ETC.
2341	011714	005237	001414			INC	BAD16	
2342	011720	005237	002420			INC	BAD18	
2343	011724	005237	003424			INC	USTAB	
2344	011730	000240			6\$:	NOP	:DONE	
2345	011732	012737	016344	000060		MOV	#NEWSVC,2#TKVEC	:NEW INTERRUPT VECTOR FOR KEYBOARD
2346	011740	005037	177776			CLR	2#PSW	
2347								
2348								
2349	011744	023737	001320	001322	CKADRS:	CMP	ENDCYL,BEGCYL	:STARTING AND ENDING CYLINDERS THE SAME ?
2350	011752	001011				BNE	5\$	:BRANCH IF BEGCYL NOT EQUAL TO ENDCYL
2351	011754	013737	001324	001330		MOV	ENDTRK,TTRKS	:END TRACK ADDRESS
2352	011762	163737	001326	001330		SUB	BEGTRK,TTRKS	:SUBTRACT THE STARTING TRACK ADDRESS
2353	011770	005237	001330			INC	TTRKS	:MAKE THE NUMBER OF TRACKS INCLUSIVE
2354	011774	000450				BR	SETPAT	:SETUP THE DATA PATTERN
2355	011776	005737	001402		5\$:	TST	WRAP	:WRAP AROUND FLAG SET ?
2356	012002	001407				BEQ	1\$	:NO
2357	012004	012702	001466			MOV	#822,R2	:INITIAL COUNTER TO 822
2358	012010	163702	001322			SUB	BEGCYL,R2	:FIRST PART OF CYL #
2359	012014	063702	001320			ADD	ENDCYL,R2	:SECOND PART OF CYL #
2360	012020	000405				BR	4\$	
2361	012022	013702	001320		1\$:	MOV	ENDCYL,R2	:ENDING CYLINDER
2362	012026	163702	001322			SUB	BEGCYL,R2	:SUBTRACT THE STARTING CYLINDER
2363	012032	005302				DEC	R2	:EXCLUSIVE LAST CYLINDER
2364	012034	012737	000005	001330	4\$:	MOV	#5,TTRKS	:INITIATE COUNTER
2365	012042	163737	001326	001330		SUB	BEGTRK,TTRKS	:# OF TRACK ON STARTING CYLINDER
2366	012050	063737	001324	001330		ADD	ENDTRK,TTRKS	:# OF TRACK ON ENDING CYLINDER
2367	012056	005237	001330			INC	TTRKS	:INCLUSIVE ONE TRACK
2368	012062	005302			3\$:	DEC	R2	:DECREMENT THE CYLINDER COUNT
2369	012064	100404				BMI	6\$	:BR IF DONE
2370	012066	062737	000005	001330		ADD	#5,TTRKS	:ADD CYLINDER WORTH OF TRACKS
2371	012074	000772				BR	3\$	:CONTINUE
2372	012076	005037	001410		6\$:	CLR	EDIT	:RESET EDIT FLAG
2373	012102	023727	001330	010011		CMP	TTRKS,#(821.*5)	:ALL TRACKS ARE FORMATTED ?
2374	012110	103402				BLO	SETPAT	:BRANCH IF NOT
2375	012112	005237	001410			INC	EDIT	:SET EDIT FLAG
2376								
2377								

;THIS CODE SETS UP THE SELECTED DATA PATTERN TO BE WRITTEN ON EACH SECTOR IMAGE

H04

MAINDEC-11-DZRNA, RM03 FORMATTER
DZRNA8.P11 21-JUL-77 15:32MACY11 30(1046) 21-JUL-77 16:06 PAGE 46
GET VALUE FOR SOFTWARE SWITCH REGISTER

SEQ 0045

```

2378
2379 012116 005737 001316      SETPAT: TST      MODE      ;VERIFY MODE ?
2380 012122 001427              BEQ      1$      ;YES
2381 012124 005037 001340      4$: CLR      PATA      ;CLEAR DATA PATTERN A
2382 012130 005037 001342      CLR      PATB      ;CLEAR DATA PATTERN B
2383 012134 005737 001336      TST      PATSEL     ;SEE IF PATTERN OF ONES
2384 012140 001420              BEQ      1$      ;BR IF SO
2385 012142 012737 177777 001340  MOV      #177777,PATA ;PATA = AB
2386 012150 012737 177777 001342  MOV      #177777,PATB ;PATB=CD
2387 012156 022737 000001 001336  CMP      #1,PATSEL    ;SEE IF PATTERN OF ZEROS
2388 012164 001406              BEQ      1$      ;BRANCH IF SO
2389 012166 012737 066666 001340  MOV      #066666,PATA ;SET UP WORST CASE
2390 012174 012737 066666 001342  MOV      #066666,PATB ;SET UP WORST CASE
2391 012202 013703 001354      1$: MOV      WC,R3      ;SET UP COUNTER
2392 012206 012700 037554      MOV      #BUF,R0      ;SET UP MEMORY POINTER
2393 012212 013701 001340      3$: MOV      PATA,R1      ;SET UP PATTERN IN R1
2394 012216 013702 001342      MOV      PATB,R2      ;SET UP PATTERN IN R2
2395 012222 010120      2$: MOV      R1,(R0)+      ;MOV 1ST PAT INTO MEM
2396 012224 010220      MOV      R2,(R0)+      ;MOV 2ND PAT INTO MEM
2397 012226 005303      DEC      R3              ;KEEP COUNT
2398 012230 005303      DEC      R3              ;KEEP COUNT
2399 012232 001373      BNE      2$              ;DO IT AGAIN IF R3 NOT 0
2400
2401      ;THIS CODE SETS UP FOR THE ACTUAL FORMAT
2402 012234 004737 015376      WRTRK: JSR      PC,SETTBL    ;GO SET UP DRIVER TABLE
2403 012240 004737 015710      JSR      PC,SETHDR    ;GO INITIALIZE HEADERS IN THE SECTOR BUFFER IN CORE
2404 012244 112737 000020 005611  MOVB     #20,FMTDPB+1 ;LOAD FMT22 BIT
2405 012252 005737 001362      TST      SEC30      ;18 BIT MODE ?
2406 012256 001002              BNE      1$      ;BR IF NOT
2407 012260 105037 005611      CLRB     FMTDPB+1    ;CLEAR THE FMT22 BIT
2408 012264 112737 000143 005612  1$: MOVB     #SETFMT,FMTDPB+2 ;'LOAD FORMAT' COMMAND
2409 012272 004037 024736      2$: JSR      R0,RM03      ;START THE COMMAND
2410 012276 005610      FMTDPB      ;DPB ADDRESS
2411 012300 000774              BR      2$      ;QUEUE FULL RETURN
2412 012302 005737 005626      3$: TST      FMTDPB+16    ;LOOK FOR DONE
2413 012306 001775              BEQ      3$      ;LOOP UNTIL FINISHED
2414
2415      ;SET UP SECCU TABLE: BAD SPOT ON CURRENT TRACK ,RETRIEVED FROM PACK
2416
2417 012310 012703 005434      WRTRKX: MOV     #SECCU,R3 ;INITILIZE THE SECCU TABLE
2418 012314 012704 000040      MOV     #32,R4      ;TOTAL 32 WORD MAX
2419 012320 012702 177777      MOV     #-1,R2      ;LOAD ALL LOCATION TO -1
2420 012324 010223      1$: MOV     R2,(R3)+ ;
2421 012326 005304      DEC     R4              ;
2422 012330 001375      BNE     1$      ;BRANCH IF NOT DONE
2423
2424      ;LOAD THE BAD SPOT INTO SECCU TABLE IF ANY
2425
2426 012332 005737 001406      WRTRKY: TST     PACK      ;BAD SPOT FILE AVAILABLE AT ALL
2427 012336 003451              BLE     5$      ;BRANCH IF NOT
2428 012340 012702 001414      MOV     #BAD16,R2 ;R2 POINTER OF MFG BAD SPOT TABLE
2429 012344 005737 001362      TST     SEC30      ;IF 18 BIT MODE CHANGE POINTER
2430 012350 100402              BMI     1$      ;BRANCH IF 16 BIT MODE
2431 012352 012702 002420      MOV     #BAD18,R2 ;
2432 012356 010201      1$: MOV     R2,R1      ;R1: LAST ADDRESS OF BAD SPOT TABLE
2433 012360 062701 000776      ADD     #510.,R1 ;

```

MAINDEC-11-DZRNA, RM03 FORMATTER
DZRNA8.P11 21-JUL-77 15:32

MACY11 30(1046) 21-JUL-77 16:06 PAGE 47
GET VALUE FOR SOFTWARE SWITCH REGISTER

SEQ 0046

2434	012364	012703	005434		MOV	#SECCU,R3	:TABLE ADDRESS
2435	012370	012704	000076		MOV	#62,R4	:LAST ADDRESS OF THE TABLE
2436	012374	060704			ADD	R3,R4	
2437	012376	012702	000010		ADD	#10,R2	:BAD SPOT STARTS FROM THE 5TH WORD OF THE FILE
2438	012402	013712	005622	2\$:	CMP	FMTDPB+12,(R2)	:BAD SPOT ON THIS CYLINDER ?
2439	012406	011007			BNE	3\$	:BRANCH IF NOT
2440	012410	123762	005621	000003	CMPB	FMTDPB+11,3(R2)	:ALSO, ON THE SAME TRK ?
2441	012416	001003			BNE	3\$	:BRANCH IF NOT
2442	012420	116205	000002		MOVB	2(R2),R5	:BAD SECTOR
2443	012424	010523			MOV	R5,(R3)+	:WORD STORAGE
2444	012426	062702	000004	3\$:	ADD	#4,R2	:ADVANCE TO NEXT SET
2445	012432	020102			CMP	R1,R2	:ALL SPOT ARE CHECKED ?
2446	012434	103412			BLO	5\$	:BRANCH IF IT IS
2447	012436	020403			CMP	R4,R3	:END OF SECCU TABLE ?
2448	012440	103401			BLO	4\$	:REJECT THE PACK
2449	012442	000757			BR	2\$	:LOOPING BACK
2450	012444	004737	016246	4\$:	JSR	PC,RESTR	:IMPROPER MFG INFORMATION
2451	012450	012737	177777	001406	MOV	#-1,PACK	:RESET MFG FILE AVAILABLE FLAG
2452	012456	000137	011172		JMP	TABINT	:AND START AGAIN
2453	012462	000240		5\$:	NOP		:DONE
2454							
2455							
2456							
2457							
2458	012464	012702	005434				
2459	012470	012703	005532				
2460	012474	012704	000037				
2461	012500	022712	177777				
2462	012504	001451					
2463	012506	021262	000002	1\$:	CMP	(R2),2(R2)	:N TH ELEMENT : N+1 TH ELEMENT
2464	012512	103430			BLO	4\$	:BRANCH IF SMALLER
2465	012514	001406			BEQ	2\$	:DELET N+1 TH ELEMENT, IF SAME
2466							:OTHERWISE, N > N+1
2467	012516	011246			MOV	(R2),-(SP)	:STORE N ELEMENT
2468	012520	016212	000002		MOV	2(R2),(R2)	:SWITCH N+1 ELEMENT TO N ELEMENT
2469	012524	012662	000002		MOV	(SP)+,2(R2)	:SWITCH N ELEMENT TO N+1 ELEMENT
2470	012530	000421			BR	4\$	:ADJUST TOP POINTER
2471	012532	022712	177777	2\$:	CMP	#-1,(R2)	:TERMINATOR ?
2472	012536	001416			BEQ	4\$	:BRANCH IF IT IS
2473	012540	010246			MOV	R2,-(SP)	:SAVE TOP POINTER
2474	012542	062702	000002		ADD	#2,R2	:DELET 2(R2)
2475	012546	016222	000002	3\$:	MOV	2(R2),(R2)+	
2476	012552	022702	005532		CMP	#SECCU+62,R2	:BOTTOM OF THE TABLE ?
2477	012556	101373			BHI	3\$	:BRANCH IF NOT
2478	012560	012737	177777	005532	MOV	#-1,SECCU+62.	:LOAD -1 TO TABLE BOTTOM
2479	012566	012602			MOV	(SP)+,R2	:RESTORE THE TOP POINTER
2480	012570	005304			DEC	R4	:DECREMENT ONE ELEMENT COUNT
2481	012572	001416			BEQ	6\$	:BRANCH IF ALL DONE
2482	012574	062702	000002	4\$:	ADD	#2,R2	:INCREMENT THE TOP POINTER
2483	012600	010302			CMP	R3,R2	:REACH THE BOTTOM ?
2484	012602	001341			BNE	1\$	:BRANCH IF NOT
2485	012604	005304		5\$:	DEC	R4	:DECREMENT ONE SORT COUNT
2486	012606	001410			BEQ	6\$	:BRANCH IF ALL DONE
2487	012610	012702	005434		MOV	#SECCU,R2	:RESET TOP POINTER
2488	012614	162703	000002		SUB	#2,R3	:UPDATE BOTTOM POINTER
2489	012620	022703	005434		CMP	#SECCU,R3	:EXIT IF TOP = BOTTOM

:THIS CODE SORTS THE BAD SECTOR TABLE "SECCU" IN ASCENDING ORDER
:AND DELETES THE DUPLICATED ONES

SORT1: MOV #SECCU,R2 :TOP OF THE TABLE
MOV #SECCU+62,R3 :BOTTOM OF THE TABLE
MOV #31,R4 :TOTAL 32 ELEMENT SORT 31 TIMES
CMP #-1,(R2) :IS THE TABLE EMPTY ?
BEQ 6\$:BRANCH IF IT IS, QUICK EXIT
1\$: CMP (R2),2(R2) :N TH ELEMENT : N+1 TH ELEMENT
BLO 4\$:BRANCH IF SMALLER
BEQ 2\$:DELET N+1 TH ELEMENT, IF SAME
:OTHERWISE, N > N+1
MOV (R2),-(SP) :STORE N ELEMENT
MOV 2(R2),(R2) :SWITCH N+1 ELEMENT TO N ELEMENT
MOV (SP)+,2(R2) :SWITCH N ELEMENT TO N+1 ELEMENT
BR 4\$:ADJUST TOP POINTER
2\$: CMP #-1,(R2) :TERMINATOR ?
BEQ 4\$:BRANCH IF IT IS
MOV R2,-(SP) :SAVE TOP POINTER
3\$: ADD #2,R2 :DELET 2(R2)
MOV 2(R2),(R2)+
CMP #SECCU+62,R2 :BOTTOM OF THE TABLE ?
BHI 3\$:BRANCH IF NOT
MOV #-1,SECCU+62. :LOAD -1 TO TABLE BOTTOM
MOV (SP)+,R2 :RESTORE THE TOP POINTER
DEC R4 :DECREMENT ONE ELEMENT COUNT
BEQ 6\$:BRANCH IF ALL DONE
4\$: ADD #2,R2 :INCREMENT THE TOP POINTER
CMP R3,R2 :REACH THE BOTTOM ?
BNE 1\$:BRANCH IF NOT
5\$: DEC R4 :DECREMENT ONE SORT COUNT
BEQ 6\$:BRANCH IF ALL DONE
MOV #SECCU,R2 :RESET TOP POINTER
SUB #2,R3 :UPDATE BOTTOM POINTER
CMP #SECCU,R3 :EXIT IF TOP = BOTTOM

J04

MAINDEC-11-DZMA, RMD3 FORMATTER
DZMA8.P11 21-JUL-77 15:32

MACY11 30(1046) 21-JUL-77 16:06 PAGE 48
GET VALUE FOR SOFTWARE SWITCH REGISTER

SEQ 0047

2490	012624	001401			BEQ	65			
2491	012626	000727			BR	1			
2492	012630	000240							; SORT THE REST ELEMENTS
2493	012632	005037	005536		65:	NOP			; DONE
2494	012636	005037	001314		WTRK1:	CLR	NEXT		; FLAG START OF A NEW TRACK
2495	012642	005037	001346			SOFSW			; CLEAR ERROR COUNTER
2496	012646	105037	005620			RETRY			; ZERO THE RETRY COUNTER
2497	012652	013737	001356	005614		CLRB	FMTDPB+10		; RESTORE SECTOR
2498	012660	012737	037554	005616		MOV	MWC,FMTDPB+4		; RESTORE WC
2499						MOV	#BUFF,FMTDPB+6		; RESTORE BA
2500									; DELET THE FOLLOWING TWO LINES FOR NEW VERIFY OPERATION
2501						TST	MODE		; 'FORMAT' OR 'CHECK' MODE ?
2502						BEQ	CKTRK		; BR IF 'CHECK' MODE
2503									
2504									; THIS CODE TO SET UP WORD COUNT AND BUFFER ADDRESS
2505	012666	022737	177777	005536	WTRKA:	CMP	#-1,NEXT		; END OF OPERATION ?
2506	012674	001001				BNE	15		; BRANCH IF NOT
2507	012676	000000				HALT			; OTHERWISE SOMETHING WRONG *****
2508	012700	113737	005536	005620	15:	MOVB	NEXT,FMTDPB+10		; STARTING SECTOR
2509	012706	012701	005434			MOV	#SECCU,R1		; R1 POINTS TO THE BAD SECTOR TABLE
2510	012712	022711	177777		25:	CMP	#-1,(R1)		; END OF TABLE ?
2511	012716	001505				BEQ	WTRKD		; BRANCH IF IT IS
2512	012720	032711	100000			BIT	#BIT15,(R1)		; THE BAD SECTOR HAS ALREADY ACCESSED ?
2513	012724	001403				BEQ	35		; BRANCH IF NOT
2514	012726	062701	000002			ADD	#2,R1		; ADJUST THE POINTER
2515	012732	000767				BR	25		; LOOPING BACK
2516	012734	123711	005620		35:	CMPB	FMTDPB+10,(R1)		; BAD SPOT = STARTING SECTOR ?
2517	012740	001414				BEQ	WTRKB		; BRANCH IF IT I
2518	012742	103401				BLO	45		; BRANCH IF START SEC < BAD SPOT
2519	012744	000000				HALT			; OTHERWISE SOMETHING WRONG
2520	012746	111102			45:	MOVB	(R1),R2		; UPDATE NEXT STARTING ADDRESS
2521	012750	023702	001364			CMP	MAXSEC,R2		; NEXT BAD SPOT IS THE LAST SECTOR?
2522	012754	101466				BLOS	WTRKD		; BRANCH IF IT IS
2523	012756	005202				INC	R2		; NEXT = BAD + 1
2524	012760	010237	005536			MOV	R2,NEXT		; NEXT STARTING ADDRESS
2525	012764	052721	100000			BIS	#BIT15,(R1)+		; MARK THE ACCESSED SECTOR
2526	012770	000435				BR	WTRKC		; TO SET UP WORD COUNT
2527	012772	105237	005620		WTRKB:	INCB	FMTDPB+10		; INCREMENT THE STARTING SECTOR
2528	012776	052721	100000			BIS	#BIT15,(R1)+		; MARK THE ACCESSED SECTOR
2529	013002	123711	005620			CMPB	FMTDPB+10,(R1)		; STILL ON ANOTHER BAD SPOT ?
2530	013006	001771				BEQ	WTRKB		; LOOP AGAIN
2531	013010	123737	005620	001364		CMPB	FMTDPB+10,MAXSEC		; IF START SECTOR > MAXSEC
2532	013016	101407				BLOS	15		; IMPORPER MFG INFORMATION
2533	013020	004737	016246			JSR	PC,RESTART		; START AGAIN AND REBUILD FMG FILE
2534	013024	012737	177777	001406		MOV	#-1,PACK		; RESET THE MFG FILE AVAILABLE FLAG
2535	013032	000137	011172			JMP	TABINT		; RESTART POINT
2536	013036	022711	177777		15:	CMP	#-1,(R1)		; TO SEE IF NEXT BAD SPOT IN TABLE
2537	013042	001433				BEQ	WTRKD		; BRANCH IF NONE
2538	013044	111102				MOVB	(R1),R2		; UPDATE THE NEXT
2539	013046	023702	001364			CMP	MAXSEC,R2		; BAD SECTOR IS THE LAST SECTOR ?
2540	013052	101427				BLOS	WTRKD		; BRANCH IF IT IS
2541	013054	005202				INC	R2		; BAD SEC +1
2542	013056	010237	005536			MOV	R2,NEXT		; NEXT STARTING ADDRESS
2543	013062	052711	100000			BIS	#BIT15,(R1)		; MARK THE ACCESSED SECTOR
2544	013066	013702	005536		WTRKC:	MOV	NEXT,R2		; CALCULATE THE WORD COUNT
2545	013072	162702	000002			SUB	#2,R2		; ENDSECTOR = NEXT - 2

K04

MAINDEC-11-DZMA, RMO3 FORMATTER
DZMAB.P11 21-JUL-77 15:32

MACY11 30(1046) 21-JUL-77 16:06 PAGE 49
GET VALUE FOR SOFTWARE SWITCH REGISTER

SEQ 0048

2546	013076	006302		ASL	R2	;WORD INDEX
2547	013100	016203	006052	MOV	WCTBL(R2),R3	;WORD CTR FOR SECTOR 0 TO ENDING SECTOR
2548	013104	113702	005620	MOVB	FMTDPB+10,R2	;STARTING SECTOR
2549	013110	006302		ASL	R2	;WORD INDEX
2550	013112	166203	006052	SUB	WCTBL(R2),R3	;WORD COUNT FOR SECTOR 0 TO STARTING SECTOR
2551	013116	062703	000402	ADD	#258.,R3	;ADJUST ONE SECTOR
2552	013122	005403		NEG	R3	;GET THE NEAGTIVE WORD COUNT
2553	013124	010337	005614	MOV	R3,FMTDPB+4	;LOAD THE WORD CTR INTO DPB
2554	013130	000423		BR	WTRKE	;LOAD THE STARTING ADDRESS
2555	013132	012737	177777	MOV	#-1,NEXT	;NO MORE BAD SPOT
2556	013140	013702	001364	MOV	MAXSEC,R2	;LAST SECTOR
2557	013144	006302		ASL	R2	;WORD INDEX
2558	013146	016203	006052	MOV	WCTBL(R2),R3	;WORD CTR FOR THE ENDING SECTOR
2559	013152	113702	005620	MOVB	FMTDPB+10,R2	;STARTING ADDRESS
2560	013156	006302		ASL	R2	;WORD INDEX
2561	013160	166203	006052	SUB	WCTBL(R2),R3	;WORD CTR FOR THE STARTING SECTOR
2562	013164	062703	000402	ADD	#258.,R3	;ADJUST ONE SECTOR
2563	013170	005403		NEG	R3	;NEGATIVE WORD COUNT
2564	013172	010337	005614	MOV	R3,FMTDPB+4	;LOAD THE WORD COUNT INTO DPB
2565	013176	000240		NOP		;DONE
2566	013200	113702	005620	WTRKE: MOVB	FMTDPB+10,R2	;STARTING ADDRESS
2567	013204	006302		ASL	R2	;LOCATE THE BUFFER ADDRESS
2568	013206	016237	005752	MOV	ADRTBL(R2),FMTDPB+6	;BUFFER ADDRESS
2569	013214	112737	000163	WTRK2: MOVB	#WTRHD,FMTDPB+2	;SET WRITE HEADER & DATA COMMAND IN TBL
2570	013222	012737	013222	MOV	#.SLPERR	;SET UP LOOP ON ERROR ADDRESS
2571	013230	012706	001100	MOV	#STACK,SP	;LOAD STACK POINT
2572	013234	004037	024736	JSR	RO,RMO3	;GO FORMAT A TRACK
2573	013240	005610		FMTDPB		;ADRS OF PARAMETERS - TBL
2574	013242	000774		BR	1\$	;WAIT FOR QUEUE IF FULL
2575	013244	005737	005626	2\$: TST	FMTDPB+16	;WAIT FOR COMMAND TO COMPLETE
2576	013250	001775		BEQ	2\$	;BRANCH IF NOT DONE
2577	013252	100024		BPL	4\$	;BRANCH IF NO ERROR
2578	013254	012737	013302	MOV	#3\$,SESCAPE	;ESCAPE TO 3\$ ON ERROR
2579	013262	004737	015230	JSR	PC,ERINDX	;SEE WHICH ERROR
2580	013266	104010		ERROR	10	;DRIVE OFFLINE
2581	013270	104011		ERROR	11	;PERSISTENT DRIVE UNSAFE ERROR
2582	013272	104012		ERROR	12	;UNCORRECTABLE MASSBUS PARITY ERROR
2583	013274	104013		ERROR	13	;SOFTWARE TIMEOUT
2584	013276	104014		ERROR	14	;DRIVE UNSAFE ERROR
2585	013300	104015		ERROR	15	;DRIVE/CONTROLLER ERROR DURING WRITE
2586	013302	004737	015330	3\$: JSR	PC,LOP.CK	;LOOP ON THE ERROR ?
2587	013306	022737	000003	CMP	#3\$,RETRY	;ERROR RETRY LIMIT ?
2588	013314	001403		BEQ	4\$	;BR IF REACHED
2589	013316	005237	001346	INC	RETRY	;COUNT THE ERROR
2590	013322	000744		BR	1\$	;TRY AGAIN
2591	013324	005037	001174	4\$: CLR	SESCAPE	;CLEAR ERROR ESCAPE ADDRESS
2592	013330	005037	001346	CLR	RETRY	;CLEAR THE RETRY COUNTER
2593						
2594						
2595						
2596	013334	112737	000153	005612	WTRK: MOVB	#WCKHD,FMTDPB+2
2597	013342	012737	013342	001124	MOV	#.SLPERR
2598	013350	012706	001100		MOV	#STACK,SP
2599	013354	004037	024736	1\$: JSR	RO,RMO3	
2600	013360	005610			FMTDPB	
2601	013362	000774			BR	1\$

L04

MAINDEC-11-DZMA, RM03 FORMATTER
DZMA8.P11 21-JUL-77 15:32MACY11 30(1046) 21-JUL-77 16:06 PAGE 50
GET VALUE FOR SOFTWARE SWITCH REGISTER

SEQ 0049

2602	013364	005737	005626	2\$:	TST	FMTDPB+16	WAIT FOR COMMAND TO COMPLETE
2603	013370	001775			BEQ	2\$	BRANCH IF NOT DONE
2604	013372	100131			BPL	8\$	BRANCH IF NO ERROR
2605	013374	113737	005546 001350		MOVB	RM.REG+RMDA, SAVSEC	GET THE SECTOR ADDRESS
2606	013402	001005			BNE	3\$	BRANCH IF NOT SECTOR 0
2607	013404	013737	001364 001350		MOV	MAXSEC, SAVSEC	RESTORE TO LAST SECTOR +1
2608	013412	005237	001350		INC	SAVSEC	INCREMENT TO LAST SECTOR +1
2609	013416	005337	001350	3\$:	DEC	SAVSEC	ADJUST SECTOR TO THE ONE THAT FAILED
2610	013422	012737	013724 001174		MOV	#10\$, \$ESCAPE	ESCAPE TO 10\$ ON ERROR
2611	013430	004737	015230		JSR	PC, ERINDX	SEE WHICH ERROR
2612	013434	104010			ERROR	10	DRIVE OFFLINE
2613	013436	104011			ERROR	11	PERSISTENT DRIVE UNSAFE ERROR
2614	013440	104012			ERROR	12	UNACCEPTABLE MASSBUS PARITY ERROR
2615	013442	104013			ERROR	13	SOFTWARE TIMEOUT
2616	013444	104014			ERROR	14	DRIVE UNSAFE ERROR
2617	013446	032737	040000 005550		BIT	#WCE, RM.REG+RMCS2	WRITE CHECK ERROR ?
2618	013454	001005			BNE	4\$	BR IF SET
2619	013456	033727	005554		BIT	RM.REG+RMER1, (PC)+	CHECK FOR DATA ERRORS
2620	013462	130620			.WORD	DCK!OPI!DTE!HCRC!	WCE!FER! ; DATA ERROR BITS
2621	013464	001001			BNE	4\$	BR IF SET
2622	013466	104016			ERROR	16	CONTROLLER/DRIVE ERROR DURING WRITE CHECK
2623	013470	005737	001314	4\$:	TST	SOF SW	RETRYING THE SECTOR ?
2624	013474	001413			BEQ	5\$	BR IF NOT
2625	013476	012737	013750 001174		MOV	#11\$, \$ESCAPE	ESCAPE TO 11\$ ON ERROR
2626	013504	004737	016616		JSR	PC, RECORD	RECORD THE BAD SPOT
2627	013510	012737	100000 001412		MOV	#BIT15, HEADX	RESET THE USER BIT
2628	013516	004737	017436		JSR	PC, RESET	RESET THE HEADER BITS
2629	013522	104017			ERROR	17	SECTOR NOT ACCEPTABLE
2630	013524	005037	001174	5\$:	CLR	\$ESCAPE	RESET ESCAPE ADDRESS
2631	013530	032737	040000 00552		BIT	#ERR, RM.REG+RMDS	DATA ERROR ?
2632	013536	001002			BNE	6\$	BR IF IT IS
2633	013540	104024			ERROR	24	WRITE CHECK ERROR
2634	013542	000401			BR	7\$	CONTINUE
2635	013544	104020		6\$:	ERROR	20	DATA ERROR DURING WRITE CHECK
2636	013546	013701	001350	7\$:	MOV	SAVSEC, R1	FAILING SECTOR ADDRESS
2637	013552	113737	001350 005620		MOVB	SAVSEC, FMTDPB+10	SECTOR ADDRESS
2638	013560	006301			ASL	R1	SETUP INDEX TO ADDRESS WORDS
2639	013562	016137	005752 005616		MOV	ADRTBL(R1), FMTDPB+6	BUFFER ADDRESS FOR FAILING SECTOR
2640	013570	013701	001364		MOV	MAXSEC, R1	R1 ENDING SECTOR
2641	013574	005737	005536		TST	NEXT	WHOLE TRACK DONE ?
2642	013600	100404			BMI	13\$	BRANCH IF IT IS
2643	013602	013701	005536		MOV	NEXT, R1	LOAD CURRENT ENDING SECTOR
2644	013606	162701	000002		SUB	#2, R1	ADJUST TWO SECTORS
2645	013612	006301		13\$:	ASL	R1	WORD INDEX
2646	013614	016102	006052		MOV	WCTBL(R1), R2	WORDCTR FOR ENDING SECTOR
2647	013620	113701	005620		MOVB	FMTDPB+10, R1	STARTING ADDRESS
2648	013624	006301			ASL	R1	WORD COUNT
2649	013626	166102	006052		SUB	WCTBL(R1), R2	WORD CTR FOR STARTING SECTOR
2650	013632	005402			NEG	R2	
2651	013634	010237	001352		MOV	R2, SAVWC	SAVE THE WORD COUNT
2652	013640	012737	177376 005614		MOV	#-258, FMTDPB+4	WORD COUNT FOR 1 SECTOR
2653	013646	005237	001314		INC	SOF SW	INDICATE THAT A RETRY IS IN PROGRESS
2654	013652	000137	013214		JMP	WTRK2	REFORMAT ERROR SECTOR
2655	013656	005737	001314	8\$:	TST	SOF SW	RETRY IN PROGRESS ?
2656	013662	001432			BEQ	11\$	BR IF NOT
2657	013664	022737	000002 001314		CMP	#2, SOFSW	SEE IF LAST RETRY

M04

MAINDEC-11-DZRNA, RM03 FORMATTER
DZRNAB.P11 21-JUL-77 15:32

MACY11 30(1046) 21-JUL-77 16:06 PAGE 51
GET VALUE FOR SOFTWARE SWITCH REGISTER

SEQ 0050

2658	013672	001403			BEQ	9\$		;BR IF IT IS
2659	013674	005237	001314		INC	SOF SW		;INCREMENT RETRY COUNT
2660	013700	000625			BR	1\$		;READ AGAIN
2661	013702	123737	001364	005620	9\$: CMPB	MAXSEC,FMTDPB+10		;SEE IF LAST SECTOR ON THE TRACK
2662	013710	001417			BEQ	11\$		;BR IF IT IS
2663	013712	004737	015662		JSR	PC,SCAWC		;SETUP TO CHECK REMAINING SECTORS ON THE TRACK
2664	013716	005037	001314		CLR	SOF SW		;CLEAR RETRY COUNTER
2665	013722	000614			BR	1\$		;FINISH CHECKING THE TRACK
2666	013724	004737	015330		10\$: JSR	PC,LOP.CK		;CHECK FOR LOOP ON ERROR
2667	013730	022737	000003	001346	CMP	#3,RETRY		;ERROR RETRY REACHED ?
2668	013736	001404			BEQ	11\$		;BR IF IT IS
2669	013740	005237	001346		INC	RETRY		;COUNT THE RETRY
2670	013744	000137	013354		JMP	1\$		;DO THE WRITE CHECK AGAIN
2671	013750	005037	001174		11\$: CLR	\$ESCAPE		;CLEAR THE ERROR RETURN ESCAPE ADDRESS
2672	013754	005037	001346		CLR	RETRY		;CLEAR THE RETRY COUNTER
2673	013760	005037	001314		CLR	SOF SW		;CLEAR THE SOFTWARE RETRY COUNT
2674	013764	005737	005536		TST	NEXT		;WHOLE TRACK DONE ?
2675	013770	100432			BMI	.+6		;BRANCH IF IT IS
2676	013772	000137	012666		JMP	WTRKA		
2677	013776	032777	000032	165150	BIT	#BIT1,JSWR		;LOOP ON CURRENT TRACK ?
2678	014004	001402			BEQ	12\$		;BR IF NOT
2679	014006	000137	012310		JMP	WTRKX		;START AGAIN ON THE SAME TRACK
2680								;RESET THE HEADER WORD OF ALL MFG
2681								;DEFINED BAD SECTOR
2682	014012	012737	040000	001412	12\$: MOV	#BIT14,HEADX		;RESET THE MFG BIT OF THE 1ST HEADER WORD
2683	014020	012704	005434		MOV	#SECCU,R4		;TABLE FOR BAD SECTOR ON CURRENT TRACK
2684	014024	022714	177777		14\$: CMV	#-1,(R4)		;END OF TABLE ?
2685	014030	001414			BEQ	16\$		;BRANCH IF SO
2686	014032	032714	040000		BIT	#BIT14,(R4)		;HAS THE SECTOR BE ASSCESSED ?
2687	014036	001006			BNE	15\$		;BRANCH IF SO
2688	014040	052714	040000		BIS	#BIT14,(R4)		;MASK THE SECTOR TABLE
2689	014044	111437	001350		MOV	(R4),SAVSEC		;SECTOR ADDRESS
2690	014050	004737	017436		JSR	PC,RESET		;RESET THE HEADER
2691	014054	062704	000002		15\$: ADD	#2,R4		;INCREMENT THE TABLE POINTER
2692	014060	000761			BR	14\$		;LOOPING BACK
2693	014062	000240			16\$: NOP			;ALL DONE
2694								
2695	014064	004737	015466		WTRKZ: JSR	PC,TRKTST		;GET UPDATED TRACK AND CYLINDER ADDRESSES
2696	014070	000402			BR	HOREAD		;RETURN HERE IF DONE - DO QUICK CHECK OF DISK
2697	014072	000137	012310		JMP	WTRKX		;CONTINUE WITH THE FORMAT
2698								
2699								
2700								
2701								
2702								
2703	014076	005737	001316		HDREAD: TST	MODE		;VERIFY MODE ?
2704	014102	001020			BNE	9\$		;NO
2705	014104	022737	066666	001340	CMP	#066666,PATA		;PATTERN HAS BEEN ROTATED TO WORST CASE ?
2706	014112	001575			BEQ	\$EOP		;YES
2707	014114	000241			CLC			;CLEAR THE CARRY BIT
2708	014116	006037	001340		ROR	PATA		;ROTATE PATTERN
2709	014122	103003			.+10			;SET SIGN BIT IF CARRY=1
2710	014124	052737	100000	001340	BIS	#BIT15,PATA		;FROM BIT 0
2711	014132	013737	001340	001342	MOV	PATA,PATB		;SET BOTH PATTERN
2712	014140	000137	011744		JMP	CKADRS		;CONTINUE THE VERIFY OPERATION
2713	014144	005037	001360		9\$: CLR	HEDERR		;CLEAR HEADER CHECK ERROR INDICATOR

;THIS CODE MAKES SURE EACH CYLINDER FORMATTED CONTAINS THE
;PROPER CYLINDER ADDRESS. THE PROGRAM IS LOOKING FOR
;POSSIBLE POSITIONER ERRORS THAT MAY HAVE OCCURRED DURING THE FORMAT.

N04

MAINDEC-11-DZMA, RMO3 FORMATTER
DZMA8.P11 21-JUL-77 15:32

MACY11 30(1046) 21-JUL-77 16:06 PAGE 52
GET VALUE FOR SOFTWARE SWITCH REGISTER

SEQ 0051

2714	014150	004737	015376		JSR	PC,SETTBL	:GO SET UP DRIVER TABLE
2715	014154	004737	015710		JSR	PC,SETHDR	:GO SET UP HEADERS IN CORE
2716	014160	013737	001322	001344	MOV	BEGCYL,CYLCK	:USE 'BEGCYL' FOR FORMAT VERIFICATION
2717	014166	012737	177776	005614	MOV	#-2,FMTDPB+4	:SET UP WORD COUNT
2718	014174	012737	037544	005616	MOV	#RBUF,FMTDPB+6	:SET UP BUFFER ADRS
2719	014202	005037	005620		CLR	FMTDPB+10	:CLEAR THE SECTOR & TRACK ADDRESS FIELD
2720	014206	013737	001322	005622	MOV	BEGCYL,FMTDPB+12	:SETUP THE CYLINDER FIELD
2721	014214	112737	000173	005612	MOV	#RDHD,FMTDPB+2	:SET UP READ HEADER & DATA COMMAND
2722	014222	012737	014222	001124	MOV	#,SLPERR	:SETUP LOOP ON ERROR ADDRESS
2723	014230	012706	001100		MOV	#STACK,SP	:LOAD STACK POINTER
2724	014234	004037	024736		JSR	RO,RMO3	:GO READ HEADER
2725	014240	005610			FMTDPB		:ADRS OF PARAMETER TBL
2726	014242	000774			BR	1\$	:WAIT IF QUEUE IS FULL
2727	014244	005737	005626		TST	FMTDPB+16	:WAIT FOR COMMAND TO COMPLETE
2728	014250	001775			BEQ	2\$	:BRANCH IF NOT DONE
2729	014252	100024			BPL	4\$	:BR IF NOT ERROR
2730	014254	012737	014336	001174	MOV	#5\$,SESCAPE	:ESCAPE TO 5\$ ON ERROR
2731	014262	004737	015230		JSR	PC,ERINDX	:SEE WHICH ERROR
2732	014266	104010			ERROR	10	:DRIVE OFFLINE
2733	014270	104011			ERROR	11	:PERSISTENT DRIVE UNSAFE ERROR
2734	014272	104012			ERROR	12	:UNCORRECTABLE MASSBUS PARITY ERROR
2735	014274	104013			ERROR	13	:SOFTWARE TIMEOUT
2736	014276	104014			ERROR	14	:DRIVE UNSAFE ERROR
2737	014300	032737	177577	005626	BIT	#1C<HCE>,FMTDPB+16	:IS ONLY ERROR A HEADER COMPARE ERROR ?
2738	014306	001401			BEQ	3\$	:BR IF YES
2739	014310	104021			ERROR	21	:CONTROLLER/DRIVE ERROR VERIFYING HEADERS
2740	014312	005037	001174		CLR	SESCAPE	:CLEAR THE ERROR ESCAPE ADDRESS
2741	014316	104022			ERROR	22	:HEADER COMPARE ERROR VERIFYING HEADERS
2742	014320	004737	015330		JSR	PC,LOP.CK	:CHECK FOR LOOP ON ERROR
2743	014324	023737	037544	037554	CMP	RBUF,BUFF	:SEE IF CYL READ EQUALS CYL EXPECTED
2744	014332	001403			BEQ	6\$	:BR IF CYL CORRECT
2745	014334	104023			ERROR	23	:CYLINDER NOT CORRECT
2746	014336	004737	015330		JSR	PC,LOP.CK	:CHECK FOR LOOP ON ERROR
2747	014342	023737	001320	001344	CMP	ENDCYL,CYLCK	:SEE IF LAST CYLINDER
2748	014350	001002			BNE	7\$	:BR IF NOT FINISHED
2749	014352	000137	014506		JMP	SEOP	:END OF FORMAT
2750	014356	005237	001344		INC	CYLCK	:INCREMENT CYLINDER ADDRESS BEING CHECKED
2751	014362	022737	001466	001344	CMP	#822.,CYLCK	:LAST CYLINDER ?
2752	014370	000041			BGT	8\$	:NO
2753	014372	122737	000004	005621	CMPB	#4,FMTDPB+11	:LAST TRACK ?
2754	014400	003035			BGT	8\$	:NO THEN BRANCH
2755	014402	001237	001344		INC	CYLCK	:INCREMENT CYLINDER ADDRESS
2756	014406	023737	001320	001344	CMP	ENDCYL,CYLCK	:LAST CYLINDER ?
2757	014414	001752			BEQ	6\$	:EXIT IF IT IS
2758	014416	005037	001344		CLR	CYLCK	:YES LAST CYLINDER
2759	014422	001037	005622		CLR	FMTDPB+12	:RESER CYLINDER # TO 0
2760	014426	005037	005620		CLR	FMTDPB+10	:RESET TRACK AND SECTOR #
2761	014432	005037	005626		CLR	FMTDPB+16	:RESET CYLINDER #
2762	014436	013746	001322		MOV	BEGCYL,-(SP)	:SAVE BEGCVL
2763	014442	013746	001326		MOV	BEGTRK,-(SP)	:SAVE BEGTRK
2764	014446	005037	001326		CLR	BEGTRK	
2765	014452	005037	001322		CLR	BEGCYL	
2766	014456	004737	015710		JSR	PC,SETHDR	:SET UP BUFFER
2767	014462	012637	001326		MOV	(SP)+,BEGTRK	:RESTORE BEGTRK
2768	014466	012637	001322		MOV	(SP)+,BEGCYL	:RESTORE BEGCVL
2769	014472	000660			BR	1\$	

MAINDEC-11-DZRNA, RM03 FORMATTER
DZRNA8.P11 21-JUL-77 15:32

MACY11 30(1046) 21-JUL-77 16:06 PAGE 53
GET VALUE FOR SOFTWARE SWITCH REGISTER

SEQ 0052

2770 014474 004737 015776
2771 014500 005237 005622
2772 014504 000653

8\$: JSR PC,UPDACY ;SET UP FOR NEXT CYL
INC FMTDPB+12 ;ADVANCE TO NEXT CYL #
BR 1\$;GO READ NEXT HEADER

.SBTTL END OF PASS ROUTINE

;INCREMENT THE PASS NUMBER (\$PASS)
;IF THERES A MONITOR GO TO IT
;IF THERE ISN'T JUMP TO DONE

2781 014506
2782 014506 005737 001410
2783 014512 001517

SEOP:
5\$: TST EDIT ;TEST LAST TRACK ?
BEQ 11\$;BRANCH IF NOT
CLR RETRY ;CLEAR RETRY COUNT
JSR PC, SORT2 ;SORT THE USTAB
JSR PC, TEXT ;SET UP THE BUFFER
MOVB #20, FMTDPB+1 ;16 BIT MODE
MOVB #SETFMT, FMTDPB+2 ;SET FORMAT COMMAND
6\$: JSR RO, RM03 ;CALL THE DRIVER

2784 014514 005037 001346
2785 014520 004737 016732
2786 014524 004737 017070

7\$: TST FMTDPB+16 ;LOOP IF QUEUE FAILS
BEQ 7\$;ALL DONE ?
MOV #822, FMTDPB+12 ;CYLINDER ADDRESS
MOVB #4, FMTDPB+11 ;TRACK NUMBER
CLRB FMTDPB+10 ;SECTOR 0

2787 014530 112737 000020 005611
2788 014536 112737 000143 005612

8\$: JSR FMTDPB 6\$;LOOP IF QUEUE FAILS
BR 6\$;ALL DONE ?
TST FMTDPB+16 ;LOOP IF QUEUE FAILS
BEQ 7\$;ALL DONE ?

2789 014544 004037 024736
2790 014550 005610
2791 014552 000774

9\$: TST FMTDPB+16 ;LOOP IF QUEUE FAILS
BEQ 9\$;ALL DONE ?
BPL 11\$;EXIT IF NO ERROR
INC RETRY ;OVER 3 TIMES ?

2792 014554 005737 005626
2793 014560 001775
2794 014562 012737 001466 005622

10\$: MOV #31, FMTDPB+10 ;LAST SECTOR ?
MOVB #31, FMTDPB+10 ;BRANCH IF ALL SET
CLRB FMTDPB+10 ;FOUND THE FALLING SPOT
MOV #-(32.*258.), FMTDPB+4 ;WORD COUNT

2795 014570 112737 000004 005621
2796 014576 105037 005620
2797 014602 012737 157700 005614

11\$: INC \$DEVCT ;INCREMENT THE DEVICE COUNT
TST MODE ;FORMAT OR CHECK MODE
BEQ 12\$;BR IF CHECK
TYPE ,MFCMPT ;FORMAT COMPLETE

2798 014610 012737 037554 005616
2799 014616 112737 000163 005612

12\$: INC \$DEVCT ;INCREMENT THE DEVICE COUNT
TST MODE ;FORMAT OR CHECK MODE
BEQ 12\$;BR IF CHECK
TYPE ,MFCMPT ;FORMAT COMPLETE

2800 014624 004037 024736
2801 014630 005610
2802 014632 000774

13\$: INC \$DEVCT ;INCREMENT THE DEVICE COUNT
TST MODE ;FORMAT OR CHECK MODE
BEQ 12\$;BR IF CHECK
TYPE ,MFCMPT ;FORMAT COMPLETE

2803 014634 005737 005626
2804 014640 001775
2805 014642 100043

14\$: INC \$DEVCT ;INCREMENT THE DEVICE COUNT
TST MODE ;FORMAT OR CHECK MODE
BEQ 12\$;BR IF CHECK
TYPE ,MFCMPT ;FORMAT COMPLETE

2806 014644 005237 001346
2807 014650 022737 000003 001346
2808 014656 101362

15\$: INC \$DEVCT ;INCREMENT THE DEVICE COUNT
TST MODE ;FORMAT OR CHECK MODE
BEQ 12\$;BR IF CHECK
TYPE ,MFCMPT ;FORMAT COMPLETE

2809 014660 122737 000037 005620
2810 014666 003431
2811 014670 113737 005546 001350

16\$: INC \$DEVCT ;INCREMENT THE DEVICE COUNT
TST MODE ;FORMAT OR CHECK MODE
BEQ 12\$;BR IF CHECK
TYPE ,MFCMPT ;FORMAT COMPLETE

2812 014676 001003
2813 014700 012737 000037 001350
2814 014706 113737 001350 005620

17\$: INC \$DEVCT ;INCREMENT THE DEVICE COUNT
TST MODE ;FORMAT OR CHECK MODE
BEQ 12\$;BR IF CHECK
TYPE ,MFCMPT ;FORMAT COMPLETE

2815 014714 113701 001350
2816 014720 006301
2817 014722 016137 005752 005616

18\$: INC \$DEVCT ;INCREMENT THE DEVICE COUNT
TST MODE ;FORMAT OR CHECK MODE
BEQ 12\$;BR IF CHECK
TYPE ,MFCMPT ;FORMAT COMPLETE

2818 014730 012737 157700 005614
2819 014736 066137 006052 005614
2820 014744 005037 001346

19\$: INC \$DEVCT ;INCREMENT THE DEVICE COUNT
TST MODE ;FORMAT OR CHECK MODE
BEQ 12\$;BR IF CHECK
TYPE ,MFCMPT ;FORMAT COMPLETE

2821 014750 000725
2822 014752 005237 001216
2823 014756 005737 001316

20\$: INC \$DEVCT ;INCREMENT THE DEVICE COUNT
TST MODE ;FORMAT OR CHECK MODE
BEQ 12\$;BR IF CHECK
TYPE ,MFCMPT ;FORMAT COMPLETE

2824 014762 001403
2825 014764 104401 073606

21\$: INC \$DEVCT ;INCREMENT THE DEVICE COUNT
TST MODE ;FORMAT OR CHECK MODE
BEQ 12\$;BR IF CHECK
TYPE ,MFCMPT ;FORMAT COMPLETE

MAINDEC-11-DZRNA, 'MO3 FORMATTER
DZRNA8.P11 21-JUL-77 15:32

MACY11 30(1046) 21-JUL-77 16:06 PAGE 54
END OF PASS ROUTINE

SEQ 0053

2826	014770	000402			BR	13\$	; FINISH THE MESSAGE
2827	014772	104401	033633	12\$:	TYPE	,MCCMPT	; CHECK COMPLETE
2828	014776	104401	033657	13\$:	TYPE	,NUMERR	; TOTAL ERRORS
2829	015002	013746	001126		MOV	\$ERTTL,-(SP)	; SAVE \$ERTTL FOR TYPEOUT
2830	015006	104405			TYPDS		; GO TYPE--DECIMAL ASCII WITH SIGN
2831	015010	104401	001203		TYPE	,SCRLF	
2832	015014	005737	001404		TST	DEVMP	; ALL ASSIGNED DEVICE DONE ?
2833	015020	001402			BEQ	14\$	; BRANCH IF ALL DONE
2834	015022	005337	001214		DEC	\$PASS	; RESTORE THE PASS COUNT
2835	015026	000240		14\$:	NOP		
2836							
2837							
2838	015030	005237	001214		INC	\$PASS	; INCREMENT THE PASS NUMBER
2839	015034	042737	100000	001214	BIC	#100000,\$PASS	; DON'T ALLOW A NEG. NUMBER
2840	015042	005327			DEC	(PC)+	; LOOP?
2841	015044	000001		\$EOPCT:	.WORD	1	
2842	015046	003013			BGT	\$DOAGN	; YES
2843	015050	012737			MOV	(PC)+,2(PC)+	; RESTORE COUNTER
2844	015052	000001		\$ENDCT:	.WORD	1	
2845	015054	015044			\$EOPCT		
2846	015056	013700	000042	\$GET42:	MOV	2#42,R0	; GET MONITOR ADDRESS
2847	015062	001405			BEQ	\$DOAGN	; BRANCH IF NO MONITOR
2848	015064	000005			RESET		; CLEAR THE WORLD
2849	015066	004710		\$ENDAD:	JSR	PC,(R0)	; GO TO MONITOR
2850	015070	000240			NOP		; SAVE ROOM
2851	015072	000240			NOP		; FOR
2852	015074	000240			NOP		; ACT11
2853	015076			\$DOAGN:			
2854	015076	000137			JMP	2(PC)+	; RETURN
2855	015100	015102		\$RTNAD:	.WORD	DONE	
2856	015102	105737	001150	DONE:	TSTB	\$AUTOB	; RUN UNDER APT
2857	015106	001405			BEQ	2\$	; NO
2858	015110	005737	001404		TST	DEVMP	; ALL DRIVES ARE DONE ?
2859	015114	001427			BEQ	4\$	; BRANCH IF ALL DONE
2860	015116	000137	007360		JMP	MSFX	; NEXT DRIVE FROM APT DEVICE MAP
2861	015122	032777	000001	164024	2\$:	BIT	#SWO,2SWR
2862	015130	001002			BNE	1\$	; BR IF SET
2863	015132	000137	010106		JMP	M0	; ASK FOR NEXT DRIVE
2864	015136	012706	001100	1\$:	MOV	#STACK,SP	; RESET STACK POINTER
2865	015142	005737	001316		TST	MODE	; VERIFY MODE ?
2866	015146	001402			BEQ	3\$	; YES
2867	015150	000137	011744		JMP	CKADRS	; DO THE FORMAT OR CHECK AGAIN
2868	015154	012737	133331	001340	3\$:	MOV	#133331,PATA
2869	015162	012737	133331	001342		MOV	#133331,PATB
2870	015170	000137	011744		JMP	CKADRS	; LOOP ON CHECK
2871	015174	005237	001214	4\$:	INC	\$PASS	; GARBAGE CODE FOR APT
2872	015200	005237	001212	5\$:	INC	\$TESTN	
2873	015204	005237	001216		INC	\$DEVCT	
2874	015210	000773			BR	5\$	
2875	015212	104401	001203	REJEC1:	TYPE	,SCRLF	
2876	015216	104401	034365		TYPE	,MESG4	
2877	015222	000000			HALT		
2878	015224	000137	000200		JMP	2#200	
2879							
2880							
2881							

```

2882
2883
2884
2885
2886
2887
2888
2889
2890 015230 010146
2891 015232 005001
2892 015234 033727 005626
2893 015240 060006
2894 015242 001025
2895 015244 033727 005626
2896 015250 010000
2897 015252 001020
2898 015254 033727 005626
2899 015260 006000
2900 015262 001013
2901 015264 033727 005626
2902 015270 001400
2903 015272 001006
2904 015274 033727 005626
2905 015300 000020
2906 015302 001001
2907 015304 005201
2908 015306 005201
2909 015310 005201
2910 015312 005201
2911 015314 005201
2912 015316 006301
2913 015320 060166 000002
2914 015324 012601
2915 015326 000207
2916
2917 015330 032777 001000 163616
2918 015336 001402
2919 015340 000177 163560
2920 015344 005037 001174
2921 015350 033727 005626
2922 015354 072002
2923 015356 001004
2924 015360 032737 004000 005554
2925 015366 001402
2926 015370 000137 014506
2927 015374 000207
2928
2929
2930
2931 015376 113737 001220 005610
2932 015404 105037 005613
2933 015410 013737 001356 005614
2934 015416 012737 037554 005616
2935 015424 105037 005620
2936 015430 113737 001326 005621
2937 015436 013737 001322 005622

```

.SBTTL SUPPORT SUBROUTINES

;*****

;THIS ROUTINE DETERMINES THE ERROR TYPE

```

ERINDX: MOV R1,-(SP) ;STORE R1
        CLR R1 ;CLEAR R1
        BIT FMTDPB+16,(PC)+ ;CHECK ERROR TYPE
        .WORD BIT14:BIT13:BIT12:BIT11 ;DRIVE OFFLINE
        BNE 5$ ;BR IF OFFLINE
        BIT FMTDPB+16,(PC)+ ;CHECK ERROR TYPE
        .WORD BIT12 ;DRIVE PERSISTENTLY UNSAFE
        BNE 4$ ;BR IF UNSAFE
        BIT FMTDPB+16,(PC)+ ;CHECK ERROR TYPE
        .WORD BIT11:BIT10 ;UNCORRECTABLE MASSIVE PARITY ERROR ?
        BNE 3$ ;BR IF PARITY ERROR
        BIT FMTDPB+16,(PC)+ ;CHECK ERROR TYPE
        .WORD BIT09:BIT08 ;SOFTWARE TIMEOUT ?
        BNE 2$ ;BR IF YES
        BIT FMTDPB+16,(PC)+ ;CHECK ERROR TYPE
        .WORD BIT04 ;DRIVE UNSAFE ERROR ?
        BNE 1$ ;BR IF YES
        INC R1 ;INCREMENT THE RETURN INDEX
1$: INC R1 ;INCREMENT THE RETURN INDEX
2$: INC R1 ;INCREMENT THE RETURN INDEX
3$: INC R1 ;INCREMENT THE RETURN INDEX
4$: INC R1 ;INCREMENT THE RETURN INDEX
5$: ASL R1 ;DOUBLE THE INCREMENT
        ADD R1,2(SP) ;DEVELOP THE RETURN ADDRESS
        MOV (SP)+,R1 ;RESTORE R1
        RTS PC ;RETURN

```

;ROUTINE TO CHECK FOR LOOP ON ERROR

```

LOP.CK: BIT #SW09,2SWR ;LOOP ON ERROR ?
        BEQ 1$ ;BR IF NOT
        JMP 2$LPERR ;GO TO THE LOOP ON ERROR ADDRESS
1$: CLR $ESCAPE ;CLEAR THE ERROR ESCAPE ADDRESS
        BIT FMTDPB+16,(PC)+ ;CHECK FOR 'FATAL' ERROR
        .WORD BIT14:BIT13:BIT12:BIT10:BIT01 ;'FATAL' ERROR BITS
        BNE 2$ ;BR IF NOT
        BIT $WLE,RM.REG+RMER1 ;WRITE LOCK ERROR ?
        BEQ 3$ ;BR IF NOT
        JMP $EOP ;TERMINATE THE FORMAT
2$: JMP $EOP
3$: RTS PC ;RETURN

```

;THIS ROUTINE SETS UP THE INPUT TABLE FOR THE DRIVER ROUTINE

```

SETTBL: MOVB DRIVE,FMTDPB ;SET UP DRIVE #
        CLRB FMTDPB+3 ;CLEAR HIGH ORDER ADRS BITS
        MOV #MC,FMTDPB+4 ;LOAD UP WORD COUNT
        MOV #BUFP,FMTDPB+6 ;LOAD UP CURRENT ADRS
        CLRB FMTDPB+10 ;SET SECTOR TO ZERO
        MOVB BEGTRK,FMTDPB+11 ;SET UP STARTING TRK ADRS
        MOV BEGCYL,FMTDPB+12 ;SET UP STARTING CYL

```

E05

MAINDEC-11-DZMA, RM03 FORMATTER
DZMA8.P11 21-JUL-77 15:32

MACY11 30(1046) 21-JUL-77 16:06 PAGE 56
SUPPORT SUBROUTINES

SEQ 0055

```

2938 015444 005037 005626          CLR      FMTDPB+16      ;CLEAR RM03 STATUS
2939 015450 013737 001326 001334    MOV      BEGTRK,TRKCNT ;SET UP PARTIAL CYL TRACK COUNT
2940 015456 013737 001330 001332    MOV      TTRKS,TTRKSC ;SET UP TOTAL TRACKS COUNTER
2941 015464 000207                    RTS      PC          ;RETURN FROM SETUP
2942
2943 ;THIS ROUTINE CONTROLS THE DISK ADDRESSING AND TOTAL TRK COUNT
2944 ;IT IS ENTERED AFTER EVERY TRK OPERATION
2945
2946 015466 005337 001332          TRKTST: DEC      TTRKSC      ;SEE IF LAST TRACK
2947 015472 001472                    BEQ      3$          ;BRANCH IF LAST TRACK
2948 015474 022737 000004 001334    CMP      #4.,TRKCNT    ;IS THIS THE LAST TRACK IN THE CYLINDER ?
2949 015502 001423                    BEQ      1$          ;BRANCH IF SO
2950 015504 005237 001334          INC      TRKCNT        ;COUNT UP TRACK WITHIN CYLINDER
2951 015510 105237 005621          INCB     FMTDPB+11      ;INCREMENT TRACK NUMBER
2952 015514 022737 001466 005622    CMP      #822.,FMTDPB+12 ;LAST CYLINDER AND LAST TRACK ?
2953 015522 003010                    BGT      5$          ;NO
2954 015524 122737 000004 005621    CMPB    #4,FMTDPB+11 ;LAST TRACK ?
2955 015532 003004                    BGT      5$          ;NO
2956 015534 005337 001332          DEC      TTRKSC        ;DECREMENT ONE TRACK
2957 015540 001447                    BEQ      3$          ;BRANCH IF LAST TRACK
2958 015542 000403                    BR       1$          ;DON'T DESTROY THE LAST TRACK OF CYL822
2959 015544 004737 016054          5$:   JSR      PC,UPDATK  ;UPDATE TRACK ADDRESS IN BUFFER
2960 015550 000441                    BR       2$          ;EXIT
2961 015552 005037 001334          1$:   CLR      TRKCNT        ;CLEAR TRACK COUNT FOR NEXT CYLINDER
2962 015556 105037 005621          CLRB    FMTDPB+11      ;RESET TRACK ADDRESS TO 0
2963 015562 005237 005622          INC      FMTDPB+12      ;UPDATE CYLINDER NUMBER
2964 015566 022737 001467 005622    CMP      #823.,FMTDPB+12 ;LAST CYLINDER REACHED ?
2965 015574 003025                    BGT      4$          ;NO
2966 015576 013746 001322          MOV      BEGCTL,-(SP)    ;SAVE BEGCTL
2967 015602 013746 001326          MOV      BEGTRK,-(SP)    ;SAVE BEGTRK
2968 015606 005037 001322          CLR      BEGCTL        ;RESET THE STARTING CYL
2969 015612 005037 001326          CLR      BEGTRK        ;RESET THE START TRACK
2970 015616 004737 015710          JSR      PC,SETHDR    ;INITIATE HEAD WORDS
2971 015622 005037 005620          CLR      FMTDPB+10      ;CLEAR TRACK AND SECTOR
2972 015626 005037 005622          CLR      FMTDPB+12      ;CLEAR CYLINDER #
2973 015632 005037 005626          CLR      FMTDPB+16      ;CLEAR STATUS
2974 015636 012637 001326          MOV      (SP)+,BEGTRK  ;RESTORE BEGTRK
2975 015642 012637 001322          MOV      (SP)+,BEGCTL  ;RESTORE BEGCTL
2976 015646 000402                    BR       2$          ;
2977 015650 004737 015776          4$:   JSR      PC,UPDACY  ;UPDATE CYLINDER NUMBER IN BUFFER
2978 015654 062716 000002          2$:   ADD      #2,(SP)    ;ADD TWO TO RETURN ADRS
2979 015660 000207                    3$:   RTS      PC          ;RETURN
2980
2981 ;THIS ROUTINE SETS UP WC & BA FOR REMAINING SECTORS
2982 ;AFTER A WRITE CHECK ERROR
2983
2984 015662 013737 001352 005614    SCALWC: MOV      SAVWC,FMTDPB+4 ;SET UP WC FOR REMAINING SECTORS
2985 015670 062737 001004 005616    ADD      #516.,FMTDPB+6 ;SET UP BA FOR REMAINING SECTORS
2986 015676 005237 005620          INC      FMTDPB+10      ;ADVANCE TO NEXT SECTOR IN TBL
2987 015702 005037 001314          CLR      SOFSW        ;RESET RETRY COUNTER
2988 015706 000207                    RTS      PC          ;RETURN TO COMPLETE TRK FORMAT
2989
2990 ;THIS ROUTINE SETS UP THE CYLINDER ADRS, FORMAT BIT, TRACK AND
2991 ;SECTOR ADRS IN MEMORY WITH THE STARTING CYL - TRK INFORMATION
2992
2993

```

2994	015710	013701	001364		SETHDR: MOV	MAXSEC, R1	;SET UP SECTOR COUNT
2995	015714	012737	000001	001212	MOV	#1, STSTN	;GARBAGE CODE FOR APT
2996	015722	012700	037554		MOV	#BUFF, R0	;SET UP HEADER POINTER IN R0
2997	015726	013702	001322		MOV	BEGCYL, R2	;PUT STARTING CYL # IN R2
2998	015732	052702	140000		BIS	#140000, R2	;BIT 15, BIT 14 ARE SET FOR 144 SPC 3/14/77
2999	015736	013703	001326		MOV	BEGTRK, R3	;PUT STARTING TRK # IN R3
3000	015742	000303			SWAB	R3	;JUSTIFY TRACK ADRS
3001	015744	005737	001362		TST	SEC30	;SEE IF 30 OR 32 SECTOR MODE
3002	015750	001402			BEQ	1\$	;BR IF 30 SECTOR MODE
3003	015752	052702	010000		BIS	#10000, R2	;SET THE 32 SECTOR FORMAT BIT
3004	015756	010220			1\$: MOV	R2, (R0)+	;WRITE IN HEADER AREA OF CORE THE CYL ADRS
3005	015760	010320			MOV	R3, (R0)+	;WRITE IN HEADER AREA OF CORE THE TRK ADRS
3006	015762	062700	001000		ADD	#512, R0	;SET UP FOR NEXT HEADER
3007	015766	005203			INC	R3	;UPDATE SECTOR ADRS FOR NEXT HEADER
3008	015770	005301			DEC	R1	;MAINTAIN COUNT OF SECTORS
3009	015772	002371			BGE	1\$	;BRANCH IF NOT LAST SECTOR
3010	015774	000207			RTS	PC	;EXIT - HEADERS ARE LOADED INTO CORE
3011							
3012							
3013							
3014	015776	013701	001364		UPDACY: MOV	MAXSEC, R1	;SET UP SECTOR COUNT
3015	016002	012700	037554		MOV	#BUFF, R0	;SET UP HEADER POINTER IN R0
3016	016006	010246			MOV	R2, -(SP)	;SAVE R2
3017	016010	005220			1\$: INC	(R0)+	;INCREMENT FOR NEXT CYLINDER
3018	016012	042720	177400		BIC	#177400, (R0)+	;RESET TRK ADRS TO 0
3019	016016	012702	001000		MOV	#512, R2	;DATA FIELD LENGTH
3020	016022	013720	001340		2\$: MOV	PATA, (R0)+	;FILL BUFFER
3021	016026	013720	001342		MOV	PATB, (R0)+	;FILL BUFFER
3022	016032	162702	000004		SUB	#4, R2	;END OF DATA FIELD ?
3023	016036	001371			BNE	2\$	;NO
3024	016040	005301			DEC	R1	;COUNT SECTORS
3025	016042	002362			BGE	1\$	;BRANCH IF NOT LAST SECTOR
3026	016044	012602			MOV	(SP)+, R2	;RESETORE THE R2
3027	016046	005237	001212		INC	STSTN	;INCREMENT TEST NUMBER FOR EACH CYLINDER CHANGE
3028							;GARBAGE CODE FOR APT
3029	016052	000207			RTS	PC	;EXIT
3030							
3031							
3032							
3033	016054	013701	001364		UPDATK: MOV	MAXSEC, R1	;SET UP SECTOR COUNT
3034	016060	012700	037554		MOV	#BUFF, R0	;SET UP HEADER POINTER IN R0
3035	016064	005720			TST	(R0)+	;POINT HEADER POINTER TO TRK - SEC ADRS
3036	016066	062710	000400		1\$: ADD	#400, (R0)	;INDEX TRK ADRS
3037	016072	062700	001004		ADD	#516, R0	;SET UP FOR NEXT HEADER
3038	016076	005301			DEC	R1	;COUNT SECTORS
3039	016100	002372			BGE	1\$	;BRANCH IF NOT LAST SECTOR
3040	016102	000207			RTS	PC	;EXIT
3041							
3042							
3043							
3044	016104	012737	016162	000004	ST.CLK: MOV	#STCLK1, @ERRVEC	;SET UP VECTOR FOR P CLK
3045	016112	005037	000006		CLR	@ERRVEC+2	;NEW PSW
3046	016116	005777	163154		TST	@SLKCSR	;CHECK FOR KW11-P
3047	016122	013746	001302		MOV	SLPVEC, -(SP)	;VECTOR ADDRESS
3048	016126	012776	016312	000000	MOV	#CLOCK, @ (SP)	;SET UP KW11-P VECTOR
3049	016134	062716	000002		ADD	#2, (SP)	;POINT TO PSW

```

3050 016140 012736 000300      MOV      #PR6,2(SP)+      ;PSW - PRI 6
3051 016144 012777 177777 163126  MOV      #-1,2$CLKCSB      ;LOAD COUNTER BUFFER
3052 016152 012777 000135 163116  MOV      #135,2$CLKCSR      ;SET CLK - CNT UP
3053 016160 000426          BR      STCLK3
3054 016162 062706 000004      STCLK1: ADD      #4,SP      ;RESTORE THE STACK POINTER
3055 016166 012737 016232 000004  MOV      #STCLK2,2$ERRVEC      ;CHANGE ERROR VECTOR
3056 016174 005777 163106      TST      2$CLKS      ;LOOK FOR KW11-L
3057 016200 013746 001310      MOV      $LLVEC,-(SP)      ;KW11-L VECTOR ADDRESS
3058 016204 012776 016312 000000  MOV      #CLOCK,2(SP)      ;SET UP KW11-L VECTOR
3059 016212 062716 000002      ADD      #2,(SP)      ;INCREMENT VECTOR ADDRESS
3060 016216 012736 000300      MOV      #PR6,2(SP)+      ;PSW - PRI 6
3061 016222 012777 000100 163056  MOV      #100,2$CLKS      ;SET KW11-L INTERRUPT ENABLE
3062 016230 000402          BR      STCLK3
3063 016232 062706 000004      STCLK2: ADD      #4,SP      ;RESTORE THE STACK POINTER
3064 016236 012737 000006 000004  STCLK3: MOV      #6,2$ERRVEC      ;RESTORE THE ERROR VECTOR
3065 016244 000207          RTS      PC
3066
3067      ;THIS CODE PRINT THE ABORT MESSAGE AND LET O.P. RESTART
3068
3069 016246 105737 001150      RESTRT: TSTB     $AUT08      ;RUN UNDER APT ?
3070 016252 001016          BNE      1$      ;BRANCH IF SO
3071 016254 004737 021772      JSR      PC,$TKINT      ;ENABLE TO READ
3072 016260 104401 034204      TYPE     ,MSG2      ;IMPROPER MFG INFORMATION
3073 016264 104411          ROLIN
3074 016266 012601          MOV      (SP)+,R1      ;LOCATE THE READ IN LINE
3075 016270 105711          TSTB     (R1)      ;TERMINATOR BY <CR>
3076 016272 001406          BEQ      1$      ;BRANCH IF SO
3077 016274 121127 000131      CMPB     (R1),#Y      ;ENTER Y ?
3078 016300 001403          BEQ      1$      ;BRANCH IF SO
3079 016302 000000          HALT
3080 016304 000137 000200      JMP      2$200      ;JUMP TO 200 IF HIT START SWITCH
3081 016310 000207      1$:      RTS      PC      ;EXIT
3082
3083      ;THIS CODE SERVICES A CLOCK INTERRUPT EVERY 16MS
3084
3085 016312 012746 000020      CLOCK: MOV      #16,-(SP)      ;PUT MILLISECONDS ON THE STACK
3086 016316 004737 030404      JSR      PC,$PTMR      ;GO REPORT TIME
3087 016322 000002          RTI      ;RETURN AND CONTINUE
3088
3089      ;ROUTINE TO INTERCEPT 'CONTROL C' TYPED DURING PARAMETER ENTRY TIME
3090
3091 016324 012706 001100      OENTER: MOV      #STACK,SP      ;INITIALIZE THE STACK
3092 016330 005737 024126      1$:      TST      TRNSWT      ;ALL ACTIVITY STOPPED ?
3093 016334 001375          BNE      1$      ;BR IF NOT
3094 016336 000005          RESET
3095 016340 000137 010046      JMP      M1B      ;CLEAR THE BUS
3096      ;START AGAIN WITH DRIVE SELECTION
3097
3098      ;ROUTINE TO TYPE THE PRESENT DISK ADDRESS. ENTERED BY TYPING 'CONTROL C'
3099
3099 016344 117746 162612      NEWSVC: MOVB     2$TKB,-(SP) ;READ FROM TTY BUFFER
3100 016350 042716 177600      BIC      #177,(SP)      ;STRIP PARITY
3101 016354 022627 000017      CMP      (SP)+,#15.      ;CONTROL-0 ?
3102 016360 001406          BEQ      1$      ;YES
3103 016362 005777 162574      TST      2$TKB      ;CLEAR DONE BIT
3104 016366 012777 000100 162564  MOV      #100,2$TKS      ;ENABLE TTY INTERRUPT
3105 016374 000002          RTI

```

H05

MAINDEC-11-DZMA, RM03 FORMATTER
DZMA8.P11 21-JUL-77 15:32MACY11 30(1046) 21-JUL-77 16:06 PAGE 59
SUPPORT SUBROUTINES

SEQ 0058

3106	016376	004737	021772		1\$: JSR	PC,STKINT	: INITIAL VECTOR
3107	016402	005037	177776		TYPADR: CLR	PSW	: SET PROCESSOR TO PRIORITY 0
3108	016406	012737	016324	001376		MOV #OENTER,CNTLC	: CHANGE 'CONTROL C' RETURN ADDRESS
3109	016414	104401	033707			TYPE ,ADDRIS	: 'PRESENT ADDRESS IS: '
3110	016420	104401	032473			TYPE C	: 'C'
3111	016424	013746	005622			MOV FMTDPB+12,-(SP)	: PUT THE CYLINDER ADDRESS ON THE STACK
3112	016430	004737	023436			JSR PC,\$S820	: CONVERT IT TO DECIMAL
3113	016434	004737	023376			JSR PC,\$SUPRS	: TYPE IT
3114	016440	104401	032503			TYPE ,LINSF	: SPACES
3115	016444	104401	032475			TYPE T	: 'T'
3116	016450	005046				CLR -(SP)	: CLEAR THE STACK
3117	016452	113716	005621			MOVB FMTDPB+11,(SP)	: PUT THE TRACK ADDRESS ON THE STACK
3118	016456	004737	023436			JSR PC,\$S220	: CONVERT IT TO DECIMAL
3119	016462	004737	023376			JSR PC,\$SUPRS	: TYPE IT
3120	016466	104401	001203			TYPE \$CRLF	: CR-LF
3121	016472	012737	016344	000060		MOV #NEWSVC,2*TKVEC	: RESTORE ENTRANCE TO THIS ROUTINE
3122	016500	000002				RTI	: RETURN
3123							
3124							
3125							
3126							
3127							
3128							
3129	016502	010346					
3130	016504	012703	005650				
3131	016510	012337	016520				
3132	016514	001436					
3133	016516	104401					
3134	016520	000000					
3135	016522	012302					
3136	016524	012305					
3137	016526	011546					
3138	016530	104405					
3139	016532	104401	032467				
3140	016536	104411					
3141	016540	012601					
3142	016542	004537	017730				
3143	016546	016510					
3144	016550	016612					
3145	016552	016566					
3146	016554	016562					
3147	016556	016566					
3148	016560	016600					
3149	016562	010215					
3150	016564	000751					
3151	016566	104401	033230				
3152	016572	162703	000006				
3153	016576	000744					
3154	016600	010215					
3155	016602	000403					
3156	016604	012703	005650				
3157	016610	000737					
3158	016612	012603					
3159	016614	000207					
3160							
3161							

;PARAMETER ENTRY ROUTINE

;CALL

; MOV #ADR,R3
; JSR PC,PARENT;PARAMETER TABLE ADDRESS
;GET THE PARAMETERSPARENT: MOV R3,-(SP)
MOV #TABLE,R3;SAVE R3
;PARAMETER TABLE ADDRESS1\$: MOV (R3)+,3\$
BEQ 9\$;ADDRESS OF PARAMETER NAME
;BR IF AT END OF TABLE

3\$: .WORD 0

;TYPE THE PARAMETER NAME
;ADDRESS OF PARAMETER NAME TEXTMOV (R3)+,R2
MOV (R3)+,R5;MAXIMUM PARAMETER VALUE
;ADDRESS OF PARAMETER

MOV (R5),-(SP)

;CURRENT VALUE OF PARAMETER
;TYPE THE CURRENT VALUE OF THE PARAMETERTYPDS
TYPE ,SLASH

; '/ '

RDLIN
MOV (SP)+,R1;READ THE KEYBOARD
;INPUT ASCII STRING ADDRESS

JSR R5,CK.DIG

;CHECK THE DIGIT(S)
;CARRIAGE RETURN ONLY ENTERED1\$
9\$;PERIOD ONLY ENTERED
;ILLEGAL INPUT6\$
5\$
6\$
7\$;TERMINATED WITH A CARRIAGE RETURN
;TERMINATED WITH A "-"
;TERMINATED WITH A "-."5\$: MOV R2,(R5)
BR 1\$;MOVE NEW VALUE TO PARAMETER LOCATION
;GET MORE PARAMETERS6\$: TYPE ,BADENT
SUB #6,R3; 'BAD ENTRY'
;DECREMENT THE TABLE POINTER1\$
7\$: MOV R2,(R5)
BR 9\$;TRY AGAIN
;NEW VALUE8\$: MOV #TABLE,R3
BR 1\$;EXIT
;RELOAD THE PARAMETER TABLE ADDRESS
;TRY AGAIN9\$: MOV (SP)+,R3
RTS PC;RESTORE R3
;RETURN

;THIS ROUTINE LOAD THE BAD SECTOR INTO USTAB TABLE

;CALL SEQ JSR PC,RECORD

```

3162
3163
3164 016616 012701 003434
3165 016622 022711 177777
3166 016626 001421
3167 016630 023711 005622
3168 016634 001010
3169 016636 123761 005621 000003
3170 016644 001004
3171 016646 123761 005620 000002
3172 016654 001416
3173 016656 062701 000004
3174 016662 022701 004424
3175 016666 101355
3176 016670 000411
3177 016672 013711 005622
3178 016676 113761 005621 000003
3179 016704 113761 001350 000002
3180 016712 000207
3181 016714 104401 001203
3182 016720 104401 034103
3183 016724 000000
3184 016726 000137 000200
3185
3186
3187
3188
3189 016732 012701 003434
3190 016736 012702 000175
3191 016742 012703 004420
3192 016746 022711 177777
3193 016752 001445
3194 016754 021161 000004
3195 016760 101013
3196 016762 103426
3197 016764 126161 000003 000007
3198 016772 101006
3199 016774 103421
3200 016776 126161 000002 000006
3201 017004 101001
3202 017006 000414
3203 017010 011146
3204 017012 016146 000002
3205 017016 016111 000004
3206 017022 016161 000006 000002
3207 017030 012661 000006
3208 017034 012661 000004
3209 017040 062701 000004
3210 017044 020103
3211 017046 001342
3212 017050 005302
3213 017052 001405
3214 017054 162703 000004
3215 017060 012701 003434
3216 017064 000733
3217 017066 000207

```

;ALL REGISTER USED ARE DESTORIED

```

RECORD: MOV      #USTAB+10,R1      ;TABLE ENTRY IN R1
1$:      CMP      #-1,(R1)         ;AN OPENING IN THE TABLE
        BEQ      3$               ;BRANCH IF IT IS
        CMP      FMTDPB+12,(R1)   ;CYLINDER NUMBER MATCHES ?
        BNE      2$               ;BRANCH IF NOT
        CMPB     FMTDPB+11,3(R1)  ;TRK NUMBER MATCHES ?
        BNE      2$               ;BRANCH IF NOT
        CMPB     FMTDPB+10,2(R1)  ;SECTOR NUMBER MATCHES ?
        BEQ      4$               ;BRANCH TO EXIT IF THE SPOT HAS RECORDED
2$:      ADD      #4,R1            ;INCREMENT 4 BYTES
        CMP      #USTAB+512.,R1   ;END OF TABLE ?
        BHI      1$               ;BRANCH IF NOT
        BR       5$               ;THROW AWAY THE PACK
3$:      MOV      FMTDPB+12,(R1)   ;LOAD THE CYLINDER #
        MOVB     FMTDPB+11,3(R1)  ;LOAD THE TRK NUMBER
        MOVB     SAVSEC,2(R1)     ;LOAD THE SECTOR NUMBER
4$:      RTS      PC              ;EXIT
5$:      TYPE     ,SCRLF           ;
        TYPE     ,MSG3           ;THROW AWAY THE PACK
        HALT
        JMP      @#200           ;RESTART IF DESIRED ?

```

;THIS ROUTINE SORT THE USTAB IN ASCENDING ORDER
;CALL SEQ JSR PC, SORT2

```

SORT2:  MOV      #USTAB+8.,R1      ;TOP POINTER
        MOV      #125.,R2         ;TOTAL 126 ELEMENTS
        MOV      #USTAB+508.,R3   ;THE LAST ELEMENT
        CMP      #-1,(R1)         ;THE TABLE IS EMPTY ?
        BEQ      4$               ;QUICK EXIT
1$:      CMP      (R1),4(R1)       ;COMPARE THE CYLINDER NUMBER
        BHI      2$               ;SWITCH THE TWO ELEMENT
        BLO      3$               ;INCREMENT POINTER
        CMPB     3(R1),7(R1)      ;COMPARE THE TRK NUMBER
        BHI      2$               ;SWITCH ELEMENT
        BLO      3$               ;INCREMENT POINTER
        CMPB     2(R1),6(R1)      ;COMPARE THE SECTOR NUMBER
        BHI      2$               ;SWITCH ELEMENT
        BR       3$               ;INCREMENT POINTER
2$:      MOV      (R1),-(SP)       ;SAVE THE N TH BLOCK
        MOV      2(R1),-(SP)      ;INTO STACK
        MOV      4(R1),(R1)       ;SWITCH THE N+1 BLOCK TO N BLOCK
        MOV      6(R1),2(R1)
        MOV      (SP)+,6(R1)
        MOV      (SP)+,4(R1)
3$:      ADD      #4,R1            ;UPDATE THE TOP POINTER
        CMP      R1,R3            ;REACH THE BOTTOM ?
        BNE      1$               ;BRANCH IF NOT
        DEC      R2               ;DECREMENT ONE SORT COUNT
        BEQ      4$               ;BRANCH IF ALL DONE
        SUB      #4,R3            ;ASJUST THE BOTTOM POINTER
        MOV      #USTAB+8.,R1     ;RESET TOP POINTER
        BR       1$               ;BRANCH IF NOT ALL DONE
4$:      RTS      PC              ;EXIT

```

```

3218
3219
3220
3221
3222
3223
3224 017070 012700 037554
3225 017074 012701 020100
3226 017100 012702 177777
3227 017104 010220
3228 017106 005301
3229 017110 001375
3230 017112 013746 001212
3231 017116 013746 001322
3232 017122 013746 001326
3233 017126 013746 001364
3234 017132 013746 001362
3235 017136 012737 000037 001364
3236 017144 012737 177777 001362
3237 017152 012737 001466 001322
3238 017160 012737 000004 001326
3239 017166 004737 015710
3240 017172 012637 001362
3241 017176 012637 001364
3242 017202 012637 001326
3243 017206 012637 001322
3244 017212 012637 001212
3245 017216 012701 0014.4
3246 017222 012702 037560
3247 017226 012703 000040
3248 017232 011112
3249 017234 016162 000002 000002
3250 017242 005062 000004
3251 017246 005062 000006
3252 017252 062702 001004
3253 017256 005303
3254 017260 001364
3255 017262 005046
3256 017264 005046
3257 017266 005046
3258 017270 012746 001414
3259 017274 012746 037560
3260 017300 012746 000005
3261 017304 012746 002420
3262 017310 012746 040564
3263 017314 012746 000005
3264 017320 012746 003424
3265 017324 012746 051630
3266 017330 005737 001362
3267 017334 100402
3268 017336 012716 052634
3269 017342 012746 000013
3270 017346 012701 004430
3271 017352 012702 052634
3272 017356 005737 001362
3273 017362 100402

```

```

;THIS ROUTINE TEXT THE LAST TRACK CYL 822, TRK 4
;SECTORS 0,2,4,6,8 16 BIT MFG
;SECTORS 1,3,5,7,9 18 BIT MFG
;SECTORS 10,12,14,16,.....30, 16 BIT USER
;SECTORS 11,13,15,17,.....31 18 BIT USER

```

```

TEXT:  MOV    #BUFF,R0      ;BUFFER ADDRESS
        MOV    #<258.*32.>,R1 ;TOTAL WORD COUNT
        MOV    #-1,R2      ;SET ALL LOCATIONS TO -1
1$:    MOV    R2,(R0)+      ;FILL THE BUFFER
        DEC    R1          ;ALL DONE ?
        BNE    1$          ;LOOP, IF NOT
        MOV    $TESTN,-(SP) ;SAVE TESTN,BEGCYL,BEGTRK
        MOV    BEGCYL,-(SP)
        MOV    BEGTRK,-(SP)
        MOV    MAXSEC,-(SP) ;SAVE FLAG MAXSEC AND SEC30
        MOV    SEC30,-(SP)
        MOV    #37,MAXSEC   ;SET MAX 31 SECTORS
        MOV    #-1,SEC30    ;ALWAYS IN 16 BIT MODE
        MOV    #822,BEGCYL ;LOAD LAST CYLINDER
        MOV    #4,BEGTRK    ;LAST TRACK
        JSR    PC,SETHOR    ;SET UP THE HEAD FIELD IN THE BUFFER
        MOV    (SP)+,SEC30  ;RESTORE SEC30 AND MAXSEC
        MOV    (SP)+,MAXSEC
        MOV    (SP)+,BEGTRK
        MOV    (SP)+,BEGCYL
        MOV    (SP)+,$TESTN
        MOV    #BAD16,R1    ;LOAD SERIAL NUMBER TO EVERY SECTOR
        MOV    #BUFF+4,R2   ;BUFFER LOCATION + 4
        MOV    #32,R3      ;TOTAL 32 SECTORS
2$:    MOV    (R1),(R2)     ;1 ST SN #
        MOV    2(R1),2(R2) ;2 ND SN #
        CLR    4(R2)       ;THIRD WORD = 0
        CLR    6(R2)       ;ID = DATA PACK
        ADD    #516.,R2    ;ADVANCE TO NEXT SECTOR
        DEC    R3          ;DECREMENT ONE SECTOR COUNT
        BNE    2$          ;BRANCH IF NOT DONE
        CLR    -(SP)       ;LOAD TABLE INTO SECTORS
        CLR    -(SP)       ;DUMMY PAIRS
        CLR    -(SP)
        MOV    #BAD16,-(SP) ;TABLE ADDRESS
        MOV    #BUFF+4,-(SP) ;BUFFER ADDRESS
        MOV    #5,-(SP)    ;SECTOR COUNT
        MOV    #BAD18,-(SP) ;TABLE ADDRESS
        MOV    #BUFF+4+516.,-(SP) ;BUFFER ADDRESS
        MOV    #5,-(SP)    ;SECTOR COUNT
        MOV    #USTAB,-(SP) ;USER DEFINED TABLE
        MOV    #BUFF+4+(516.*10.),-(SP) ;BUFFER ADDRESS
        TST    SEC30
        BMI    3$          ;BRANCH IF IN 16 BIT MODE
        MOV    #BUFF+4+(516.*11.),-(SP) ;RESET THE BUFFER ADDRESS
3$:    MOV    #11,-(SP)    ;SECTOR COUNT
        MOV    #USSAV,R1   ;TABLE ADDRESS IN R1
        MOV    #BUFF+4+(516.*11.),R2 ;BUFFER ADDRESS IN R2
        TST    SEC30
        BMI    4$          ;BRANCH IF IN 16 BIT MODE
        MOV    #4$

```

K05

MAINDEC-11-DZMA, RMO3 FORMATTER
DZMA8.P11 21-JUL-77 15:32

MACY11 30(1046) 21-JUL-77 16:06 PAGE 62
SUPPORT SUBROUTINES

SEQ 0061

3274 017364 012702 051630
3275 017370 012703 000013
3276 017374 010105
3277 017376 012704 000400
3278 017402 012122
3279 017404 005304
3280 017406 001375
3281 017410 010501
3282 017412 062702 001010
3283 017416 005303
3284 017420 001366
3285 017422 012603
3286 017424 012602
3287 017426 012601
3288 017430 010105
3289 017432 001361
3290 017434 000207

```

MOV      #BUF+4+(516.*10.),R2 ;BUFFER ADDRESS FOR 18 BIT MODE
4$:      MOV      #11.,R3      ;SECTOR COUNT IN R3
          MOV      R1,R5        ;TEMPOR STORAGE
          MOV      #256.,R4     ;WORD COUNT = DATA FIELD OF ONE SECTOR
5$:      MOV      (R1)+(R2)+    ;LOAD TABLE INTO SECTOR DATA FIELD
          DEC      R4           ;ALL DONE ?
          BNE      6$          ;BRANCH IF NOT
          MOV      R5,R1        ;RELOAD THE TABLE ADDRESS
          ADD      #520.,R2     ;SKIP ONE SECTOR
          DEC      R3           ;DECREMENT ONE SECTOR COUNT
          BNE      5$          ;BRANCH IF NOT DONE
          MOV      (SP)+,R3     ;RETRIEVE NEXT SET OF SECTOR #
          MOV      (SP)+,R2     ;BUFFER ADDRESS
          MOV      (SP)+,R1     ;TABLE ADDRESS
          MOV      R1,R5        ;TEMPOR STORAGE
          BNE      5$          ;BRANCH IF NOT DUMMY PAIR
          RTS      PC          ;EXIT

```

```

;THIS ROUTINE RESET:
;RESET THE BIT 14 OF THE 1 ST HEADER WORD OF THE BAD SPOT
;CALLING SEQ;          JSR      PC,RESET
;

```

3291
3292
3293
3294
3295
3296
3297 017436 013737 005622 037544
3298 017444 113737 005621 037547
3299 017452 113737 001350 037546
3300 017460 053737 001412 037544
3301 017466 113737 001220 005630
3302 017474 012737 177776 005634
3303 017502 112737 000163 005632
3304 017510 012737 037544 005636
3305 017516 113737 001350 005640
3306 017524 113737 005621 005641
3307 017532 013737 005622 005642
3308 017540 052737 010000 037544
3309 017546 005737 001362
3310 017552 100403
3311 017554 042737 010000 037544
3312 017562 004037 024736
3313 017566 005630
3314 017570 000774
3315 017572 005737 005646
3316 017576 001775
3317 017600 000207

```

RESET:   MOV      FMTDPB+12,RBUF ;CYLINDER ADDRESS
          MOV      FMTDPB+11,RBUF+3 ;TRACK ADDRESS
          MOV      SAVSEC,RBUF+2 ;SECTOR ADDRESS
          BIS      HEADX,RBUF     ;SET MFG BIT,RESET USER BIT
          MOV      DRIVE,FMTX    ;SETUP THE DPB FOR OPERATION
          MOV      #-2,FMTX+4     ;WORD COUNT =2
          MOV      #WRTHD,FMTX+2 ;WRITE HEAD AND DATA COMMAND
          MOV      #RBUF,FMTX+6  ;BUFFER ADDRESS
          MOV      SAVSEC,FMTX+10 ;SECTOR ADDRESS
          MOV      FMTDPB+11,FMTX+11 ;TRACK ADDRESS
          MOV      FMTDPB+12,FMTX+12 ;CYLINDER ADDRESS
          BIS      #10000,RBUF    ;ASSUM 16 BIT MODE
          TST      SEC30         ;IN 16 BIT MODE ?
          BMI      1$           ;BRANCH IF IT IS
          BIC      #10000,RBUF   ;RESET THE FMT BIT
          JSR      R0,RMO3       ;CALL THE DRIVER
          FMTX     1$           ;DPB ADDRESS
          BR       1$           ;BRANCH IF QUEUE FAILS
          TST      FMTX+16       ;DONE ?
          BEQ      2$           ;BRANCH IF NOT
          RTS      PC          ;EXIT
2$:

```

```

;THIS ROUTINE IS USED TO CHECK IF AN
;ASCII CHARACTER IS A DIGIT BETWEEN 0 AND 7.
;CALL

```

```

MOV      #ADR,R1 ;ADDRESS OF ASCII CHARACTER
JSR      R5,CK.OCT ;CHECK THE CHARACTER
RETURN1  ;CHARACTER IS NOT BETWEEN 0-7
RETURN2  ;CHARACTER IS IN R2 AS A
          ;OCTAL DIGIT

```

3318
3319
3320
3321
3322
3323
3324
3325
3326
3327
3328
3329

3330 017602 121127 000060
3331 017606 103407
3332 017610 121127 000067
3333 017614 101004
3334 017616 111102
3335 017620 042702 177770
3336 017624 005725
3337 017626 000205

CK.OCT: CMPB (R1),#0 ;LESS THAN ZERO?
BLO 1\$;YES -- BRANCH
CMPB (R1),#7 ;GREATER THAN SEVEN?
BHI 1\$;YES -- BRANCH
MOVB (R1),R2 ;GET THE CHARACTER
BIC #07,R2 ;STRIP AWAY THE ASCII
TST (R5)+ ;ADJUST FOR RETURN
1\$: RTS R5 ;RETURN

; THIS ROUTINE IS USED TO CHECK AN ASCII CHARACTER
; AND DETERMINE IF IT IS A DIGIT BETWEEN 0 AND 9.

;CALL
; MOV #ADR,R1 ;ADDRESS OF ASCII CHARACTER
; JSR R5,CK.DEC ;CHECK THE CHARACTER
; RETURN1 ;NOT BETWEEN 0 AND 9
; RETURN2 ;BETWEEN 0 AND 9
; R2 = DIGIT

3348 017630 121127 000060
3349 017634 103407
3350 017636 121127 000071
3351 017642 101004
3352 017644 111102
3353 017646 042702 000060
3354 017652 005725
3355 017654 000205

CK.DEC: CMPB (R1),#0 ;LESS THAN ZERO?
BLO 1\$;YES -- BRANCH
CMPB (R1),#9 ;GREATER THAN NINE?
BHI 1\$;YES -- BRANCH
MOVB (R1),R2 ;GET THE CHARACTER
BIC #0,R2 ;STRIP AWAY THE ASCII
TST (R5)+ ;ADJUST FOR RETURN
1\$: RTS R5 ;RETURN

; THIS ROUTINE WILL CHECK AN ASCII CHARACTER TO
; DETERMINE WHAT IT IS.

;CALL
; MOV #ADR,R1 ;ADDRESS OF ASCII CHARACTER
; JSR R5,CK.CHR ;CHECK CHARACTER
; RETURN ADR1 ;UNKNOWN CHARACTER
; RETURN ADR2 ;CARRIAGE RETURN * (R1)=ADR+1
; RETURN ADR3 ;COMMA * (R1)=ADR+1
; RETURN ADR4 ;PERIOD * (R1)=ADR+1
; RETURN ADR5 ;DIGIT BETWEEN 0 AND 7.
; RETURN ADR6 ;DIGIT BETWEEN 8 AND 9.
; R2 = DIGIT * (R1)=ADR+1

3370 017656 105711
3371 017660 001417
3372 017662 121127 000054
3373 017666 001413
3374 017670 121127 000056
3375 017674 001407
3376 017676 004537 017630
3377 017702 000410
3378 017704 004537 017602
3379 017710 005725
3380 017712 005725
3381 017714 005725
3382 017716 005725
3383 017720 005725
3384 017722 005201
3385 017724 011505

CK.CHR: TSTB (R1) ;"CARRIAGE RETURN"?
BEQ 3\$;YES -- BRANCH
CMPB (R1),#', ;"COMMA"?
BEQ 2\$;YES -- BRANCH
CMPB (R1),#'. ;"PERIOD"?
BEQ 1\$;YES -- BRANCH
JSR R5,CK.DEC ;"DIGIT"?
BR 4\$;NO -- BRANCH
JSR R5,CK.OCT ;OCTAL?
TST (R5)+ ;DIGIT BETWEEN 8-9
TST (R5)+ ;DIGIT BETWEEN 0-7
1\$: TST (R5)+ ;PERIOD
2\$: TST (R5)+ ;COMMA
3\$: TST (R5)+ ;CARRIAGE RETURN
INC R1 ;MOVE POINTER TO NEXT CHARACTER
4\$: MOV (R5),R5 ;UNKNOWN CHARACTER

3386	017726	000205	RTS	R5	;RETURN
3387					
3388			;THIS ROUTINE CHECKS AN ASCII STRING FOR LEGAL		
3389			;CHARACTERS AND FORMS A DECIMAL VALUE BINARY NUMBER IN R2.		
3390			CALL		
3391			MOV	#ADR,R1	;ADDRESS OF ASCII STRING
3392			MOV	#NUM,R2	;MAX. MAGNITUDE OF INPUT NUMBER
3393			JSR	R5,CK.DIG	;CHECK DIGITS
3394			RETURN	ADR1	"CR" ONLY ENTERED -- R2=0
3395			RETURN	ADR2	"PERIOD" ONLY ENTERED -- R2=0
3396			RETURN	ADR3	ILLEGAL CHARACTER OR INPUT TOO LARGE -- R2=?
3397			RETURN	ADR4	"CR" -- R2 = NUMBER
3398			RETURN	ADR5	"COMMA" -- R2 = NUMBER
3399			RETURN	ADR6	"PERIOD" -- R2 = NUMBER
3400					
3401	017730	010446	CK.DIG: MOV	R4,-(SP)	;SAVE R4
3402	017732	010346	MOV	R3,-(SP)	;SAVE R3
3403	017734	010246	MOV	R2,-(SP)	;SAVE THE MAX. SIZE ON THE STACK
3404	017736	005002	CLR	R2	;START WITH 0
3405	017740	005003	CLR	R3	
3406	017742	005004	CLR	R4	
3407	017744	004537	JSR	R5,CK.CHR	;CHECK ONE CHARACTER
3408	017750	020044	BS		ILLEGAL CHARACTER
3409	017752	020052	BS		CARRIAGE RETURN
3410	017754	020044	BS		" "
3411	017756	020046	BS		" "
3412	017760	017764	BS		DIGIT 0-7
3413	017762	017764	BS		DIGIT 8-9
3414	017764	062705	1S: ADD	#4,R5	STEP RETURN POINTER PAST "CR" & "PERIOD" RETURNS
3415	017770	006303	2S: ASL	R3	INPUT NUMBER *2
3416	017772	010346	MOV	R3,-(SP)	SAVE *2
3417	017774	006303	ASL	R3	*4
3418	017776	006303	ASL	R3	*8
3419	020000	062603	ADD	(SP)+,R3	(*2)+(*8) = *10
3420	020002	060203	ADD	R2,R3	UPDATE THE INPUT NUMBER
3421	020004	004537	JSR	R5,CK.CHR	CHECK ONE CHARACTER
3422	020010	020050	BS		ILLEGAL CHARACTER
3423	020012	020034	BS		CARRIAGE RETURN
3424	020014	020032	BS		" "
3425	020016	020024	BS		" "
3426	020020	017770	BS		DIGIT 0-7
3427	020022	017770	BS		DIGIT 8-9
3428	020024	105711	3S: TSTB	(R1)	DOES A "CR" FOLLOW THE "PERIOD"
3429	020026	001010	BNE	BS	BR IF NOT
3430	020030	005724	TST	(R4)+	INCREMENT THE RETURN
3431	020032	005724	4S: TST	(R4)+	INCREMENT THE RETURN
3432	020034	005724	5S: TST	(R4)+	INCREMENT THE RETURN
3433	020036	020316	CMP	R3,(SP)	CHECK THE MAGNITUDE OF THE NUMBER
3434	020040	002004	BGE	BS	BR IF ENTERED NUMBER TOO LARGE
3435	020042	000402	BR	BS	BYPASS INCREMENT
3436	020044	005725	6S: TST	(R5)+	INCREMENT RETURN PAST INVALID RETURN
3437	020046	005725	7S: TST	(R5)+	INCREMENT RETURN
3438	020050	060405	8S: ADD	R4,R5	SETUP RETURN POINTER
3439	020052	010302	9S: MOV	R3,R2	ENTERED VALUE
3440	020054	005726	TST	(SP)+	CLEAN MAX. SIZE OFF OF STACK
3441	020056	012603	MOV	(SP)+,R3	RESTORE R3

N05

MAINDEC-11-DZMA, RM03 FORMATTER
DZMA8.P11 21-JUL-77 15:32

MACY11 30(1046) 21-JUL-77 16:06 PAGE 65
SUPPORT SUBROUTINES

SEQ 0064

3442 020060 012604
3443 020062 011505
3444 020064 000205

MOV (SP)+,R4 ;RESTORE R4
MOV (R5),R5 ;GET RETURN ADDRESS
RTS R5 ;RETURN

.SBTTL MACRO ROUTINES

.SBTTL ERROR HANDLER ROUTINE

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

\$ERROR:

CKSWR
MOV R1,DORIVE ;:TEST FOR CHANGE IN SOFT-SWR
MOV R3,ATTN ;:DRIVE ADDRESS IF DRIVE ERROR CALL
MOV FMTDPB+12,DS.CYL ;:ATTENTION REGISTER CONTENTS
MOV FMTDPB+11,DS.TRK ;:CURRENT CYLINDER ADDRESS
TST RM.REG+RMB4 ;:REQUESTED TRACK ADDRESS
BEQ 7\$;:NON-ZERO BUFFER ADDRESS ?
MOV RM.REG+RMB4,-(SP) ;:BR IF NO BUFFER ADDRESS
SUB #2,(SP) ;:BUFFER ADDRESS
MOV @2(SP)+,\$GDDAT ;:DECREMENT THE ADDRESS
INC \$ERFLG ;:GET THE BUFFER WORD WHICH DIDN'T COMPARE
BEQ 7\$;:SET THE ERROR FLAG
MOV \$TSTNM,\$DISPLAY ;:DON'T LET THE FLAG GO TO ZERO
BIT #BIT10,\$SWR ;:DISPLAY TEST NUMBER AND ERROR FLAG
BEQ 1\$;:BELL ON ERROR?
TYPE \$BELL ;:NO - SKIP
INC \$ERTTL ;:RING BELL
MOV (SP), \$ERRPC ;:COUNT THE NUMBER OF ERRORS
SUB #2,\$ERRPC ;:GET ADDRESS OF ERROR INSTRUCTION
MOV \$ERRPC,\$ITEMB ;:STRIP AND SAVE THE ERROR ITEM CODE
BIT #BIT13,\$SWR ;:SKIP TYPEOUT IF SET
BNE 20\$;:SKIP TYPEOUTS
JSR PC,TYPERR ;:GO TO USER ERROR ROUTINE
TYPE \$CRLF
20\$: CMPB #APTENV,\$ENV ;:RUNNING IN APT MODE
BNE 2\$;:NO SKIP APT ERROR REPORT
MOV \$ITEMB,21\$;:SET ITEM NUMBER AS ERROR NUMBER
JSR PC,\$ATY4 ;:REPORT FATAL ERROR TO APT
21\$: .BYTE 0
22\$: .BYTE 0
2\$: BR 22\$;:APT ERROR LOOP
TST \$SWR ;:HALT ON ERROR
BPL 3\$;:SKIP IF CONTINUE

3463 020066
3464 020066 104407
3465 020070 010137 001372
3466 020074 010337 001374
3467 020100 013737 005622 001366
3468 020106 113737 005621 001370
3469 020114 005737 005544
3470 020120 001406
3471 020122 013746 005544
3472 020126 162716 000002
3473 020132 013637 001140
3474 020136 105237 001117
3475 020142 001775
3476 020144 013777 001116 161004
3477 020152 032777 002000 160774
3478 020160 001402
3479 020162 104401 001176
3480 020166 005237 001126
3481 020172 011637 001132
3482 020176 162737 000002 001132
3483 020204 117737 160722 001130
3484 020212 032777 020000 160734
3485 020220 001004
3486 020222 004737 020322
3487 020226 104401 001203
3488 020232
3489 020232 122737 000001 001226
3490 020240 001007
3491 020242 113737 001130 020254
3492 020250 004737 021052
3493 020254 000
3494 020255 000
3495 020256 000777
3496 020260 005777 160670
3497 020264 100002

```

3498 020266 000000          HALT                                ;; HALT ON ERROR!
3499 020270 104407          CKSWR                               ;; TEST FOR CHANGE IN SOFT-SWR
3500 020272 032777 001000 160654 3$: BIT    #BIT09,2SWR       ;; LOOP ON ERROR SWITCH SET?
3501 020300 001402          BEQ     4$                          ;; BR IF NO
3502 020302 013716 001124   MOV     $LPERR,(SP)                ;; FUDGE RETURN FOR LOOPING
3503 020306 005737 001174   4$: TST     $ESCAPE                ;; CHECK FOR AN ESCAPE ADDRESS
3504 020312 001402          BEQ     5$                          ;; BR IF NONE
3505 020314 013716 001174   MOV     $ESCAPE,(SP)                ;; FUDGE RETURN ADDRESS FOR ESCAPE
3506 020320          5$:                                         ;;
3507 020320 000002          RTI                                ;; RETURN
3508
3509
3510                                     ; THIS ROUTINE USES THE "ITEM CONTROL BYTE" ($ITEMB) TO DETERMINE
3511                                     ; WHICH ERROR IS TO BE REPORTED. IT THEN OBTAINS, FROM THE "ERROR
3512                                     ; TABLE" ($ERRTB), AND REPORTS THE APPROPRIATE INFORMATION
3513                                     ; CONCERNING THE ERROR.
3514
3515 020322 104412          TYPERR: SAVREG                        ; SAVE R0-R5
3516 020324 005000          CLR     R0                          ; CLEAR R0 FOR ERROR NUMBER
3517 020326 113700 001130   MOVB    $ITEMB,R0                  ; ERROR NUMBER
3518 020332 005300          DEC     R0                          ; FORM INDEX FOR ERROR TABLE
3519 020334 006300          ASL     R0
3520 020336 006300          ASL     R0
3521 020340 006300          ASL     R0
3522 020342 062700 006152   1$: ADD     # $ERRTB,R0              ; FORM ADDRESS
3523 020346 012037 020362   MOV     (R0)+,2$                  ; GET ERROR MESSAGE (EM) POINTER
3524 020352 001404          BEQ     3$                          ; BRANCH IF THERE ISN'T ONE
3525 020354 104401 001203   TYPE    , $CRLF                    ; "CARRIAGE RETURN - LINE FEED"
3526 020360 104401          TYPE
3527 020362 000000          2$: .WORD 0                          ; "EM" POINTER GOES HERE
3528 020364 012037 020400   3$: MOV     (R0)+,4$                  ; PICK UP DATA HEADER (DH) POINTER
3529 020370 001404          BEQ     5$                          ; BRANCH IF NONE
3530 020372 104401 001203   TYPE    , $CRLF                    ; CARRIAGE RETURN-LINE FEED
3531 020376 104401          TYPE
3532 020400 000000          4$: .WORD 0                          ; "DH" POINTER GOES HERE
3533 020402 012001 5$: MOV     (R0)+,R1                          ; PICKUP DATA TABLE (DT) POINTER
3534 020404 001460          BEQ     20$                          ; BRANCH IF NONE
3535 020406 005005          CLR     R5                          ; SET INDENT SWITCH
3536 020410 012000          MOV     (R0)+,R0                      ; DATA FORMAT (DF) POINTER
3537 020412 012002          MOV     (R0)+,R2                      ; NUMBER OF DH'S TO TYPE
3538 020414 001451          BEQ     17$                          ; BRANCH IF DH NUMBER IS 0
3539 020416 005105          COM     R5                          ; NO INDENT
3540 020420 104401 001203   TYPE    , $CRLF                    ; CARRIAGE RETURN-LINE FEED
3541 020424 112003          10$: MOVB   (R0)+,R3                 ; NUMBER OF DATA WORDS TO TYPE
3542 020426 112004          MOVB   (R0)+,R4                     ; AND HOW TO TYPE THEM
3543 020430 006004          11$: ROR     R4                      ; OCTAL OR DECIMAL?
3544 020432 103403          BCS     12$                          ; DECIMAL--BRANCH
3545 020434 013146          MOV     2(R1)+,-(SP)                ; SAVE 2(R1)+ FOR TYPEOUT
3546 020436 104402          TYPOC                                ; GO TYPE--OCTAL ASCII(ALL DIGITS)
3547 020440 000402          BR      13$
3548 020442          12$:
3549 020442 013146          MOV     2(R1)+,-(SP)                ; SAVE 2(R1)+ FOR TYPEOUT
3550 020444 104405          TYPDS                                ; GO TYPE--DECIMAL ASCII WITH SIGN
3551 020446 005303          13$: DEC     R3                      ; MORE NUMBERS TO TYPE?
3552 020450 001403          BEQ     14$                          ; NO--BRANCH
3553 020452 104401 032503   TYPE    ,LINSF                     ; YES--TYPE SEPERATORS

```

3554 020456 000764
3555 020460 005302
3556 020462 003431
3557 020464 104401 001203
3558 020470 005760 000002
3559 020474 001404
3560 020476 005105
3561 020500 001002
3562 020512 104401 032503
3563 020516 012037 020514
3564 020518 104401
3565 020514 000000
3566 020516 005710
3567 020520 001003
3568 020522 062700 000004
3569 020526 000754
3570 020530 104401 001203
3571 020534 005705
3572 020536 001332
3573 020540 104401 032503
3574 020544 000727
3575 020546 104413
3576 020550 000207

14\$: BR 11\$
DEC R2
BLE 20\$
TYPE \$CRLF
TST 2(RJ)
BEQ 15\$
COM R5
BNE 15\$
TYPE LINS
MOV (R0)+,16\$
15\$: TYPE
16\$: .WORD 0
TST (R0)
BNE 21\$
ADD #4,R0
BR 14\$
21\$: TYPE \$CRLF
TST R5
BNE 10\$
17\$: TYPE LINS
BR 10\$
20\$: RESREG
RTS PC

LOOP
MORE DH'S?
NO--BRANCH
YES--START A NEW LINE
ONLY A 'DH' IN THIS REQUEST?
BR IF YES - BYPASS THE INDENT
INDENT?
NO--BRANCH
YES--TYPE SPACES
GET NEXT DH
AND TYPE IT
DH POINTER GOES HERE
TYPE A 'DT' ?
BR IF A 'DT'
INCREMENT THE 'DF' POINTER
SEE IF END OF 'DF' BLOCK
CARRIAGE RETURN-LINE FEED
INDENT?
NO--BRANCH
YES--TYPE SPACES
LOOP
RESTORE R0-R5
RETURN

.SBTTL TYPE ROUTINE

ROUTINE TO TYPE ASCIZ MESSAGE. MESSAGE MUST TERMINATE WITH A 0 BYTE.
THE ROUTINE WILL INSERT A NUMBER OF NULL CHARACTERS AFTER A LINE FEED.
NOTE1: \$NULL CONTAINS THE CHARACTER TO BE USED AS THE FILLER CHARACTER.
NOTE2: \$FILLS CONTAINS THE NUMBER OF FILLER CHARACTERS REQUIRED.
NOTE3: \$FILLC CONTAINS THE CHARACTER TO FILL AFTER.
*
CALL:
*1) USING A TRAP INSTRUCTION
* TYPE ,MESADR ; MESADR IS FIRST ADDRESS OF AN ASCIZ STRING
*OR
* TYPE
* MESADR
*

3595 020552 105737 001173
3596 020556 100002
3597 020560 000000
3598 020562 000430
3599 020564 010046
3600 020566 017600 000002
3601 020572 122737 000001 001226
3602 020600 001011
3603 020602 132737 000100 001227
3604 020610 001405
3605 020612 010037 020622
3606 020616 004737 021042
3607 020622 000000
3608 020624 132737 000040 001227
3609 020632 001003

STYPE: TSTB \$TPFLG
BPL 1\$
HALT
BR 3\$
1\$: MOV R0, -(SP)
MOV 22(SP), R0
CMPB #APTENV, \$ENV
BNE 62\$
BITB #APTSPool, \$ENV
BEQ 62\$
MOV R0, 61\$
JSR PC, \$ATY3
61\$: .WORD 0
62\$: BITB #APTCsup, \$ENV
BNE 60\$

IS THERE A TERMINAL?
BR IF YES
HALT HERE IF NO TERMINAL
LEAVE
SAVE R0
GET ADDRESS OF ASCIZ STRING
RUNNING IN APT MODE
NO GO CHECK FOR APT CONSOLE
SPOOL MESSAGE TO APT
NO GO CHECK FOR CONSOLE
SETUP MESSAGE ADDRESS FOR APT
SPOOL MESSAGE TO APT
MESSAGE ADDRESS
APT CONSOLE SUPPRESSED
YES, SKIP TYPE OUT

3610	020634	112046		25:	MOVB	(RO)+, -(SP)	:: PUSH CHARACTER TO BE TYPED ONTO STACK
3611	020636	001005			BNE	45	:: BR IF IT ISN'T THE TERMINATOR
3612	020640	005726			TST	(SP)+	:: IF TERMINATOR POP IT OFF THE STACK
3613	020642	012600		605:	MOV	(SP)+, RO	:: RESTORE RO
3614	020644	02716	000002	35:	ADD	#2, (SP)	:: ADJUST RETURN PC
3615	020650	02002			RTI		:: RETURN
3616	020652	122716	000011	45:	CMPB	#HT, (SP)	:: BRANCH IF <HT>
3617	020656	001430			BEQ	85	
3618	020658	122716	000200		CMPB	#CRLF, (SP)	:: BRANCH IF NOT <CRLF>
3619	020664	021006			BNE	55	
3620	020656	005726			TST	(SP)+	:: POP <CR><LF> EQUIV
3621	020670	104401			TYPE		:: TYPE A CR AND LF
3622	020672	001203			\$CRLF		
3623	020674	105037	021030		CLRB	\$CHARCNT	:: CLEAR CHARACTER COUNT
3624	020700	000755			BR	25	:: GET NEXT CHARACTER
3625	020702	004737	020764	55:	JSR	PC, \$TYPEC	:: GO TYPE THIS CHARACTER
3626	020706	123726	001172	65:	CMPB	\$FILLC, (SP)+	:: IS IT TIME FOR FILLER CHARS.?
3627	020712	001350			BNE	25	:: IF NO GO GET NEXT CHAR.
3628	020714	013746	001170		MOV	\$NULL, -(SP)	:: GET # OF FILLER CHARS. NEEDED
3629							:: AND THE NULL CHAR.
3630	020720	105366	000001	75:	DECB	1(SP)	:: DOES A NULL NEED TO BE TYPED?
3631	020724	002770			BLT	65	:: BR IF NO--GO POP THE NULL OFF OF STACK
3632	020726	004737	020764		JSR	PC, \$TYPEC	:: GO TYPE A NULL
3633	020732	105337	021030		DECB	\$CHARCNT	:: DO NOT COUNT AS A COUNT
3634	020736	000770			BR	75	:: LOOP
3635							
3636							
3637							
3638	020740	112716	000040		85:	MOVB	#', (SP)
3639	020744	004737	020764		95:	JSR	PC, \$TYPEC
3640	020750	132737	000007 021030			BITB	#7, \$CHARCNT
3641	020756	001372			BNE	95	:: BRANCH IF NOT AT
3642	020760	005726			TST	(SP)+	:: TAB STOP
3643	020762	000724			BR	25	:: POP SPACE OFF STACK
3644	020764	105777	160174	\$TYPEC:	TSTB	\$STPS	:: GET NEXT CHARACTER
3645	020770	100375			BPL	\$TYPEC	:: WAIT UNTIL PRINTER IS READY
3646	020772	116677	000002 160166		MOVB	2(SP), \$STPB	:: LOAD CHAR TO BE TYPED INTO DATA REG.
3647	021000	122766	000015 000002		CMPB	#CR, 2(SP)	:: IS CHARACTER A CARRIAGE RETURN?
3648	021006	001003			BNE	15	:: BRANCH IF NO
3649	021010	105037	021030		CLRB	\$CHARCNT	:: YES--CLEAR CHARACTER COUNT
3650	021014	000406			BR	\$TYPEX	:: EXIT
3651	021016	122766	000012 000002	15:	CMPB	#LF, 2(SP)	:: IS CHARACTER A LINE FEED?
3652	021024	001402			BEQ	\$TYPEX	:: BRANCH IF YES
3653	021026	105227			INCB	(PC)+	:: COUNT THE CHARACTER
3654	021030	000000		\$CHARCNT:	.WORD	0	:: CHARACTER COUNT STORAGE
3655	021032	000207		\$TYPEX:	RTS	PC	
3656							
3657							
3658							
3659							
3660							
3661							
3662	021034	112737	000001 021300	\$ATY1:	MOVB	#1, \$FFLG	:: TO REPORT FATAL ERROR
3663	021042	112737	000001 021276	\$ATY3:	MOVB	#1, \$MFLG	:: TO TYPE A MESSAGE
3664	021050	000403			BR	\$ATYC	
3665	021052	112737	000001 021300	\$ATY4:	MOVB	#1, \$FFLG	:: TO ONLY REPORT FATAL ERROR

.SBTTL APT COMMUNICATIONS ROUTINE

```

SATYC:  MOV      RO,-(SP)          ;; PUSH RO ON STACK
        MOV      R1,-(SP)          ;; PUSH R1 ON STACK
        TSTB     $MFLG             ;; SHOULD TYPE A MESSAGE?
        BEQ      5$                ;; IF NOT: BR
        CMPB     #APTENV,$ENV       ;; OPERATING UNDER APT?
        BNE      3$                ;; IF NOT: BR
        BITB     #APTSPool,$ENVm    ;; SHOULD SPOOL MESSAGES?
        BEQ      3$                ;; IF NOT: BR
        MOV      @4(SP),RO          ;; GET MESSAGE ADDR.
        ADD      #2,4(SP)           ;; BUMP RETURN ADDR.
1$:      TST      $MSGTYPE           ;; SEE IF DONE W/ LAST XMISSION?
        BNE      1$                ;; IF NOT: WAIT
        MOV      RO,$MSGAD          ;; PUT ADDR IN MAILBOX
2$:      TSTB     (RO)+             ;; FIND END OF MESSAGE
        BNE      2$
        SUB      $MSGAD,RO          ;; SUB START OF MESSAGE
        ASR      RO                ;; GET MESSAGE LNTH IN WORDS
        MOV      RO,$MSGGLT         ;; PUT LENGTH IN MAILBOX
        MOV      #4,$MSGTYPE        ;; TELL APT TO TAKE MSG.
        BR       5$
3$:      MOV      @4(SP),4$          ;; PUT MSG ADDR IN JSR LINKAGE
        ADD      #2,4(SP)           ;; BUMP RETURN ADDRESS
        MOV      177776,-(SP)       ;; PUSH 177776 ON STACK
        JSR      PC,$TYPE           ;; CALL TYPE MACRO
4$:      .WORD    0
5$:
10$:     TSTB     $FFLG             ;; SHOULD REPORT FATAL ERROR?
        BEQ      12$               ;; IF NOT: BR
        TST      $ENV              ;; RUNNING UNDER APT?
        BEQ      12$               ;; IF NOT: BR
11$:     TST      $MSGTYPE           ;; FINISHED LAST MESSAGE?
        BNE      11$               ;; IF NOT: WAIT
        MOV      @4(SP),$FATAL      ;; GET ERROR #
        ADD      #2,4(SP)           ;; BUMP RETURN ADDR.
        INC      $MSGTYPE           ;; TELL APT TO TAKE ERROR
12$:     CLRB     $FFLG             ;; CLEAR FATAL FLAG
        CLRB     $LFLG             ;; CLEAR LOG FLAG
        CLRB     $MFLG             ;; CLEAR MESSAGE FLAG
        MOV      (SP)+,R1           ;; POP STACK INTO R1
        MOV      (SP)+,RO          ;; POP STACK INTO RO
        RTS      PC                ;; RETURN
$MFLG:   .BYTE    0                ;; MESSG. FLAG
$LFLG:   .BYTE    0                ;; LOG FLAG
$FFLG:   .BYTE    0                ;; FATAL FLAG
        .EVEN
APTSIZE=200
APTENV=001
APTSPool=100
APTCsup=040

.SBTTL  BINARY TO OCTAL (ASCII) AND TYPE

;*****
;THIS ROUTINE IS USED TO CHANGE A 16-BIT BINARY NUMBER TO A 6-DIGIT
;OCTAL (ASCII) NUMBER AND TYPE IT.

```

F06

MAINDEC-11-DZMA, RM03 FORMATTER
DZMA8.P11 21-JUL-77 15:32

MACY11 30(1046) 21-JUL-77 16:06 PAGE 70
BINARY TO OCTAL (ASCII) AND TYPE

SEQ 0069

```

3722  ;*STYPOS---ENTER HERE TO SETUP SUPPRESS ZEROS AND NUMBER OF DIGITS TO TYPE
3723  ;*CALL:
3724  ;*      MOV      NUM,-(SP)      ;;NUMBER TO BE TYPED
3725  ;*      TYPOS      ;;CALL FOR TYPEOUT
3726  ;*      .BYTE  N      ;;N=1 TO 6 FOR NUMBER OF DIGITS TO TYPE
3727  ;*      .BYTE  M      ;;M=1 OR 0
3728  ;*                      ;;1=TYPE LEADING ZEROS
3729  ;*                      ;;0=SUPPRESS LEADING ZEROS
3730  ;*
3731  ;*STYPON---ENTER HERE TO TYPE OUT WITH THE SAME PARAMETERS AS THE LAST
3732  ;*STYPOS OR STYPOC
3733  ;*CALL:
3734  ;*      MOV      NUM,-(SP)      ;;NUMBER TO BE TYPED
3735  ;*      TYPON      ;;CALL FOR TYPEOUT
3736  ;*
3737  ;*STYPOC---ENTER HERE FOR TYPEOUT OF A 16 BIT NUMBER
3738  ;*CALL:
3739  ;*      MOV      NUM,-(SP)      ;;NUMBER TO BE TYPED
3740  ;*      TYPOC      ;;CALL FOR TYPEOUT
3741  ;*
3742  021302 017646 000000 021525 STYPOS: MOV      2(SP),-(SP)      ;;PICKUP THE MODE
3743  021306 116637 000001 021525 MOV      1(SP),SOFILL      ;;LOAD ZERO FILL SWITCH
3744  021314 112637 021527 MOV      (SP)+,SOMODE+1      ;;NUMBER OF DIGITS TO TYPE
3745  021320 062716 000002 ADD      #2,(SP)      ;;ADJUST RETURN ADDRESS
3746  021324 000406 BR      STYPON
3747  021326 112737 000001 021525 STYPOC: MOV      #1,SOFILL      ;;SET THE ZERO FILL SWITCH
3748  021334 112737 000006 021527 MOV      #6,SOMODE+1      ;;SET FOR SIX(6) DIGITS
3749  021342 112737 000005 021524 STYPON: MOV      #5,SOCNT      ;;SET THE ITERATION COUNT
3750  021350 010346 MOV      R3,-(SP)      ;;SAVE R3
3751  021352 010446 MOV      R4,-(SP)      ;;SAVE R4
3752  021354 010546 MOV      R5,-(SP)      ;;SAVE R5
3753  021356 113704 021527 MOV      SOMODE+1,R4      ;;GET THE NUMBER OF DIGITS TO TYPE
3754  021362 005404 NEG      R4
3755  021364 062704 000006 ADD      #6,R4      ;;SUBTRACT IT FOR MAX. ALLOWED
3756  021370 110437 021526 MOV      R4,SOMODE      ;;SAVE IT FOR USE
3757  021374 113704 021525 MOV      SOFILL,R4      ;;GET THE ZERO FILL SWITCH
3758  021400 016605 000012 MOV      12(SP),R5      ;;PICKUP THE INPUT NUMBER
3759  021404 005003 CLR      R3      ;;CLEAR THE OUTPUT WORD
3760  021406 006105 1$: ROL      R5      ;;ROTATE MSB INTO "C"
3761  021410 000404 BR      3$      ;;GO DO MSB
3762  021412 006105 2$: ROL      R5      ;;FORM THIS DIGIT
3763  021414 006105 ROL      R5
3764  021416 006105 ROL      R5
3765  021420 010503 MOV      R5,R3
3766  021422 006103 3$: ROL      R3      ;;GET LSB OF THIS DIGIT
3767  021424 105337 021526 DECB      SOMODE      ;;TYPE THIS DIGIT?
3768  021430 100016 BPL      7$      ;;BR IF NO
3769  021432 042703 177770 BIC      #177770,R3      ;;GET RID OF JUNK
3770  021436 001002 BNE      4$      ;;TEST FOR 0
3771  021440 005704 TST      R4      ;;SUPPRESS THIS 0?
3772  021442 001403 BEQ      5$      ;;BR IF YES
3773  021444 005204 4$: INC      R4      ;;DON'T SUPPRESS ANYMORE 0'S
3774  021446 052703 000060 BIS      #'0,R3      ;;MAKE THIS DIGIT ASCII
3775  021452 052703 000040 5$: BIS      #' ,R3      ;;MAKE ASCII IF NOT ALREADY
3776  021456 110337 021522 MOV      R3,8$      ;;SAVE FOR TYPING
3777  021462 104401 021522 TYPE      ,8$      ;;GO TYPE THIS DIGIT

```

G06

MAINDEC-11-DZMA, RM03 FORMATTER
DZMA8.P11 21-JUL-77 15:32

MACY11 30(1046) 21-JUL-77 16:06 PAGE 71
BINARY TO OCTAL (ASCII) AND TYPE

SEQ 0070

3778	021466	105337	021524	7\$:	DECB	\$OCNT	::COUNT BY 1
3779	021472	003347			BGT	2\$	::BR IF MORE TO DO
3780	021474	002402			BLT	6\$	::BR IF DONE
3781	021476	005204			INC	R4	::INSURE LAST DIGIT ISN'T A BLANK
3782	021500	000744			BR	2\$	::GO DO THE LAST DIGIT
3783	021502	012605		6\$:	MOV	(SP)+,R5	::RESTORE R5
3784	021504	012604			MOV	(SP)+,R4	::RESTORE R4
3785	021506	012603			MOV	(SP)+,R3	::RESTORE R3
3786	021510	016666	000002 000004		MOV	2(SP),4(SP)	::SET THE STACK FOR RETURNING
3787	021516	012616			MOV	(SP)+,(SP)	
3788	021520	000002			RTI		::RETURN
3789	021522	000		8\$:	.BYTE	0	::STORAGE FOR ASCII DIGIT
3790	021523	000			.BYTE	0	::TERMINATOR FOR TYPE ROUTINE
3791	021524	000		\$OCNT:	.BYTE	0	::OCTAL DIGIT COUNTER
3792	021525	000		\$OFILL:	.BYTE	0	::ZERO FILL SWITCH
3793	021526	000000		\$OMODE:	.WORD	0	::NUMBER OF DIGITS TO TYPE
3794							
3795							
3796							
3797							
3798							
3799							
3800							
3801							
3802							
3803							
3804							
3805							
3806							
3807	021530						
3808	021530	010046					
3809	021532	010146					
3810	021534	010246					
3811	021536	010346					
3812	021540	010546					
3813	021542	012746	020200				
3814	021546	016605	000020				
3815	021552	100004					
3816	021554	005405					
3817	021556	112766	000055 000001				
3818	021564	005000					
3819	021566	012703	021744	1\$:	CLR	R0	::ZERO THE CONSTANTS INDEX
3820	021572	112723	000040		MOV	\$OBLK,R3	::SETUP THE OUTPUT POINTER
3821	021576	005002			MOVB	#',(R3)+	::SET THE FIRST CHARACTER TO A BLANK
3822	021600	016001	021734	2\$:	CLR	R2	::CLEAR THE BCD NUMBER
3823	021604	160105			MOV	\$DTBL(R0),R1	::GET THE CONSTANT
3824	021606	002402		3\$:	SUB	R1,R5	::FORM THIS BCD DIGIT
3825	021610	005202			BLT	4\$	::BR IF DONE
3826	021612	000774			INC	R2	::INCREASE THE BCD DIGIT BY 1
3827	021614	060105		4\$:	BR	3\$	
3828	021616	005702			ADD	R1,R5	::ADD BACK THE CONSTANT
3829	021620	001002			TST	R2	::CHECK IF BCD DIGIT=0
3830	021622	105716			BNE	5\$	::FALL THROUGH IF 0
3831	021624	100407			TSTB	(SP)	::STILL DOING LEADING 0'S?
3832	021626	106316			BMI	7\$	::BR IF YES
3833	021630	103003		5\$:	ASLB	(SP)	::MSD?
					BCC	6\$	::BR IF NO

.SBTTL CONVERT BINARY TO DECIMAL AND TYPE ROUTINE

::*****
 ::THIS ROUTINE IS USED TO CHANGE A 16-BIT BINARY NUMBER TO A 5-DIGIT
 ::SIGNED DECIMAL (ASCII) NUMBER AND TYPE IT. DEPENDING ON WHETHER THE
 ::NUMBER IS POSITIVE OR NEGATIVE A SPACE OR A MINUS SIGN WILL BE TYPED
 ::BEFORE THE FIRST DIGIT OF THE NUMBER. LEADING ZEROS WILL ALWAYS BE
 ::REPLACED WITH SPACES.
 ::CALL:
 ::* MOV NUM,-(SP) ::PUT THE BINARY NUMBER ON THE STACK
 ::* TYPDS ::GO TO THE ROUTINE

\$TYPDS:
 MOV R0,-(SP) ::PUSH R0 ON STACK
 MOV R1,-(SP) ::PUSH R1 ON STACK
 MOV R2,-(SP) ::PUSH R2 ON STACK
 MOV R3,-(SP) ::PUSH R3 ON STACK
 MOV R5,-(SP) ::PUSH R5 ON STACK
 MOV #20200,-(SP) ::SET BLANK SWITCH AND SIGN
 MOV 20(SP),R5 ::GET THE INPUT NUMBER
 BPL 1\$::BR IF INPUT IS POS.
 NEG R5 ::MAKE THE BINARY NUMBER POS.
 MOVB #'-,1(SP) ::MAKE THE ASCII NUMBER NEG.
 1\$:
 CLR R0 ::ZERO THE CONSTANTS INDEX
 MOV \$OBLK,R3 ::SETUP THE OUTPUT POINTER
 MOVB #'',(R3)+ ::SET THE FIRST CHARACTER TO A BLANK
 2\$:
 CLR R2 ::CLEAR THE BCD NUMBER
 MOV \$DTBL(R0),R1 ::GET THE CONSTANT
 3\$:
 SUB R1,R5 ::FORM THIS BCD DIGIT
 BLT 4\$::BR IF DONE
 INC R2 ::INCREASE THE BCD DIGIT BY 1
 BR 3\$
 4\$:
 ADD R1,R5 ::ADD BACK THE CONSTANT
 TST R2 ::CHECK IF BCD DIGIT=0
 BNE 5\$::FALL THROUGH IF 0
 TSTB (SP) ::STILL DOING LEADING 0'S?
 BMI 7\$::BR IF YES
 5\$:
 ASLB (SP) ::MSD?
 BCC 6\$::BR IF NO

H06

MAINDEC-11-DZMA, RM03 FORMATTER
DZMA8.P11 21-JUL-77 15:32

MACY11 30(1046) 21-JUL-77 16:06 PAGE 72
CONVERT BINARY TO DECIMAL AND TYPE ROUTINE

SEQ 0071

3834	021632	116663	000001	177777		MOV B	1(SP),-1(R3)	;; YES--SET THE SIGN
3835	021640	052702	000060		6S:	BIS	#'0,R2	;; MAKE THE BCD DIGIT ASCII
3836	021644	052702	000040		7S:	BIS	#' ,R2	;; MAKE IT A SPACE IF NOT ALREADY A DIGIT
3837	021650	110223				MOV B	R2,(R3)+	;; PUT THIS CHARACTER IN THE OUTPUT BUFFER
3838	021652	005720				TST	(R0)+	;; JUST INCREMENTING
3839	021654	020027	000010			CMP	R0,#10	;; CHECK THE TABLE INDEX
3840	021660	002746				BLT	2S	;; GO DO THE NEXT DIGIT
3841	021662	003002				BGT	8S	;; GO TO EXIT
3842	021664	010502				MOV	R5,R2	;; GET THE LSD
3843	021666	000764				BR	6S	;; GO CHANGE TO ASCII
3844	021670	105726			8S:	TST B	(SP)+	;; WAS THE LSD THE FIRST NON-ZERO?
3845	021672	100003				BPL	9S	;; BR IF NO
3846	021674	116663	177777	177776		MOV B	-1(SP),-2(R3)	;; YES--SET THE SIGN FOR TYPING
3847	021702	105013			9S:	CLRB	(R3)	;; SET THE TERMINATOR
3848	021704	012605				MOV	(SP)+,R5	;; POP STACK INTO R5
3849	021706	012603				MOV	(SP)+,R3	;; POP STACK INTO R3
3850	021710	012602				MOV	(SP)+,R2	;; POP STACK INTO R2
3851	021712	012601				MOV	(SP)+,R1	;; POP STACK INTO R1
3852	021714	012600				MOV	(SP)+,R0	;; POP STACK INTO R0
3853	021716	104401	021744			TYPE	\$DBLK	;; NOW TYPE THE NUMBER
3854	021722	016666	000002	000004		MOV	2(SP),4(SP)	;; ADJUST THE STACK
3855	021730	012616				MOV	(SP)+,(SP)	
3856	021732	000002				RTI		;; RETURN TO USER
3857	021734	023420			\$DTBL:	10000.		
3858	021736	001750				1000.		
3859	021740	000144				100.		
3860	021742	000012				10.		
3861	021744	000004			\$DBLK:	.BLKW 4		
3862								
3863					.SBTTL	TTY INPUT ROUTINE		
3864								
3865					;;*****			
3866					.ENABL	LSB		
3867	021754	000000			\$TKCNT:	.WORD 0		;; NUMBER OF ITEMS IN QUEUE
3868	021756	000000			\$TKQIN:	.WORD 0		;; INPUT POINTER
3869	021760	000000			\$TKQOUT:	.WORD 0		;; OUTPUT POINTER
3870	021762	000007			\$TKQSRT:	.BLKB 7		;; TTY KEYBOARD QUEUE
3871		021771			\$TKQEND=.			
3872		021772			.EVEN			
3873								
3874					;;*TK INITIALIZE ROUTINE			
3875					;;*THIS ROUTINE WILL INITIALIZE THE TTY KEYBOARD INPUT QUEUE			
3876					;;*SETUP THE INTERRUPT VECTOR AND TURN ON THE KEYBOARD INTERRUPT			
3877					;;			
3878					;;*CALL:			
3879					;;*	JSR	PC,\$TKINT	
3880					;;*	RETURN		
3881					;;			
3882	021772	005037	021754		\$TKINT:	CLR	\$TKCNT	;; CLEAR COUNT OF ITEMS IN QUEUE
3883	021776	012737	021762	021756		MOV	\$TKQSRT,\$TKQIN	;; MOVE THE STARTING ADDRESS OF THE
3884	022004	013737	021756	021760		MOV	\$TKQIN,\$TKQOUT	;; QUEUE INTO THE INPUT & OUTPUT POINTERS.
3885	022012	012737	022042	000060		MOV	\$TKSRV,\$TKVEC	;; INITIALIZE THE KEYBOARD VECTOR
3886	022020	012737	000200	000062		MOV	#200,\$TKVEC+2	;; "BR" LEVEL 4
3887	022026	005777	157130			TST	\$TKB	;; CLEAR DONE FLAG
3888	022032	012777	000100	157120		MOV	#100,\$TKS	;; ENABLE TTY KEYBOARD INTERRUPT
3889	022040	000207				RTS	PC	;; RETURN TO CALLER

```

3890
3891
3892
3893
3894
3895
3896
3897
3898 022042 117746 157114
3899 022046 042716 177600
3900 022052 021627 000003
3901 022056 001007
3902 022060 104401 023163
3903 022064 004737 021772
3904 022070 005726
3905 022072 000177 157300
3906 022076 021627 000007
3907 022102 001004
3908 022104 022737 000176 001154
3909 022112 001500
3910
3911 022114
3912 022114 022737 000007 021754
3913 022122 001004
3914 022124 104401 001176
3915 022130 005726
3916 022132 000451
3917 022134 021627 000023
3918 022140 001021
3919 022142 005077 157012
3920 022146 005726
3921 022150 105777 157004
3922 022154 100375
3923 022156 117746 157000
3924 022162 042716 177600
3925 022166 022627 000021
3926 022172 001366
3927 022174 012777 000100 156756
3928 022202 000002
3929 022204 005237 021754
3930 022210 021627 000140
3931 022214 002405
3932 022216 021627 000175
3933 022222 003002
3934 022224 042716 000040
3935 022230 112677 177522
3936 022234 005237 021756
3937 022240 023727 021756 021771
3938 022246 001003
3939 022250 012737 021762 021756
3940 022256 000002
3941
3942
3943
3944
3945

; *TK SERVICE ROUTINE
; *THIS ROUTINE WILL SERVICE THE TTY KEYBOARD INTERRUPT
; *BY READING THE CHARACTER FROM THE INPUT BUFFER AND PUTTING
; *IT IN THE QUEUE.
; *IF THE CHARACTER IS A "CONTROL-C" (1C) STKINT IS CALLED AND
; *UPON RETURN EXIT IS MADE TO THE "CONTROL-C" RESTART ADDRESS (2CNTLC)

$TKSRV: MOVB 2$TKB, -(SP) ; PICKUP THE CHARACTER
        BIC 1C177, (SP) ; STRIP THE JUNK
        CMP (SP), #3 ; IS IT A CONTROL C?
        BNE 1$ ; BRANCH IF NO
        TYPE 2$CNTLC ; TYPE A CONTROL-C (1C)
        JSR PC, STKINT ; INIT THE KEYBOARD
        TST (SP)+ ; CLEAN UP STACK
        JMP 2$CNTLC ; CONTROL C RESTART
1$: CMP (SP), #7 ; IS IT A CONTROL G?
    BNE 2$ ; BRANCH IF NO
    CMP #SWREG, SWR ; IS SOFT-SWR SELECTED?
    BEQ 6$ ; GO TO SWR CHANGE
2$: CMP #7, $TKCNT ; IS THE QUEUE FULL?
    BNE 3$ ; BRANCH IF NO
    TYPE $BELL ; RING THE TTY BELL
    TST (SP)+ ; CLEAN CHARACTER OFF OF STACK
    BR 5$ ; EXIT
3$: CMP (SP), #23 ; IS IT A CONTROL-S?
    BNE 32$ ; BRANCH IF NO
    CLR 2$TKS ; DISABLE TTY KEYBOARD INTERRUPTS
    TST (SP)+ ; CLEAN CHAR OFF STACK
31$: TSTB 2$TKS ; WAIT FOR A CHAR
    BPL 31$ ; LOOP UNTIL ITS THERE
    MOVB 2$TKB, -(SP) ; GET THE CHARACTER
    BIC 1C177, (SP) ; MAKE IT 7-BIT ASCII
    CMP (SP)+, #21 ; IS IT A CONTROL-Q?
    BNE 31$ ; BRANCH IF NO
    MOV #100, 2$TKS ; REENABLE TTY KEYBOARD INTERRUPTS
    RTI ; RETURN
32$: INC $TKCNT ; COUNT THIS CHARACTER
    CMP (SP), #140 ; IS IT UPPER CASE?
    BLT 4$ ; BRANCH IF YES
    CMP (SP), #175 ; IS IT A SPECIAL CHAR?
    BGT 4$ ; BRANCH IF YES
    BIC #40, (SP) ; MAKE IT UPPER CASE
    MOVB (SP)+, 2$TKQIN ; AND PUT IT IN QUEUE
    INC $TKQIN ; UPDATE THE POINTER
    CMP $TKQIN, $STKQEND ; GO OFF THE END?
    BNE 5$ ; BRANCH IF NO
    MOV #STKQSR, $TKQIN ; RESET THE POINTER
5$: RTI ; RETURN

; *****
; *SOFTWARE SWITCH REGISTER CHANGE ROUTINE.
; *ROUTINE IS ENTERED FROM THE TRAP HANDLER, AND WILL
; *SERVICE THE TEST FOR CHANGE IN SOFTWARE SWITCH REGISTER TRAP

```

```

3946      : *CALL WHEN OPERATING IN TTY INTERRUPT MODE.
3947 022260 022737 000176 001154 $CKSWR: CMP      #SWREG,SWR      ;; IS THE SOFT-SWR SELECTED
3948 022266 001124      BNE      15$      ;; EXIT IF NOT
3949 022270 105777 156664      TSTB     2$TKS      ;; IS A CHAR WAITING?
3950 022274 100121      BPL      15$      ;; IF NOT, EXIT
3951 022276 117746 156660      MOVB     2$TKB,-(SP)      ;; YES
3952 022302 042716 177600      BIC      #1C177,(SP)      ;; MAKE IT 7-BIT ASCII
3953 022306 021627 000007      CMP      (SP),#7      ;; IS IT A CONTROL-G?
3954 022312 001300      BNE      2$      ;; IF NOT, PUT IT IN THE TTY QUEUE
3955      ;; AND EXIT
3956
3957      ;*****
3958      ;*CONTROL IS PASSED TO THIS POINT FROM EITHER THE TTY INTERRUPT SERVICE
3959      ;*ROUTINE OR FROM THE SOFTWARE SWITCH REGISTER TRAP CALL, AS A RESULT OF A
3960      ;*CONTROL-G BEING TYPED, AND THE SOFTWARE SWITCH REGISTER BEING SELECTED.
3961 022314 123727 001150 000001 6$: CMPB     $AUTOB,#1      ;; ARE WE RUNNING IN AUTO-MODE?
3962 022322 001674      BEQ      2$      ;; BRANCH IF YES
3963 022324 005726      TST      (SP)+      ;; CLEAR CONTROL-G OFF STACK
3964 022326 004737 021772      JSR      PC,$TKINT      ;; FLUSH THE TTY INPUT QUEUE
3965 022332 005077 156622      CLR      2$TKS      ;; DISABLE TTY KEYBOARD INTERRUPTS
3966 022336 112737 000001 001151      MOVB     #1,$INTAG      ;; SET INTERRUPT MODE INDICATOR
3967
3968 022344 104401 023175      TYPE      , $CNTLG      ;; ECHO THE CONTROL-G (1G)
3969 022350 104401 023202      TYPE      , $MSWR      ;; TYPE CURRENT CONTENTS
3970 022354 013746 000176      MOV      $SWREG,-(SP)      ;; SAVE SWREG FOR TYPEOUT
3971 022360 104402      TYPOC      ;; GO TYPE--OCTAL ASCII(ALL DIGITS)
3972 022362 104401 023213      TYPE      , $MNEW      ;; PROMPT FOR NEW SWR
3973 022366 005046      19$: CLR      -(SP)      ;; CLEAR COUNTER
3974 022370 005046      CLR      -(SP)      ;; THE NEW SWR
3975 022372 105777 156562      7$: TSTB     2$TKS      ;; CHAR THERE?
3976 022376 100375      BPL      7$      ;; IF NOT TRY AGAIN
3977
3978 022400 117746 156556      MOVB     2$TKB,-(SP)      ;; PICK UP CHAR
3979 022404 042716 177600      BIC      #1C177,(SP)      ;; MAKE IT 7-BIT ASCII
3980
3981 022410 021627 000003      CMP      (SP),#3      ;; IS IT A CONTROL-C?
3982 022414 001015      BNE      9$      ;; BRANCH IF NOT
3983 022416 104401 023163      TYPE      , $CNTLC      ;; YES, ECHO CONTROL-C (1C)
3984 022422 062706 000006      ADD      #6,SP      ;; CLEAN UP STACK
3985 022426 123727 001151 000001      CMPB     $INTAG,#1      ;; REENABLE TTY KEYBOARD INTERRUPTS?
3986 022434 001003      BNE      8$      ;; BRANCH IF NO
3987 022436 012777 000100 156514      MOV      #100,2$TKS      ;; ALLOW TTY KEYBOARD INTERRUPTS
3988 022444 000177 156726      8$: JMP      2CNTLC      ;; CONTROL-C RESTART
3989
3990
3991 022450 021627 000025      9$: CMP      (SP),#25      ;; IS IT A CONTROL-U?
3992 022454 001005      BNE      10$      ;; BRANCH IF NOT
3993 022456 104401 023170      TYPE      , $CNTLU      ;; YES, ECHO CONTROL-U (1U)
3994 022462 062706 000006      20$: ADD      #6,SP      ;; IGNORE PREVIOUS INPUT
3995 022466 000737      BR      19$      ;; LET'S TRY IT AGAIN
3996
3997
3998 022470 021627 000015      10$: CMP      (SP),#15      ;; IS IT A <CR>?
3999 022474 001022      BNE      16$      ;; BRANCH IF NO
4000 022476 005766 000004      TST      4(SP)      ;; YES, IS IT THE FIRST CHAR?
4001 022502 001403      BEQ      11$      ;; BRANCH IF YES

```

K06

MAINDEC-11-DZRM, RM03 FORMATTER
DZRMAB.P11 21-JUL-77 15:32

MACY11 30(1046) 21-JUL-77 16:06 PAGE 75
TTY INPUT ROUTINE

SEQ 0074

4002	022504	016677	000002	156442		MOV	2(SP), 2SWR	;; SAVE NEW SWR
4003	022512	062706	000006		11S:	ADD	#6, SP	;; CLEAR UP STACK
4004	022516	104401	001203		14S:	TYPE	\$CRLF	;; ECHO <CR> AND <LF>
4005	022522	123727	001151	000001		CMPB	\$INTAG, #1	;; RE-ENABLE TTY KBD INTERRUPTS?
4006	022530	001003				BNE	15S	;; BRANCH IF NOT
4007	022532	012777	000100	156420		MOV	#100, 2STKS	;; RE-ENABLE TTY KBD INTERRUPTS
4008	022540	000002			15S:	RTI		;; RETURN
4009	022542	004737	020764		16S:	JSR	PC, \$TYPEC	;; ECHO CHAR
4010	022546	021627	000060			CMP	(SP), #60	;; CHAR < 0?
4011	022552	002420				BLT	18S	;; BRANCH IF YES
4012	022554	021627	000067			CMP	(SP), #67	;; CHAR > 7?
4013	022560	003015				BGT	18S	;; BRANCH IF YES
4014	022562	042726	000060			BIC	#60, (SP)+	;; STRIP-OFF ASCII
4015	022566	005766	000002			TST	2(SP)	;; IS THIS THE FIRST CHAR
4016	022572	001403				BEQ	17S	;; BRANCH IF YES
4017	022574	006316				ASL	(SP)	;; NO, SHIFT PRESENT
4018	022576	006316				ASL	(SP)	;; CHAR OVER TO MAKE
4019	022600	006316				ASL	(SP)	;; ROOM FOR NEW ONE.
4020	022602	005266	000002		17S:	INC	2(SP)	;; KEEP COUNT OF CHAR
4021	022606	056616	177776			BIS	-2(SP), (SP)	;; SET IN NEW CHAR
4022	022612	000667				BR	7S	;; GET THE NEXT ONE
4023	022614	104401	001202		18S:	TYPE	\$QUES	;; TYPE ?<CR><LF>
4024	022620	000720				BR	20S	;; SIMULATE CONTROL-U
4025						.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

4036	022622	011646			\$RDCHR:	MOV	(SP), -(SP)	;; PUSH DOWN THE PC AND
4037	022624	016666	000004	000002		MOV	4(SP), 2(SP)	;; THE PS
4038	022632	005066	000004			CLR	4(SP)	;; GET READY FOR A CHARACTER
4039	022636	005046				CLR	-(SP)	;; PUT NEW PS ON STACK
4040	022640	012746	022646			MOV	#64S, -(SP)	;; PUT NEW PC ON STACK
4041	022644	000002				RTI		;; POP NEW PC AND PS
4042	022646				64S:			
4043	022646	005737	021754		1S:	TST	\$STKCNT	;; WAIT ON A CHARACTER
4044	022652	001775				BEQ	1S	
4045	022654	005337	021754			DEC	\$STKCNT	;; DECREMENT THE COUNTER
4046	022660	117766	177074	000004		MOVB	2\$STKQOUT, 4(SP)	;; GET ONE CHARACTER
4047	022666	005237	021760			INC	\$STKQOUT	;; UPDATE THE POINTER
4048	022672	023727	021760	021771		CMP	\$STKQOUT, \$STKQEND	;; DID IT GO OFF C THE END?
4049	022700	001003				BNE	2S	;; BRANCH IF NO
4050	022702	012737	021762	021760		MOV	\$STKQSR, \$STKQOUT	;; RESET THE POINTER
4051	022710	000002			2S:	RTI		;; RETURN

; THIS ROUTINE WILL INPUT A STRING FROM THE TTY
; CALL:
; * RDLIN ; INPUT A STRING FROM THE TTY
; * RETURN HERE ; ADDRESS OF FIRST CHARACTER WILL BE ON THE STACK
; * ; TERMINATOR WILL BE A BYTE OF ALL 0'S

Address	Offset	Hex	Label	Instruction	Comment
4058	022712	010346		SRDLIN: MOV R3, -(SP)	;; SAVE R3
4059	022714	005046		CLR -(SP)	;; CLEAR THE RUBOUT KEY
4060	022716	012703	023146	1S: MOV #STTYIN, R3	;; GET ADDRESS
4061	022722	022703	023163	2S: CMP #STTYIN+15, R3	;; BUFFER FULL?
4062	022726	101456		BLOS 4S	;; BR IF YES
4063	022730	104410		RDCHR	;; GO READ ONE CHARACTER FROM THE TTY
4064	022732	112613		MOVB (SP)+, (R3)	;; GET CHARACTER
4065	022734	122713	000177	10S: CMPB #177, (R3)	;; IS IT A RUBOUT
4066	022740	001022		BNE 5S	;; BR IF NO
4067	022742	005716		TST (SP)	;; IS THIS THE FIRST RUBOUT?
4068	022744	001007		BNE 6S	;; BR IF NO
4069	022746	112737	000134 023144	MOVB #'\\, 9S	;; TYPE A BACK SLASH
4070	022754	104401	023144	TYPE 9S	
4071	022760	012716	177777	MOV #-1, (SP)	;; SET THE RUBOUT KEY
4072	022764	005303		6S: DEC R3	;; BACKUP BY ONE
4073	022766	020327	023146	CMP R3, #STTYIN	;; STACK EMPTY?
4074	022772	103434		BLO 4S	;; BR IF YES
4075	022774	111337	023144	MOVB (R3), 9S	;; SETUP TO TYPEOUT THE DELETED CHAR.
4076	022780	104401	023144	TYPE 9S	;; GO TYPE
4077	022800	000746		BR 2S	;; GO READ ANOTHER CHAR.
4078	023006	005716		5S: TST (SP)	;; RUBOUT KEY SET?
4079	023010	001406		BEQ 7S	;; BR IF NO
4080	023012	112737	000134 023144	MOVB #'\\, 9S	;; TYPE A BACK SLASH
4081	023020	104401	023144	TYPE 9S	
4082	023024	005016		CLR (SP)	;; CLEAR THE RUBOUT KEY
4083	023026	122713	000025	7S: CMPB #25, (R3)	;; IS CHARACTER A CTRL U?
4084	023032	001003		BNE 8S	;; BR IF NO
4085	023034	104401	023170	TYPE \$CNTLU	;; TYPE A CONTROL "U"
4086	023040	000726		BR 1S	;; GO START OVER
4087	023042	122713	000022	8S: CMPB #22, (R3)	;; IS CHARACTER A "+R"?
4088	023046	001011		BNE 3S	;; BRANCH IF NO
4089	023050	105013		CLRB (R3)	;; CLEAR THE CHARACTER
4090	023052	104401	001203	TYPE \$SCRLF	;; TYPE A "CR" & "LF"
4091	023056	104401	023146	TYPE \$STTYIN	;; TYPE THE INPUT STRING
4092	023062	000717		BR 2S	;; GO PICKUP ANOTHER CHARACTER
4093	023064	104401	001202	4S: TYPE \$QUES	;; TYPE A '?'
4094	023070	000712		BR 1S	;; CLEAR THE BUFFER AND LOOP
4095	023072	111337	023144	3S: MOVB (R3), 9S	;; ECHO THE CHARACTER
4096	023076	104401	023144	TYPE 9S	
4097	023102	122723	000015	CMPB #15, (R3)+	;; CHECK FOR RETURN
4098	023106	001305		BNE 2S	;; LOOP IF NOT RETURN
4099	023110	105063	177777	CLRB -1(R3)	;; CLEAR RETURN (THE 15)
4100	023114	104401	001204	TYPE \$LF	;; TYPE A LINE FEED
4101	023120	005726		TST (SP)+	;; CLEAR RUBOUT KEY FROM THE STACK
4102	023122	012603		MOV (SP)+, R3	;; RESTORE R3
4103	023124	011646		MOV (SP), -(SP)	;; ADJUST THE STACK AND PUT ADDRESS OF THE
4104	023126	016666	000004 000002	MOV 4(SP), 2(SP)	;; FIRST ASCII CHARACTER ON IT
4105	023134	012766	023146 000004	MOV #STTYIN, 4(SP)	
4106	023142	000002		RTI	;; RETURN
4107	023144	000		9S: .BYTE 0	;; STORAGE FOR ASCII CHAR. TO TYPE
4108	023145	000		.BYTE 0	;; TERMINATOR
4109	023146	000015		\$TTYIN: .BLKB 15	;; RESERVE 15 BYTES FOR TTY INPUT
4110	023163	136 006503	000012	\$CNTLC: .ASCIZ /C<15><12>	;; CONTROL "C"
4111	023170	052536 005015	000	\$CNTLU: .ASCIZ /U<15><12>	;; CONTROL "U"
4112	023175	136 006507	000012	\$CNTLG:	

M06

MAINDEC-11-DZRNA, RM03 FORMATTER
DZRNA8.P11 21-JUL-77 15:32

MACY11 30(1046) 21-JUL-77 16:06 PAGE 77
TTY INPUT ROUTINE

SEQ 0076

```

4114 023202 005015 053523 020122 SMSWR: .ASCIZ <15><12>/SWR = /
4115 023210 020075 000 000
4116 023213 040 047040 053505 SMNEW: .ASCIZ / NEW = /
4117 023220 036440 000040
4118
4119 .SBTTL POWER DOWN AND UP ROUTINES
4120
4121 ;*****
4122 ;POWER DOWN ROUTINE
4123 023224 012737 023370 000024 $PWRDN: MOV $SILLUP,2#$PWRVEC ;;SET FOR FAST UP
4124 023232 012737 000340 000026 MOV $340,2#$PWRVEC+2 ;;PRIO:7
4125 023240 010046 MOV R0,-(SP) ;;PUSH R0 ON STACK
4126 023242 010146 MOV R1,-(SP) ;;PUSH R1 ON STACK
4127 023244 010246 MOV R2,-(SP) ;;PUSH R2 ON STACK
4128 023246 010346 MOV R3,-(SP) ;;PUSH R3 ON STACK
4129 023250 010446 MOV R4,-(SP) ;;PUSH R4 ON STACK
4130 023252 010546 MOV R5,-(SP) ;;PUSH R5 ON STACK
4131 023254 017746 155674 MOV $SWR,-(SP) ;;PUSH $SWR ON STACK
4132 023260 010637 023374 MOV SP,$SAVR6 ;;SAVE SP
4133 023264 012737 023276 000024 MOV $PWRUP,2#$PWRVEC ;;SET UP VECTOR
4134 023272 000000 HALT
4135 023274 000776 BR .-2 ;;HANG UP
4136
4137 ;*****
4138 ;POWER UP ROUTINE
4139 023276 012737 023370 000024 $PWRUP: MOV $SILLUP,2#$PWRVEC ;;SET FOR FAST DOWN
4140 023304 013706 023374 MOV $SAVR6,$3 ;;GET SP
4141 023310 005037 023374 CLR $SAVR6 ;;WAIT LOOP FOR THE TTY
4142 023314 005237 023374 15: INC $SAVR6 ;;WAIT FOR THE INC
4143 023320 001375 BNE 15 OF WORD
4144 023322 012677 155626 MOV (SP)+,$SWR ;;POP STACK INTO $SWR
4145 023326 012605 MOV (SP)+,R5 ;;POP STACK INTO R5
4146 023330 012604 MOV (SP)+,R4 ;;POP STACK INTO R4
4147 023332 012603 MOV (SP)+,R3 ;;POP STACK INTO R3
4148 023334 012602 MOV (SP)+,R2 ;;POP STACK INTO R2
4149 023336 012601 MOV (SP)+,R1 ;;POP STACK INTO R1
4150 023340 012600 MOV (SP)+,R0 ;;POP STACK INTO R0
4151 023342 012737 023224 000024 MOV $PWRDN,2#$PWRVEC ;;SET UP THE POWER DOWN VECTOR
4152 023350 012737 000340 000026 MOV $340,2#$PWRVEC+2 ;;PRIO:7
4153 023356 104401 TYPE REPORT THE POWER FAILURE
4154 023360 033734 SPWRMG: .WORD PMSG POWER FAIL MESSAGE POINTER
4155 023362 012716 MOV (PC)+,(SP) ;;RESTART AT BEGIN2
4156 023364 006436 SPWRAD: .WORD BEGIN2 ;;RESTART ADDRESS
4157 023366 000002 RTI
4158 023370 000000 $SILLUP: HALT ;;THE POWER UP SEQUENCE WAS STARTED
4159 023372 000776 BR .-2 ;;BEFORE THE POWER DOWN WAS COMPLETE
4160 023374 000000 $SAVR6: 0 ;;PUT THE SP HERE
4161
4162 .SBTTL TYPE NUMERICAL ASCIZ STRING SUPPRESS LEADING ZEROS
4163
4164 ;*****
4165 ;THIS ROUTINE IS USED TO TYPE AN ASCIZ NUMBER SUPPRESSING THE
4166 ;LEADING NUMBERS.
4167 ;CALL
4168 ;* MOV $NUMADR,-(SP) ;;FIRST ADDRESS OF ASCIZ STRING
4169 ;* JSR PC,2#$SUPRS

```

MAINDEC-11-DZMA, RM03 FORMATTER
DZMAB.P11 21-JUL-77 15:32

MACY11 30(1046) 21-JUL-77 16:06 PAGE 78
TYPE NUMERICAL ASCIZ STRING SUPPRESS LEADING ZEROS

SEQ 0077

4170
4171
4172 023376 010046
4173 023400 016600 000004
4174 023404 105710
4175 023406 001403
4176 023410 122720 000060
4177 023414 001773
4178 023416 005300
4179 023420 010037 023426
4180 023424 104401
4181 023426 000000
4182 023430 012600
4183 023432 012616
4184 023434 000207
4185
4186
4187
4188
4189
4190
4191
4192
4193
4194
4195
4196
4197 023436 016637 000002 023466
4198 023444 012746 023466
4199 023450 004737 023472
4200 023454 062716 000005
4201 023460 012666 000002
4202 023464 000207
4203 023466 000000 000000
4204
4205
4206
4207
4208
4209
4210
4211
4212
4213
4214
4215
4216
4217
4218 023472 104412
4219 023474 016602 000002
4220 023500 012700 023652
4221 023504 010066 000002
4222 023510 012201
4223 023512 012202
4224 023514 012737 000012 023570
4225 023522 012704 023602

```

$SUPRS: MOV    RO, -(SP)          ;; SAVE RO
          MOV    4(SP), RO        ;; PICKUP THE POINTER
1$:      TSTB   (RO)              ;; TERMINATE OR?
          BEQ    2$               ;; BR IF YES
          CMPB   #'0, (RO)+       ;; IS THIS AN ASCII "0" ?
          BEQ    1$               ;; BR IF YES
2$:      DEC    RO                ;; BACKUP BY "1"
          MOV    RO, 3$           ;; SAVE FOR TYPING
          TYPE   PC               ;; GO TYPE
3$:      .WORD   0                ;; ASCII POINTER GOES HERE
          MOV    (SP)+, RO        ;; RESTORE RO
          MOV    (SP)+, (SP)      ;; RESTORE THE STACK
          RTS    PC               ;; RETURN

.SBTTL SINGLE LENGTH BINARY TO DECIMAL ASCIZ ROUTINE

;*****
;THIS ROUTINE WILL CONVERT A 16-BIT UNSIGNED BINARY NUMBER TO AN
;UNSIGNED DECIMAL ASCIZ NUMBER.
;CALL
;*
;*      MOV    NUMBER, -(SP)      ;; PUT BINARY NUMBER ON THE STACK
;*      JSR    PC, 0#$S820       ;; CALL
;*      RETURN                    ;; ADDRESS OF THE 1ST ASCIZ CHAR. IS ON THE STACK

$S820:   MOV    2(SP), 1$         ;; SAVE BINARY NUMBER
          MOV    1$, -(SP)        ;; SET POINTER
          JSR    PC, 0#$S0820     ;; CALL DOUBLE LENGTH CONVERT
          ADD    5, (SP)          ;; ONLY ALLOW FIVE CHARACTERS
          MOV    (SP)+, 2(SP)     ;; PICKUP POINTER
          RTS    PC               ;; RETURN
1$:      .WORD   0, 0

.SBTTL DOUBLE LENGTH BINARY TO DECIMAL ASCII CONVERT ROUTINE

;*****
;THIS ROUTINE WILL CONVERT A 32-BIT BINARY NUMBER TO AN UNSIGNED
;DECIMAL (ASCII) NUMBER. THE SIGN OF THE BINARY NUMBER MUST BE
;POSITIVE.
;CALL
;*
;*      MOV    #PNTR, -(SP)      ;; POINTER TO LOW WORD OF BINARY NUMBER
;*      JSR    PC, 0#$S0820     ;; CALL
;*      RETURN                    ;; THE FIRST ADDRESS OF ASCII
;                               ;; IS ON THE STACK

$S0820:   SAVREG                  ;; SAVE REGISTERS
          MOV    2(SP), R2        ;; PICKUP THE DATA POINTER
          MOV    #S0ECVL, RO      ;; GET ADDRESS OF "S0ECVL" STRING
          MOV    RO, 2(SP)        ;; PUT ADDRESS OF ASCII STRING ON STACK
          MOV    (R2)+, R1        ;; PICKUP THE BINARY NUMBER
          MOV    (R2)+, R2
          MOV    #10, 4$         ;; SET UP TO DO 10 CONVERSIONS
          MOV    #STNPOWER, R4   ;; ADDRESS OF TEN POWER

```

4226 023526 012705 023604
 4227 023532 005003
 4228 023534 161401
 4229 023536 005602
 4230 023540 161502
 4231 023542 002402
 4232 023544 005203
 4233 023546 000772
 4234 023550 062401
 4235 023552 005502
 4236 023554 062402
 4237 023556 022525
 4238 023560 052703 000060
 4239 023564 110320
 4240 023566 005327
 4241 023570 000000
 4242 023572 001357
 4243 023574 105020
 4244 023576 104413
 4245 023600 000207
 4246 023602 145000
 4247 023604 035632
 4248 023606 160400
 4249 023610 002765
 4250 023612 113200
 4251 023614 000230
 4252 023616 041100
 4253 023620 000017
 4254 023622 103240
 4255 023624 000001
 4256 023626 023420
 4257 023630 000000
 4258 023632 001750
 4259 023634 000000
 4260 023636 000144
 4261 023640 000000
 4262 023642 000012
 4263 023644 000000
 4264 023646 000001
 4265 023650 000000
 4266 023652 000014

```

MOV      #STNPWR+2,R5
1$:      CLR      R3          ;; CLEAR PARTIAL
2$:      SUB      (R4),R1     ;; SUBTRACT TEN POWER
        SBC      R2
        SUB      (R5),R2
        BLT      3$          ;; BR IF TEN POWER TO LARGE
        INC      R3          ;; ADD 1 TO PARTIAL
        BR      2$          ;; LOOP
3$:      ADD      (R4)+,R1     ;; RESTORE SUBTRACTED VALUE
        ADC      R2
        ADD      (R4)+,R2
        CMP      (R5)+,(R5)+  ;; MOVE TO NEXT TEN POWER
        BIS      #'0,R3      ;; CHANGE PARTIAL TO ASCII
        MOVB     R3,(R0)+    ;; SAVE IT
        DEC      (PC)+       ;; DONE?
4$:      .WORD    0
        BNE     1$          ;; BR IF NO
        CLRB     (R0)+       ;; TERMINATOR
        RESREG
        RTS      PC          ;; RETURN
STNPWR:  145000             ;; 1.0E09
        35632              ;; 1.0E08
        160400
        2765
        113200             ;; 1.0E07
        230
        041100             ;; 1.0E06
        17
        103240             ;; 1.0E05
        1
        23420              ;; 1.0E04
        0
        1750               ;; 1.0E03
        0
        144                ;; 1.0E02
        0
        12                 ;; 1.0E01
        0
        1                  ;; 1.0E00
        0
SDECVL:  .BLKB   12.        ;; RESERVE STORAGE FOR ASCII STRING

```

.SBTTL SAVE AND RESTORE R0-R5 ROUTINES

```

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

```

```

4282
4283
4284
4285 023666
4286 023666 010046
4287 023670 010146
4288 023672 010246
4289 023674 010346
4290 023676 010446
4291 023700 010546
4292 023702 016646 000022
4293 023706 016646 000022
4294 023712 016646 000022
4295 023716 016646 000022
4296 023722 000002
4297
4298
4299
4300
4301 023724
4302 023724 012666 000022
4303 023730 012666 000022
4304 023734 012666 000022
4305 023740 012666 000022
4306 023744 012605
4307 023746 012604
4308 023750 012603
4309 023752 012602
4310 023754 012601
4311 023756 012600
4312 023760 000002
4313
4314
4315
4316
4317
4318
4319
4320
4321
4322 023762 010046
4323 023764 016600 000002
4324 023770 005740
4325 023772 111000
4326 023774 006300
4327 023776 016000 024016
4328 024002 000200
4329
4330
4331
4332
4333 024004 011646
4334 024006 016666 000004 000002
4335 024014 000002
4336
4337

```

```

;+12---R1
;+14---R0

```

SSAVREG:

```

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

```

;*RESTORE RO-R5

;*CALL:

;* RESREG

\$RESREG:

```

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

```

.SBTTL TRAP DECODER

```

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

```

```

$TRAP: MOV 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		
4343		ROUTINE		
4344	024016 024004	STRPAD: .WORD STRAP2		
4345	024020 020552	STYPE ;; CALL=TYPE	TRAP+1(104401)	TTY TYPEOUT ROUTINE
4346	024022 021326	STYPOC ;; CALL=TYPOC	TRAP+2(104402)	TYPE OCTAL NUMBER (WITH LEADING ZEROS)
4347	024024 021302	STYPOS ;; CALL=TYPOS	TRAP+3(104403)	TYPE OCTAL NUMBER (NO LEADING ZEROS)
4348	024026 021342	STYPON ;; CALL=TYPON	TRAP+4(104404)	TYPE OCTAL NUMBER (AS PER LAST CALL)
4349	024030 021530	STYPDS ;; CALL=TYPDS	TRAP+5(104405)	TYPE DECIMAL NUMBER (WITH SIGN)
4350				
4351	024032 022350	SGTSWR ;; CALL=GTSWR	TRAP+6(104406)	GET SOFT-SWR SETTING
4352				
4353	024034 022260	SKCSWR ;; CALL=CKSWR	TRAP+7(104407)	TEST FOR CHANGE IN SOFT-SWR
4354	024036 022622	SRDCHR ;; CALL=RDCHR	TRAP+10(104410)	TTY TYPEIN CHARACTER ROUTINE
4355	024040 022712	SRDLIN ;; CALL=RDLIN	TRAP+11(104411)	TTY TYPEIN STRING ROUTINE
4356	024042 023666	SSAVREG ;; CALL=SAVREG	TRAP+12(104412)	SAVE RD-RS ROUTINE
4357	024044 023724	SRESREG ;; CALL=RESREG	TRAP+13(104413)	RESTORE RD-RS ROUTINE

.SBTTL SINGLE/DUAL PORT RH70/RM03 DRIVER (REV 1.0)

; COPYRIGHT (C) 1976
; DIGITAL EQUIPMENT CORP.
; MAYNARD, MA 01754
; AUTHOR(S): JIM LACEY/CHUCK HESS/C. CHEN

; *****

; STORAGE FOR RMDS, RMER1, RMER2, AND RMMR2 ON AN ERROR "2"

; RMERRS = RMDS
; RMERRS+2 = RMER1
; RMERRS+4 = RMER2
; RMERRS+6 = RMMR2

4376 024046 000000 000000 000000 RMERRS: .WORD 0,0,0,0

; TABLE OF DRIVE ACTIVE INDICATORS (DRVACT=8 BYTES)

; DRVACT=0 IF DRIVE IS IDLE
; DRVACT>0 IF DRIVE IS ACTIVE WITH A COMMAND
; DRVACT<0 IF DRIVE IS ACTIVE WITH AN ERROR RECOVERY OPERATION

4384	024056	000	DRVACT: .BYTE 0	: DRIVE 0
4385	024057	000	.BYTE 0	: DRIVE 1
4386	024060	000	.BYTE 0	: DRIVE 2
4387	024061	000	.BYTE 0	: DRIVE 3
4388	024062	000	.BYTE 0	: DRIVE 4
4389	024063	000	.BYTE 0	: DRIVE 5
4390	024064	000	.BYTE 0	: DRIVE 6
4391	024065	000	.BYTE 0	: DRIVE 7

; TABLE OF DRIVE STATUS INDICATORS (DRVSTA=8 BYTES)

E07

MAINDEC-11-DZMA, RM03 FORMATTER
DZRMW8.P11 21-JUL-77 15:32

MACY11 30(1046) 21-JUL-77 16:06 PAGE 82
SINGLE/DUAL PORT RH70/RM03 DRIVER (REV 1.0)

SEQ 0081

```

4394
4395
4396
4397
4398 024066 000
4399 024067 000
4400 024070 000
4401 024071 000
4402 024072 000
4403 024073 000
4404 024074 000
4405 024075 000
4406
4407
4408
4409
4410
4411
4412 024076 000
4413 024077 000
4414 024100 000
4415 024101 000
4416 024102 000
4417 024103 000
4418 024104 000
4419 024105 000
4420
4421
4422
4423
4424
4425 024106 000
4426 024107 000
4427 024110 000
4428 024111 000
4429 024112 000
4430 024113 000
4431 024114 000
4432 024115 000
4433
4434
4435
4436
4437
4438 024116 000
4439 024117 000
4440 024120 000
4441 024121 000
4442 024122 000
4443 024123 000
4444 024124 000
4445 024125 000
4446
4447
4448
4449

```

```

;DRVSTA=0 IF DRIVE IS OFFLINE OR NONEXISTENT
;DRVSTA>0 IF DRIVE IS ONLINE
;DRVSTA<0 IF DRIVE IS UNSAFE

DRVSTA: .BYTE 0 ;DRIVE 0
        .BYTE 0 ;DRIVE 1
        .BYTE 0 ;DRIVE 2
        .BYTE 0 ;DRIVE 3
        .BYTE 0 ;DRIVE 4
        .BYTE 0 ;DRIVE 5
        .BYTE 0 ;DRIVE 6
        .BYTE 0 ;DRIVE 7

;TABLE OF DRIVE TYPES (DRVTYP=8 BYTES)
;DRVTYP=0 IF DRIVE IS NONEXISTENT (DRVSTA=0, ALSO)
;DRVTYP=4 IF DRIVE IS RM03
;DRVTYP=-1 IF NOT RM03

DRVTYP: .BYTE 0 ;DRIVE 0
        .BYTE 0 ;DRIVE 1
        .BYTE 0 ;DRIVE 2
        .BYTE 0 ;DRIVE 3
        .BYTE 0 ;DRIVE 4
        .BYTE 0 ;DRIVE 5
        .BYTE 0 ;DRIVE 6
        .BYTE 0 ;DRIVE 7

;TABLE OF DUAL PORT INITIALIZATION INDICATORS
;DPINT=0 IF INITIALIZATION IS NOT ACTIVE ON THE DRIVE
;DPINT<0 IF INITIALIZATION IS IN PROGRESS

DPINT: .BYTE 0 ;DRIVE 0
        .BYTE 0 ;DRIVE 1
        .BYTE 0 ;DRIVE 2
        .BYTE 0 ;DRIVE 3
        .BYTE 0 ;DRIVE 4
        .BYTE 0 ;DRIVE 5
        .BYTE 0 ;DRIVE 6
        .BYTE 0 ;DRIVE 7

;TABLE OF PENDING DUAL PORT REQUESTS
;DPRQS=0 IF THAT A DUAL PORT REQUEST IS NOT PENDING FOR THAT DRIVE
;DPRQS<0 IF THAT A DUAL PORT REQUEST IS PENDING FOR THAT DRIVE

DPRQS: .BYTE 0 ;DRIVE 0
        .BYTE 0 ;DRIVE 1
        .BYTE 0 ;DRIVE 2
        .BYTE 0 ;DRIVE 3
        .BYTE 0 ;DRIVE 4
        .BYTE 0 ;DRIVE 5
        .BYTE 0 ;DRIVE 6
        .BYTE 0 ;DRIVE 7

;TRANSFER WAIT FLAG (TRNSWT=1 WORD)
;THIS IS A ONE WORD QUEUE. IT WILL CONTAIN THE ADDRESS OF
;"DPB" OF THE I/O OPERATION.

```

F07

MAINDEC-11-DZRNA, RM03 FORMATTER
DZRNAB.P11 21-JUL-77 15:32

MACY11 30(1046) 21-JUL-77 16:06 PAGE 83
SINGLE/DUAL PORT RH70/RM03 DRIVER (REV 1.0)

SEQ 0082

```

4450
4451 024126 000000      TRNSWT: .WORD 0
4452
4453      ;SEARCH WAIT KEYS (SRCHWT=1 WORD)
4454      ;THIS IS A ONE WORD QUEUE THAT WILL CONTAIN A KEY FOR EACH OF
4455      ;THE DRIVES THAT ARE PERFORMING A SEARCH COMMAND FOR THE I/O
4456      ;REQUEST THAT IS AT THE TOP OF THEIR REQUEST QUEUE.
4457      ;EACH DRIVE IS ASSIGNED ONE BIT, STARTING AT BIT00 FOR DRIVE 0.
4458
4459 024130 000000      SRCHWT: .WORD 0
4460
4461      ;RM03 DRIVER ACTIVE FLAG (ACTDRV=1 BYTE)
4462      ;ACTDRV=0 IF DRIVER IS INACTIVE
4463      ;ACTDRV>0 IF DRIVER IS ACTIVE
4464
4465 024132 000         ACTDRV: .BYTE 0
4466
4467      ;SOFTWARE TIMER ROUTINE ACTIVE FLAG (ACTSTR=1 BYTE)
4468      ;ACTSTR=0 IF SOFTWARE TIMER ROUTINE IS INACTIVE
4469      ;ACTSTR>0 IF SOFTWARE TIMER ROUTINE IS ACTIVE
4470
4471 024133 000         ACTSTR: .BYTE 0
4472
4473      ;UNLOAD FLAG (ULDFLG=8 BYTES)
4474      ;ULDFLG=0 IF NO UNLOAD COMMAND
4475      ;ULDFLG>0 IF UNLOAD COMMAND IN PROGRESS
4476      ;ULDFLG<0 IF UNLOAD COMMAND IN WAIT QUEUE
4477
4478 024134 000         ULDFLG: .BYTE 0      ;DRIVE 0
4479 024135 000         .BYTE 0      ;DRIVE 1
4480 024136 000         .BYTE 0      ;DRIVE 2
4481 024137 000         .BYTE 0      ;DRIVE 3
4482 024140 000         .BYTE 0      ;DRIVE 4
4483 024141 000         .BYTE 0      ;DRIVE 5
4484 024142 000         .BYTE 0      ;DRIVE 6
4485 024143 000         .BYTE 0      ;DRIVE 7
4486
4487      ;LOOK AHEAD COUNT (LACNT=8 BYTES)
4488      ;LACNT WILL INDICATE THE NUMBER OF LOOK AHEADS PERFORMED
4489
4490 024144 000         LACNT: .BYTE 0      ;DRIVE 0
4491 024145 000         .BYTE 0      ;DRIVE 1
4492 024146 000         .BYTE 0      ;DRIVE 2
4493 024147 000         .BYTE 0      ;DRIVE 3
4494 024150 000         .BYTE 0      ;DRIVE 4
4495 024151 000         .BYTE 0      ;DRIVE 5
4496 024152 000         .BYTE 0      ;DRIVE 6
4497 024153 000         .BYTE 0      ;DRIVE 7
4498
4499      ;SAVE REGISTERS FLAG (SAVEFG =1 WORD)
4500      ;SAVEFG <0 IF SAVE THE RH70/RM03 REGISTERS WHEN THE
4501      ;OPERATION IS COMPLETED AS PER (DPB+14).
4502      ;SAVEFG=0 IF SAVE THE RH70/RM03 REGISTERS, AS PER
4503      ;(DPB+14), AFTER AN ERROR.
4504
4505 024154 000000      SAVEFG: .WORD 0

```

G07

MAINDEC-11-DZRNA, RM03 FORMATTER
DZRNAB.P11 21-JUL-77 15:32MACY11 30(1046) 21-JUL-77 16:06 PAGE 84
SINGLE/DUAL PORT RH70/RM03 DRIVER (REV 1.0)

SEQ 0083

```

4506
4507
4508
4509
4510
4511
4512
4513 024156 177777
4514
4515
4516
4517
4518 024160 177777
4519 024162 177777
4520 024164 177777
4521 024166 177777
4522 024170 177777
4523 024172 177777
4524 024174 177777
4525 024176 177777
4526
4527
4528
4529
4530
4531 024200 177777
4532
4533
4534
4535
4536
4537 024202 001
4538 024203 002
4539 024204 004
4540 024205 010
4541 024206 020
4542 024207 040
4543 024210 100
4544 024211 200
4545
4546
4547
4548
4549 024212 000003
4550
4551
4552
4553
4554 024214 176700
4555 024216 000254 000240
4556
4557
4558
4559 024222 000004
4560
4561

```

```

;SEEK FLAG (SEEKFG=1 WORD)
;SEEKFG=0 IF WHEN THE DISK ADDRESS ISN'T IN THE WINDOW
;FOR A DATA TRANSFER START A SEARCH COMMAND
;SEEKFG<0 IF DATA TRANSFER WILL DO IMPLIED SEEKS,
;DISREGARD THE WINDOW

SEEKFG: .WORD -1

;TIMEOUT TABLE (TIMER=8 WORDS)
;THIS TABLE CONTAINS THE TIME ALLOWED FOR AN OPERATION

TIMER: .WORD -1 ;DRIVE 0
        .WORD -1 ;DRIVE 1
        .WORD -1 ;DRIVE 2
        .WORD -1 ;DRIVE 3
        .WORD -1 ;DRIVE 4
        .WORD -1 ;DRIVE 5
        .WORD -1 ;DRIVE 6
        .WORD -1 ;DRIVE 7

;DATA TRANSFER UNDERWAY INDICATOR (DTUW=1 WORD)
;DTUW<0 IF NO DATA TRANSFER UNDERWAY
;DTUW=+N (WHERE N=0 TO 7) IMPLIES DATA TRANSFER UNDERWAY ON DRIVE N

DTUW: .WORD -1

;ATTENTION BITS TABLE (ATABIT=8 BYTES)
;THIS TABLE CONTAINS THE CORRESPONDING BIT TO EACH DRIVES
;ATTENTION BIT

ATABIT: .BYTE 1 ;DRIVE 0
        .BYTE 2 ;DRIVE 1
        .BYTE 4 ;DRIVE 2
        .BYTE 10 ;DRIVE 3
        .BYTE 20 ;DRIVE 4
        .BYTE 40 ;DRIVE 5
        .BYTE 100 ;DRIVE 6
        .BYTE 200 ;DRIVE 7

;FSRM03 TO RH70 "MASSBUS CONTROL BUS PARITY ERRORS" (MCPE) ALLOWED BEFORE
;CALLING IT FATAL (MCPEMX=1 WORD)

MCPEMX: .WORD 3

;STORAGE FOR RMADR (THE FIRST ADDRESS (776700) OF THE RH70/RM03),
;RMVEC (THE VECTOR ADDRESS (254)), AND RMVEC+2 (THE BR LEVEL (5)).

RMADR: .WORD 176700
RMVEC: .WORD 254,5*32.

;MAXIMUM NUMBER OF LOOK AHEADS ALLOWED IS 4 (MXLACT=1 WORD)

MXLACT: .WORD 4
;MAXIMUM DELTA DELAY IS 8 SECTORS (MXDLTA=1 WORD)

```

H07

MAINDEC-11-DZRNA, RMO3 FORMATTER
DZRNAB.P11 21-JUL-77 15:32

MACY11 30(1046) 21-JUL-77 16:06 PAGE 85
SINGLE/DUAL PORT RH70/RMO3 DRIVER (REV 1.0)

SEQ 0084

4562 024224 001000

4563

4564

4565 024226 000200

4566

4567

4568 024230 000005

4569

4570

4571

4572 000000

4573 000002

4574 000004

4575 000006

4576 000010

4577 000012

4578 000014

4579 000016

4580 000020

4581 000022

4582 000024

4583 000026

4584 000030

4585 000032

4586 000034

4587 000036

4588 000040

4589 000042

4590 000044

4591 000046

4592

4593

4594

4595

4596

4597

4598

4599

4600

4601

4602

4603

4604

4605

4606 024232 104412

4607 024234 013746 177776

4608 024240 012737 000240 177776

4609 024246 004737 032046

4610 024252 012701 024046

4611 024258 012702 024156

4612 024262 005021

4613 024264 020102

4614 024266 103775

4615 024270 012702 024200

4616 024274 012721 177777

4617 024300 020102

MXDLTA: .WORD 8.*64.

;MINIMUM DELTA DELAY IS 2 SECTORS (MNDLTA=1 WORD)

MNDLTA: .WORD 2.*64.

;MAXIMUM SEARCH FOR I/O WINDOW IS 5 SECTORS (MXWINDOW=1 WORD)

MXWINDOW: .WORD 5

;DEFINITIONS OF THE RH70/RMO3 ADDRESS INDEXES

RMCS1=0

RMWC=2

RMAA=4

RMDA=6

RMCS2=10

RMD5=12

RMER1=14

RMA5=16

RMLA=20

RMD8=22

RMMR1=24

RMDT=26

RMSN=30

RMOF=32

RMDC=34

RMHR=36

RMMR2=40

RMER2=42

RMEC1=44

RMEC2=46

;CONTROL AND STATUS REGISTER #1 (DRIVE REG. 00)

;WORD COUNT REGISTER (NOT A DRIVE REG)

;UNIBUS ADDRESS REGISTER (NOT A DRIVE REG)

;DESIRED SECTOR/TRACK ADDRESS REGISTER (DRIVE REG. 05)

;CONTROL AND STATUS REGISTER #2 (NOT A DRIVE REG)

;DRIVE STATUS REGISTER (DRIVE REG 01)

;ERROR REGISTER #1 (DRIVE REG. 02)

;ATTENTION SUMMARY PSEUDO REGISTER (DRIVE REG. 04)

;LOOK AHEAD REGISTER (DRIVE REG. 07)

;DATA BUFFER REGISTER (NOT A DRIVE REG.)

;MAINTAINABILITY REGISTER (DRIVE REG. 03)

;DRIVE TYPE REGISTER (DRIVE REG. 06)

;SERIAL NUMBER REGISTER (DRIVE REG. 10)

;OFFSET REGISTER (DRIVE REG. 11)

;DESIRED CYLINDER ADDRESS REGISTER (DRIVE REG. 12)

;DUMMY ADDRESS REGISTER (DRIVE REG. 13)

;MAINTENANCE REGISTER #2

;ERROR REGISTER #2 (DRIVE REG. 15)

;ECC POSITION REGISTER (DRIVE REG. 16)

;ECC PATTERN REGISTER (DRIVE REG. 17)

;RH70/RMO3 DRIVER INITIALIZATION CODE

;THIS ROUTINE WILL DETERMINE WHICH RMO3 DRIVES ARE
;AVAILABLE FOR TESTING AND SET THE DRVSTA INDICATOR
;TO THE PROPER STATE FOR EACH DRIVE.
;NOTE: THIS ROUTINE CALLS DRVINT

;CALL

JSR PC,RMINIT

RETURN

;NOTE: THE 'P' OR 'L' CLOCK MUST BE STARTED

RMINIT: SAVREG

MOV @#PS, -(SP)

MOV #(<5*32.> @#PS

JSR PC, CLRQUE

MOV #RMERRS, R1

MOV #SEEKFG, R2

1\$: CLR (R1)+

CMP R1, R2

BLO 1\$

MOV #DTUM, R2

2\$: MOV #-1, (R1)+

CMP R1, R2

;SAVE R0 - R5

;SAVE THE PRESENT PROCESSOR STATUS

;CHANGE THE PRIORITY TO 5

;CLEAR ALL REQUEST QUEUES

;FIRST ADDRESS TO BE CLEARED

;LAST ADDRESS TO BE CLEARED

;CLEAR

;ARE WE DONE?

;BRANCH IF NO

;LAST ADDRESS

;INITIALIZE

;DONE?

MAINDEC-11-DZMA, RM03 FORMATTER
DZRMAB.P11 21-JUL-77 15:32

MACY11 30(1046) 21-JUL-77 16:06 PAGE 86
SINGLE/DUAL PORT RH70/RM03 DRIVER (REV 1.0)

SEQ 0085

```

4618 024302 101774          BLOS 2$          ;LOOP IF NO
4619 024304 005037 024066    CLR  DRVSTA      ;SET ALL DRIVES TO OFFLINE
4620 024310 005037 024070    CLR  DRVSTA+2
4621 024314 005037 024072    CLR  DRVSTA+4
4622 024320 005037 024074    CLR  DRVSTA+6
4623 024324 013703 024216    MOV  RMVEC,R3      ;SETUP THE RH70/RM03 VECTOR
4624 024330 012723 027020    MOV  #ISR,(R3)+
4625 024334 013713 024220    MOV  RMVEC+2,(R3)
4626 024340 013704 024214    MOV  RMAOR,R4      ;FIRST ADDRESS OF RH70/RM03
4627 024344 012764 000040 000010 MOV  #BIT05,RMCS2(R4) ;MASSBUS INIT
4628 024352 005001          CLR  R1             ;START WITH DRIVE 0
4629 024354 004037 024444    3$: JSR  RO,DRVINT ;INIT THE DRIVE
4630 024360 000401          BR   4$             ;'DVA' NOT SET OR PARITY ERROR
4631 024362 000402          BR   5$             ;NORMAL RETURN
4632 024364 105061 024066    4$: CLRB DRVSTA(R1) ;SET DRIVE STATUS TO OFFLINE
4633 024370 005201          5$: INC  R1          ;GO TO NEXT DRIVE
4634 024372 042701 177770    BIC  #1C7,R1      ;MASK OUT UNUSED BITS
4635 024376 001366          BNE  3$             ;BR IF MORE DRIVES TO GO
4636 024400 012701 000007    MOV  #7,R1         ;START WITH DRIVE 7
4637 024404 005037 177776    CLR  #PS         ;CLEAR THE PROCESSOR STATUS
4638 024410 105761 024106    6$: TSTB DPINT(R1) ;WAITING FOR DRIVE TO SWITCH PORTS ?
4639 024414 001405          BEQ  8$             ;BR NOT WAITING
4640 024416 004737 031502    JSR  PC,SET.IE     ;SET INTERRUPT
4641 024422 105761 024106    7$: TSTB DPINT(R1) ;DRIVE SWITCHED PORTS ?
4642 024426 001375          BNE  7$             ;BR IF NOT
4643 024430 005301          8$: DEC  R1          ;GO TO THE NEXT DRIVE
4644 024432 100366          BPL  6$             ;CHECK NEXT DRIVE
4645 024434 012637 177776    MOV  (SP)+,#PS    ;RESTORE THE PROCESSOR STATUS
4646 024440 104413          RESREG ;RESTORE R0 - R5
4647 024442 000207          RTS  PC             ;BYE-BYE

;DRIVE INITIALIZATION ROUTINE
;THIS ROUTINE DETERMINES IF A DRIVE EXIST AND IF IT IS
;AN RM03. IF IT IS, A "READ-IN PRESET" IS ISSUED AND FMT22
;IS SET TO A "1". THEN MOL, DPR, DRY, AND VV ARE CHECKED TO
;INSURE THEY ARE ALL ON A "1". AND DEPENDING ON THEIR STATE,
;DRVSTA IS SET TO THE PROPER CONDITION.

;CALL
MOV  #DRVNUM,R1 ;DRIVE NUMBER TO R1
MOV  RMAOR,R4   ;UNIBUS ADDRESS OF RH70/RM03 (RMCS1)
JSR  RO,DRVINT  ;CALLED BY A JSR
RETURN1 ;ERROR OCCURRED (PARITY)
RETURN2 ;NORMAL RETURN

DRVINT: MOV  R5,-(SP) ;SAVE R5
DULP:  CLRB DRVSTA(R1) ;START DRIVE STATUS AS OFFLINE
        CLRB DRVSTYP(R1) ;CLEAR THE DRIVE TYPE INDICATOR
        CLRB ULDFLG(R1) ;CLEAR THE UNLOAD FLAG
        MOV  R1,RMCS2(R4) ;SELECT A DRIVE
        MOVB #11,RMCS1(R4) ;DO A DRIVE CLEAR COMMAND (& SEIZE DRIVE)
        BIT  #BIT12,RMCS2(R4) ;NONEXISTENT DRIVE?
        BEQ  1$         ;NO---BRANCH
        JSR  PC,SET.IE  ;GO SET "IE" WITHOUT A "TRE"
        BR   6$         ;LEAVE THIS ROUTINE
1$:     CLRB DRVSTA(R1) ;SET DRIVE STATUS TO OFFLINE

```

J07

MAINDEC-11-DZRNA, RM03 FORMATTER
DZRNA8.P11 21-JUL-77 15:32MACY11 30(1046) 21-JUL-77 16:06 PAGE 87
SINGLE/DUAL PORT RH70/RM03 DRIVER (REV 1.0)

SEQ 0086

4674	024516	032764	004000	000000	BIT	#BIT11,RMCS1(R4)	;SEE IF DRIVE AVAILABLE
4675	024524	001750			BEQ	DULP	;BR IF DRIVE NOT AVAILABLE
4676	024536	004037	031012		JSR	RO,RO.RM	;READ THE DRIVE TYPE REG.
4677	024548	000026			RMDT		
4678	024534	024732			B\$		;ERROR RETURN ADDRESS
4679	024536	012605			MOV	(SP)+,R5	;PUT DRIVE TYPE IN R5
4680	024540	112761	000004	024076	MOVB	#4,DRVSTYP(R1)	;SET RM03 INDICATOR
4681	024546	022705	020024		CMP	#20024,R5	;SINGLE PORT RM03 ?
4682	024552	001407			BEQ	2\$	;BR IF YES
4683	024554	022705	024024		CMP	#24024,R5	;DUAL PORT RM03 ?
4684	024556	001404			BEQ	2\$	;BR IF YES
4685	024558	112761	177777	024076	MOVB	#-1,DRVSTYP(R1)	;SET INDICATOR TO 'OTHER'
4686	024560	000446			BR	6\$	;EXIT
4687	024562	012746	000121		MOV	#121,-(SP)	;DO A "READ-IN PRESET"
4688	024564	004037	031172		JSR	RO,WRT.RM	
4689	024566	000000			RMCS1		
4690	024568	024732			B\$		
4691	024570	012746	010000		MOV	#BIT12,-(SP)	;SET FMT22=1
4692	024572	004037	031172		JSR	RO,WRT.RM	
4693	024574	000332			RMOF		
4694	024576	024732			B\$		
4695	024578	004037	031012		JSR	RO,RO.RM	;READ RMD5
4696	024580	000012			RMD5		
4697	024582	024732			B\$		
4698	024584	012605			MOV	(SP)+,R5	;AND SAVE IT IN R5
4699	024586	100015			BPL	4\$	;BRANCH IF ATA=0
4700	024588	116164	024202	000016	MOVB	ATABIT(R1),RMAS(R4)	;CLEAR ATTENTION BIT
4701	024590	004037	031012		JSR	RO,RO.RM	;FIND OUT WHY ATA=1
4702	024592	000014			RMR1		
4703	024594	024732			B\$		
4704	024596	006126			ROL	(SP)+	;IS IT UNSAFE?
4705	024598	100004			BPL	4\$	;BR IF NOT
4706	024600	112761	177777	024066	MOVB	#-1,DRVSTA(R1)	;SET UNSAFE INDICATOR
4707	024602	000407			BR	6\$	;EXIT
4708	024604	005105			COM	R5	;CHECK MOL, DPR, DRY, AND VV
4709	024606	042705	167077		BIC	#1C<BIT12!BIT08!BIT07!BIT06>,R5	
4710	024608	001003			BNE	6\$	;BRANCH IF MOL, DPR, DRY, OR VV IS CLEAR
4711	024610	112761	000001	024066	MOVB	#1,DRVSTA(R1)	;SET DRIVE STATUS TO ONLINE
4712	024612	005720			TST	(R0)+	;STEP OVER THE ERROR RETURN
4713	024614	000410			BR	8\$	;EXIT
4714	024616	006301			ASL	R1	;CHANGE INDEX TO ADDRESS WORDS
4715	024618	012761	003720	024160	MOV	#2000.,TIMER(R1)	;START 2 SEC TIMER
4716	024620	006201			ASR	R1	;RESTORE R1
4717	024622	112761	177777	024106	MOVB	#-1,DPINT(R1)	;SET PORT INITIALIZE INIDICATOR
4718	024624	012605			MOV	(SP)+,R5	;RESTORE R5
4719	024626	000200			RTS	RO	;EXIT
4720							
4721							
4722							
4723							
4724							
4725							
4726							
4727							
4728							
4729							

;REQUEST PRE-PROCESSOR-HANDLES SUBSYSTEM REQUEST

;CALL

```

JSR    RO,#RM03
PNTADR
RETURN1
RETURN2

```

```

;CALL THE RM03 DRIVER
;ADDRESS OF POINTER OF DRIVES PARAMETER BLOCK
;RETURN HERE IF QUEUE IS FULL
;RETURN HERE IF REQUEST IS IN QUEUE OR THERE
;IS AN ERROR CONDITION

```

K07

MAINDEC-11-DZAMA, RM03 FORMATTER
DZAMAB.P11 21-JUL-77 15:32

MACY11 30(1046) 21-JUL-77 16:06 PAGE 88
SINGLE/DUAL PORT RM70/RM03 DRIVER (REV 1.0)

SEQ 0087

4730											
4731	024736	013746	177776			RM03:	MOV	2#PS, -(SP)		SAVE THE CALLING STATUS	
4732	024742	013737	024220	177776			MOV	RMVEC+2, 2#PS		DON'T ALLOW ANY RM03 INTERRUPTS	
4733	024750	112737	000001	024132			MOVB	#1, ACTDRV		SET "ACTIVE DRIVER" FLAG	
4734	024756	104412					SAVREG			SAVE R0 - R5	
4735	024760	011002					MOV	(R0), R2		PICKUP THE DRIVE PARAMETER BLOCK POINTER	
4736	024762	005062	000016				CLR	16(R2)		CLEAR THE STATUS/ERROR INDICATOR	
4737	024766	111201					MOVB	(R2), R1		PICKUP THE DRIVE NUMBER	
4738	024770	013704	024214				MOV	RMADR, R4		UNIBUS ADDRESS OF RMCS1	
4739	024774	105761	024066				TSTB	DRVSTA(R1)		CHECK DRIVES STATUS	
4740	025000	003014					BGT	15		BRANCH IF ONLINE	
4741	0 5002	105761	024134				TSTB	ULDFLG(R1)		UNLOAD COMMAND IN QUEUE?	
4742	0 5006	001036					BNE	35		BRANCH IF YES	
4743	0 5010	105761	024106				TSTB	DPINT(R1)		TRYING TO INIT THE DRIVE	
4744	0 5014	001042					BNE	55		BR IF YES	
4745	025016	004037	024444				JSR	R0, DRVINT		GO INIT. THE DRIVE	
4746	0 5022	000434					BR	45		ERROR RETURN	
4747	0 5024	105761	024066				TSTB	DRVSTA(R1)		IS DRIVE STATUS ONLINE?	
4748	0 5030	003445					BLE	65		BR IF NOT	
4749	0 5032	105761	024116			15:	TSTB	DPRQS(R1)		OUTSTANDING PORT REQUEST FOR THE DRIVE ?	
4750	0 5036	001031					BNE	55		BR IF YES	
4751	0 5040	010164	000010				MOV	R1, RMCS2(R4)		SELECT THE DRIVE	
4752	0 5044	004037	032144				JSR	R0, DRVQUE		PUT THIS REQUEST IN QUEUE	
4753	0 5050	000460					BR	95		QUEUE IS FULL	
4754	0 5052	122762	000103	000002			CMPB	#103, 2(R2)		IS THIS REQ. FOR AN UNLOAD?	
4755	0 5056	001003					BNE	25		BR IF NO	
4756	0 5060	112761	177777	024134			MOVB	#-1, ULDFLG(R1)		SET THE "UNLOAD IN QUEUE" FLAG	
4757	0 5064	105761	024056			25:	TSTB	DRVACT(R1)		IS THIS DRIVE ACTIVE?	
4758	0 5068	001043					BNE	85		BR IF YES	
4759	0 5072	004737	025230				JSR	PC, OPT		CALL THE OPTIMIZER	
4760	0 5076	000440					BR	85			
4761	0 5080	012762	120000	000016		35:	MOV	#BIT15!BIT13, 16(R2)		SET THE "UNLOAD IN QUEUE" ERROR FLAG	
4762	0 5084	000434					BR	85		EXIT	
4763	0 5088	004737	026310			45:	JSR	PC, CI7		GO HANDLE THE PARITY ERROR	
4764	0 5092	000431					BR	85			
4765	0 5096	004037	032144			55:	JSR	R0, DRVQUE		PUT REQUEST IN QUEUE	
4766	0 5100	000431					BR	95		QUEUE IS FULL	
4767	0 5104	032714	000100				BIT	#BIT06, (R4)		IE BIT SET ?	
4768	0 5108	001023					BNE	85		YES	
4769	0 5112	004737	031502				JSR	PC, SET.IE		SET THE INTERRUPT	
4770	0 5116	000420					BR	85		RETURN	
4771	0 5120	105761	024066			65:	TSTB	DRVSTA(R1)		SEE IF DRIVE OFFLINE OR UNSAFE	
4772	0 5124	002412					BLT	75		BR IF UNSAFE	
4773	0 5128	012762	140000	000016			MOV	#BIT15!BIT14, 16(R2)		SET OFFLINE ERROR INDICATOR	
4774	0 5132	105761	024076				TSTB	DRVSTP(R1)		SEE IF OFFLINE OR NONEXISTENT	
4775	0 5136	001007					BNE	85		BR IF OFFLINE	
4776	0 5140	012762	100002	000016			MOV	#BIT15!BIT01, 16(R2)		REPORT DRIVE NONEXISTENT	
4777	0 5144	000403					BR	85		GO TO EXIT	
4778	0 5148	012762	110000	000016		75:	MOV	#BIT15!BIT12, 16(R2)		DRIVE IS UNSAFE	
4779	0 5152	104413				85:	RESREG			RESTORE R0 - R5	
4780	0 5156	005720					TST	(R0)+		SETUP FOR NORMAL RETURN	
4781	0 5160	000401					BR	105		FINISH UP, THEN EXIT	
4782	0 5164	104413				95:	RESREG			RESTORE R0 - R5	
4783	0 5168	005720				105:	TST	(R0)+		CORRECT THE RETURN ADDRESS	
4784	025216	105037	024132				CLRB	ACTDRV		CLEAR "ACTIVE DRIVER" FLAG	
4785	025222	012637	177776				MOV	(SP)+, 2#PS		RETURN "PS" TO USER LEVEL	

L07

 MAINDEC-11-DZRNA, RM03 FORMATTER
 DZRNAB.P11 21-JUL-77 15:32

 MACY11 30(1046) 21-JUL-77 16:06 PAGE 89
 SINGLE/DUAL PORT RH70/RM03 DRIVER (REV 1.0)

SEQ 0088

```

4786 025226 000200          RTS      RO          ;RETURN TO CALLER
4787
4788          ;OPTIMIZER-CALLED FOR A PARTICULAR DRIVE
4789
4790          ;CALL
4791          ;MOV      #DRVNUM,R1          ;DRIVE NUMBER TO R1
4792          ;JSR      PC,OPT          ;SETUP A COMMAND
4793
4794          OPT:  SAVREG          ;SAVE R0 - R5
4795          MOV      @#PS, -(SP)          ;SAVE PROC. STATUS
4796          BICB     ATABIT(R1),SRCHWT          ;CLEAR LA SEARCH FLAG
4797          CLRB     DPRQS(R1)          ;RESET THE PORT REQ FLAG ****
4798          JSR      PC,GETREQ          ;GET "DPB" POINTER OF REQUEST
4799          TST      R2          ;IS THERE A REQUEST IN QUEUE?
4800          BEQ      7$          ;NO--BRANCH TO EXIT
4801          MOV      R1,RMCS2(R4)          ;LOAD THE DRIVE ADDRESS *****
4802          MOV      #111,RMCS1(R4)          ;CLEAR THE DRIVE
4803          BIT      #BIT11,RMCS1(R4)          ;DVA SET ?
4804          BEQ      5$          ;TO PROT REQUEST, IF NOT
4805          TSTB     DRVSTA(R1)          ;IS DRIVE ONLINE?
4806          BGT      1$          ;YES--BRANCH
4807          JSR      PC,POPUQ          ;NO--REMOVE REQUEST FROM QUEUE
4808          MOV      #BIT15:BIT14,16(R2)          ;SET OFFLINE STATUS/ERROR INDICATOR
4809          TSTB     DRVSTA(R1)          ;IS DRIVE UNSAFE ?
4810          BPL      8$          ;BR TO EXIT IF NOT
4811          MOV      #BIT15:BIT12,16(R2)          ;SET UNSAFE STATUS/ERROR INDICATOR
4812          BR       8$          ;BRANCH TO EXIT
4813
4814          1$:  MOV      #111, -(SP)          ;LOAD COMMAND ONTO THE STACK
4815          JSR      RO,WRT.RM          ;LOAD THE REGISTER
4816          RMCS1     6$          ;REGISTER INCREMENT
4817          BIT      #BIT11,(R4)          ;DRIVE AVAILABLE ?
4818          BEQ      9$          ;BR IF NOT
4819          CMPB     #150,2(R2)          ;IS THE REQUEST FOR I/O?
4820          BLT      2$          ;YES--BRANCH
4821          JSR      PC,C14          ;CALL THE COMMAND INITIATOR
4822          BR       8$          ;BRANCH TO EXIT
4823          TST      DTUW          ;DATA TRANSFER UNDERWAY?
4824          BGE      4$          ;YES--GO START A SEARCH
4825          TST      SEEKFG          ;DO IMPLIED SEEKS?
4826          BMI      3$          ;YES---BRANCH
4827          JSR      RO,LA          ;NO--DO LOOK AHEAD
4828          BR       8$          ;RETURN HERE ON A PARITY ERROR
4829          BR       4$          ;GO START A SEARCH
4830          JSR      PC,C11          ;START A DATA TRANSFER
4831          BR       8$
4832          JSR      PC,C13          ;START A SEARCH
4833          BR       8$          ;GO TO THE EXIT
4834          MOVB     #-1,DPRQS(R1)          ;SET PORT REQUEST INDICATOR
4835          MOV      R1,R3          ;SET UP TO ADDRESS WORDS
4836          ASL      R3          ;CONVERT TO WORD INDEX
4837          MOV      #10000.,TIMER(R3)          ;START 10 SEC TIMER
4838          BR       7$          ;EXIT
4839          JSR      PC,C17          ;PROCESS THE PARITY ERROR
4840          BIT      #BIT06,(R4)          ;SEE IF 'IE' ALREADY SET
4841

```

M07

MAINDEC-11-DZRNA, RM03 FORMATTER
DZRNA8.P11 21-JUL-77 15:32

MACY11 30(1046) 21-JUL-77 16:06 PAGE 90
SINGLE/DUAL PORT RH70/RM03 DRIVER (REV 1.0)

SEQ 0089

4842 025450 001002
4843 0 2 004737 031502
4844 0 3456 012637 177776
4845 0 5462 104413
4846 025464 000207
4847
4848
4849
4850
4851
4852
4853
4854
4855
4856
4857
4858
4859 025466 004737 032242
4860 025472 010237 024126
4861 025476 010203
4862 025500 013704 024214
4863 025504 010164 000010
4864 025510 062703 000004
4865 025514 062704 000002
4866 0 30 012324
4867 0 322 012324
4868 025524 012346
4869 0 326 004037 031172
4870 0 5532 000006
4871 0 534 026310
4872 0 336 012346
4873 0 540 004037 031172
4874 025544 000034
4875 025546 026310
4876 0 550 016246 000002
4877 025554 004037 031172
4878 025560 000000
4879 0 62 026310
4880 025564 010137 024200
4881 025570 000137 026252
4882 025574 013704 024214
4883 025600 010164 000010
4884 025604 016246 000012
4885 025610 004037 031172
4886 025614 000034
4887 025616 026310
4888 025620 116203 000010
4889
4890
4891
4892 025624 010346
4893 025626 042716 177740
4894 025632 116266 000011 000001
4895 025640 004037 031172
4896 025644 000006
4897 025646 026310

```

      BNE      BS
      JSR      PC,SET.IE
BS:    MOV      (SP)+,2#PS
      RESREG
      RTS      PC

;COMMAND INITIATOR
;CALL
      MOV      #DRVNUM,R1
      MOV      #DPB,R2
      JSR      PC,C1?
;DRIVE NUMBER
;ADDRESS OF DPB
;C1?= C11,C13, OR C14
;WHERE:
;C11=DATA TRANSFER
;C12=SEARCH REQUESTED BY DATA XFER
;C14=NOT DATA TRANSFER

C11:   JSR      PC,POPQUE
      MOV      R2,TRANSWT
      MOV      R2,R3
      MOV      RMDR,R4
      MOV      R1,RMCS2(R4)
      ADD      #4,R3
      ADD      #2,R4
      MOV      (R3)+,(R4)+
      MOV      (R3)+,(R4)+
      MOV      (R3)+,-(SP)
      JSR      RO,WRT.RM
      RMDR
      CI7
      MOV      (R3)+,-(SP)
      JSR      RO,WRT.RM
      RMDC
      CI7
      MOV      2(R2),-(SP)
      JSR      RO,WRT.RM
      RMCS1
      CI7
      MOV      R1,DTUW
      JMP      C15
;REMOVE REQUEST FROM "DRIVES WAIT" QUEUE
;PUT REQ. IN TRANSFER WAIT QUEUE
;DPB ADDRESS TO R3
;RMCS1 ADDRESS
;SELECT DRIVE
;DESIRED WORD COUNT
;RMC ADDRESS
;LOAD WORD COUNT
;LOAD BUFFER ADDRESS
;LOAD SECTOR AND TRACK
;CALL THE LOAD(WRITE) ROUTINE
;INDEX OF REGISTER TO LOAD
;ERROR RETURN ADDRESS
;LOAD CYLINDER ADDRESS

;LOAD "COMMAND+GO", "A17&A16", AND "PSEL"

C13:   MOV      RMDR,R4
      MOV      R1,RMCS2(R4)
      MOV      12(R2),-(SP)
      JSR      RO,WRT.RM
      RMDC
      CI7
      MOV      10(R2),R3
      SUB      MXWINDW,R3
      BGE      15
      ADD      #32,R3
      MOV      R3,-(SP)
      BIC      #177740,(SP)
      MOV      11(R2),1(SP)
      JSR      RO,WRT.RM
      RMDR
      CI7
;SET "DATA TRANSFER UNDERWAY"
;RMCS1 ADDRESS
;SELECT DRIVE
;DESIRED CYLINDER ADDRESS

;PICKUP SECTOR ADDRESS
;BACKUP BY MAX. SEARCH FOR I/O WINDOW

;COMBINE THE ADJUSTED SECTOR WITH
;CHOP OF ALL EXTENDED BITS ,IF ANY
;THE DESIRED TRACK
;LOAD DESIRED TRACK & SECTOR

```

NO7

MAINDEC-11-DZRNA, RM03 FORMATTER
DZRNA8.P11 21-JUL-77 15:32

MACY11 30(1046) 21-JUL-77 16:06 PAGE 91
SINGLE/DUAL PORT RM70/RM03 DRIVER (REV 1.0)

SEQ 0090

4898	025650	012746	000131		MOV	#131, -(SP)	; START A SEARCH
4899	025654	004037	031172		JSR	RD, WRT.RM	
4900	025660	000000			RMCS1		
4901	025662	026310			CI7		
4902	025664	156137	024202	024130	BISB	ATABIT(R1), SRCHWT	; SET "SEARCH WAIT" KEY
4903	025672	000567			BR	CI5	
4904	025674	013704	024214		MOV	RMADR, R4	; RMCS1 ADDRESS
4905	025700	010164	000010	CI4:	MOV	R1, RMCS2(R4)	; SELECT DRIVE
4906	025704	116203	000002		MOVB	2(R2), R3	; PICKUP THE REQUESTED COMMAND
4907	025710	122703	000131		CMPB	#131, R3	; IS IT A SEARCH COMMAND?
4908	025714	001007			BNE	1\$	; BRANCH IF NO
4909	025716	016246	000010		MOV	10(R2), -(SP)	; LOAD DESIRED TRACK & SECTOR
4910	025722	004037	031172		JSR	RD, WRT.RM	
4911	025726	000006			RMDA		
4912	025730	026310			CI7		
4913	025732	000403			BR	2\$	; GO LOAD CYLINDER
4914	025734	122703	000105	1\$:	CMPB	#105, R3	; IS IT A SEEK COMMAND
4915	025740	001007			BNE	3\$	; BRANCH IF NO
4916	025742	016246	000012	2\$:	MOV	12(R2), -(SP)	; LOAD DESIRED CYLINDER
4917	025746	004037	031172		JSR	RD, WRT.RM	
4918	025752	000034			RMDC		
4919	025754	026310			CI7		
4920	025756	000546			BR	CI6	
4921	025760	122703	000115	3\$:	CMPB	#115, R3	; IS IT AN "OFFSET" COMMAND?
4922	025764	001013			BNE	4\$	; BR IF NO
4923	025766	004037	031012		JSR	RD, RD.RM	; MERGE THE OFFSET VALUE INTO RMOF
4924	025772	000032			RMOF		; BUT DON'T CHANGE THE UPPER
4925	025774	026310			CI7		
4926	025776	116216	000001		MOVB	1(R2), (SP)	; BYTE WHEN LOADING THE
4927	026002	004037	031172		JSR	RD, WRT.RM	; REGISTER (RMOF)
4928	026006	000032			RMOF		
4929	026010	026310			CI7		
4930	026012	000530			BR	CI6	; GO START THE COMMAND
4931	026014	122703	000107	4\$:	CMPB	#107, R3	; IS IT A "RECALIBRATE" COMMAND?
4932	026020	001525			BEQ	CI6	; BRANCH IF YES
4933	026022	122703	000117		CMPB	#117, R3	; IS IT A RETURN TO CENTER?
4934	026026	001522			BEQ	CI6	; BRANCH IF YES
4935	026030	122703	000103		CMPB	#103, R3	; IS IT AN "UNLOAD" COMMAND?
4936	026034	001016			BNE	5\$	; BRANCH IF NO
4937	026036	112761	000001	024056	MOVB	#1, DRVACT(R1)	; SET THE DRIVE ACTIVE INDICATOR
4938	026044	105061	024066		CLRB	DRVSTA(R1)	; PUT DRIVE STATUS TO OFFLINE
4939	026050	112761	000001	024134	MOVB	#1, ULDFLG(R1)	; SET "UNLOAD IN PROGRESS" FLAG
4940	026056	010346			MOV	R3, -(SP)	; START THE "UNLOAD" COMMAND
4941	026060	004037	031172		JSR	RD, WRT.RM	
4942	026064	000000			RMCS1		
4943	026066	026310			CI7		
4944	026070	000207			RTS	PC	; RETURN TO USER
4945	026072	122703	000143	5\$:	CMPB	#143, R3	; IS IT A "SET FORMAT" COMMAND?
4946	026076	001014			BNE	6\$	; BRANCH IF NO
4947	026100	004037	031012		JSR	RD, RD.RM	; READ THE OFFSET REGISTER
4948	026104	000032			RMOF		
4949	026106	026310			CI7		
4950	026110	116266	000001	000001	MOVB	1(R2), 1(SP)	; COMBINE "FMT22", "ECI", AND "HCI"
4951	026116	004037	031172		JSR	RD, WRT.RM	; LOAD "FMT22", "ECI", AND/OR "HCI".
4952	026122	000032			RMOF		
4953	026124	026310			CI7		

4954	026126	000436			BR	12\$	
4955	026130	122703	000141		6\$: CMPB	#141,R3	; IS IT A "GET REGISTER" COMMAND?
4956	026134	001023			BNE	10\$	; BRANCH IF NO
4957	026136	016203	000006		7\$: MOV	6(R2),R3	; POINTS TO 1ST ADDRESS OF WHERE
4958							; TO PUT THE REGISTER(S)
4959	026142	116237	000010	026160	MOVB	10(R2),9\$	; INIT. THE INDEX FOR THE FIRST REG.
4960	026150	116205	000011		MOVB	11(R2),R5	; INDEX OF LAST REG. TO MOVE
4961	026154	004037	031012		8\$: JSR	RO,RO.RM	; READ RH70/RMO3 REGISTER
4962	026160	000000			9\$: RMCS1		; INDEX OF REG. TO READ
4963	026162	026310			CI7		
4964	026164	012623			MOV	(SP)+,(R3)+	; GET THE CONTENTS OF RH70//RMO3 REG.
4965	026166	023705	026160		CMP	9\$,R5	; LAST REG. BEEN READ?
4966	026172	001414			BEQ	12\$	; GET OUT IF YES
4967	026174	062737	000002	026160	ADD	#2,9\$	; INCREASE THE INDEX BY 2
4968	026202	000764			BR	8\$	; LOOP--MORE TO READ
4969	026204	122703	000145		10\$: CMPB	#145,R3	; IS IT A "SELECT DRIVE" COMMAND?
4970	026210	001405			BEQ	12\$	; BRANCH IF YES
4971	026212	010346			11\$: MOV	R3,-(SP)	; LOAD THE COMMAND
4972	026214	004037	031172		JSR	RO,WRT.RM	
4973	026220	000000			RMCS1		
4974	026222	026310			CI7		
4975	026224	004737	032242		12\$: JSR	PC,POPQUE	; REMOVE REG. FROM QUEUE
4976	026230	052762	000200	000016	BIS	#BIT07,16(R2)	; SET THE "DONE" BIT
4977	026236	005737	024154		TST	SAVEFG	; SAVE THE RH70/RMO3 REGISTERS?
4978	026242	100002			BPL	13\$	; BRANCH IF NO
4979	026244	004737	031364		JSR	PC,SVRH70	; YES--GO SAVE THE REGISTERS
4980	026250	000207			13\$: RTS	PC	; RETURN TO USER
4981	026252	006301			CI5:	R1	
4982	026254	012761	001750	024160	ASL	#1000.,TIMER(R1)	; SET A ONE SECOND TIMER
4983	026262	006201			MOV	R1	
4984	026264	112761	000001	024056	ASR	R1	
4985	026272	000207			MOVB	#1,DRVACT(R1)	; SET THE DRIVE ACTIVE
4986	026274	010346			RTS	PC	; RETURN TO THE USER
4987	026276	004037	031172		CI6: MOV	R3,-(SP)	; LOAD THE COMMAND
4988	026302	000000			JSR	RO,WRT.RM	
4989	026304	026310			RMCS1		
4990	026306	000761			CI7		
4991	026310	032764	010000	000010	BR	CI5	
4992	026316	001034			CI7: BIT	#BIT12,RMCS2(R4)	; DRIVE NON-EXISTENT ?
4993	026320	005702			BNE	CI8	; BR IF YES
4994	026322	001405			1\$: TST	R2	; ANYTHING IN QUEUE ?
4995	026324	012762	104000	000016	BEQ	CI7B	; BR IF NOT
4996	026332	004737	031364		MOV	#BIT15:BIT11,16(R2)	; SET "PARITY" ERROR INDICATOR
4997	026336	012746	000111		JSR	PC,SVRH70	; GO SAVE THE RH70/RMO3 REGISTERS
4998	026342	004037	031172		CI7B: MOV	#111,-(SP)	; DO A "DRIVE CLEAR"
4999	026346	000000			JSR	RO,WRT.RM	
5000	026350	026410			RMCS1		
5001	026352	004737	032124		CI8		
5002	026356	105061	024134		JSR	PC,EMPTYQ	; EMPTY THE QUEUE
5003	026362	105061	024056		CLR B	ULDFLG(R1)	; CLEAR THE UNLOAD IN QUEUE FLAG
5004	026366	020137	024200		CLR B	DRVACT(R1)	; DRIVE IS IDLE
5005	026372	001005			CMP	R1,DTUW	; IF THIS DRIVE HAD AN I/O REQUEST
5006	026374	005037	024126		BNE	1\$	; IN PROGRESS CLEAR ALL OF THE FLAGS
5007	026400	012737	177777	024200	CLR	TRNSWT	
5008	026406	000207			1\$: MOV	#-1,DTUW	
5009	026410	104412			CI8: RTS	PC	
					SAVREG		; SAVE RO - R5

```

5010 026412 032764 010000 000010      BIT      #BIT12,RMCS2(R4)  ;IS 'NED' SET ?
5011 026412 001002                      BNE      1$              ;BR IF YES
5012 026412 005001                      CLR      R1
5013 026424 005003                      CLR      R3
5014 026426 105761 024056      1$:    TSTB     DRVACT(R1)      ;DRIVE ACTIVE?
5015 026432 001443                      BEQ      5$              ;BRANCH IF NO
5016 026434 013702 024126      MOV      TRANSW,R2      ;GET THE "TRANSFER WAIT" QUEUE
5017 026440 020137 024200      CMP      R1,DTUM        ;DID THIS DRIVE HAVE AN I/O IN PROGRESS?
5018 026444 001402                      BEQ      2$              ;BRANCH IF YES
5019 026446 004737 032220      JSR      PC,GETREQ      ;GET THE DPB POINTER
5020 026452 005702                      TST      R2              ;QUEUE ENTRY FOR DRIVE ?
5021 026454 001415                      BEQ      4$              ;BR IF NOT
5022 026456 032764 010000 000010      BIT      #BIT12,RMCS2(R4)  ;'NED' SET ?
5023 026464 001404                      BEQ      3$              ;BR IF NOT
5024 026466 012762 100002 000016      MOV      #BIT15:BIT01,16(R2) ;SET 'DRIVE NON-EXISTENT' INDICATOR
5025 026474 000405                      BR       4$              ;CONTINUE
5026 026476 012762 102000 000016      3$:    MOV      #BIT15:BIT10,16(R2) ;SET "NON-CLEARABLE PARITY" ERROR INDICATOR
5027 026504 004737 031364      JSR      PC,SVRH70      ;SAVE RH70/RMD3 REGISTERS
5028 026510 012763 177777 024160      4$:    MOV      #-1,TIMER(R3)    ;STOP THE TIMER
5029 026516 105061 024056      CLRB     DRVACT(R1)      ;SET "DRIVE ACTIVE" TO IDLE
5030 026522 020137 024200      CMP      R1,DTUM        ;IS THIS DRIVE SETUP FOR A TRANSFER
5031 026526 001005                      BNE      5$              ;BR IF NOT
5032 026530 012737 177777 024200      MOV      #-1,DTUM        ;RESET THE INDICATOR
5033 026536 005037 024126      CLR      TRANSW      ;CLEAR THE TRANSFER QUEUE
5034 026542 105061 024134      5$:    CLRB     ULDFLG(R1)    ;CLEAR UNLOAD FLAG
5035 026546 032764 010000 000010      BIT      #BIT12,RMCS2(R4)  ;'NED' SET ?
5036 026554 001021                      BNE      6$              ;BR IF YES
5037 026556 005201                      INC      R1              ;MOVE TO THE NEXT DRIVE
5038 026560 062703 000002                      ADD      #2,R3
5039 026564 042701 177770                      BIC      #1C7,R1
5040 026570 001316                      BNE      1$              ;BRANCH IF MORE DRIVES
5041 026572 012737 177777 024200      MOV      #-1,DTUM        ;NO DATA TRANSFERS UNDERWAY
5042 026600 005037 024126      CLR      TRANSW      ;CLEAR THE 'TRANSFER WAIT' QUEUE
5043 026604 004737 032046      JSR      PC,CLIQUE      ;CLEAR ALL OF THE REQUEST QUEUES
5044 026610 012764 000040 000010      MOV      #BIT05,RMCS2(R4) ;DO A MASSBUS INIT.
5045 026616 000406                      BR       7$              ;CONTINUE
5046 026620 004737 032124      6$:    JSR      PC,EMPTYQ      ;CLEAR THE DRIVE'S QUEUE
5047 026624 105061 024066      CLRB     DRVSTA(R1)    ;SET DRIVE TO OFFLINE
5048 026630 105061 024076      CLRB     DRVSTYP(R1)   ;CLEAR THE DRIVE TYPE INDICATOR
5049 026634 004737 031502      7$:    JSR      PC,SET.IE    ;SET "IE" WITHOUT "TRE"
5050 026640 104413                      RESREG    PC              ;RESTORE R0 - R5
5051 026642 000207                      RTS      PC              ;RETURN
5052
5053      ;LOOK AHEAD ROUTINE
5054
5055      ;CALL
5056      ;
5057      ;      MOV      #DRVNUM,R1      ;DRIVE NUMBER
5058      ;      MOV      #DPB,R2        ;POINT TO DPB
5059      ;      JSR      R0,LA          ;GO CHECK THE WINDOW
5060      ;      RETURN1      ;ERROR RETURN
5061      ;      RETURN2      ;START A SEARCH
5062      ;      RETURN3      ;START A DATA TRANSFER
5063 026644 013704 024214      LA:    MOV      RMADR,R4      ;GET RMCS1'S ADDRESS
5064 026650 010164 000010      MOV      R1,RMCS2(R4)    ;SELECT DRIVE
5065 026654 004037 031012      JSR      R0,RD.RM        ;READ DRIVE STATUS

```

5066	026660	000012			RMO3			
5067	026662	027012			4\$		; ERROR RETURN ADDRESS	
5068	026664	042716	157577		BIC	#1C020200, (SP)	; ON CYLINDER ?	
5069	026670	022726	000200		CMP	#200, (SP)+	; PIP=0, DRY=1?	
5070	026674	001044			BNE	3\$	; NO	
5071	026676	105261	024144		INCB	LACNT(R1)	; INCREMENT THE LOOK AHEAD COUNT	
5072	026702	126137	024144	024222	CMPB	LACNT(R1), MXLACT	; EXCEED MAX?	
5073	026710	003033			BGT	2\$	; BRANCH IF YES	
5074	026712	116203	000010		MOVB	10(R2), R3	; GET DESIRED SECTOR ADDRESS AND	
5075	026716	000303			SWAB	R3	; MULT. BY 64--ALIGN WITH	
5076	026720	000203			ASR	R3	; LOOK AHEAD REGISTER	
5077	026722	006203			ASR	R3		
5078	026724	012737	000340	177776	MOV	#340, 2#PS	; PRIORITY LEVEL "7"	
5079	026732	004037	031012		JSR	RO, RO.RM	; READ LOOK AHEAD REGISTER	
5080	026736	000020			RMLA			
5081	026740	027012			4\$			
5082	026742	021664	000020		CMP	(SP), RMLA(R4)	; CORRECT LA NUMBER ?	
5083	026746	001402			BEQ	7\$	; YES	
5084	026750	005726			TST	(SP)+	; NO, CLEAR STACK	
5085	026752	000415			BR	3\$		
5086	026754	162603			SUB	(SP)+, R3	; CALCULATE THE DELTA	
5087	026756	002002			BGE	1\$		
5088	026760	062703	004000		ADD	#(32.*64.), R3	; MAKE THE DELTA POSITIVE	
5089	026764	023703	024224		CMP	MXDLTA, R3	; CHECK THE DELTA TO SEE	
5090	026770	002406			BLT	3\$	; IF IT IS WITHIN THE	
5091	026772	023703	024226		CMP	MXDLTA, R3	; WINDOW---IF YES, ZERO	
5092	026776	002003			BGE	3\$	; THE LOOK AHEAD COUNT	
5093	027000	105061	024144		CLRB	LACNT(R1)	; AND TAKE THE I/O EXIT	
5094	027004	005720			TST	(RO)+		
5095	027006	005720			TST	(RO)+	; ADJUST THE RETURN ADDRESS	
5096	027010	000402			BR	5\$	; EXIT	
5097	027012	004737	026310		JSR	PC, CI7	; PROCESS THE ERROR	
5098	027016	000200			RTS	RO	; RETURN	
5099								
5100							; INTERRUPT SERVICE ROUTINE	
5101								
5102	027020	112737	000001	024132	ISR:	MOVB	#1, ACTDRV	; SET "ACTIVE DRIVER" FLAG
5103	027026	104412			SAVREG		; SAVE RO - R5	
5104	027030	013704	024214		MOV	RMAOR, R4	; ADDRESS OF RHSCSI	
5105	027034	013701	024200		MOV	DTUW, R1	; GET "DATA TRANSFER UNDERWAY" INDICATOR	
5106	027040	002403			BLT	1\$	; BRANCH IF NO DATA TRANSFER UNDERWAY	
5107	027042	004737	027064		JSR	PC, TD	; CALL TRANSFER DONE	
5108	027046	000402			BR	2\$	; EXIT	
5109	027050	004737	027234		JSR	PC, SC	; CALL SPECIAL CONDITIONS	
5110	027054	104413			RESREG		; RESTORE RO - R5	
5111	027056	105037	024132		CLRB	ACTDRV	; CLEAR "ACTIVE DRIVER" FLAG	
5112	027062	000002			RTI		; RETURN	
5113								
5114							; TRANSFER DONE ROUTINE	
5115								
5116	027064	105061	024056		TD:	CLRB	DRVACT(R1)	; SET DRIVE ACTIVE INDICATOR TO IDLE
5117	027070	012737	177777	024200	MOV	#-1, DTUW	; NO DATA TRANSFERS UNDERWAY	
5118	027076	006301			ASL	R1		
5119	027100	012761	177777	024160	MOV	#-1, TIMER(R1)	; CANCEL TIMEOUT	
5120	027106	006201			ASR	R1		
5121	027110	013702	024126		MOV	TRANST, R2	; GET "DPB" ADDRESS FROM THE	

5122	027114	005037	024126		CLR	TRANSW	;TRANSFER WAIT QUEUE--CLEAR QUEUE
5123	027120	052762	000200	000016	BIS	#BIT07,16(R2)	;SET DONE
5124	027126	010164	000010		MOV	R1,RMC52(R4)	;SELECT THE DRIVE
5125	027132	004037	031012		JSR	RO,RO.RM	;TRANSFER ERROR(TRE=1)?
5126	027136	000000			RMCS1		
5127	027140	026310			CI7		
5128	027142	006126			ROL	(SP)+	
5129	027144	100417			BMI	3\$	;BR IF YES
5130	027146	005737	024154		TST	SAVEFG	;SAVE THE RH70/RMD3 REGISTERS?
5131	027152	100002			BPL	1\$	;BRANCH IF NO
5132	027154	004737	031364		JSR	PC,SVRH70	;YES--SAVE THE REGISTERS
5133	027160			1\$:			
5134	027160	004737	032220		JSR	PC,GETREQ	;GET DPB POINTER
5135	027164	005702			TST	R2	;ENTRY FOR DRIVE ?
5136	027166	001403			BEQ	2\$	;BR IF NOT
5137	027170	004737	025230		JSR	PC,OPT	;CALL OPTIMIZER
5138	027174	000417			BR	SC	;CHECK OTHER DRIVES
5139							;THE RELEASE DRIVE COMMAND IS FORCED TO ENTER FOR DUAL PORT OPERATION
5140	027176	012714	000113		2\$:	MOV	#113,(R4)
5141	027202	000414			BR	SC	;CHECK FOR OTHER DRIVES
5142	027204	052762	100100	000016	3\$:	BIS	#BIT15:BIT06,16(R2)
5143	027212	004737	032124		JSR	PC,EMPTYQ	;EMPTY THE "DRIVE'S WAIT" QUEUE
5144	027216	004737	031364		JSR	PC,SVRH70	;SAVE THE RH70/RMD3 REGISTERS
5145	027222	012714	040111		MOV	#40111,(R4)	;ISSUE A "DRIVE CLEAR"
5146	027226	012714	000113		MOV	#113,(R4)	;ISSUE A RELEASE TO THE DRIVE
5147	027232	000400			BR	SC	;CHECK FOR OTHER DRIVES
5148							
5149							
5150							
5151							
5152							;SPECIAL CONDITION ROUTINE
5153	027234	116403	000016		SC:	MOVB	RMA5(R4),R3
5154	027240	001014			BNE	2\$	;BRANCH IF ANY 'ATA' BITS SET
5155	027242	004037	031012		JSR	RO,RO.RM	;READ CONTROL AND STATUS REGISTER
5156	027246	000000			RMCS1		
5157	027250	026410			CI8		
5158	027252	106126			ROLB	(SP)+	;IS "IE"=1?
5159	027254	100405			BMI	1\$	;YES, NO DRIVES TO CHECK
5160	027256	004037	032310		JSR	RO,ES.SAV	;SAVE THE ADDRESS IN 'SESCAPE'
5161	027262	104001			ERROR	1	;REPORT AN ILLEGAL INTERRUPT
5162	027264	004737	031502		JSR	PC,SET.IE	;SET INTERRUPT ENABLE
5163	027270	000207		1\$:	RTS	PC	;RETURN
5164	027272	005046		2\$:	CLR	-(SP)	;PROCESS ALL DRIVES THAT HAVE
5165	027274	110316			MOVB	R3,(SP)	;AN "ATA"=1
5166	027276	012703	000001		MOV	#1,R3	
5167	027302	005001			CLR	R1	
5168	027304	030316			SC3:	BIT	R3,(SP)
5169	027306	001005			BNE	SC5	;ATA=1?
5170	027310	005201			SC4:	INC	R1
5171	027312	106303			ASLB	R3	;MOVE TO THE NEXT DRIVE
5172	027314	001373			BNE	SC3	;BRANCH IF MORE TO CHECK?
5173	027316	005726			TST	(SP)+	;CLEAN OFF THE STACK
5174	027320	000207			RTS	PC	;RETURN TO USER
5175	027322	105761	024106		SC5:	TSTB	DPINT(R1)
5176	027326	001402			BEQ	1\$	;BR IF NOT
5177	027330	000137	030246		JMP	SC13	;PROCESS THE DRIVE

F08

 MAINDEC-11-DZRNA, RM03 FORMATTER
 DZRNA8.P11 21-JUL-77 15:32

 MACY11 30(1046) 21-JUL-77 16:06 PAGE 96
 SINGLE/DUAL PORT RH70/RM03 DRIVER (REV 1.0)

SEQ 0095

5178	027334	105761	024116	15:	TSTB	DPRQS(R1)	:PORT REQUEST OUTSTANDING ?
5179	027340	001402			BEQ	25	:BR IF NOT
5180	027342	000137	030246		JMP	SC13	:START THE OUTSTANDING COMMAND
5181	027346	105761	024066	25:	TSTB	DRVSTA(R1)	:CHECK THE DRIVE STATUS
5182	027352	003025			BGT	55	:BRANCH IF ONLINE
5183	027354	105761	024134		TSTB	ULDFLG(R1)	:UNLOAD IN PROGRESS?
5184	027360	003422			BLE	55	:BRANCH IF NOT
5185	027362	004737	032220		JSR	PC,GETREQ	:GET DPB POINTER
5186	027366	004737	031364		JSR	PC,SVRH70	:SAVE THE RH70/RM03 REGISTERS
5187	027372	004737	030176		JSR	PC,SC12	:SAVE RMD5, RMR1, RMR2, AND RMMR2
5188							:ALSO DO A DRIVE INIT (DRVINT)
5189							:DID DRIVE COME ONLINE?
5189	027376	105761	024066		TSTB	DRVSTA(R1)	
5190	027402	003416			BLE	65	:NO---BRANCH
5191	027404	032737	040000 024046		BIT	#BIT14,RMERRS	:WAS THERE AN ERROR?
5192	027412	001002			BNE	35	:BR IF ERROR
5193	027414	000137	030066		JMP	SC11	:NO ERROR
5194	027420	013705	024050	35:	MOV	RMERRS+2,R5	:YES -- PICKUP RMR1 AND
5195	027424	000502			BR	SC6A	:GO PROCESS THE ERROR
5196	027426	105761	024056	55:	TSTB	DRVACT(R1)	:DRIVE ACTIVE WITH COMMAND OR ERROR RECOVERY ?
5197	027432	001033			BNE	SC6	:BR IF EITHER
5198	027434	004737	030176		JSR	PC,SC12	:SAVE RMD5, RMR1, RMR2, AND RMMR2
5199							:ALSO DO A DRVINT
5200	027440	105761	024106	65:	TSTB	DPINT(R1)	:TRYING TO INIT THE DRIVE ?
5201	027444	001321			BNE	SC4	:BR IF YES, CHECK ON MORE DRIVES
5202	027446	105761	024066		TSTB	DRVSTA(R1)	:CHECK ON DRIVE'S STATUS
5203	027452	100412			BMI	75	:BR IF UNSAFE
5204	027454	032737	020000 024052		BIT	#BIT13,RMERRS+4	:ADDRESS PLUG CHANGED ?
5205	027462	001013			BNE	85	:BR IF YES
5206	027464	012746	000113		MOV	#113,-(SP)	:RELEASE COMMAND
5207	027470	004037	031172		JSR	RD,WRD.RM	:WRITE THE COMMAND INTO RMCS1
5208	027474	000000			RMCS1		:REGISTER INDEX
5209	027476	030036			SC8		:PARITY EXIT ADDRESS
5210	027500	011605		75:	MOV	(SP),R5	:PICKUP (RMAS) BEFORE THE ERROR CALL
5211	027502	004037	032310		JSR	RD,ES.SAV	:SAVE THE ADDRESS IN 'SESCAPE'
5212	027506	104002			ERROR	2	:REPORT THE UNEXPECTED ATTENTION
5213	027510	000677			BR	SC4	:GO CHECK FOR MORE ATA'S
5214	027512			85:			
5215	027512	004037	032310		JSR	RD,ES.SAV	:SAVE THE ADDRESS IN 'SESCAPE'
5216	027516	104005			ERROR	5	:REPORT THE ADDRESS PLUG CHANGE
5217	027520	000673			BR	SC4	:CHECK FOR MORE DRIVES
5218	027522	006301		SC6:	ASL	R1	:SETUP TO ADDRESS WORDS
5219	027524	012761	177777 024160		MOV	#-1,TIMER(R1)	:STOP THE TIMER
5220	027532	006201			ASR	R1	:RESTORE THE DRIVE ADDRESS
5221	027534	004737	032220		JSR	PC,GETREQ	:GET THE DPB POINTER FROM THE QUEUE
5222	027540	010164	000010		MOV	R1,RMCS2(R4)	:SELECT DRIVE
5223	027544	004037	031012		JSR	RD,RD.RM	:READ THE RM03'S STATUS REG.
5224	027550	000012			RMD5		
5225	027552	030036			SC8		
5226	027554	011605			MOV	(SP),R5	:AND PUT IT IN R5
5227	027556	006126			ROL	(SP)+	:WAS THERE AN ERROR?
5228	027560	100407			BMI	15	:BR IF ERROR
5229	027562	105761	024056		TSTB	DRVACT(R1)	:CHECK DRIVE'S STATE
5230	027566	003137			BGT	SC11	:BR IF DRIVE ACTIVE WITH ORDER
5231	027570	052762	100210 000016		BIS	#BIT15!BIT07!BIT03,16(R2)	:INFORM USER OF ERROR RECOVER COMPLETION
5232	027576	000470			BR	SC7	
5233	027600	004037	031012	15:	JSR	RD,RD.RM	:READ ERROR REGISTER #1

5234	027604	000014				RMER1		
5235	027606	030036				SC8		
5236	027610	012605				MOV	(SP)+,R5	;AND SAVE IT IN R5
5237	027612	004737	031364			JSR	PC,SVRH70	;SAVE RH70/RMO3 REGISTERS
5238	027616	012746	000111			MOV	#11,-(SP)	;ISSUE A DRIVE CLEAR
5239	027622	004037	031172			JSR	RO,WRT.RM	
5240	027626	000000				RMCS1		
5241	027630	030036				SC8		
5242	027632	006105			SC6A:	ROL	R5	;WAS "UNSAFE" CONDITION =1?
5243	027634	100406				BMI	1\$	;BRANCH IF YES
5244	027636	005702				TST	R2	;ANYTHING IN QUEUE ?
5245	027640	001447				BEQ	SC7	;BR IF NOT
5246	027642	052762	100240	000016		BIS	#BIT15!BIT07!BIT05,16(R2)	;INFORM USER OF ERROR
5247	027650	000443				BR	SC7	
5248	027652	004037	031012		1\$:	JSR	RO,RD.RM	;READ DRIVE STATUS REG. #1
5249	027656	000012				RMD5		
5250	027660	030036				SC8		
5251	027662	011605				MOV	(SP),R5	;SAVE RMD5 IN R5
5252	027664	006126				ROL	(SP)+	;ERR=1?
5253	027666	100011				BPL	2\$	;BR IF NO--UNSAFE CLEARED
5254	027670	112761	177777	024066		MOVB	#-1,DRVSTA(R1)	;DRIVE IS UNSAFE
5255	027676	004737	031364			JSR	PC,SVRH70	;SAVE RH70/RMO3 REGISTERS
5256	027702	052762	110000	000016		BIS	#BIT15!BIT12,16(R2)	;INFORM USER OF UNSAFE ERROR
5257	027710	000423				BR	SC7	
5258	027712	032705	010000		2\$:	BIT	#BIT12,R5	;MOL = 1 ?
5259	027716	001015				BNE	3\$	;BR IF YES
5260	027720	112761	177777	024056		MOVB	#-1,DRVACT(R1)	;ACTIVE ERROR RECOVER
5261	027726	112761	000001	024066		MOVB	#1,DRVSTA(R1)	;ONLINE
5262	027734	006301				ASL	R1	
5263	027736	012761	072460	024160		MOV	#30000.,TIMER(R1)	;START 30 SECOND TIMER
5264	027744	006201				ASR	R1	
5265	027746	000137	027310			JMP	SC4	
5266	027752	052762	100220	000016	3\$:	BIS	#BIT15!BIT07!BIT04,16(R2)	;INFORM USER OF ERROR
5267	027760	105061	024056		SC7:	CLRB	DRVACT(R1)	;DRIVE IS IDLE
5268	027764	004737	032124			JSR	PC,EMPTYQ	;DUMP THE QUEUE
5269	027770	105761	024134			TSTB	ULDFLG(R1)	;UNLOAD IN PROGRESS OR QUEUE?
5270	027774	003002				BGT	1\$	;BR IF NOT
5271	027776	105061	024134			CLRB	ULDFLG(R1)	;CLEAR UNLOAD FLAG
5272	030002	116164	024202	000016	1\$:	MOVB	ATABIT(R1),RMAS(R4)	;CLEAR ATTENTION BIT
5273	030010	105761	024066			TSTB	DRVSTA(R1)	;IS THE DRIVE UNSAFE ?
5274	030014	100406				BMI	2\$	;BR IF IT IS
5275	030016	012746	000113			MOV	#113,-(SP)	;RELEASE COMMAND
5276	030022	004037	031172			JSR	RO,WRT.RM	;WRITE THE COMMAND INTO RPCS1
5277	030026	000000				RMCS1		;REGISTER INDEX
5278	030030	030036				SC8		;PARITY EXIT ADDRESS
5279	030032	000137	027310		2\$:	JMP	SC4	;CHECK FOR MORE DRIVES
5280	030036	105761	024056		SC8:	TSTB	DRVACT(R1)	;IS DRIVE IDLE?
5281	030042	001405				BEQ	1\$	;YES--BRANCH
5282	030044	004737	032220			JSR	PC,GETREQ	;GET DPB POINTER
5283	030050	004737	026310			JSR	PC,C17	;PROCESS THE PARITY ERROR
5284	030054	000402				BR	2\$	;CONTINUE
5285	030056	004737	026336		1\$:	JSR	PC,C17B	;PROCESS THE UNCORRECTABLE PARITY ERROR
5286	030062	000137	027310		2\$:	JMP	SC4	;CHECK MORE DRIVES
5287	030066	105761	024134		SC11:	TSTB	ULDFLG(R1)	;UNLOAD IN PROGRESS?
5288	030072	003402				BLE	1\$	;BRANCH IF NO
5289	030074	105061	024134			CLRB	ULDFLG(R1)	;CLEAR UNLOAD FLAG

H08

 MAINDEC-11-DZRNA, RMO3 FORMATTER
 DZRNA8.P11 21-JUL-77 15:32

 MACY11 30(1046) 21-JUL-77 16:06 PAGE 98
 SINGLE/DUAL PORT RH70/RMO3 DRIVER (REV 1.0)

SEQ 0097

```

5290 030100 105061 024056 1$: CLRB DRVACT(R1) ;SET DRIVE IDLE
5291 030104 136137 024202 024130 BITB ATABIT(R1),SRCHWT ;DOING A SEARCH OPERATION FOR
5292 ;AN I/O COMMAND?
5293 030112 001012 BNE 2$ ;BRANCH IF YES
5294 030114 004737 032242 JSR PC,POPQUE ;REMOVE REQUEST FROM QUEUE
5295 030120 052762 000200 000016 BIS #BIT07,16(R2) ;SET "DONE" BIT
5296 030126 005737 024154 TST SAVEFG ;SAVE THE REGISTERS?
5297 030132 100002 BPL 2$ ;BRANCH IF NO
5298 030134 004737 031364 JSR PC,SVRH70 ;YES--SAVE ALL OF THE RH70/RMO3 REG'S
5299 030140 116164 024202 000016 2$: MOVB ATABIT(R1),RMAS(R4) ;CLEAR ATTENTION BIT
5300 030146 146137 024202 024130 BICB ATABIT(R1),SRCHWT ;CLEAR IMPLIED SEEK SET
5301 030154 006301 ASL R1 ;WORD INDEX
5302 030156 012761 177777 024160 MOV #1,TIMER(R1) ;STOP CLOCK
5303 030164 006201 ASR R1 ;RESTORE R1
5304 030166 004737 025230 JSR PC,OPT ;START A REQUEST
5305 030172 000137 027310 JMP SC4 ;CHECK FOR MORE DRIVES
5306 030176 010164 000010 SC12: MOV R1,RMCS2(R4) ;SELECT DRIVE
5307 030202 016437 000012 024046 MOV RMO5(R4),RMERRS ;SAVE THE FOUR REGISTERS THAT
5308 030210 016437 000014 024050 MOV RMER1(R4),RMERRS+2 ;WILL TELL US SOMETHING
5309 030216 016437 000042 024052 MOV RMER2(R4),RMERRS+4
5310 03 124 016437 000040 024054 MOV RMMR2(R4),RMERRS+6
5311 030232 004037 024444 JSR RC,DRVINT ;INIT. THE STATE OF THE DRIVE
5312 030236 000401 BR 1$ ;TAKE ERROR EXIT
5313 030240 000207 RTS PC ;RETURN
5314 030242 005726 1$: TST (SF)+ ;POP PC OFF OF THE STACK
5315 030244 000674 SC8 ;PROCESS THE PARITY ERROR
5316 030246 006301 SC13: ASL R1 ;SETUP TO ADDRESS WORDS
5317 03 250 012761 177777 024160 MOV #1,TIMER(R1) ;STOP THE TIMER
5318 030256 006201 ASR R1
5319 030260 010164 000010 MOV R1,RMCS2(R4) ;SELECT THE DRIVE
5320 03 64 116164 024202 000016 MOVB ATABIT(R1),RMAS(R4) ;CLEAR THE ATTENTION BIT
5321 030272 105761 024106 1$: TSTB DPINT(R1) ;INITIALIZING THE DRIVE ?
5322 030276 001424 BEQ 2$ ;BR IF NOT
5323 03 300 105061 024106 CLRB DPINT(R1) ;CLEAR THE INIT INDICATOR
5324 030304 004037 024444 JSR RC,DRVINT ;GO INIT THE DRIVE
5325 030310 000240 NOP ;DUMMY PARITY ERROR RETURN
5326 030312 105761 024066 TSTB DRVSTA(R1) ;DRIVE ONLINE ?
5327 030316 003014 BGT 2$ ;BR IF YES -- START ORDER
5328 030320 005702 TST R2 ;QUEUE ENTRY FOR THE DRIVE
5329 030322 001426 BEQ 3$ ;BR IF NOT
5330 030324 004737 032220 JSR PC,GETREQ ;GET DPB ADDRESS
5331 030330 052762 140000 000016 BIS #BIT15:BIT14,16(R2) ;INFORM USER THAT DRIVE OFFLINE
5332 030336 004737 031364 JSR PC,SVRH70 ;SAVE THE REGISTERS
5333 030342 004737 032124 JSR PC,EMPTYQ ;EMPTY THE REQUEST QUEUE
5334 030346 000414 BR 3$
5335 030350 032764 004000 000000 2$: BIT #BIT11,RMCS1(R4) ;DVA SET ?
5336 030356 001006 BNE 4$ ;SET THEN CALL OPT
5337 030360 006301 ASL R1
5338 030362 012761 023420 024160 MOV #10000.,TIMER(R1)
5339 030370 006201 ASR R1
5340 030372 000402 BR 3$
5341 030374 004737 025230 4$: JSR PC,OPT ;START THE PENDING REQUEST
5342 030400 000137 027310 3$: JMP SC4 ;PROCESS OTHER DRIVES
5343
5344 ;/RMO3 TIMER ROUTINE
5345 ;CALL

```

MAINDEC-11-DZRNA, RMO3 FORMATTER
DZRNA8.P11 21-JUL-77 15:32

MACY11 30(1046) 21-JUL-77 16:06 PAGE 99
SINGLE/DUAL PORT RH70/RMO3 DRIVER (REV 1.0)

SEQ 0098

```

5346      ;      MOV      #TIME -(SP)      ;ELAPSED TIME IN MILLISECONDS ON THE STACK
5347      ;      JSR      PC,RPTMR          ;CALL RMO3 TIME ROUTINE
5348
5349 030404 005737 024132      RPTMR: TST      ACTDRV          ;CHECK "ACTDRV & ACTSTR"
5350 030410 001027              BNE      4$              ;IF NON ZERO EXIT
5351 030412 112737 000001 024133      MOV      #1,ACTSTR      ;SET "ACTSTR"
5352 030420 104412              SAVREG          ;SAVE R0 - R5
5353 030422 005001              CLR      R1              ;START WITH DRIVE 0
5354 030424 005003              CLR      R3
5355 030426 005763 024160      1$: TST      TIMER(R3)        ;IS THE TIMER RUNNING?
5356 030432 002406              BLT      2$              ;BRANCH IF NO
5357 030434 166663 000002 024160      SUB      2(SP),TIMER(R3) ;COUNT THE INTERVAL
5358 030442 003602              BGT      2$              ;BR IF NO SOFTWARE TIMEOUT
5359 030444 004737 030474      JSR      PC,ST0          ;CALL SOFTWARE TIMEOUT ROUTINE
5360 030450 005201      2$: INC      R1              ;MOVE TO NEXT DRIVE
5361 030452 005723              TST      (R3)+
5362 030454 022701 000010      CMP      #8,R1              ;OUT OF DRIVES?
5363 030460 003362              BGT      1$              ;BRANCH IF NO
5364 030462 104413      3$: RESREG          ;RESTORE R0 - R5
5365 030464 105037 024133      CLR      ACTSTR          ;ZERO ACTIVE SOFTWARE TIMEOUT ROUTINE FLAG
5366 030470 012616      4$: MOV      (SP)+,(SP)      ;ADJUST THE STACK
5367 030472 000207      RTS      PC              ;RETURN
5368
5369      ; SOFTWARE TIMEOUT ROUTINE
5370
5371      ; NOTE: THIS ROUTINE MUST BE ENTERED AT PRIORITY 6
5372      ; OR GREATER
5373
5374      ; CALL: ST0
5375      ; MOV      #DRVNUM,R1          ;DRIVE NUMBER
5376      ; JSR      PC,ST0              ;CALL
5377      ; RETURN
5378
5379 030474 010146      ST0: MOV      R1,-(SP)          ;SAVE R1
5380 030476 010246      MOV      R2,-(SP)          ;SAVE R2
5381 030500 010346      MOV      R3,-(SP)          ;SAVE R3
5382 030502 010446      MOV      R4,-(SP)          ;SAVE R4
5383 030504 013704 024214      MOV      RADDR,R4          ;GET ADDRESS OF "RMCS1"
5384 030510 010164 000010      MOV      R1,RMCS2(R4)        ;SELECT THE DRIVE
5385 030514 004037 031012      JSR      R0,RD.RM          ;READ "DRIVE STATUS REG"
5386 030520 000012      RMO3
5387 030522 030674      STOS
5388 030524 105726      TST      (SP)+
5389 030526 100434      BMI      ST02
5390 030528 105761 024106      ST01: TST      DPINT(R1)        ;TRYING TO INITIALIZE THE DRIVE ?
5391 030534 001031      BNE      ST02
5392 030536 105761 024116      TST      DPRQS(R1)        ;OUTSTANDING PORT REQUEST FOR THE DRIVE ?
5393 030542 001026      BNE      ST02
5394 030544 013702 024126      MOV      TRNSWT,R2        ;PICKUP TRANSFER WAIT QUEUE
5395 030550 020137 024200      CMP      R1,DTUW          ;TRANSFER UNDERWAY ON THIS DRIVE?
5396 030554 001404      BEQ      1$
5397 030556 000137 031000      JMP      ST09
5398 030562 004737 032220      JSR      PC,GETREQ        ;GET DPB ADDRESS
5399 030566 052762 101000 000016 1$: BIS      #BIT15!BIT09,16(R2) ;SET THE ERROR FLAGS
5400 030574 004737 031364      JSR      PC,SVRH70        ;SAVE RH70/RMO3 REGISTERS
5401 030600 012764 000040 000010      MOV      #BIT05,RMCS2(R4) ;"INIT" THE MASS BUS

```

5402	030606	105061	024056		CLRB	DRVACT(R1)	;DRIVE IS IDLE
5403	030612	105061	024134		CLRB	ULDFLG(R1)	;CLEAR THE UNLOAD FLAG
5404	030616	000470			BR	ST09	;DON'T BOTHER OTHER DRIVES
5405	030620	116405	000016	ST02:	MOVB	RMA5(R4),R5	;READ ATTENTION REG
5406	030624	136105	024202		BITB	ATABIT(R1),R5	;IS ATTENTION FOR THIS DRIVE UP ?
5407	030630	001007			BNE	ST03	;YES--BRANCH
5408	030632	105761	024106		TSTB	DPINT(R1)	;TRYING TO INITIALIZE THE DRIVE ?
5409	030636	001021			BNE	ST06	;BR IF YES - DRIVE NOT ONLINE
5410	030640	105761	024116		TSTB	DPRQS(R1)	;OUTSTANDING PORT REQUEST FOR THE DRIVE ?
5411	030644	001035			BNE	ST07	;BR IF YES - NO RESPONSE TO REQUEST
5412	030646	000454			BR	ST09	;OTHER WISE EXIT
5413	030650	105761	024106	ST03:	TSTB	DPINT(R1)	;INITIALIZING THE DRIVE ?
5414	030654	001003			BNE	IS	;BR IF INIT PENDING
5415	030656	105761	024116		TSTB	DPRQS(R1)	;PORT REQUEST PENDING ?
5416	030660	001446			BEQ	ST09	;BR IF NOT
5417	030664	012763	177777	024160	IS:	MOV	#-1,TIMER(R3)
5418	030672	000442			BR	ST09	;EXIT
5419	030674	004737	026410	ST05:	JSR	PC,CIB	;GO HANDLE THE PARITY ERROR
5420	030700	000437			BR	ST09	
5421	030702	105061	024106	ST06:	CLRB	DPINT(R1)	;CLEAR THE INITIALIZE INDICATOR
5422	030706	105061	024066		CLRB	DRVSTA(R1)	;SET UNIT OFFLINE
5423	030712	012763	177777	024160	MOV	#-1,TIMER(R3)	;STOP THE TIMER
5424	030720	004737	032220		JSR	PC,GETREQ	;GET THE DPB ADDRESS
5425	030724	005702			TST	R2	;REQUEST IN QUEUE ?
5426	030726	001424			BEQ	ST09	;BR IF NOT
5427	030730	052762	140000	000016	BIS	#BIT15:BIT14,16(R2)	;INFORM THE USER DRIVE NOT AVAILABLE
5428	030736	000414			BR	ST08	;FINISH
5429	030740	012763	177777	024160	ST07:	MOV	#-1,TIMER(R3)
5430	030746	105061	024116		CLRB	DPRQS(R1)	;CLEAR PORT REQUEST INDICATOR
5431	030752	004737	032220		JSR	PC,GETREQ	;GET DPB ADDRESS
5432	030756	005702			TST	R2	;QUEUE ENTRY FOR DRIVE ?
5433	030760	001407			BEQ	ST09	;BR IF NONE
5434	030762	012762	100004	000016	MOV	#BIT15:BIT2,16(R2)	;INFORM USER OF PORT REQUEST ERROR
5435	030770	004737	032124	ST08:	JSR	PC,EMPTYQ	;CLEAR THE QUEUE FOR THE DRIVE
5436	030774	004737	031364		JSR	PC,SVRH70	;SAVE THE REGISTERS
5437	030800	012604		ST09:	MOV	(SP)+,R4	;RESTORE R4
5438	030802	012603			MOV	(SP)+,R3	;RESTORE R3
5439	031004	012602			MOV	(SP)+,R2	;RESTORE R2
5440	031006	012601			MOV	(SP)+,R1	;RESTORE R1
5441	031010	000207			RTS	PC	;RETURN
5442							
5443							
5444							
5445					CALL		
5446					JSR	RO,RD.RM	;GO READ A REGISTER
5447					INDEX		;REG. INDEX FROM BASE
5448					ERRADR		;ERROR ADDRESS--PROCESS ERROR STARTING
5449							;AT THIS ADDRESS
5450					RETURN		;CONTENTS OF REG. IS ON THE STACK
5451							
5452	031012	013737	024212	031160	RD.RM:	MOV	MCPEMX,RD.RM2
5453	031020	011646			MOV	(SP)-,(SP)	;MAX. RETRY ALLOWED
5454	031022	013737	024214	031036	MOV	RMA0,RD.ADR	;SAVE RO FOR RETURN
5455	031030	062037	031036		ADD	(RD)+,RD.ADR	;FORM THE DESIRED ADDRESS
5456	031034	013727			RD.RM1:	MOV	2(PC)+,(PC)+
5457	031036	000000			RD.ADR:	.WORD	0

K08

MAINDEC-11-DZMA, RM03 FORMATTER
DZMA8.P11 21-JUL-77 15:32

MACY11 30(1046) 21-JUL-77 16:06 PAGE 101
SINGLE/DUAL PORT RH70/RM03 DRIVER (REV 1.0)

SEQ 0100

5458	031040	000000			RD.WRD: .WORD 0	; REG. CONTENTS PUT HERE
5459	031042	013766	031040	000002	MOV RD.WRD, 2(SP)	; RETURN IT TO THE USER
5460	031050	013746	024214		MOV RMADR, -(SP)	; PUT THE ADDRESS ON THE STACK
5461	031054	062716	000010		ADD #RMCS2, (SP)	; FORM THE ADDRESS OF RMCS2
5462	031060	032736	010000		BIT #BIT12, 2(SP)+	; CHECK THE 'MED' BIT
5463	031064	001037			BNE RD.RM3	; BR IF DRIVE NON-EXISTENT
5464	031066	017746	173122		MOV 2RMADR, -(SP)	; READ RMCS1
5465	031072	032716	020000		BIT #BIT13, (SP)	; DID MCPE SET?
5466	031076	001002			BNE 1\$	; BRANCH IF YES
5467	031100	022620			CMP (SP)+, (RD)+	; ADJUST FOR RETURN
5468	031102	000432			BR RD.RM4	; EXIT
5469	031104					
5470	031104	004037	032310		1\$: JSR RD, ES.SAV	; SAVE THE ADDRESS IN 'SESCAPE'
5471	031110	104003			ERROR 3	; REPORT "MCPE" ERROR
5472	031112	005737	024200		TST DTUW	; DATA TRANSFER UNDERWAY?
5473	031116	100405			BMI 2\$	; NO--BRANCH
5474	031120	032716	040000		BIT #BIT14, (SP)	; NO--"TRE"=1?
5475	031124	001402			BEQ 2\$	; NO--BRANCH
5476	031126	005726			TST (SP)+	; YES--CLEAN OFF THE STACK AND
5477	031130	000415			BR RD.RM3	; TAKE THE FATAL ERROR EXIT
5478	031132	052716	040000		2\$: BIS #BIT14, (SP)	; CLEAR "MCPE" BY SENDING A "1" TO "TRE"
5479	031136	000316			SWAB (SP)	; POSITION BEFORE WRITING
5480	031140	013737	024214	031154	MOV RMADR, 3\$	; FORM ADDRESS OF HIGH BYTE
5481	031146	005237	031154		INC 3\$	
5482	031152	112637			MOVB (SP)+, 2(PC)+	; WRITE THE HIGH BYTE OF RMCS1
5483	031154	000000			3\$: .WORD 0	; ADDRESS STORAGE
5484	031156	005327			DEC (PC)+	; EXCEEDED MAX. RETRYS
5485	031160	000003			RD.RM2: .WORD 3	
5486	031162	002324			BGE RD.RM1	; BRANCH IF NO
5487	031164	011000			RD.RM3: MOV (RD), RD	; FATAL ERROR EXIT
5488	031166	012616			MOV (SP)+, (SP)	
5489	031170	000200			RD.RM4: RTS RD	
5490						
5491					; ROUTINE TO WRITE A REGISTER	
5492					; CALL	
5493					MOV DATA, -(SP)	; DATA TO BE LOADED ON THE STACK
5494					JSR RD, WRT.RM	; CALL THE ROUTINE TO LOAD(WRITE) THE REG.
5495					INDEX	; INDEX OF THE REGISTER TO BE LOADED
5496					ERRADR	; ADDRESS TO RETURN TO ON AN ERROR
5497					RETURN	; ERROR FREE RETURN
5498						
5499						
5500	031172	013737	024212	031350	WRT.RM: MOV MCPEMX, WRT.R2	; MAX RETRYS ALLOWED
5501	031200	016637	000002	031260	MOV 2(SP), WRT.WD	; SAVE THE WORD TO WRITE
5502	031206	012616			MOV (SP)+, (SP)	; ADJUST THE STACK
5503	031210	012037	031262		MOV (RD)+, WRT.AD	; GET INDEX OF REGISTER TO BE WRITTEN
5504	031214	001015			BNE 1\$	; BRANCH IF NOT RMCS1
5505	031216	122737	000150	031260	CMPB #150, WRT.WD	; IS THE COMMAND FOR DATA TRANSFERS?
5506	031224	002411			BLT 1\$	; YES--DON'T GET THE OLD A16 & A17, & PSEL
5507	031226	004037	031012		JSR RD, RD.RM	; NO---COMBINE A16&A17, & PSEL WITH
5508	031232	000000			RMCS1	; THE COMMAND BEFORE SENDING IT TO
5509	031234	031354			WRT.R3	; THE RH70/RM03
5510	031236	000316			SWAB (SP)	
5511	031240	042716	177770		BIC #C7, (SP)	
5512	031244	112637	031261		MOVB (SP)+, WRT.WD+1	
5513	031250	063737	024214	031262	1\$: ADD RMADR, WRT.AD	; FORM THE ADDRESS OF THE DISK REG.

L08

MAINDEC-11-DZRNA, RM03 FORMATTER
DZRNA8.P11 21-JUL-77 15:32

MACY11 30(1046) 21-JUL-77 16:06 PAGE 102
SINGLE/DUAL PORT RH70/RM03 DRIVER (REV 1.0)

SEQ 0101

5514	031256	012737				WRT.R1: MOV	(PC)+, 2(PC)+	; LOAD THE DESIRED REG.
5515	031260	000000				WRT.WD: .WORD	0	; WORD TO WRITE GOES HERE
5516	031262	000000				WRT.AD: .WORD	0	; ADDRESS IS FORMED HERE
5517	031264	013746	024214			MOV	RMADR, -(SP)	; PUT THE ADDRESS ON THE STACK
5518	031270	062716	000010			ADD	#RMCS2, (SP)	; FORM THE ADDRESS OF RMCS2
5519	031274	032736	010000			BIT	#BIT12, 2(SP)+	; CHECK THE 'NED' BIT
5520	031300	001025				BNE	WRT.R3	; BR IF DRIVE NON-EXISTENT
5521	031302	004037	031012			JSR	RO, RD.RM	; CHECK FOR PARITY ERROR ON WRITE
5522	031306	000014				RMER1		
5523	031310	031354				WRT.R3		
5524	031312	032726	000010			BIT	#BIT03, (SP)+	
5525	031316	001420				BEQ	WRT.R4	; BRANCH IF "PAR=0"
5526	031320	016037	177776	031332		MOV	-2(RO), 1\$	; PICKUP THE INDEX
5527	031326	004037	031012			JSR	RO, RD.RM	; READ THE REG.
5528	031332	000000				1\$: .WORD	0	; REG. INDEX
5529	031334	031354				WRT.R3		; RETURN TO THIS ADDRESS ON ERROR
5530	031336	004037	032310			JSR	RO, ES.SAV	; SAVE THE ADDRESS IN 'SESCAPE'
5531	031342	104004				ERROR	4	; REPORT THE PARITY ON WRITE ERROR
5532	031344	005726				TST	(SP)+	; CLEAR OFF THE STACK
5533	031346	005327				DEC	(PC)+	; DECREMENT THE ERROR COUNT
5534	031350	000003				WRT.R2: .WORD	3	; RETRY COUNTER
5535	031352	002341				BGE	WRT.R1	; TRY AGAIN IF NOT FINISHED
5536	031354	011000				WRT.R3: MOV	(RO), RO	; TAKE THE "PARITY ON WRITE" ERROR EXIT
5537	031356	000401				BR	WRT.R5	; EXIT
5538	031360	005720				WRT.R4: TST	(RO)+	; ADJUST FOR ERROR FREE EXIT
5539	031362	000200				WRT.R5: RTS	RO	
5540								
5541								; ROUTINE TO SAVE THE RH70/RP04/5/RM03 REGISTERS AS PER DPB+14
5542								
5543								
5544								
5545								
5546								
5547	031364	104412				SVRH70: SAVREG		; SAVE RD - R5
5548	031366	005702				TST	R2	; QUEUE ENTRY FOR THE DRIVE ?
5549	031370	001442				BEQ	6\$	; BR IF NONE
5550	031372	013704	024214			MOV	RMADR, R4	
5551	031376	111264	000010			MOVB	(R2), RMCS2(R4)	; SELECT DRIVE
5552	031402	016203	000014			MOV	14(R2), R3	; GET THE ERROR TABLE POINTER
5553	031406	001433				BEQ	6\$	; EXIT IF NO ADDRESS
5554	031410	005037	031444			CLR	3\$	; COUNTER & POINTER
5555	031414	023727	031444	000022		1\$: CMP	3\$, #RMD8	; REACHED THE BUFFER REGISTER ?
5556	031422	001006				BNE	2\$	; BR IF NOT
5557	031424	032764	000200	000010		BIT	#BIT07, RMCS2(R4)	; 'OR' SET ?
5558	031432	001002				BNE	2\$	; BR IF SET
5559	031434	005023				CLR	(R3)+	; STORE RMD8 AS ZEROES
5560	031436	000405				BR	4\$	; CONTINUE
5561	031440	004037	031012			2\$: JSR	RO, RD.RM	; READ THE SELECTED REGISTER
5562	031444	000000				3\$: .WORD	0	; REGISTER INDEX
5563	031446	031472				5\$		; ERROR RETURN ADDRESS
5564	031450	012623				MOV	(SP)+, (R3)+	; STORE THE REGISTER CONTENTS
5565	031452	023727	031444	000046		4\$: CMP	3\$, #RMEC2	; REACHED THE END ?
5566	031460	001406				BEQ	6\$	; BR IF YES
5567	031462	062737	000002	031444		ADD	#2, 3\$	; INCREMENT THE REGISTER INDEX
5568	031470	000751				BR	1\$	; CONTINUE READING THE REGISTERS
5569	031472	004737	026310			5\$: JSR	PC, CI7	; PROCESS THE UNCORRECTABLE PARITY ERROR

M08

MAINDEC-11-DZRNA, RM03 FORMATTER
DZRNA8.P11 21-JUL-77 15:32

MACY11 30(1046) 21-JUL-77 16:06 PAGE 103
SINGLE/DUAL PORT RH70/RM03 DRIVER (REV 1.0)

SEQ 0102

5570 031476 104413
5571 031500 000207
5572
5573
5574
5575
5576
5577
5578
5579 031502 010446
5580 031504 013704 024214
5581 031510 010164 000010
5582 031514 011446
5583 031516 052716 040000
5584 031522 000316
5585 031524 112714 000100
5586 031530 032764 010000 000010
5587 031536 001002
5588 031540 005726
5589 031542 000402
5590 031544 112664 000001
5591 031550 012604
5592 031552 000207
5593
5594
5595 031554 000
5596 031555 000
5597 031556 000
5598 031557 000
5599 031560 000
5600 031561 000
5601 031562 000
5602 031563 000
5603
5604
5605
5606 031564 031646
5607 031566 031666
5608 031570 031706
5609 031572 031726
5610 031574 031746
5611 031576 031766
5612 031600 032006
5613 031602 032026
5614
5615
5616
5617 031604 031646
5618 031606 031666
5619 031610 031706
5620 031612 031726
5621 031614 031746
5622 031616 031766
5623 031620 032006
5624 031622 032026
5625

6S: RESREG ;RESTORE R0 - R5
RTS PC ;RETURN
;ROUTINE TO SET THE INTERRUPT WITHOUT GETTING A "TRE"
;CALL
;MOV #DRVNUM,R1 ;DRIVE NUMBER TO R1
;JSR PC,SET.IE ;SET "IE"
;RETURN
SET.IE: MOV R4, -(SP) ;SAVE R4
MOV RMAOR, R4 ;PICKUP ADDRESS OF RMCS1
MOV R1, RMCS2(R4) ;SELECT DRIVE
MOV (R4), -(SP) ;READ RMCS1
BIS #BIT14, (SP) ;SET THE "TRE" BIT OF THE WORD READ
SWAB (SP) ;ADJUST FOR DAT0
MOVB #BIT06, (R4) ;SET "IE"
BIT #BIT12, RMCS2(R4) ;IS "NED"=1?
BNE 1S ;YES--CLEAR "TRE"
TST (SP)+ ;CLEAN OFF THE STACK
BR 2S
1S: MOVB (SP)+, 1(R4) ;CLEAR "TRE"
2S: MOV (SP)+, R4 ;RESTORE R4
RTS PC ;RETURN TO CALLER
;QUEUE COUNT
QCN: .BYTE 0 ;DRIVE 0
; .BYTE 0 ;DRIVE 1
; .BYTE 0 ;DRIVE 2
; .BYTE 0 ;DRIVE 3
; .BYTE 0 ;DRIVE 4
; .BYTE 0 ;DRIVE 5
; .BYTE 0 ;DRIVE 6
; .BYTE 0 ;DRIVE 7
;QUEUE INPUT POINTERS
QINPT: .WORD QDRV0 ;DRIVE 0
; .WORD QDRV1 ;DRIVE 1
; .WORD QDRV2 ;DRIVE 2
; .WORD QDRV3 ;DRIVE 3
; .WORD QDRV4 ;DRIVE 4
; .WORD QDRV5 ;DRIVE 5
; .WORD QDRV6 ;DRIVE 6
; .WORD QDRV7 ;DRIVE 7
;QUEUE OUTPUT POINTERS
QOUTPT: .WORD QDRV0 ;DRIVE 0
; .WORD QDRV1 ;DRIVE 1
; .WORD QDRV2 ;DRIVE 2
; .WORD QDRV3 ;DRIVE 3
; .WORD QDRV4 ;DRIVE 4
; .WORD QDRV5 ;DRIVE 5
; .WORD QDRV6 ;DRIVE 6
; .WORD QDRV7 ;DRIVE 7

MAINDEC-11-DZRNA, RMO3 FORMATTER
DZRNA8.P11 21-JUL-77 15:32

N08

MACY11 30(1046) 21-JUL-77 16:06 PAGE 104
SINGLE/DUAL PORT RH70/RMO3 DRIVER (REV 1.0)

SEQ 0103

5626 031624 031646
5627 031626 031666
5628 031630 031706
5629 031632 031726
5630 031634 031746
5631 031636 031766
5632 031640 032006
5633 031642 032026
5634 031644 032046

```
QSTART: .WORD QDRV0      ;DRIVE 0 START ADDRESS
QSTOP:  .WORD QDRV1      ;DRIVE 0 STOP ADDRESS & DRIVE 1 START ADDRESS
        .WORD QDRV2      ;STOP DRIVE 1--START DRIVE 2
        .WORD QDRV3      ;STOP DRIVE 2--START DRIVE 3
        .WORD QDRV4      ;STOP DRIVE 3--START DRIVE 4
        .WORD QDRV5      ;STOP DRIVE 4--START DRIVE 5
        .WORD QDRV6      ;STOP DRIVE 5--START DRIVE 6
        .WORD QDRV7      ;STOP DRIVE 6--START DRIVE 7
        .WORD QTERM      ;STOP DRIVE 7
```

;DRIVE REQUEST QUEUES

5635
5636
5637
5638 031646 000010
5639 031666 000010
5640 031706 000010
5641 031726 000010
5642 031746 000010
5643 031766 000010
5644 032006 000010
5645 032026 000010
5646 032046

```
QDRV0:  .BLKW 10
QDRV1:  .BLKW 10
QDRV2:  .BLKW 10
QDRV3:  .BLKW 10
QDRV4:  .BLKW 10
QDRV5:  .BLKW 10
QDRV6:  .BLKW 10
QDRV7:  .BLKW 10
QTERM=.
```

;ROUTINE TO CLEAR ALL OF THE REQUEST QUEUES

5647
5648
5649
5650
5651
5652
5653 032046 104412
5654 032050 012702 031554
5655 032054 005022
5656 032056 005022
5657 032060 005022
5658 032062 005022
5659 032064 012703 000010
5660 032070 012701 031624
5661 032074 012122
5662 032076 005303
5663 032100 001375
5664 032102 012703 000010
5665 032106 012701 031624
5666 032112 012122
5667 032114 005303
5668 032116 001375
5669 032120 104413
5670 032122 000207

```
;CALL
      JSR      PC,CLRQUE
CLRQUE: SAVREG      ;SAVE R0 - R5
      MOV      #QCNT,R2      ;ZERO THE QUEUE COUNTS
      CLR      (R2)+          ;DRIVES 0 & 1
      CLR      (R2)+          ;DRIVES 2 & 3
      CLR      (R2)+          ;DRIVES 4 & 5
      CLR      (R2)+          ;DRIVES 6 & 7
      MOV      #R3,R3        ;MOVE THE STARTING
      MOV      #QSTART,R1    ;ADDRESS OF THE QUEUE INTO
1$:    MOV      (R1)+,(R2)+   ;THE QUEUE INPUT POINTER
      DEC      R3
      BNE      1$
      MOV      #R3,R3        ;MOVE THE STARTING ADDRESS
      MOV      #QSTART,R1    ;OF THE QUEUE INTO THE
2$:    MOV      (R1)+,(R2)+   ;QUEUE OUTPUT POINTER
      DEC      R3
      BNE      2$
      RESREG
      RTS      PC            ;RESTORE R0 - R5
```

;EMPTY THE QUEUE SPECIFIED BY R1

5671
5672
5673
5674
5675
5676
5677
5678 032124 105061 031554
5679 032130 006301
5680 032132 016161 031564 031604
5681 032140 006201

```
;CALL
      MOV      DRVNUM,R1      ;DRIVE NUMBER TO R1
      JSR      PC,EMPTYQ
EMPTYQ: CLRB      QCNT(R1)    ;CLEAR NUMBER OF ITEMS IN QUEUE
      ASL      R1
      MOV      QINPT(R1),QOUTPT(R1) ;SET OUTPUT QUEUE POINTER=INPUT POINTER
      ASR      R1
```

```

5682 032142 000207          RTS    PC
5683
5684          ;ROUTINE TO PUT A REQUEST IN QUEUE
5685          ;CALL
5686          ;      MOV    #DRVNUM,R1      ;DRIVE NUMBER
5687          ;      MOV    #DPB,R2        ;ADDRESS OF PARAMETER BLOCK
5688          ;      JSR     RD,DRVQUE      ;GO PUT REQUEST IN QUEUE
5689          ;      RETURN1                ;RETURN HERE IF QUEUE IS FULL
5690          ;      RETURN2                ;RETURN HERE IF REQUEST IS IN QUEUE
5691
5692
5693 032144 122761 000010 031554 DRVQUE: CMPB    #10,QCNT(R1)    ;IS QUEUE FULL?
5694 032152 001421                BEQ     2$                ;BR IF YES-TAKE RETURN1
5695 032154 105261 031554                INCB    QCNT(R1)        ;INCREMENT QUEUE COUNT
5696 032160 006301                ASL     R1
5697 032162 010271 031564                MOV    R2,QOINPT(R1)    ;PUT THIS REQUEST IN QUEUE
5698 032166 062761 000002 031564                ADD    #2,QINPT(R1)    ;UPDATE THE QUEUE POINTER
5699 032174 026161 031564 031626                CMP    QINPT(R1),QSTOP(R1) ;TIME TO RESET THE POINTER
5700 032202 001003                BNE     1$                ;BRANCH IF NO
5701 032204 016161 031624 031564                MOV    QSTART(R1),QINPT(R1) ;YES--RESET POINTER
5702 032212 006201                1$:   ASR     R1
5703 032214 005720                TST     (R0)+                ;TAKE RETURN 2
5704 032216 000200                2$:   RTS     R0                ;RETURN TO USER
5705
5706          ;ROUTINE TO GET THE "DPB" ADDRESS OF NEXT REQUEST IN QUEUE
5707          ;CALL
5708          ;      MOV    #DRVNUM,R1      ;DRIVE NUMBER TO R1
5709          ;      JSR     PC,GETREQ      ;GO GET THE REQUEST
5710          ;      RETURN                ;R2="DPB" ADDRESS OF THE REQUEST
5711          ;      ;R2=0 IF NO REQUEST IN QUEUE
5712
5713
5714 032220 005002                GETREQ: CLR     R2
5715 032222 105761 031554                TSTB    QCNT(R1)        ;IS THERE ANY REQUEST IN QUEUE?
5716 032226 001404                BEQ     2$                ;NO---BRANCH
5717 032230 006301                1$:   ASL     R1
5718 032232 017102 031604                MOV    QOOUTPT(R1),R2    ;PICKUP "DPB" POINTER FOR THIS DRIVE
5719 032236 006201                ASR     R1
5720 032240 000207                2$:   RTS     PC                ;RETURN TO USER
5721
5722          ;ROUTINE TO "POP" THE REQUEST FROM QUEUE
5723          ;CALL
5724          ;      MOV    #DRVNUM,R1      ;DRIVE NUMBER TO R1
5725          ;      JSR     PC,POPQUE      ;CALL TO REMOVE REQUEST
5726          ;      RETURN                ;R2=ADDRESS OF DPB REMOVED
5727
5728
5729 032242 105361 031554                POPQUE: DECB    QCNT(R1)    ;DECREMENT QUEUE COUNT
5730 032246 006301                ASL     R1
5731 032250 017102 031604                MOV    QOOUTPT(R1),R2    ;GET THE "DPB" POINTER
5732 032254 005071 031604                CLR     QOOUTPT(R1)    ;REMOVE DPB ADDRESS FROM THE QUEUE
5733 032260 062761 000002 031604                ADD    #2,QOOUTPT(R1)    ;UPDATE THE QUEUE POINTER
5734 032266 026161 031604 031626                CMP    QOOUTPT(R1),QSTOP(R1) ;TIME TO RESET THE POINTER?
5735 032274 001003                BNE     1$                ;NO--BRANCH TO EXIT
5736 032276 016161 031624 031604                MOV    QSTART(R1),QOOUTPT(R1) ;YES--RESET THE POINTER
5737 032304 006201                1$:   ASR     R1

```

C09

MAINDEC-11-DZRNA, RMO3 FORMATTER
DZRNA8.P11 21-JUL-77 15:32

MACY11 30(1046) 21-JUL-77 16:06 PAGE 106
SINGLE/DUAL PORT RH70/RMO3 DRIVER (REV 1.0)

SEQ 0105

5738 032306 000207
5739
5740
5741
5742
5743
5744
5745
5746
5747
5748 032310 012037 032324
5749 032314 013746 001174
5750 032320 005037 001174
5751 032324 000000
5752 032326 012637 001174
5753 032332 000200
5754
5755
5756
5757

RTS PC ;RETURN TO USER

;ROUTINE TO SAVE THE CONTENTS OF 'SESCAPE' WHEN THE DRIVER
;REPORTS AN ERROR DIRECTLY.

```
CALL
      JSR      RO,ES.SFV
      ERROR    N
      RETURN
      ;THE ERROR CALL
      ;THE RETURN IS PAST THE ERROR CALL

ES.SAV: MOV      (RO)+,1$
        MOV      $ESCAPE,-(SP)
        CLR      $ESCAPE
1$:      .WORD    0
        MOV      (SP)+,$ESCAPE
        RTS      RO
      ;GET THE ERROR CALL
      ;SAVE THE ADDRESS IN 'SESCAPE'
      ;CLEAR THE ESCAPE RETURN
      ;THE ERROR CALL IS MOVED HERE
      ;RESTORE THE ESCAPE ADDRESS
      ;RETURN
```

;LIST BEX

.S8TTL TELETYPE MESSAGES

032334	047440	043106	044514	UNTOFF:	.ASCIZ	/ OFFLINE/
032345	040	047117	044514	UNTON:	.ASCIZ	/ ONLINE/
032355	040	047516	020124	NOTRM:	.ASCIZ	@ NOT AN RMO3@
032372	047040	052117	050040	NOTPRS:	.ASCIZ	/ NOT PRESENT/
032407	040	047125	040523	NOTSAF:	.ASCIZ	/ UNSAFE/
032417	122	030120	000064	RMO3B:	.ASCIZ	/RPO4/
032424	050122	032460	000	RPOS:	.ASCIZ	/RPO5/
032431	122	030115	000063	RMO3A:	.ASCIZ	/RMO3/
032436	047125	052111	051440	SYSTAT:	.ASCIZ	/UNIT STATUS/<CR><LF><LF>
032455	015	042012	044522	MUNIT:	.ASCIZ	<CR><LF>/DRIVE: /
032467	040	020057	000	SLASH:	.ASCIZ	@ / @
032473	103	000		C:	.ASCIZ	/C/
032475	124	000		T:	.ASCIZ	/T/
032477	060	000		MORVD:	.ASCIZ	/O/
032501	040	040		LIN4SP:	.ASCII	/ /
032503	040	000040		LINSP:	.ASCIZ	/ /
032506	047105	042524	020122	ENTADR:	.ASCIZ	/ENTER ADDRESS LIMITS:<CR><LF>
032536	020040	051104	053111	MOFFLN:	.ASCIZ	/ DRIVE OFFLINE/<CR><LF>
032560	042040	044522	042526	MORNP:	.ASCIZ	/ DRIVE NOT PRESENT/<CR><LF>
032605	040	051104	053111	MER11:	.ASCIZ	/ DRIVE NOT AVAILABLE/<CR><LF>
032634	042040	044522	042526	MNRMO3:	.ASCIZ	@ DRIVE NOT AN RMO3@<CR><LF>
032661	040	051104	053111	MUSDR:	.ASCIZ	/ DRIVE 'UNSAFE'<CR><LF>
032701	040	042523	042514	MSELD:	.ASCIZ	/ SELECTED/
032713	015	050012	047522	MMODE:	.ASCIZ	<CR><LF>/PROGRAM MODE (C OR F): /
032745	040	047506	046522	MFORMAT:	.ASCIZ	/ FORMAT & VERIFY/
032766	044103	041505	020113	MHECK:	.ASCIZ	/CHECK ONLY/
033001	106	051117	040515	MORMAT:	.ASCIZ	/FORMAT & VERIFY/
033021	015	005012	050117	MSIZE:	.ASCIZ	<CR><LF><LF>/OPERATE IN 32 SECTOR MODE (Y OR N) ? /
033072	050117	051105	052101	MSEC22:	.ASCIZ	/OPERATION WILL BE IN 32 SECTOR (16 BIT) MODE/<CR><LF>
033151	117	042520	040522	MSEC20:	.ASCIZ	/OPERATION WILL BE IN 30 SECTOR (18 BIT) MODE/<CR><LF>
033230	047111	040526	044514	BADENT:	.ASCIZ	/INVALID ENTRY/<CR><LF>
033250	047105	044504	043516	MADRER:	.ASCII	/ENDING DSK ADRS MUST BE EQUAL TO OR GREATER/<CR><LF>
033325	124	040510	020116		.ASCIZ	/THAN STARTING ADRS/<CR><LF>

MAINDEC-11-DZMA, RM03 FORMATTER
DZMA8.P11 21-JUL-77 15:32

MACY11 30(1046) 21-JUL-77 16:06 PAGE 107
TELETYPE MESSAGES

SEQ 0106

033352	042523	042514	052103	MSLP:	.ASCII	/SELECT DATA PATTERN (BY ENTERING 0,1 OR 2)/<CR>.<LF>
033426	024040	024460	055040		.ASCII	/ (0) ZERO'S <CR><LF>
033444	024040	024461	047440		.ASCII	/ (1) ONE'S/<CR><LF>
033460	024040	024462	053440		.ASCII	/ (2) WORST CASE: /
033502	047527	051522	020124	MPATD:	.ASCII	/WORST CASE/
033515	015	005012	052123	MSFOU:	.ASCII	<CR><LF><LF>/STARTING FORMAT ON DRIVE /
033552	005015	051412	040524	MSCHK:	.ASCII	<CR><LF><LF>/STARTING CHECK ON DRIVE /
033606	005015	043012	051117	MFCMPT:	.ASCII	<CR><LF><LF>/FORMAT COMPLETE, /
033633	015	005012	044103	MCCMPT:	.ASCII	<CR><LF><LF>/CHECK COMPLETE, /
033657	124	052117	046101	NUMERR:	.ASCII	/TOTAL ERRORS DETECTED: /
033707	120	042522	042523	ACORIS:	.ASCII	/PRESENT ADDRESS IS: /
033734	005015	047520	042527	PMSG:	.ASCII	<CR><LF>/POWER ON, PROGRAM STARTS AT 200/<LF>
033777	105	052116	051105	MESG1:	.ASCII	/ENTER THE SERIAL NUMBER (MAXIMUM 10 OCTAL DIGITS) <CR><LF><LF>
034065	015	020012	047123	SN1:	.ASCII	<CR><LF>/ SN # : /
034103	015	047412	042526	MESG3:	.ASCII	<CR><LF>/OVER 126 BAD SECTORS HAVE BEEN DETECTED/<CR><LF>
034156	040520	045503	047040		.ASCII	/PACK NOT ACCEPTABLE !/
034204	005015	047111	047503	MESG2:	.ASCII	<CR><LF>/INCORRECT MANUFACTURE DEFINED BAD SECTOR/
034256	044440	043116	051117		.ASCII	/ INFORMATION/<CR><LF>
034274	005015	047111	052111		.ASCII	<CR><LF>/INITIALIZE THE BAD SECTOR FILE ?/
034336	040450	051516	042527		.ASCII	/(ANSWER: Y OR N <CR>)/
034365	015	040412	044514	MESG4:	.ASCII	<CR><LF>/ALIGNMENT PACK, PROGRAM HALT !/
				.EVEN		

(1) ;*****

.SBTTL ERROR MESSAGES

(1) ;*****

034426	044122	030461	044440	EM1:	.ASCII	/RH11 INTERRUPT OCCURRED (RMAS=0)/
034467	125	042516	050130	EM2:	.ASCII	/UNEXPECTED ATTENTION OCCURRED/
034525	115	051501	041123	EM3:	.ASCII	/MASSBUS PARITY ERROR (MCPE=1)/
034563	115	051501	041123	EM4:	.ASCII	/MASSBUS PARITY ERROR (PAR=1)/
034620	042101	051104	051505	EM5:	.ASCII	/ADDRESS PLUG CHANGE BIT SET/
034654	044122	030461	042040	EM6:	.ASCII	/RH11 DIDN'T RESPOND TO ADDRESSING/
034716	051104	053111	020105	EM10:	.ASCII	/DRIVE OFFLINE/
034734	042520	051522	051511	EM11:	.ASCII	/PERSISTENT DRIVE UNSAFE ERROR/
034772	047125	047503	051122	EM12:	.ASCII	/UNCORRECTABLE MASSBUS PARITY ERROR/
035035	123	043117	053524	EM13:	.ASCII	/SOFTWARE TIMEOUT/
035056	051104	053111	020105	EM14:	.ASCII	/DRIVE UNSAFE ERROR/
035101	103	047117	051124	EM15:	.ASCII	@CONTROLLER/DRIVE ERROR DURING WRITE@
035145	103	047117	051124	EM16:	.ASCII	@CONTROLLER/DRIVE ERROR DURING WRITE CHECK@
035217	122	052105	044522	EM17:	.ASCII	@RETRIES NOT SUCCESSFUL - SECTOR NOT ACCEPTABLE@
035275	104	052101	020101	EM20:	.ASCII	@DATA ERROR DURING WRITE CHECK@
035333	103	047117	051124	EM21:	.ASCII	@CONTROLLER/DRIVE ERROR VERIFYING HEADERS@
035404	042510	042101	051105	EM22:	.ASCII	@HEADER COMPARE ERROR VERIFYING HEADERS@
035453	103	046131	047111	EM23:	.ASCII	@CYLINDER FIELD IN HEADER IS NOT CORRECT@
035523	127	044522	042524	EM24:	.ASCII	@WRITE CHECK ERROR@

(1) ;*****

035545	122	040515	004523	DH1:	.ASCII	/RMAS	RMCS1	RMER1	RMER2	RMDS	RMDC	RMDA/
035613	104	044522	042526	DH2:	.ASCII	/DRIVE	RMDS	RMER1	RMER2	RMAS	RMCS1/	
035665	104	044522	042526	DH3:	.ASCII	/DRIVE	REG ADR	DATA	RMCS1	RMER1	RMER2/	
035736	051104	053111	020105	DH4:	.ASCII	/DRIVE	REG ADR	GOOD	BAD	RMCS1	RMER1	RMER2/

```

DH6:      .ASCIIZ  /RH ADDRESS/
DH10:     .ASCIIZ  /DRIVE   ERR PC  RMCS1   RMCS2   RMD5    RMER1   RMER2/
DH10A:    .ASCIIZ  /RMEC1   RMEC2   RMWC    RMBA    RMDA    RMA5    RMLA/
DH10B:    .ASCIIZ  /RMD5    RMR1    RMDT    RMSN    RMOF    CYLINDER TRACK/
DH17:     .ASCIIZ  /DRIVE   ERR PC  CYLINDER TRACK SECTOR/
DH20:     .ASCIIZ  /DRIVE   ERR PC  CYLINDER TRACK SECTOR/
DH20A:    .ASCIIZ  /RMCS1   RMCS2   RMD5    RMER1   RMER2   RMR2    RMEC1   RMEC2/
DH20B:    .ASCIIZ  /RMWC    RMBA    RMDA    RMA5    RMLA    RMD5    RMR1    RMDT/
DH20C:    .ASCIIZ  /RMSN    RMOF    CYLINDER TRACK/
DH23:     .ASCII   /
DH23:     .ASCII   /
DH24:     .ASCIIZ  /DRIVE   ERR PC  CYLNR  ACTUAL  HEADER/
DH24:     .ASCII   /
DH24:     .ASCIIZ  /DRIVE   ERR PC  CYLNR  TRACK   SECTOR  DATA   DISK/<CR><LF>
.LIST     BEX

```

Address	Offset	Value	Label	Comment
5758				
5759				
5760	037100	001374	005540	005554 DT1: .WORD ATTN, RM. REG, RM. REG+14, RM. REG+42, RM. REG+12, RM. REG+34, RM. REG+6
5761	037106	005602	005552	005574
5762	037114	005546		
5763	037116	001372	024046	024050 DT2: .WORD DORIVE, RMERRS, RMERRS+2, RMERRS+4, ATTN, RMCS1
5764	037124	024052	001374	000000
5765	037132	001372	031036	031040 DT3: .WORD DORIVE, RD. ADR, RD. WRD, RM. REG, RM. REG+14, RM. REG+42
5766	037140	005540	005554	005602
5767	037146	001372	031262	031260 DT4: .WORD DORIVE, WRT. AD, WRT. WD, RD. WRD, RM. REG, RM. REG+14, RM. REG+42
5768	037154	031040	005540	005554
5769	037162	005602		
5770	037164	001272		
5771	037166	001220	001132	005540 DT6: .WORD \$RMAOR
5772	037174	005550	005552	005554 DT10: .WORD DRIVE, \$ERRPC, RM. REG, RM. REG+10, RM. REG+12, RM. REG+14, RM. REG+42
5773	037202	005602		
5774	037204	005604	005606	005542 .WORD RM. REG+44, RM. REG+46, RM. REG+2, RM. REG+4, RM. REG+6, RM. REG+16, RM. REG+20
5775	037212	005644	005546	005556
5776	037220	005660		
5777	037222	005662	005564	005566 .WORD RM. REG+22, RM. REG+24, RM. REG+26, RM. REG+30, RM. REG+32, DS. CYL, DS. TRK
5778	037230	005570	005572	001366
5779	037236	001370		
5780	037240	001220	001132	001366 DT17: .WORD DRIVE, \$ERRPC, DS. CYL, DS. TRK, SAVSEC
5781	037246	001370	001350	
5782	037252	001220	001132	001366 DT20: .WORD DRIVE, \$ERRPC, DS. CYL, DS. TRK, SAVSEC
5783	037260	001370	001350	
5784	037264	005540	005550	005552 .WORD RM. REG, RM. REG+10, RM. REG+12, RM. REG+14, RM. REG+42, RM. REG+40, RM. REG+44, RM. REG+46
5785	037272	005554	005602	005600
5786	037300	005604	005606	
5787	037304	005542	005544	005546 .WORD RM. REG+2, RM. REG+4, RM. REG+6, RM. REG+16, RM. REG+20, RM. REG+22, RM. REG+24, RM. REG+26
5788	037312	005556	005560	005562
5789	037320	005564	005566	
5790	037324	005570	005572	001366 .WORD RM. REG+30, RM. REG+32, DS. CYL, DS. TRK
5791	037332	001370		
5792	037334	001220	001132	037554 DT23: .WORD DRIVE, \$ERRPC, BUFP, RBUF, RBUF+2
5793	037342	037544	037546	
5794	037346	001220	001132	001366 DT24: .WORD DRIVE, \$ERRPC, DS. CYL, DS. TRK, SAVSEC, \$GDDAT, RM. REG+22
5795	037354	001370	001350	001140
5796	037362	005562		
5797	037364	005540	005550	005552 .WORD RM. REG, RM. REG+10, RM. REG+12, RM. REG+14, RM. REG+42, RM. REG+40, RM. REG+44, RM. REG+46
5798	037372	005554	005602	005600
5799	037400	005604	005606	

F09

MAINDEC-11-DZMA, RM03 FORMATTER
DZMA8.P11 21-JUL-77 15:32

MACY11 30(1046) 21-JUL-77 16:06 PAGE 109
ERROR MESSAGES

SEQ 0108

5800	037404	005542	005544	005546	.WORD	RM.REG+2, RM.REG+4, RM.REG+6, RM.REG+16, RM.REG+20, RM.REG+22, RM.REG+24, RM.REG
5801	037412	005556	005560	005562		
5802	037420	005564	005566			
5803	037424	005570	005572	001366	.WORD	RM.REG+30, RM.REG+32, DS.CYL, DS.TRK
5804	037432	001370				
5805						
5806	037434	000001			DF1:	.WORD 1
5807	037436	007	000			.BYTE 7,0
5808	037440	000001			DF2:	.WORD 1
5809	037442	006	000			.BYTE 6,0
5810	037444	000001			DF3:	.WORD 1
5811	037446	006	000			.BYTE 6,0
5812	037450	000001			DF4:	.WORD 1
5813	037452	007	000			.BYTE 7,0
5814	037454	000001			DF6:	.WORD 1
5815	037456	001	000			.BYTE 1,0
5816	037460	000003			DF10:	.WORD 3
5817	037462	007	000			.BYTE 7,0
5818	037464	036121				.WORD DH10A
5819	037466	007	000			.BYTE 7,0
5820	037470	036206				.WORD DH10B
5821	037472	007	140			.BYTE 7,140
5822	037474	000001			DF17:	.WORD 1
5823	037476	005	034			.BYTE 5,34
5824	037500	000004			DF20:	.WORD 4
5825	037502	005	034			.BYTE 5,34
5826	037504	036401				.WORD DH20A
5827	037506	010	000			.BYTE 8,0
5828	037510	036476				.WORD DH20B
5829	037512	010	000			.BYTE 8,0
5830	037514	036574				.WORD DH20C
5831	037516	004	014			.BYTE 4,14
5832	037520	000001			DF23:	.WORD 1
5833	037522	005	000			.BYTE 5,0
5834	037524	000004			DF24:	.WORD 4
5835	037526	007	034			.BYTE 7,34
5836	037530	036401				.WORD DH20A
5837	037532	010	000			.BYTE 8,0
5838	037534	036476				.WORD DH20B
5839	037536	010	000			.BYTE 8,0
5840	037540	036574				.WORD DH20C
5841	037542	004	014			.BYTE 4,14
5842						
5843						
5844						
5845						
5846						
5847						
5848						
5849	037544	000000	000000	000000	RBUF:	.WORD 0,0,0,0 ;BUFFER FOR HEADER CHECK
5850	037552	000000				
5851						
5852	037554				BUF:	;FORMAT AND CHECK BUFFER STARTS HERE
5853						;AND WILL OVERLAY ALL REMAINING CODE IN PROGRAM
5854						
5855					.NLIST	BEX

MAINDEC-11-DZRNA, RM03 FORMATTER
DZRNA8.P11 21-JUL-77 15:32

MACY11 30(1046) 21-JUL-77 16:06 PAGE 110
ERROR MESSAGES

SEQ 0109

037554 005015 046412 044501
037601 122 030115 020063
037623 120 047522 051107
037660 005015 047524 023440
037746 040520 045503 047440
040044 047101 020104 042522

TITLE: .ASCII <CR><LF><LF>/MAINDEC-11-DZRNA/<CR><LF>
.ASCII 2RM03 FORMATTER 2<CR><LF><LF>
.ASCII /PROGRAM NEEDS 17 K MEMORY/<CR><LF><LF>
LOADRV: .ASCII <CR><LF>/TO 'FORMAT' OR 'CHECK' DRIVE 0, REPLACE THE 'XXDP'<CR><LF>
.ASCII /PACK ON DRIVE 0 WITH ANOTHER PACK, CLEAR MEMORY LOCATION 40,<CR><LF>
.ASCII /AND RESTART THE PROGRAM/<CR><LF>
.LIST .ASCII
BEX

5856
5857
5858
5859
5860
5861
5862
5863
5864
5865
5866
5867
5868
5869
5870
5871
5872
5873
5874
5875
5876
5877
5878
5879
5880
5881
5882
5883
5884
5885
5886
5887
5888
5889
5890
5891
5892
5893
5894
5895
5896
5897
5898
5899
5900
5901
5902
5903
5904

.SBTTL BUSADR - GET BUS ADDRESS AND VECTOR ADDRESS FOR RH11
;THIS ROUTINE IS USED TO INSURE THE BUS ADDRESS
;OF THE RH11 IS SETUP FOR THE PROPER ADDRESS.
;IT WILL ALSO READ THE ADDRESS FROM THE TTY IF
;REQUIRED.
;NOTE: THIS ROUTINE DESTROYS R0-R4
;CALL

```

      JSR      PC,BUSADR
      RETURN
BUSADR: TST      CHGADR      ;INPUT FROM TTY REQUESTED?
        BEQ      3$         ;NO--BRANCH
        CLR      CHGADR      ;YES--CLEAR THE REQUEST FLAG
1$:     MOV      $RMADR,R0    ;FIRST ADDRESS
        TYPE     MRMCS1      ;"RMCS1="
        MOV      (R0),-(SP)   ;PRESENT RMCS1 ADDRESS
        TYPOC    ,LINSR      ;TYPE IT
        TYPE     ,LINSR      ;2 SPACES
        RDLIN    (SP)+,R1     ;GET THE ENTRY
        MOV      R5,CK.NUM    ;ADDRESS OF ASCII TEXT
        JSR      1$          ;ENTER AND STORE NEW ADDRESS
        BR       1$          ;ERROR RET
2$:     MOV      $RMVEC,R0    ;CHECK VERC
        TYPE     MRHVEC      ;"RHVEC="
        MOV      (R0),-(SP)   ;PRESENT RH11 VECTOR ADDRESS ON THE STACK
        TYPOC    ,LINSR      ;TYPE IT
        TYPE     ,LINSR      ;2 SPACES
        RDLIN    (SP)+,R1     ;READ THE ENTRY
        MOV      R5,CK.NUM    ;ASCII TEXT ADDRESS
        JSR      2$          ;ENTER AND STORE NEW ADDRESS
        BR       2$          ;ERROR RET
3$:     MOV      $RMADR,R0    ;FIRST ADDRESS OF NEW PARAMETERS
        MOV      $RMADR,R1    ;FIRST ADDRESS OF WHERE TO PUT THEM
        MOV      (R0)+,(R1)+  ;BUS ADDRESS
        MOV      (R0)+,(R1)+  ;VECTOR ADDRESS
        RTS      PC          ;RETURN

```

040212 046522 051503 020061 MRMCS1: .ASCIIZ 2RMCS1 = 2
040220 020075 000 000
040223 122 053110 041505 MRHVEC: .ASCIIZ 2RHVEC = 2
040230 036440 000040

.SBTTL CK.NUM - CHECK NUMBER (OCTAL)
;THIS ROUTINE CHECKS AN ASCII STRING FOR LEGAL CHARACTERS
;AND FORMS AN OCTAL NUMBER IN R2
;CALL:

H09

MAINDEC-11-DZRNA, RMO3 FORMATTER
DZRNAB.P11 21-JUL-77 15:32

MACY11 30(1046) 21-JUL-77 16:06 PAGE 111
CK.NUM - CHECK NUMBER (OCTAL)

SEQ 0110

5905			:	MOV	#ADR, R1	: ADDRESS OF ASCIZ STRING
5906			:	RO=	ADDRESS TO STORE THE	NUMBER
5907			:	JSR	R5, CK.NUM	
5908			:	RET1	ERROR RETURN	
5909			:	RET2	NORMAL RET	
5910						
5911						
5912						
5913	040234	010246	CK.NUM:	MOV	R2, -(SP)	; SAVE R2
5914	040236	010346		MOV	R3, -(SP)	; SAVE R3
5915	040240	010446		MOV	R4, -(SP)	; SAVE R4
5916	040242	012703	000006	MOV	#6, R3	; MAX OCTAL DIGITS IN THE NUMBER
5917	040246	005002		CLR	R2	; FINAL OCTAL VALUE
5918	040250	112104	1\$:	MOVB	(R1)+, R4	; GET CURRENT POINTED BYTE
5919	040252	001424		BEQ	3\$	; BRANCH IF TERMINATOR DETECTED
5920	040254	120427	000060	CMPB	R4, #'0	; SMALLER THAN ASCII-0 ?
5921	040260	103425		BLO	5\$	; YES, ERROR EXIT
5922	040262	120427	000067	CMPB	R4, #'7	; LARGER THAN ASCII-7 ?
5923	040266	101022		BHI	5\$	; YES, ERROR EXIT
5924	040270	006302		ASL	R2	; SHIFT LEFT
5925	040272	103420		BCS	5\$	
5926	040274	006302		ASL	R2	; ONE
5927	040276	103416		BCS	5\$	
5928	040300	006302		ASL	R2	; OCTAL DIGIT
5929	040302	103414		BCS	5\$	; ERROR IF CARRY BIT SET
5930	040304	042704	177770	BIC	#177770, R4	; CHOP OFF HIGHER BITS
5931	040310	060402		ADD	R4, R2	; APPENDING CURRENT DIGIT
5932	040312	005303		DEC	R3	; DECREMENT BYTE COUNT
5933	040314	001401		BEQ	2\$	; BRANCH IF LAST DIGIT
5934	040316	000754		BR	1\$	; LOOPING BACK
5935	040320	112104	2\$:	MOVB	(R1)+, R4	; CHECK TERMINATOR
5936	040322	001004		BNF	5\$	; BRANCH IF NOT FOUND
5937	040324	005702	3\$:	TST	R2	; FINAL VALUE = 0 ?
5938	040326	001401		BEQ	4\$	; YES
5939	040330	010210		MOV	R2, (R0)	; REPLACE THE ORIGINAL VALUE
5940	040332	005725	4\$:	TST	(R5)+	; ADJUST FOR NORMAL RET
5941	040334	012604	5\$:	MOV	(SP)+, R4	; RESTORE 4
5942	040336	012603		MOV	(SP)+, R3	; RESTORE R3
5943	040340	012602		MOV	(SP)+, R2	; RESTORE R2
5944	040342	000205		RTS	R5	; EXIT
5945						
5946						
5947						
5948						
5949						
5950						
5951		000001				
			.END			

MAINDEC-11-DZRMA, RM03 FORMATTER
DZRMA8.P11 21-JUL-77 15:32

MACY11 30(1046) 21-JUL-77 16:06 PAGE 114
CROSS REFERENCE TABLE -- USER SYMBOLS

ATESTN=	000000	1389	1393															
ATTN	001374	1474#	3466*	5760	5763													
ATO	= 000001	1248#																
AT1	= 000002	1249#																
AT2	= 000004	1250#																
AT3	= 000010	1251#																
AT4	= 000020	1252#																
AT5	= 000040	1253#																
AT6	= 000100	1254#																
AT7	= 000200	1255#																
AUNIT	= 000000	1389	1396															
AUSWR	= 000000	1389	1403															
AVECT1=	000254	1079#	1389	1428														
AVECT2=	000000	1389	1429															
A16	= 000400	1152#																
A17	= 001000	1153#																
BADENT	033230	3151	5757#															
BAD16	001414	1487#	2236	2256	2291*	2292*	2301	2302*	2303*	2304*	2305*	2306*	2307*	2308*				
		2309*	2311*	2313*	2315*	2316*	2317*	2318*	2319*	2320*	2323*	2329	2331	2333				
		2334	2335	2336	2337	2338	2341*	2428	3245	3258								
		1490#	2235	2262	2333*	2338*	2342*	2431	3261									
BAD18	002420	1170#																
BAI	= 000010	1450#	1561	1955*	2045*	2101*	2109	2349	2358	2362	2716	2720	2762	2765*				
BEGCYL	001322	2768*	2937	2966	2968*	2975*	2997	3231	3237*	3243*								
BEGIN	006412	1115	1811#															
BEGIN1	006422	1114	1813#	1858														
BEGIN2	006436	1812	1817#	4156														
BEGTRK	001326	1452#	1562	1954*	2044*	2100*	2113	2352	2365	2763	2764*	2767*	2936	2939				
		2967	2969*	2974*	2999	3232	3238*	3242*										
BIT0	= 000001	1063#																
BIT00	= 000001	1053#	1063	1930	2097													
BIT01	= 000002	1052#	1062	1933	2097	2922	4776	5024										
BIT02	= 000004	1051#	1061															
BIT03	= 000010	1050#	1060	5231	5524													
BIT04	= 000020	1049#	1059	2903	5266													
BIT05	= 000040	1048#	1058	4627	5044	5246	5401											
BIT06	= 000100	1047#	1057	4709														

SEQ 0113

[illegible]

DH20	036332	1773	5757#
DH20A	036401	5757#	5826
DH20B	036476	5757#	5828
DH20C	036574	5757#	5830
DH23	036627	1794	5757#
DH24	036725	1801	5757#
DH3	035665	1682	5757#
DH4	035736	1689	5757#
DH6	036021	1703	5757#
DISPLA	001156	1367#	1839*
DISPRE	000174	1100#	1847
DLT =	100000	1182#	
DMD =	000001	1238#	
DOME	015102	2855	2856#
DPINT	024106	4425#	4638
DPR =	000400	1208#	
DPRS	024116	4438#	4749
DRIVE =	001220	1445#	1988*
		5794	
DRQ =	004000	1271#	
DRVACT	024056	4384#	4757
		5290*	5402*
DRVCLR=	000111	1321#	
DRVINT	024444	4629	4663#
DRVQUE	032144	4752	4765
DRVSTA	024066	1912	1990
		4739	4747
		5326	5422*
DRVTP	024076	1915	1930
DRY =	000200	1207#	
DSMR =	177570	984#	1366
DS.CYL	001366	1471#	3467*
DS.TRK	001370	1472#	3468*
DTE =	010000	1231#	2620
DTSY =	000200	1244#	
DTUM	024200	4531#	4615
		5472	
DT00 =	000001	1262#	
DT01 =	000002	1263#	
DT02 =	000004	1264#	
DT03 =	000010	1265#	
DT04 =	000020	1266#	
DT05 =	000040	1267#	
DT06 =	000100	1268#	
DT07 =	000200	1269#	
DT08 =	000400	1270#	
DT1	037100	1669	5760#
DT10	037166	1718	1725
DT17	037240	1767	5780#
DT2	037116	1676	5763#
DT20	037252	1774	5782#
DT23	037334	1795	5792#
DT24	037346	1802	5794#
DT3	037132	1683	5765#
DT4	037146	1690	5767#
DT6	037164	1704	5770#

SEO 0115

[illegible]

NO9

MAINDEC-11-DZRNA, RMD3 FORMATTER
DZRNA8.P11 21-JUL-77 15:32

MACY11 30(1046) 21-JUL-77 16:06 PAGE 118
CROSS REFERENCE TABLE -- USER SYMBOLS

SEQ 0116

[illegible]

SEQ 0117

[illegible]

[illegible]

SEQ 0119

RMCS1 =	000000	4572#	4668#	4674	4689	4802#	4803	4878	4900	4942	4962	4973	4988	4999
RMCS2 =	000010	5126	5156	5208	5240	5277	5335	5508	5763					
		2617	4576#	4627*	4667*	4669	4751*	4801*	4863*	4883*	4905*	4991	5010	5022
		5035	5044*	5064*	5124*	5222*	5306*	5319*	5384*	5401*	5461	5518	5551*	5557
		5581*	5586											
RMDA =	000006	2605	2811	4575#	4870	4896	4911							
RMD8 =	000022	4581#	5555											
RMDC =	000034	4586#	4874	4886	4918									
RMD5 =	000012	2631	4577#	4696	5066	5224	5249	5307	5386					
RMDT =	000026	4583#	4677											
RMEC1 =	000044	4590#												
RMEC2 =	000046	4591#	5565											
RMERRS	024046	4376#	4610	5191	5194	5204	5307*	5308*	5309*	5310*	5763			
RMER1 =	000014	2619	2924	4578#	4702	5234	5308	5522						
RMER2 =	000042	4589#	5309											
RMHR =	000036	4587#												
RMINIT	024232	1895	2078	4606#										
RMLA =	000020	4580#	5080	5082										
RMHR1 =	000024	4582#												
RMHR2 =	000040	4588#	5310											
RMOF =	000032	4585#	4693	4924	4928	4948	4952							
RMR =	000004	1221#												
RMSN =	000030	4584#												
RMVEC	024216	1879*	1889*	4555#	4623	4625	4732							
RMWC =	000002	4573#												
RM.REG	005540	1506#	1539	2605	2617	2619	2631	2811	2924	3469	3471	5760	5765	5767
		5771	5774	5777	5784	5787	5790	5794	5797	5800	5803			
		2177	2185	2409	2572	2599	2724	2789	2800	3312	4731#			
		1935	5757#											
		1929	5757#											
RM03	024736	1318#												
RM03A	032431	3086	5349#											
RM03B	032417	1932	5757#											
RNOP =	000101	1318#												
RPTMR	030404	3086	5349#											
RPOS	032424	1932	5757#											
RTC =	000117	1324#												
SAVEFG	024154	1896*	2079*	4505#	4977	5130	5296							
SAVREG=	104412	3515	4218	4356#	4606	4734	4794	5009	5103	5352	5547	5653		
SAVSEC	001350	1461#	2605*	2607*	2608*	2609*	2636	2637	2689*	2811*	2813*	2814	2815	3179
		3299	3305	5780	5782	5794								
SAVMC	001352	1462#	2651*	2984										
SC	027234	5109	5138	5141	5147	5153#								
SCAWC	015662	2663	2984#											
SC01 =	000100	1278#												
SC02 =	000200	1279#												
SC04 =	000400	1280#												
SC10 =	001000	1281#												
SC11	030066	5193	5230	5287#										
SC12	030176	5187	5198	5306#										
SC13	030246	5177	5180	5316#										
SC20 =	002000	1282#												
SC3	027304	5168#	5172											
SC4	027310	5170#	5201	5213	5217	5265	5279	5286	5305	5342				
SC5	027322	5169	5175#											
SC6	027522	5197	5218#											
SC6A	027632	5195	5242#											
SC7	027760	5232	5245	5247	5257	5267#								
SC8	030036	5209	5225	5235	5241	5250	5278	5280#	5315					

MAINDEC-11-DZRMA, RM03 FORMATTER
DZRMA8.P11 21-JUL-77 15:32

MACY11 30(1046) 21-JUL-77 16:06 PAGE 122
CROSS REFERENCE TABLE -- USER SYMBOLS

[illegible]

SEQ 0121

[illegible]

MAINDEC-11-DZRMA, RM03 FORMATTER
DZRMA8.P11 21-JUL-77 15:32

MACY11 30(1046) 21-JUL-77 16:06 PAGE 124
CROSS REFERENCE TABLE -- USER SYMBOLS

UNTON	032345	1926	5757#																	
UPDACY	015776	2770	2977	3014#																
UPDATK	016054	2959	3033#																	
UPE =	020000	1180#																		
USSAV	004430	1496#	2234	2271	2335*	2337*	2340*	3270												
USTAB	003424	1493#	2237	2334*	2336*	2343*	3164	3174	3189	3191	3215	3264								
US1 =	000001	1167#																		
US2 =	000002	1168#																		
US4 =	000004	1169#																		
VV =	000100	1206#																		
WC =	001354	1463#	1966*	1970*	2034*	2040*	2391													
WCE =	040000	1181#	2617																	
WCF =	000040	1224#																		
WCKD =	000151	1331#																		
WCKHD =	000153	1332#	2596																	
WCTBL	006052	1615#	2547	2550	2558	2561	2646	2649	2819											
WLE =	004000	1230#	2924																	
WRAP	001402	1478#	1958*	2046*	2108*	2117*	2355													
WRL =	004000	1211#																		
WRTDAT=	000161	1333#																		
WRTHD =	000163	1334#	2569	2799	3303															
WTRK	012234	2402#																		
WTRKA:	012666	2505#	2676																	
WTRKB	012772	2517	2527#	2530																
WTRKC	013066	2526	2544#																	
WTRKD	013132	2511	2522	2537	2540	2555#														
WTRKE	013200	2554	2566#																	
WTRKX	012310	2417#	2679	2697																
WTRKY	012332	2426#																		
WTRKZ	014064	2695#																		
WTRK1	012636	2494#																		
WTRK2	013214	2569#	2654																	
WRT.AO	031262	5503*	5513*	5516#	5767															
WRT.RM	031172	4688	4692	4869	4873															
		4972	4987	4998	5207	4877	4885	4895	4899	4910	4917	4927	4941	4951						
					</															

SEQ 0123

[illegible]

SMAOR3	001250	1424#																
SMAOR4	001254	1427#																
SMAIL	001206	1135	1139	1390#	1816	1850	1867	3489	3601									
\$MAMS1	001236	1411#																
\$MAMS2	001242	1419#																
\$MAMS3	001246	1422#																
\$MAMS4	001252	1425#																
SMBADR	001102	1135#																
\$MFLG	021276	3663*	3669	3704*	3708#													
\$MNEW	023213	3972	4116#															
\$MSGAD	001222	1397#	3679*	3682														
\$MSGLG	001224	1398#	3684*															
\$MSGTY	001206	1391#	3677	3685*	3697	3701*												
\$MSWR	023202	3969	4114#															
\$MTYP1	001237	1412#																
\$MTYP2	001243	1420#																
\$MTYP3	001247	1423#																
\$MTYP4	001253	1426#																
\$NULL	001170	1372#	3628	3657														
\$NWTST=	000000	1814#																
\$OCNT	021524	3749#	3778*	3791#														
\$OMODE	021526	3744#	3748*	3753	3756*	3767*	3793#											
\$PASS	001214	1394#	1850*	2834*	2838*	2839*	2856	2871*										
\$PASTH	001106	1137#																
\$PWRAD	023364	4156#																
\$PWRON	023224	1830	4123#	4151														
\$PWRCG	023360	4154#																
\$PWRUP	023276	4133	4139#															
\$QUES	001202	1378#	2011	2031	2070	2142	3508	3657	4023	4094	4111							
\$PDCHR	022622	4036#	4354															
\$FDOEC=	***** U	4356																
\$FDLIN	022712	4059#	4355															
\$RDOCT=	***** U	4356																
\$POSZ =	000015	4052#																
\$FSRE	023724	4301#	4357															
\$SMAOR	001272	1438#	1878	1887*	1888	5770	5872	5890										
\$RMVEC	001274	1439#	1879	1884*	1889	5881												
\$RTNAD	015100	2855#																
\$R2A =	***** U	4358																
\$SAVRE	023666	4285#	4356															
\$SAVR6	023374	4132#	4140	4141*	4142*	4160#												
\$S820	023436	3112	3118	4197#														
\$SETUP=	000116	1811#	1825	1826	1828	1830	1832	1834	1864	2838	3464	3499	3507	3906				
		3911	3912	3942	4118													
\$STUP =	177777	1811#																
\$SUPRS	023376	3113	3119	4172#														
\$SVPC =	000200	1107#	1112															
\$SWR =	123000	958#	968	1085	1086	1087	1088	1089	1376	1377	1816	1832	2778	2838				
		2848	2854	2856	3455	3456	3457	3458	3459	3477	3484	3496	3500	3508				
		4157																
\$SWREG	001230	1402#	1853															
\$TESTN	001212	1393#	1816*	2872*	2995*	3027*	3230	3244*										
\$TKB	001162	1369#	3099	3103	3866	3887	3898	3923	3951	3978								
\$TKCNT	021754	3867#	3882*	3912	3929*	4043	4045*											
\$TKINT	021772	1863	2056	2286	3071	3106	3882#	3903	3964									
\$TKQEN=	021771	3871#	3937	4048														

MAINDEC-11-DZRMA, RM03 FORMATTER
DZRMA8.P11 21-JUL-77 15:32

MACY11 30(1046) 21-JUL-77 16:06 PAGE 127
CROSS REFERENCE TABLE -- USER SYMBOLS

STKQIN	021756		3868#	3883#	3884	3935*	3936*	3937	3939*							
STKQOU	021760		3869#	3884#	4046	4047*	4048	4050*								
STKQSR	021762		3870#	3883	3939	4050										
STKS	001160		1368#	3104*	3866	3888*	3919*	3921	3927*	3949	3965*	3975	3987*	4007*		
STKSRV	022042		3885	3898#												
STN =	000002		958#	968	1814	1816#										
STNPWR	023602		4225	4226	4246#											
STPB	001166		1371#	3646*	3657											
STPFLG	001173		1375#	3595	3657											
STPS	001164		1370#	3644	3657											
STRAP	023762		1828	4322#												
STRAP2	024004		4333#	4344												
STRP =	000014		4337#	4346#	4347#	4348#	4349#	4350#	4351	4352#	4353	4354#	4355#	4356#	4357#	
			4358#													
STRPAD	024016		4327	4344#												
STSTM	001104		1136#													
STSTMH	001116		1348#	3476	3508											
STTYIN	023146		4061	4062	4074	4092	4106	4110#								
STYPBN=	***** U		4350													
STYPOS	021530		3807#	4349												
STYPE	020552		3595#	3690	4337	4345										
STYPEC	020764		3625	3632	3639	3644#	3645	4009								
STYFEX	021032		3650	3652	3655#											
STYPOC	021326		3747#	4346												
STYPON	021342		3746	3749#	4348											
STYPOS	021302		3742#	4347												
SUNIT	001220		1396#	1445												
SUNITM	001110		1138#													
SUSAR	001232		1403#													
\$VECT1	001256		1428#	1882	1884											
\$VECT2	001260		1429#													
\$SGET4=	000000		2848#													
\$OFILL	021525		3743#	3747*	3757	3792#										
\$4OCAT=	***** U		3486													
.	= 040344		1095#	1099#	1107	1108#	1110#	1112#	1113#	1117#	1123	1124#	1126#	1128#	1140	
			1345#	1381	1487#	1490#	1493#	1496#	1499#	1823	1883	1886	1961	1975	2183	
			2223	2570	2597	2675	2709	2722	2856	3508	3657	3711#	3861#	3866	3870#	
			3871	3872#	4110#	4111	4118	4135	4159	4266#	5638#	5639#	5640#	5641#	5642#	
			5643#	5644#	5645#	5646										

MAINDEC-11-DZRNA, RMO3 FORMATTER
DZRNAB.P11 21-JUL-77 15:32

MACY11 30(1046) 21-JUL-77 16:06 PAGE 129
CROSS REFERENCE TABLE -- MACRO NAMES

[illegible]

MAINDEC-11-DZRMAB, RMO3 FORMATTER
DZRMAB.P11 21-JUL-77 15:32

MACY11 30(1046) 21-JUL-77 16:06 PAGE 130
CROSS REFERENCE TABLE -- MACRO NAMES

SEQ 0127

.SAPTY	958#	3659
.SCATC	958#	1093
.SCHTA	958#	1339
.SDB2D	958#	4205
.SEOP	958#	2774
.SERR0	958#	3449
.SERRT	958#	
.SPOWE	958#	4119
.SREAO	958#	3863
.SSAVE	958#	4268
.SSB2D	958#	4186
.SSUPR	958#	4162
.STRAP	958#	4314
.STYPD	958#	3795
.STYPE	958#	3578
.STYPO	958#	3717

. ABS. 040344 000

ERRORS DETECTED: 0

DSKZ:DZRMAB,DSKZ:DZRMAB.SEQ/NL:MD:MC:CND:TOC/LI:ME/DOC/SOL/CRF=DSKM:RMDRV4.P11,DSKM:DZRMAB.P11
RUN-TIME: 29 17 1 SECONDS
RUN-TIME RATIO: 1024/48=20.9
CORE USED: 38K (75 PAGES)

DOCUMENT PAGES: 127