

TM03/TE16

DATA RELIABILITY TEST
CZTEDB0

AH-A801B-MC
COPYRIGHT © 77-78
FICHE 1 OF 1

JAN 1979
digital
MADE IN USA

The image shows a microfiche card with a grid of 120 frames. Each frame contains a small, illegible image of a document page, likely a data reliability test report. The frames are arranged in a regular grid pattern on a dark background.

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

.REM %

IDENTIFICATION

PRODUCT CODE: AC-A800B-MC
PRODUCT NAME: CZTEDBO TM03-TE16/TU77 DATA RELIBILITY PROGRAM
DATE CREATED: 15 NOV 1978
MAINTAINER: DIAGNOSTIC GROUP
AUTHOR: J. G. ADAMS

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

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

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

COPYRIGHT (©) 1977,1978 BY DIGITAL EQUIPMENT CORPORATION

TABLE OF CONTENTS

	PARAGRAPH	SUBJECT	PAGE
46	1.	ABSTRACT	3
47	2.	REQUIREMENTS	3
48	3.	LOADING PROCEDURE	3
49	4.	STARTING PROCEDURE	4
50	5.	DATA PATTERNS	11
51	6.	RANDOMIZATION	12
52	7.	DYNAMIC PARAMETERS	13
53	8.	CONSOLE SWITCH	14
54	9.	ERROR PRINTOUTS	19
55	10.	STATISTICS PRINTOUT	27
56	11.	AUTO SEQUENCE	28
57	12.	TESTING PROCEDURES	30
58	13.	LISTING	32
59			
60			
61			
62			

63
64
65
66
67
68
69
70
71
72
73
74
75
76
77
78
79
80
81
82
83
84
85
86
87
88
89
90
91
92
93
94
95
96
97
98
99
100
101
102
103
104
105
106
107

1. ABSTRACT

THIS PROGRAM IS DESIGNED TO BE USED BY AN EXPERIENCED ENGINEER /TECHNICIAN FOR EVALUATION AND DEBUGGING OF MAG TAPE DRIVES. THE PROGRAM IS CAPABLE OF EXERCISING THE TE16 MAGNETIC ON A MASSBUS THROUGH THE TMO3 MAG TAPE CONTROLLER. ANY COMBINATION OF TMO3'S & TE16'S UP TO A MAXIMUM OF EIGHT (8), MAY BE TESTED BY A SINGLE EXECUTION OF THE PROGRAM. THIS FLEXIBILITY IS POSSIBLE BECAUSE THE PROGRAM HAS NO FIXED PARAMETERS OR TESTING SEQUENCE. THE ENTIRE TEST PLAN, INCLUDING PARAMETERS AND OPERATING SEQUENCE, IS DETERMINED BY THE OPERATOR THROUGH RESPONSES TO TELETYPE REQUESTS AND SETTING OF CONSOLE SWITCHES.

THE PROGRAM PROVIDES FOR TESTING OF ALL TAPE DRIVE FUNCTIONS SUCH AS WRITING,READING,REWINDING,TAPE POSITIONING,EOT - BOT SENSING AND ASSUMES A GOOD RH AND TMO3.

HOWEVER; THE RH AND TMO3 ARE TESTED SOMEWHAT INTRINSICALLY DURING THE TEST CYCLE IN ORDER TO PROVIDE FULL INFORMATION ABOUT ANY ERROR CONDITIONS DETECTED.

DURING A TEST CYCLE, CHECKS ARE MADE FOR STATUS ERRORS,DATA ERRORS, POSITION ERRORS,WORD COUNT AND CURRENT MEMORY ADDRESS ERRORS WHEREVER APPLICABLE AS DETECTED BY THE RH OR TMO3.

2. REQUIREMENTS (HARDWARE)

- A. ANY PDP-11 PROCESSER
- B. 8K OF CORE
- C. TELETYPE
- D. TMO3 TAPE CONTROLLER
- E. 1 TO 8 MAG TAPE DRIVES
- F. MASSBUS CONTROLLER

3. LOADING PROCEDURE

USE STANDARD PROCEDURE FOR LOADING BINARY TAPES

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

4. STARTING PROCEDURE

THERE ARE FOUR (4) STARTING ADDRESSES THAT MAY BE USED;
200(8), 204(8), 210(8), AND 240(8):

- A. 200(8): THIS ADDRESS MUST BE USED ON INITIAL START FROM LOAD AS ALL PARAMETERS ARE ENTERED FROM HERE. REQUESTS ARE PRINTED ON THE TELETYPE FOR ENTRY OF RH STARTING ADDRESS, VECTOR ADDRESS, DRIVE NUMBER(TMO3 ADDRESS), SLAVE NUMBER, DENSITY, PARITY, FORMAT, RECORD COUNT, CHARACTER COUNT, PATTERN NUMBER, TAPE MARK AND STALL FOR READ, WRITE, AND TURNAROUND. ALL REPOSSES SHOULD BE MADE IN OCTAL AND WITHIN THE LIMITS OF THE PARAMETER. A QUESTION MARK (?) WILL BE TYPED IF ANY CHARACTER ENTERED IS NOT BETWEEN 0 THRU 7 (OCTAL). THE CHARACTER MAY BE RETYPED FOLLOWING THE QUESTION MARK. IF THE RESPONSE IS NOT WITHIN ITS LIMITS. A QUESTION MARK (?) IS TYPED AND THE ENTIRE RESPONSE MAY BE RENTERED. SOME RESPONSES REQUIRE MORE THAN ONE (1) CHARACTER, BUT NONE REQUIRES MORE THAN SIX (6). RESPONSES OF MORE THAN ONE CHARACTER NEED NOT HAVE LEADING ZEROS AND SHOULD BE TERMINATED BY A CARRIAGE RETURN IF LESS THAN THE MAXIMUM NUMBER OF CHARACTERS IS INPUT.
- B. 204(8): THIS ADDRESS SHOULD BE USED ANYTIME A RESTART OF THE PROGRAM IS NECESSARY AND THE PARAMETERS ENTERED AT THE INITIAL START OF 200(8) NEED NOT BE CHANGED. ALSO NOTE THAT ANY DATA PATTERN WHICH HAD BEEN GENERATED BY SETTING THE RANDOM DATA SWITCH (CONSOLE SWITCH EIGHT) WILL NOT BE OVERWRITTEN AND THERFORE IS HELD IN CORE FOR USE UNTIL CONSOLE SWITCH EIGHT(8) IS AGAIN SET AND THAT ALL STATISTICS WIL
- C. 210(8): THIS ADDRESS IS THE SAME AS USING 204(8) IN THAT THE PREVIOUSLY SET PARAMETERS ARE USED; HOWEVER, THE DATA PATTERN IS RETURNED TO THE FIXED PATTERN ORIGINALLY CALLED FOR AT THE 200(8) START AND ALL STATISTICS ARE CLEARED TO
- D. 240(8): THIS IS A SPECIAL ADDRESS WHICH WILL CAUSE THE PROGRAM TO EXECUTE A PREDETERMINED TEST PLAN ON ALL AVAILABLE DRIVES AND SLAVES. THE ONLY INPUT REQUIRED BY THE OPERATOR IS A RESPONSE TO REQUESTS FOR THE RH ADDRESS, VECTOR ADDRESS, CONTINUOUS OPERATION OF THE SEQUENCE, AND NRZ ONLY.
- E. 300(8): THIS ADDRESS IS TO BE USED AS A RESTART ONLY AND WILL PERFORM JUST AS IN 200(8) EXCEPT THAT THE PARAMETER INPUT LIST IS SHORTENED. THE SHORT PARAMETER LIST CONSISTS OF DRIVE NUMBER, SLAVE NUMBER, DENSITY, PARITY, FORMAT, RECORD COUNT, CHARACTER COUNT, PATTERN, TAPE MARK, AND

164
165
166
167
168
169
170
171
172
173
174
175
176
177
178
179
180
181
182
183
184
185
186
187
188
189
190
191
192
193
194
195
196
197
198
199
200
201
202
203
204
205
206
207
208
209
210
211
212
213
214
215
216
217
218
219

INTERCHANGE READ.
**NOTE SEE ALSO SECTION 8-CONSOLE SWITCH SETTINGS

THE FOLLOWING IS AN EXPLANATION OF THE INITIAL START (200 OCTAL) REQUESTS AND RESPONSES:

REGISTER START: THE RESPONSE REQUIRED FOR THIS REQUEST IS TO ENTER THE ADDRESS OF THE FIRST RH REGISTER (CS1) AS A SIX DIGIT UNIBUS ADDRESS.

VECTOR ADDRESS: THE RESPONSE FOR THIS REQUEST IS TO ENTER THE INTERRUPT VECTOR ADDRESS USED BY THE RH AS A THREE (3) DIGIT ADDRESS.

DRIVE NUMBER: THE DRIVE NUMBER (MASSBUS ADDRESS OF THE TM03) IS ENTERED AS ONE (1) OCTAL CHARACTER AND MUST BE WITHIN THE LIMITS OF 0 THROUGH 7.

SLAVE NUMBER: THE SLAVE NUMBER IS ENTERED AS ONE (1) OCTAL CHARACTER AND MUST BE WITHIN THE LIMITS OF 0 THROUGH 7. WHEN THE SLAVE NUMBER HAS BEEN ENTERED AND IS LEGAL, THE PROGRAM TESTS FOR THE PRESENCE OF A SLAVE OF THAT NUMBER. IF THE SLAVE IS AVAILABLE A PRINTOUT OF 7 CHANNEL, IF APPLICABLE, AND ITS SERIAL NUMBER (IN BCD) WILL BE MADE TO ASSIST THE OPERATOR IN SETTING OF DENSITY, PARITY, AND FORMAT. A CHECK IS MADE FOR THE PROPER SETTING OF THE DRIVE TYPE REGISTER; IF WRONG, A MESSAGE IS PRINTED FOR INFORMATION ONLY. IF THE SLAVE IS NOT AVAILABLE, A MESSAGE STATING SO WILL BE PRINTED AND A NEW SLAVE NUMBER REQUEST WILL BE ISSUED. WHEN A GOOD SLAVE NUMBER HAS BEEN ENTERED, REQUESTS FOR OPERATING DENSITY PARITY AND FORMAT ARE MADE FOR THAT SLAVE AND SHOULD BE RESPONDED TO ACCORDING TO THAT PARTICULAR SLAVE'S NEEDS. AS MANY AS EIGHT (8) SLAVE NUMBER REQUESTS MAY BE USED, HOWEVER, AT LEAST ONE MUST BE USED. THE SLAVE NUMBERS AND THEIR RESPECTIVE DENSITY, PARITY AND FORMAT MAY BE ENTERED IN ANY ORDER. THE INFORMATION FOR EACH SLAVE ENTERED IS LOADED INTO A TABLE FOR REFERENCE IN TESTING. IF LESS THAN EIGHT(8) SLAVES ARE REQUIRED, THEN RESPONDING TO THE SLAVE NUMBER REQUEST WITH A CARRIAGE RETURN WILL TERMINATE THE SLAVE ENTRIES AND CONTINUE TO THE NEXT PARAMETER. IT SHOULD BE REMEMBERED

220
221
222
223
224
225
226
227
228
229
230
231
232
233
234
235
236
237
238
239
240
241
242
243
244
245
246
247
248
249
250
251
252
253
254
255
256
257
258
259
260
261
262
263
264
265
266
267
268
269
270

THAT AT LEAST ONE SLAVE NUMBER REQUEST
MUST BE ENTERED. IF THE FIRST
REQUEST IS RESPONDED TO BY A CARRIAGE
RETURN, THEN THE REQUEST WILL BE REPEATED.

DENSITY: THE DENSITY REQUEST IS RESPONDED TO BY ONE (1) OCTAL
CHARACTER AND MUST BE WITHIN THE LIMITS OF 0 THRU 4.
AS EACH SLAVE NUMBER IS ENTERED, A REQUEST FOR THE
OPERATING DENSITY FOR THAT SLAVE IS TYPED. THE
RESPONSE MEANINGS ARE AS FOLLOWING:

- A. 3 = 800BPI, NRZI
- B. 4 = 1600BPI, PE (9 CHANNEL ONLY)

PARITY: THE PARITY REQUEST IS RESPONDED TO BY ONE (1)
OCTAL CHARACTER AND MUST BE EITHER 0 OR 1.

- A. 1 = EVEN PARITY
- B. 0 = ODD PARITY

FORMAT: THE FORMAT REQUEST IS RESPONDED
TO BY TWO (2) CHARACTERS
AND SHOULD BE AS FOLLOWS

- A. 14 = 9 CHANNEL NORMAL (TWO FRAMES PER WORD)
- B. 15 = CORE DUMP (FOUR FRAMES PER WORD)
- C. 16 = PDP-15 OR IBM COMPATABLE (TWO FRAMES PER
(DATA IS BYTE SWAPPED ON TAPE)

RECORD COUNT: THIS REQUEST IS RESPONDED TO BY A SIX (6) CHARACTER
OCTAL NUMBER FROM 1 TO 177777. REMEMBER LEADING
ZEROS ARE NOT REQUIRED AND IF LESS THAN SIX
CHARACTERS ARE ENTERED, A CARRIAGE RETURN
WILL TERMINATE THE RESPONSE. THE RECORD COUNT
IS USED IN CONJUNCTION WITH THE CHARACTER COUNT
TO ESTABLISH A BLOCKING FACTOR FOR USE IN READ OR
WRITE CYCLES.

CHARACTER COUNT: THIS RESPONSE IS ENTERED AS FOUR (4) OCTAL
CHARACTERS WITHIN THE LIMITS OF 20 THRU 4000. AGAIN
LEADING ZEROS ARE NOT REQUIRED AND A CARRIAGE
RETURN TERMINATES A LESS THAN FOUR (4) CHARACTER
RESPONSE. THE CHARACTER COUNT IN CONJUNCTION
WITH THE RECORD COUNT IS USED TO ESTABLISH
THE BLOCK SIZE (CHARACTERS PER RECORD, AND
RECORDS PER BLOCK) USED IN READ AND WRITE CYCLES.
THE SAME BLOCKING IS USED ON ALL AVAILABLE UNITS.

271
272
273
274
275
276
277
278
279
280
281
282
283
284
285
286
287
288
289
290
291
292
293
294
295
296
297
298
299
300
301
302
303
304
305
306
307
308
309
310
311
312
313
314
315
316
317
318
319
320
321
322
323
324

PATTERN NUMBER: THIS RESPONSE IS A TWO (2) CHARACTER OCTAL NUMBER WITHIN THE LIMITS OF 0 THRU 15(8). THE NUMBER ENTERED WILL CAUSE A SPECIFIC DATA PATTERN TO BE USED FOR ALL READING AND WRITING. THIS DATA PATTERN IS NOT CHANGED UNLESS RANDOM DATA IS REQUESTED BY SETTING CONSOLE SWITCH EIGHT (8) TO A ONE. RESETTING OF THE RANDOM DATA SWITCH DOES NOT CAUSE REVERSION TO THE FIXED PATTERN, BUT WILL HOLD THE LAST GENERATED PATTERN UNTIL A RESTART IS DONE FROM LOCATION 200(8), 210(8), OR 300(8). WHEN OPERATING IN NRZ MODE (DENSITY 0-3) THE PROGRAM CONSTRUCTS AND SAVES BOTH AN EXPECTED CRC CHARACTER AND AN LRC CHARACTER FOR COMPARISONS WITH THE HARDWARE GENERATED CHECK CHARACTER IN BOTH READ AND WRITE. THE SELECTION OF DATA PATTERN ZERO (0) HAS A SPECIAL USE. PATTERN NUMBER ZERO (0) WILL CAUSE TO BE READ IN AT THE HIGH SPEED PAPER TAPE READER ANY DATA PATTERN DESIRED. THE EXTERNAL INPUT DATA THROUGH THE READER IS DONE BY PREPARING A PAPER TAPE WITH A PROGRAM CALLED DTC. (MAINDEC-11-DZTUF-A-D) ANY CONFIGURATION OF BITS AND CHARACTERS MAY BE USED AND A LIMIT OF 377(8) CHARACTERS IS IMPOSED. WHEN EXTERNAL DATA IS INPUT, THE ENTIRE WRITE BUFFER IN CORE IS FILLED WITH THE PATTERN SO THAT ANY SIZE RECORD MAY BE USED. DATA PATTERN PATTERN ZERO (0) EXTERNAL PAPER TAPE NEED ONLY BE READ ONCE AT INITIAL START OF 200(8), AND NEED NOT BE READ AGAIN UNLESS OVERWRITTEN BY RANDOM DATA. BE SURE TO LOAD THE READER BEFORE PRESSING START.

TAPE MARK: THE TAPE MARK REQUEST IS USED TO DETERMINE IF THE OPERATOR WISHES TO HAVE EACH DATA BLOCK SEPERATED BY A TAPE MARK. IF RESPONDED TO BY A ONE (1) THE TAPE MARK WILL BE WRITTEN AND WHEN READING WILL BE EXPECTED AT THE END OF DATA BLOCK. A ZERO (0) RESPONSE WILL DISALLOW TAPE MARK. PLEASE NOTE THAT THE TAPE MARK RECORD INCREASES THE BLOCK SIZE BY ONE (1) RECORD; IN OTHER WORDS, A BLOCK OF 100 RECORDS WILL HAVE THE TAPE MARK AS RECORD 101.

INTERCHANGE READ: THIS REQUEST IS RESPONDED TO BY A SINGLE CHARACTER INPUT OF EITHER ONE (1) OR ZERO (0). A RESPONSE OF ONE (1) WILL CAUSE ALL READING TO BE DONE IN THE INTERCHANGE MODE. A ZERO RESPONSE WILL CAUSE READING IN NORMAL MODE.

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

SINGLE PASS: THIS REQUEST IS RESPONDED TO BY EITHER A ONE (1) OR A ZERO (0). RESPONSE OF 1, WILL CAUSE THE TEST TO BE STOPPED AFTER THE LAST AVAILABLE DRIVE REACHES END OF TAPE. A RESPONSE OF 0, WILL ALLOW CONTINUOUS RUNNING THROUGH MULTIPLE PASSES. TO RESTART AT END OF PASS, PRESS CONTINUE, OR RESTART AT THE CONSOLE.

STALLS: THE STALL REQUESTS ARE RESPONDED TO BY A SIX (6) CHARACTER OCTAL NUMBER WITHIN THE LIMITS OF 1 THRU 177777. LEADING ZEROS ARE NOT REQUIRED AND AN ENTRY OF LESS THAN SIX (6) CHARACTERS SHOULD BE TERMINATED BY A CARRIAGE RETURN. EACH INCREMENT OF THE VALUE ADDS ABOUT 2.6 MICSEC TO THE DELAY.

READ: THE TIME DELAY BETWEEN EACH RECORD READ

WRITE: THE TIME DELAY BETWEEN EACH RECORD WRITTEN

TURN AROUND: TIME DELAY BETWEEN CHANGES OF TAPE DIRECTION (FORWARD, TO REVERSE, ETC.) AND BETWEEN BLOCKS.

FIXED PARAMETERS: IT SHOULD BE NOTED THAT ALL PARAMTERS EXCEPT FOR THE SLAVE DESCRIPTION VALUES (SLAVE NUMBER, DENSITY, PARITY, AND FORMAT) HAVE NOMINAL VALUES ALREADY STORED IN THE PROGRAM. COUNT, CHARACTER COUNT, TAPE MARK AND STALLS) IS TYPED. ITS PRESENT STORED VALUE IS ALSO PRINTED. IF THESE VALUES NEED NOT BE CHANGED, SIMPLY TYPE A CARRIAGE RETURN AS RESPONSE AND NO CHANGE WILL BE MADE. EACH START OF THE PROGRAM AT 200(8) WILL SHOW THE CURRENT VALUES OF THESE PARAMETERS AS PER THE LAST ENTRY. WHEN A FRESH LOAD OF THE PAPER TAPE IS DONE, THE PARAMETERS WILL REFLECT THE FIXED VALUES STORED IN THE PROGRAM.

- A. RECORD COUNT = 100
- B. CHARACTER COUNT = 200
- C. PATTERN NUMBER = 1
- D. TM=0
- E. INTERCHANGE READ = 0
- F. SINGLE PASS = 0
- G. READ STALL = 1
- H. WRITE STALL = 1
- I. TURN AROUND STALL = 1

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

SAMPLE START AT 200(8):

THE FOLLOWING IS A SAMPLE OF THE
PRINTED REQUESTS AND THEIR RESPONSES.
RESPONSES ARE ENCLOSED IN PARENS FOR
CLARITY ONLY AND (CR) MEANS CARRIAGE RETURN

LOAD ADDRESS 200(8), SET CONSOLE SWITCHES, PRESS START SWITCH:

TE16 TAPE DRIVE TEST

REGISTER START=172440(172440)
VECTOR ADDRESS=224(CR)
DRIVE NUMBER (4)
SLAVE NUMBER=(5) SN: 5009
DENSITY=(3)
PARITY=(0)
FORMAT=(14)
SLAVE NUMBER=(2) 9 CHAN SN: 0022
DENSITY=(3)
PARITY=(1)
FORMAT=(15)
SLAVE NUMBER=(CR)
RECORD COUNT=100 (500)(CR)
CHARACTER COUNT=200 (38)?(7)(CR)
PATTERN NUMBER=1 (22)
?
(6)(CR)
TM=(0)
INTERCHANGE READ=(1)
SINGLE PASS=(0)

ENTER STALLS
READ=1 (CR)
WRITE=1 (CR)
TURN AROUND=1 (3000)(CR)

THE PROGRAM WILL NOW PERFORM THE TEST CYCLE SET IN
THE CONSOLE SWITCHES ON SLAVE FIVE (5) THEN TWO (2),
ONE BLOCK ON EACH UNIT PER CYCLE, USING DATA PATTERN
NUMBER SIX (6) WITH A BLOCKING FACTOR OF 37 CHARACTERS
PER RECORD AND 500 RECORDS PER BLOCK. THE DELAYS ARE SET
FOR MINIMUM ON READ AND WRITE, AND APPROXIMATELY .75
SECONDS ON TURN AROUND.

NO TAPE MARKS WILL BE WRITTEN AND ALL READING
WILL BE DONE IN INTERCHANGE MODE (MAINT MODE 0001).

426
427
428
429
430
431
432
433
434
435
436
437
438
439
440
441
442
443
444
445
446
447
448
449
450
451
452
453
454
455
456
457
458
459
460
461
462
463
464
465
466
467
468
469
470
471
472
473
474
475
476
477
478
479
480
481

4.1 AUTOMATIC MODE OPERATION

IF THE PROGRAM IS LOADED AND RUN IN AUTOMATIC (CHAIN) MODE THE AUTO ACCEPT SEQUENCE TEST PLAN IS RUN. SEE SEC 11. BELOW; THE SOFTWARE SWR IS INVOKED WITH A SWITCH SETTING OF 000000 IF LOADED VIA ACT11. NO OPERATOR INTERVENTION IS REQUIRED.

**EXCEPTION: IF THIS PROGRAM IS LOADED VIA TMDP CHAIN MODE THE PROGRAM WILL TEST ALL SLAVES ON THE FIRST AVAILABLE DRIVE EXCEPT SLAVE 0.

**NOTE: IN ORDER TO CHANGE THE DEFAULT SETTING OF THE SOFTWARE SWR, CHANGE LOC: 176(SWREG:) TO THE DESIRED SETTING.

5. DATA PATTERNS

THERE ARE FIFTEEN DATA PATTERN GENERATORS STORED IN CORE AND ANY ONE OF THESE MAY BE SELECTED. THE ONE UNIQUE CASE IS PATTERN ZERO(0); SELECTION OF PATTERN ZERO(0) REQUIRES THAT A PREVIOUSLY PREPARED PAPER TAPE BE ENTERED AT THE HIGH SPEED READER. THIS TAPE CONTAINS A DATA PATTERN OF NO MORE THAN 377 OCTAL CHARACTERS. THE FIRST CHARACTER READ IN IS THE NUMBER OF ACTUAL DATA CHARACTERS THAT ARE CONTAINED ON THE TAPE. EACH DATA CHARACTER MAY BE ANY COMBINATION OF BITS AND WILL BE LOADED INTO CORE AS THEY APPEAR ON THE TAPE. NO MATTER HOW MANY CHARACTERS ARE ON TAPE, THE ENTIRE WRITE BUFFER (4000 CHARACTERS) WILL BE FILLED WITH THE PATTERN ENTERED SO THAT ANY SIZE RECORD CAN BE USED. (SEE DTC MAINDEC-11-DZTUF-A-D) THE PROGRAM GENERATES A CYLIC REDUNDENCY CHECK CHARACTER (CRC) AND A LONGITUDINAL REDUNDENCY CHECK CHARACTER (LRC) FOR COMPARISONS AGAINST THE CRC AND LRC GENERATED BY THE HARDWARE IN NRZI READS OR WRITES.

THE FOLLOWING IS A LIST OF THE DATA PATTERNS AVAILABLE:

- DATA0: EXTERNAL INPUT THRU HIGH SPEED READER (SEE DTC)
- DATA1: ALL ONE BITS IN ALL CHARACTERS
- DATA2: ALL ZERO BITS IN ALL CHARACTERS
- DATA3: A ONE BIT WALKING FROM RIGHT TO LEFT IN A FIELD OF ZEROS
- DATA4: A ZERO BIT WALKING FROM RIGHT TO LEFT IN A FIELD OF ONES.
- DATA5: ALTERNATING ONE AND ZERO BITS IN EACH CHARACTER
- DATA6: ALTERNATING ZERO AND ONE BITS IN EACH CHARACTER
- DATA7: SAME AS DATA5 BUT WITH EVERY OTHER CHARACTER COMPLEMENTED
- DATA10: WALKING ONE/ALL ONE IN ALTERNATING CHARACTERS
- DATA11: INCREMENTING CHARACTERS (000-377)
- DATA12: DECREMENTING CHARACTERS (377-000)
- DATA13: ALTERNATING CHARACTERS OF ALL ZERO AND ALL ONE BITS
- DATA14: WALKING ZERO/ALL ZERO IN ALTERNATING CHARACTERS
- DATA15: AUTO SEQUENCE PATTERN 0,0,-1,-1,-1,0,0

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

6. RANDOMIZATION

THERE ARE THREE (3) VALUES THAT MAY BE GENERATED RANDOMLY; DATA, CHARACTER COUNT, AND RECORD COUNT. THESE ARE NORMALLY SET TO SOME FIXED VALUE BUT MAY BE RANDOMIZED BY SETTING THE APPROPRIATE CONSOLE SWITCHES.

- A. RANDOM DATA: (CONSOLE SWITCH 8)
GENERATES AN ENTIRE BUFFER, CHARACTER BY CHARACTER, OF RANDOM DATA WHEN SWITCH 8 IS SET TO A ONE. ONCE SET, THE RESETTING OF SWITCH 8 CAUSES THE LAST GENERATED PATTERN TO BE RETAINED IN CORE. A RESTART AT LOCATION 200(8) OR 210(8) WILL CAUSE REVERSION OF THE DATA TO THE FIXED PATTERN REQUESTED INITIALLY. A RESTART AT LOCATION 204(8) WILL HOLD THE LAST GENERATED PATTERN IN CORE UNTIL SWITCH 8 IS AGAIN SET.
ALTHOUGH THE DATA IS GENERATED AS RANDOM, THE PROGRESSION OF RANDOM CHARACTERS IS ALWAYS THE SAME FROM THE OUTSET OF RANDOMIZATION. THEREFORE IT IS POSSIBLE TO GENERATE ONE TAPE REEL OF RANDOM DATA ON ONE UNIT, RESTART THE PROGRAM TO RE-ESTABLISH THE OUTSET POINT, AND READ THE RANDOM TAPE REEL ON ANOTHER UNIT FOR COMPATABILITY TESTING. IN MULTIDRIVE SYSTEMS THE SAME BLOCK OF DATA, WHETHER RANDOM OR FIXED, IS WRITTEN OR READ ON EACH AVAILABLE UNIT IN THE ORDER THAT THEY WERE ENTERED, BEFORE BEING CHANGED.
- B. RANDOM CHARACTER COUNT: (CONSOLE SWITCH 7)
GENERATES A DIFFERENT NUMBER OF CHARACTERS PER RECORD TO BE WRITTEN ON EACH BLOCK CYCLE. THE SAME NUMBER OF CHARACTERS PER RECORD IS WRITTEN OR READ ON EACH AVAILABLE UNIT BEFORE BEING CHANGED. RESETTING SWITCH 7 HOLDS THE LAST VALUE GENERATED.
- C. RANDOM RECORD COUNT: (CONSOLE SWITCH 6)
GENERATES A DIFFERENT NUMBER OF RECORDS FOR EACH BLOCK OF DATA WRITTEN OR READ ON EACH BLOCK CYCLE. THE SAME NUMBER OF RECORDS IS WRITTEN OR READ ON EACH AVAILABLE UNIT BEFORE BEING CHANGED. RESETTING SWITCH 6 HOLDS LAST VALUE GENERATED.

531
532
533
534
535
536
537
538
539
540
541
542
543
544
545
546
547
548
549
550

7. DYNAMIC PARAMETERS:

THE THREE (3) STALL VALUES ARE CONSIDERED TO BE DYNAMIC PARAMETERS AS THEY MAY BE CHANGED WHILE THE PROGRAM IS RUNNING BY TYPING A CONTROL B CHARACTER AT THE TELETYPE. AS SOON AS THE BUS IS RELEASED BY THE MAG TAPE OPERATION IN PROGRESS, THE PROGRAM WILL RESPOND TO THE CONTROL C INPUT BY TYPING A REQUEST FOR NEW STALL PARAMETERS. THE LAST VALUES THAT WERE ENTERED WILL BE PRINTED AS THE STORED VALUES AND MAY BE CHANGED BY ENTERING NEW VALUES OR LEFT UNCHANGED BY TYPING A CARRIAGE RETURN. THE YOZZLE STALL IS ALSO DYNAMIC AND CAN BE CHANGED BY TYPING A CONTROL B WHILE DOING A YOZZLE. A YOZZLE STALL REQUEST WILL BE PRINTED AND SHOULD BE RESPONDED TO WITH THE DESIRED VALUE.

551
552
553
554
555
556
557
558
559
560
561
562
563
564
565
566
567
568
569
570
571
572
573
574
575
576
577
578
579
580
581
582
583
584
585
586
587
588
589
590
591
592
593
594
595
596
597
598
599
600
601
602
603
604
605
606

8. CONSOLE SWITCH SETTINGS

CONTROL:

- 1) CONTROL G <^G>:
SELECTS SOFTWARE SWR AND ALLOWS USER TO SELECT NEW SWITCHES.
THE MACHINE WILL THEN TYPE: SWR=XXXXXXNEW=
WHERE: XXXXXX IS THE OCTAL CONTENTS OF THE SOFTWARE SWR.
AFTER THE 'NEW=' HAS BEEN TYPED THEN THE OPERATOR CAN DO ONE
OF THE FOLLOWING AT THE TTY:
A) TYPE A NUMBER TO BE LOADED INTO THE SOFTWARE SWR
B) IF A <CR> IS THE FIRST KEY DEPRESSED THE SOFTWARE SWR
CONTENTS WILL NOT BE CHANGED.
- 2) CONTROL A <^A>:
ALTERNATES USAGE OF THE SWR BETWEEN THE HARDWARE SWR & SOFTWARE SWR.
- 3) CONTROL B <^B>:
SEE SECTION 7 DYNAMIC PARAMETERS
- 4) CONTROL U <^U>:
DELETES ALL CHARACTERS TYPED IN RESPONSE TO A REQUEST.

THE CONSOLE SWITCHES ARE USED TO SET UP THE TEST CYCLE
DESIRED, TO GENERATE RANDOM VALUES, AND TO CONTROL ERROR
RESPONSES. THE SWITCHES SHOULD BE SET IN THE DESIRED
MANNER BEFORE PRESSING THE START SWITCH BECAUSE THEY
ARE ALL DYNAMIC AND WILL RUN THE PROGRAM IN ANY
CONFIGURATION. ALL SWITCHES SET TO ZERO(0) IS NORMAL.

- SW15: 1=STOP ON ERROR
0=CONTINUE ON ERROR
- SW14: 1=PRINT READ/WRITE STATISTICS
0=DO NOT PRINT STATS
- SW13: 1=DO NOT CHECK DATA ERRORS
0=CHECK DATA ERRORS
- SW12: 1=DO NOT CHECK WRITE STATUS ERRORS (NOR CLEAR THEM IF THEY DO OCCUR)
0=CHECK WRITE STATUS ERRORS
- SW11: 1=DO NOT CHECK READ STATUS ERRORS (NOR CLEAR THEM IF THEY DO OCCUR)
0=CHECK READ STATUS ERRORS
- SW10: 1=DO NOT PRINT ANY ERRORS (EXCEPT CATASTROPHIC ERRORS)
0=PRINT ALL ERRORS
- SW9: 1=REWIND ALL AVAILABLE TAPES
0=DO NOT REWIND
- SW8: 1=GENERATE RANDOM DATA
0=USED FIXED DATA

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

SW7: 1=GENERATE RANDOM CHARACTER COUNT
0=USE FIXED CHARACTER COUNT

SW6: 1=GENERATE RANDOM RECORD COUNT
0=USED FIXED RECORD COUNT

SW5: 1=YOZZLE ON CURRENT RECORD
0=DO NOT YOZZLE ON RECORD

SW4: 1=DO WRITE/READ RETRIES
0=DO NOT RETRY

SW3: 1=DO NOT READ FORWARD
0=READ FORWARD

SW2: 1=DO NOT READ REVERSE
0=READ REVERSE

SW1: 1=READ FORWARD FIRST
0=READ REVERSE FIRST

SW0: 1=DO NOT WRITE
0=WRITE

632
633
634
635
636
637
638
639
640
641
642
643
644
645
646
647
648
649
650
651
652
653
654
655
656
657
658
659
660
661
662
663
664
665
666
667
668
669
670
671
672
673
674
675
676
677

SWITCH EXPLANATION AND EXAMPLES:

SW0-3: THESE SWITCHES ARE USED TO CONTROL THE SEQUENCE OF MAG TAPE OPERATIONS PERFORMED ON EACH AVAILABLE UNIT. THE BLOCK OF DATA DESCRIBED THROUGH THE RESPONSES TO TELETYPE REQUESTS AT INITIAL START WILL BE EITHER WRITTEN OR READ FROM EACH AVAILABLE UNIT IN THE ORDER THAT THEY WERE ENTERED. THE SEQUENCE OF OPERATIONS IS CALLED A CYCLE, AND WILL BE PERFORMED CONTINUOUSLY UNTIL STOPPED BY THE OPERATOR. WHEN END OF TAPE IS REACHED, THE UNIT WILL BE REWOUND AND FLAGGED AS UNAVAILABLE FOR TEST UNTIL ALL UNITS HAVE REACH EOT, AT WHICH TIME TESTING IS RESUMED ON ALL AVAILABLE UNITS.

EXAMPLES: 0-3

- A. SW0=0, SW1=0, SW2=1, SW3=1
WRITE ONLY X RECORDS OF Y CHARACTERS
- B. SW0=0, SW1=0, SW2=1, SW3=0
WRITE THEN BACKSPACE AND READ FORWARD X RECORDS
- C. SW0=0, SW1=0, SW2=0, SW3=1
WRITE THEN READ REVERSE X RECORDS.
- D. SW0=0, SW1=0, SW2=0, SW3=0
WRITE THEN READ REVERSE AND READ FORWARD X RECORDS
- E. SW0=0, SW1=1, SW2=0, SW3=0
WRITE THEN BACKSPACE AND READ FORWARD THEN REVERSE
- F. SW0=1, SW1=0, SW2=1, SW3=0
READ TAPE FORWARD X RECORDS
- G. SW0=1, SW1=0, SW2=0, SW3=1
READ TAPE REVERSE X RECORDS
- H. SW0=1, SW1=0, SW2=0, SW3=0
READ TAPE REVERSE THEN FORWARD
- I. SW0=1, SW1=1, SW2=0, SW3=0
READ TAPE FORWARD THEN REVERSE

678
679
680
681
682
683
684
685
686
687
688
689
690
691
692
693
694
695
696
697
698
699
700
701
702
703
704
705
706
707
708
709
710
711
712
713
714
715
716
717
718
719
720
721
722
723
724
725
726
727
728
729
730
731

SW4: SWITCH FOUR (4), WHEN SET TO A ONE (1), WILL CAUSE ANY DATA RELATED ERROR TO BE RETRIED. THE WRITE RETRY SCHEME CONSISTS OF REWRITING THE RECORD IN THE SAME SPOT ON TAPE FOUR (4) TIMES. IF ALL FOUR (4) REPEATS ARE SUCCESSFUL, THE RECORD IS CONSIDERED AS RECOVERED, AND A TAPE WRITE ERROR IS LOGGED. IF ANY OF THE FOUR (4) REPEATS IS UNSUCCESSFUL, A SKIP ERASE IS DONE, A SUPECTED BAD TAPE SPOT IS LOGGED AT THIS BLOCK AND RECORD NUMBER, AND A SECOND RETRY OF FOUR REPEATS IS DONE. IF AFTER FOUR (4) RETRIES, THE RECORD CANNOT BE RECOVERED A NOTIFICATION IS PRINTED, AND TESTING IS RESUMED ON THE NEXT RECORD. IF 20(8) BAD TAPE SPOTS ARE FOUND, THE SLAVE WILL BE REWOUND AND REMOVED FROM TESTING WITH AN APPROPRIATE MESSAGE PRINTED. THE READ RETRY SCHEME CONSISTS OF REREADING THE RECORD UP TO EIGHT TIMES. IF ALL EIGHT REREADS ARE BAD, IT IS A HARD ERROR. IF ANY REREAD IS SUCCESSFUL, THIS IS A SOFT ERROR. IF THE ORIGINAL ERROR IS OF THE NON-RETRYABLE TYPE (IE: ILF,RMR,ILR,NEF,CBUSPE), THE RETRY SCHEME IS NOT ENTERED AND A MESSAGE IS PRINTED.

SW5: SWITCH FIVE (5) WHEN SET DURING A READ FORWARD OR REVERSE WILL CAUSE THE TAPE TO CONTINUOUSLY READ THE CURRENT RECORD BY SPACING EITHER FORWARD OR REVERSE AND REREADING THAT RECORD. THIS TAPE MOVEMENT IS CALLED YOZZLING. THERE IS A SOFTWARE DELAY EXECUTED BETWEEN EACH SPACE/READ OF THE RECORD AND IT MAY BE VARIED BY TYPING CONTROL C ON THE TELETYPE DURING THE EXECUTION OF THE YOZZLE AND RESPONDING TO THE PRINTED REQUEST WITH A SIX (6) DIGIT VALUE. THE YOZZLE STALL IS PRESET TO A VALUE OF 3000 IN THE PROGRAM TO PREVENT EXCESSIVE TAPE WEAR, BUT MAY BE SET TO ANY VALUE THROUGH THE TELETYPE.

SW6-8: THESE THREE (3) SWITCHES CONTROL THE RANDOMIZATION OF DATA AND BLOCK SIZE AND MAY BE SET AND RESET AT ANY TIME. THE ACTUAL CHANGE WILL TAKE PLACE BETWEEN BLOCK CYCLES.

SW9: SWITCH NINE (9) WHEN SET WILL CAUSE ALL AVAILABLE TAPE UNITS TO BE REWOUND AT THE END OF THE CURRENT BLOCK CYCLE. TESTING WILL BE RESUMED AT A BLOCK COUNT OF ONE (1) WHEN ALL UNITS HAVE REACHED BOT.

732
733
734
735
736
737
738
739
740
741
742
743
744
745
746
747
748
749
750
751
752
753
754
755
756
757
758
759
760
761
762
763
764
765
766
767
768
769
770
771
772
773
774
775
776
777
778

SW10-13:

THESE SWITCHES ARE USED TO CONTROL THE ERROR HANDLING TO BE DONE ON THE TAPE OPERATION DESCRIBED BY SWITCHES 0-3.

- A. SWITCH TEN (10) WHEN SET TO A ONE WILL DISALLOW ANY ERROR PRINTOUTS MADE ON THE OPERATION IN PROGRESS. CATASTROPHIC FAILURES AND INFORMATION PRINTOUTS WILL STILL OCCUR. IE: UNIT NOT AVAILABLE, ILLEGAL BOT, DROP OR PICK OVERFLOW, AND EOT REWIND.
- B. SWITCH ELEVEN (11) WHEN SET TO A ONE WILL DISALLOW THE CHECKING FOR STATUS ERRORS ON READ (FORWARD OR REVERSE) OPERATIONS.
- C. SWITCH TWELVE (12) WHEN SET TO A ONE WILL DISALLOW THE CHECKING FOR STATUS ERRORS ON WRITE OPERATIONS.
- D. SWITCH THIRTEEN (13) WHEN SET TO A ONE WILL DISALLOW THE CHECKING OF READ DATA. THIS SWITCH HAS NO EFFECT ON STATUS CHECKING.

**NOTE THAT WHEN SW11 OR 12 ARE SET, NOT ONLY ARE ERRORS NOT CHECKED, BU
***THEREFOR USE CAUTION TO ASSURE THAT OPERATIONS ARE NOT UNEXECUTED DUE
****DO NOT SET SW 11 OR 12 TO A ONE (1), DURING A RETRY SEQUENCE.

SW14:

SWITCH FOURTEEN (14) WHEN SET TO A ONE (1) WILL PRINT THE ACCUMULATED READ/WRITE STATISTICS FOR THE SELECTED SLAVE UNDER TEST AT THE END OF THE CURRENT BLOCK CYCLE. THE STATISTICS PRINTED ARE THE NUMBER OF BITS DROPPED OR PICKED, THE NUMBER OF RETRIES, WRITE ERRORS, READ ERRORS, AND DATA ERRORS.

SW15:

SWITCH FIFTEEN (15) WHEN SET TO A ONE, WILL CAUSE THE PROGRAM TO HALT ON ANY ERROR DETECTED BY THE OPERATION IN PROGRESS. IF BOTH SWITCH TEN (10) AND FIFTEEN (15) ARE SET, THE ACTUAL ERROR DETECTED WILL NOT BE PRINTED BUT WILL CAUSE A HALT. IF SWITCH TEN (10) IS RESET BEFORE PRESSING CONTINUE, THE ERROR WHICH CAUSED THE HALT WILL BE PRINTED BEFORE TESTING IS RESUMED.

779
780
781
782
783
784
785
786
787
788
789
790
791
792
793
794
795
796
797
798
799
800
801
802
803
804
805
806
807
808
809
810
811
812
813
814
815
816
817
818
819
820
821
822
823
824
825
826
827
828

9. ERROR PRINTOUTS

THERE ARE THREE TYPES OF ERROR PRINTOUTS MADE BY THE PROGRAM: OPERATION ERRORS, DATA ERRORS, AND CONDITION ERRORS. EACH ERROR MESSAGE PRINTED IS PROCEEDED BY A TWO LINE HEADER WHICH CONTAINS THE DRIVE NUMBER, SLAVE NUMBER, DENSITY, PARITY, AND FORMAT ON THE FIRST LINE, AND THE BLOCK NUMBER, RECORD NUMBER, RECORD SIZE, AND ERROR TYPE ON THE SECOND.

A. OPERATION ERRORS:

THESE ARE ERRORS WHICH CAN OCCUR AS A DIRECT RESULT OF A TAPE OPERATION.

1. READ/WRITE STATUS ERRORS: THESE ARE DETECTED BY EITHER THE TM03 ITSELF OR BY THE MASSBUS CONTROLLER. ALL STATUS ERRORS WILL BE REPORTED.
2. TAPE POSITION ERRORS: THESE ARE INDICATED BY AN INCORRECT SPACE OR REWIND OPERATION IN WHICH TAPE POSITION BECOMES UNRELIABLE.

B. DATA ERRORS:

DATA ERRORS WILL OCCUR WHEN TAPE IS BEING READ AND THE DATA FROM TAPE DOES NOT MATCH THE EXPECTED DATA. WHEN READING IN THE REVERSE DIRECTION, THE RECORD NUMBERS WILL BE COUNTED DOWN FROM LAST TO FIRST. THE CHARACTER NUMBERS IN REVERSE READS WILL ALSO BE COUNTED DOWN IN ORDER TO REFLECT TAPE POSITION RATHER THAN THE ORDER TRANSFERRED.

BECAUSE DATA RECORDS CAN BE UP TO FOUR THOUSAND CHARACTERS LONG, AN ERROR CONDITION WHICH WILL CAUSE THE ENTIRE RECORD TO READ INCORRECTLY COULD CAUSE A VERY LENGTHY PRINTOUT. THEREFORE, A COUNTER OF SUCCESSIVE BAD CHARACTERS IS EMPLOYED. IF TEN (10) CHARACTERS IN SUCCESSION ARE BAD, A NOTIFICATION IS PRINTED (BAD RECORD) AND THE NEXT TWENTY FIVE (25) CHARACTERS ARE SKIPPED BEFORE CHECKING IS RESUMED. IF THE BAD RECORD CONDITION OCCURS THREE (3) TIMES IN ONE RECORD, THE REST OF THE RECORD IS SKIPPED, DOWN TO THE LAST TEN (10) CHARACTERS WHICH WILL BE CHECKED. THE SKIPPING AND RESUMPTION OF CHECKING WILL ONLY BE DONE ON RECORDS WHICH ARE LONG ENOUGH TO ALLOW IT.

829
830
831
832
833
834
835
836
837
838
839
840
841
842
843
844
845
846
847
848
849
850
851
852
853
854
855
856
857
858
859
860
861
862
863
864
865
866
867
868
869
870
871
872
873
874
875
876
877
878
879
880
881
882
883
884

C. CONDITION ERRORS: (CATASTROPHIC)

THESE PRINTOUTS REFLECT THE STATE OF THE TAPE SYSTEM
EITHER BEFORE OR AFTER AN OPERATION

1. EOT: WHEN EOT (END OF TAPE) IS ENCOUNTERED DURING
EITHER A READ OR WRITE, THE CYCLE IS COMPLETED
ON THE SHORTENED BLOCK AFTER WHICH THE SLAVE
WILL BE REWOUND AND FLAGGED AS UNAVAILABLE
FOR TESTING UNTIL ALL SLAVES HAVE REACHED EOT AND
ARE REWOUND. WHEN THE LAST AVAILABLE SLAVE
HAS REACHED EOT AND BEEN REWOUND TO BOT,
TESTING WILL BE RESUMED ON ALL SLAVES.
2. ILLEGAL BOT: WHEN A SLAVE ENCOUNTERS BOT DURING
A READ, WRITE, OR SPACE OPERATION, AN ERROR
IS PRINTED AND THE PROGRAM HALTED. THIS IS
A CATASTROPHIC ERROR. TESTING MAY BE RESUMED
BY PRESSING CONTINUE; BUT A RESTART IS
SUGGESTED.
3. NO INTERRUPT RETURNED: EACH TAPE OPERATION SHOULD BE
TERMINATED BY THE SETTING OF AN INTERRUPT IN
THE CPU. IF NO INTERRUPT IS RETURNED WITHIN
THE APPROPRIATE TIME, AN ERROR IS PRINTED.
4. NO MEDIUM ON-LINE: BEFORE AN OPERATION IS ATTEMPTED,
THE TMO3 IS CHECKED FOR MOL. IF IT IS NOT
SET, AN ERROR IS PRINTED, AND THE PROGRAM STOPPED.
TESTING MAY BE RESUMED BY PRESSING CONTINUE.
5. NO BOT ON REWIND: AS EACH SLAVE IS REWOUND A CHECK
IS MADE TO ASSURE THAT PROPER POSITION AT BOT
IS ESTABLISHED. IF BOT IS NOT SET UPON COMPLETION OF
A REWIND, AN ERROR IS PRINTED AND THE PROGRAM
WILL HALT. PRESS CONTINUE TO RESUME TESTING.
6. POSITION ERROR: IF POSITION IS LOST DURING A RETRY,
A MESSAGE IS PRINTED, THE TAPE REWOUND,
AND REMOVED FROM TESTING UNTIL ALL ARE
RESTARTED AT BLOCK ONE.
7. BAD TAPE OVERFLOW: IF 20(8) BAD TAPE SPOTS ARE FOUND,
A MESSAGE IS PRINTED, THE TAPE REWOUND,
AND REMOVED FROM TESTING UNTIL ALL ARE
RESTARTED AT BLOCK ONE.
8. HARD READ ERROR: IF ANY HARD READ ERROR IS ENCOUNTERED
DURING A RETRY, A MESSAGE IS PRINTED
REGARDLESS OF THE SETTING OF SW10.
9. NON-RETRYABLE: IF ANY NON-RETRYABLE ERROR IS ENCOUNTERED, A
MESSAGE IS PRINTED REGARDLESS OF THE SETTING OF SW10.

885
886
887
888
889
890
891
892
893
894
895
896
897
898
899
900
901
902
903
904
905
906
907
908
909
910
911
912
913
914
915
916

D. EXAMPLES:

GLOSSARY:

BN = CURRENT BLOCK NUMBER
RN = CURRENT RECORD NUMBER
RS = RECORD SIZE, IN FRAMES
WE = WRITE STATUS ERROR
RE = READ STATUS ERROR
SE = SPACE ERROR
TM = TAPE MARK
F = FORWARD
R = REVERSE
CS1 = RH/TE16 CONTROL REGISTER
WC = RH WORD COUNT
BA = RH BUS ADDRESS
FC = TE16 FRAME COUNT
CS2 = RH CONTROLLER STATUS
DS = TE16 DRIVE STATUS
ER = TE16 ERROR REGISTER
AS = ATTENTION SUMMARY
CK = TE16 CHECK CHARACTER
DB = RH DATA BUFFER
MR = TE16 MAINTENANCE REGISTER
DT = TE16 DRIVE TYPE
SN = TE16 SERIAL NUMBER
TC = TE16 TEST CONTROL
*F = DATA FORMAT
*P = PARITY
*D = DENSITY
*PATRN = DATA PATTERN NUMBER (R = RANDOM)

917
918
919
920
921
922
923
924
925
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
958
959
960
961
962

EXAMPLE 1: IN THIS EXAMPLE SLAVE 1 ON TMO3 0 WAS OPERATING AT 1600 BPI IN ODD PARITY USING THE NINE CHANNEL NORMAL DATA FORMAT. A WRITE STATUS ERROR WAS DETECTED. THE BAD STATUS INDICATES THAT AN UNCORRECTABLE DATA ERROR (BIT 6 OF ER) AND A PE FORMAT ERROR (BIT 7 OF ER) OCCURED DURING THE WRITE OPERATION OF THE SIXTH (6) RECORD OF THE FIFTY (50) RECORDS IN BLOCK (2). THE SIZE OF THE RECORD WAS TWO HUNDRED (200) FRAMES. THE CHECK CHARACTER REFLECTS THE BAD TRACK.

DRIVE NO. 0 *SLAVE NO. 1 *D 4 *P 0 *F 14 *PATRN 1
*BN 2 *RN 6-50 *RS = 200 *WE
CS1 144260
CS2 100
DS 150640
ER 300
WC 0
CK 4

EXAMPLE 2: IN THIS EXAMPLE SLAVE 3 ON TMO3 1 WAS OPERATING AT 800 BPI IN EVEN PARITY USING THE NINE CHANNEL NORMAL DATA FORMAT. A READ STATUS ERROR WAS DETECTED DURING THE REVERSE READ OF THE TENTH (10) RECORD OF THE 25 RECORDS IN THIS BLOCK (12). THE SIZE OF THE RECORD IS TWENTY (20) FRAMES. THE PRINTOUT INDICATES THE DETECTION OF A VERTICAL PARITY ERROR (VPE: BIT 6 OF ER) AND A CYCLIC REDUNDENCY ERROR (CRC: BIT 15 OF ER). THE CRC CHARACTER, AS RECEIVED, IS NOT AS EXPECTED AND IS PRINTED SHOWING BOTH THE ACTUAL (FIRST) AND THE EXPECTED (LAST).

DRIVE NO. 2 *SLAVE NO. 3 *D 3 *P 1 *F 14 *PATRN 3
*BN 12 *RN 10-25 *RS 20 *RE R
CS1 144276
CS2 100
DS 150600
ER 100100
WC 0
CRC 767-777

963
964
965
966
967
968
969
970
971
972
973
974
975
976
977
978
979
980
981
982
983
984
985
986
987
988
989
990
991
992
993
994
995
996
997
998
999
1000
1001
1002
1003
1004
1005
1006
1007
1008

EXAMPLE 3: IN THIS EXAMPLE, THE HEADER IS THE SAME AS IN EXAMPLE TWO (2) EXCEPT THAT THE ERROR TYPE REFLECTS A READ ERROR IN THE FORWARD DIRECTION. IT IS NORMAL FOR THE SYSTEM TO DETECT AN ERROR IN THE FORWARD AND REVERSE DIRECTION AT THE SAME RECORD. REMEMBER THAT IN REVERSE OPERATIONS THE RECORD NUMBER IS COUNTED DOWN SO THAT RECORD NUMBER TEN (10) WILL SHOWN IN THE PROPER POSITION IN BOTH FORWARD AND REVERSE.

DRIVE NO. 2 *SLAVE NO. 3 *D 3 *P 1 *F 14 *PATRN 2
*BN 12 *RN 10-25 *RS 20 *RE F
CS1 144270
CS2 100
DS 150600
ER 100100
WC 0
CRC 767-777

EXAMPLE 4: IN EXAMPLES 2 AND 3 THE READ OPERATION RESULTED IN BAD STATUS, HOWEVER THE DATA ASSOCIATED WITH THE OPERATION WAS NOT BAD (OR WAS NOT CHECKED: SW 13=1). THIS EXAMPLE (4) SHOWS A PRINTOUT REFLECTING A READ STATUS ERROR ACCOMPANIED BY BAD DATA IN CHARACTERS FOUR (4) AND SIX (6).

DRIVE NO. 2 *SLAVE NO. 3 *D 3 *P 1 *F 14 *PATRN 2
*BN 12 *RN 10-25 *RS 20 *RE F
CS1 144270
CS2 100
DS 150600
ER 100100
WC 0
CRC 767-777
CN 4
G 11111111
B 10111111
CN 6
G 11111111
B 10111111

1009
1010
1011
1012
1013
1014
1015
1016
1017
1018
1019
1020
1021
1022
1023
1024
1025
1026
1027
1028
1029
1030
1031
1032
1033
1034
1035
1036
1037
1038
1039
1040
1041
1042
1043
1044
1045
1046
1047
1048
1049
1050
1051
1052
1053

EXAMPLE 5: THIS EXAMPLE SHOWS A READ DATA ERROR
WHICH OCCURRED, WITHOUT AN ACCOMPANING
STATUS ERROR, WHICH RESULTED IN A BAD RECORD.

DRIVE NO. 3 *SLAVE NO. 1 *D 4 *P 0 *F 14 *PATRN R
*BN 100 *RN 66-200 *RS 2000 *DE F

CN 0
G 11111111
B 00000000
CN 1
G 11111111
B 00000000
CN 2
G 11111111
B 00000000
CN 3
G 11111111
B 00000000
CN 4
G 11111111
B 00000000
CN 5
G 11111111
B 00000000
CN 6
G 11111111
B 00000000
CN 7
G 11111111
B 00000000

BAD RECORD

EXAMPLE 6: THE FOLLOWING EXAMPLE SHOWS THE
RESULT OF A SPACE OPERATION THAT
SHOULD HAVE SPACED REVERSE OVER
AN ENTIRE 100 RECORD BLOCK BUT
WHICH TERMINATED AT THE END OF 40
RECORDS. LEAVING A POSITION ERROR OF 40

DRIVE NO. 2 *SLAVE NO. 6 *D 2 *P 0 *F 14
*BN 3 *RN 100-100 *RS 1000 *SE R
ERR AMT 40

1054
1055
1056
1057
1058
1059
1060
1061
1062
1063
1064
1065
1066
1067
1068
1069
1070
1071
1072
1073
1074
1075
1076
1077
1078
1079
1080
1081
1082
1083
1084
1085
1086
1087
1088
1089
1090
1091
1092
1093
1094
1095
1096
1097
1098
1099
1100
1101

EXAMPLE 7: THIS EXAMPLE REFLECTS AN ERROR DETECTED WHILE WRITING A TAPE MARK (TM) AT THE END OF THE CURRENT DATA BLOCK PER OPTION RESPONSE TM=1. NOTE THAT THE TM RECORD NUMBER IS ONE GREATER THAN THE TOTAL NUMBER OF DATA RECORDS IN THE CURRENT BLOCK.

DRIVE NO. 1 *SLAVE NO. 1 *D 2 *P 0 *F 14
*BN 67 *RN 101-100 *RS 36 *WE TM
CS1 144226
CS2 300
DS 150604
ER 1000
WC 0

EXAMPLE 8: THIS EXAMPLE SHOWS TWO (2) PRINTOUTS REFLECTING A WRITE RETRY WHICH WAS NOT SUCCESSFUL THE FIRST TIME, BUT WHICH DID RECOVER ON THE SECOND. THE UNSUCCESSFUL RETRY IS LOGGED AS A SUSPECTED BAD TAPE SPOT BY ITS BLOCK AND RECORD NUMBER.

DRIVE NO. 0 *SLAVE NO. 2 *D 4 *P 0 *F 14 *PATRN 6
*BN 2 *RN 12-20 *RS 667 *WE
CS1 144260
CS2 100
DS 150640
ER 100
WC 0
ORIGINAL ERROR

DRIVE NO. 0 SLAVE NO. 2 *D 4 *P 0 *F 14 *PATRN 6
*BN 2 *RN 12-20 *RS 667 *WE
CS1 144260
CS2 100
DS 150640
ER 100
WC 0
SUSPECT BAD TAPE
RETRY: 0
REPT: 0
RECOVERED
RETRY: 1

1102
1103
1104
1105
1106
1107
1108
1109
1110
1111
1112
1113
1114
1115
1116
1117
1118
1119
1120
1121
1122
1123
1124
1125
1126
1127
1128
1129
1130
1131
1132
1133
1134
1135
1136

EXAMPLE 9: IF , DURING A WRITE RETRY THE BACKSPACE OR THE ERASE OPERATION RESULT IN AN ERROR, THE ERROR WILL BE PRINTED AND THE PROGRAM HALTED. THIS EXAMPLE SHOWS THE ERROR PRINT FOR A SPACE AND AN ERASE (2 EXAMPLES)

DRIVE NO. 1 *SLAVE NO. 1 *D 3 *P 0 *F 14
BN 12 *RN 8-64 *RS 500 *SE RTRY
ERR AMT 1

DRIVE NO. 1 *SLAVE NO. 1 *D 3 *P 0 *F 14
*BN 12 *RN 8-64 *RS 500 *ERASE
CS1 144224
CS2 100
DS 150600
ER 400
WC 0

EXAMPLE 10: THIS EXAMPLE SHOWS THE PRINTOUT FROM A REWIND OPERATION WHICH DOES NOT HAVE BOT SET AT THE END.

DRIVE NO. 2 *SLAVE NO. 3 *D 3 *P 0 *F 14
*BN 66 *RN 15-20 *RS 1000
NOT BOT ON REWIND: HALT

EXAMPLE 11: THIS EXAMPLE SHOWS THE PRINTOUT MADE WHEN THERE IS NO INTERRUPT RETURNED AT THE END OF AN OPERATION.

DRIVE NO. 7 *SLAVE NO. 7 *D 2 *P 1 *F 14
*BN 1 *RN 25-26 *RS 1200
NO INTERRUPT

1137
1138
1139
1140
1141
1142
1143
1144
1145
1146
1147
1148
1149
1150
1151
1152
1153
1154
1155
1156
1157
1158
1159
1160
1161
1162
1163
1164
1165
1166
1167
1168
1169
1170
1171
1172
1173
1174
1175
1176
1177
1178
1179
1180
1181
1182
1183
1184
1185
1186

10. STATISTICS PRINTOUT

THE PROGRAM, THROUGH ITS ERROR CHECKING, IS ABLE TO GATHER CERTAIN STATISTICS ABOUT THE PERFORMANCE OF EACH UNIT UNDER TEST. THIS INFORMATION IS PRINTED OUT WHENEVER A UNIT IS REWOUND FROM END OF TAPE, OR BECAUSE IT IS TO BE REMOVED FROM TESTING DUE TO SOME CATASTROPHIC ERROR. (POSITION LOST, BAD TAPE OVERFLOW) THE STATISTICS MAY BE PRINTED AT ANY TIME BY SETTING SWITCH 14 TO A ONE (1). THIS PRESENTS A PICTURE OF PERFORMANCE UP TO THIS TIME. THE STATISTICS WILL BE CLEARED UPON REWIND OF THE UNIT; BUT NOT BY SETTING SW 14.

STATISTICS PRINT EXAMPLE (A HEADER WILL PRECEED THE STATS)

DROPS: 0 3 0 0 0 6 45 0
PICKS: 1 0 0 0 0 0 0 2
RETRY: 1
WTERR: 2
REFWD: 3
SOFT: 2
HARD: 1
DEFWD: 0
REREV: 4
SOFT: 1
HARD: 3
DEREV: 0
2 BAD TAPE SPOTS
0 *BN 1 *RN 2
1 *BN 15 *RN 100

** NOTE ** DROPS AND PICKS REFLECT CORE BIT POSITIONS.
THE FOLLOWING IS A TABLE OF CORE BITS TO TRACK NUMBER.

TRACK NO.	7	6	5	3	9	1	8	2
CORE BIT	7	6	5	4	3	2	1	0

DROPS: NUMBER OF DATA BITS DROPPED: PER CORE BIT(SEE NOTE ABOVE)
PICKS: NUMBER OF DATA BITS PICKED UP: PER CORE BIT(SEE NOTE ABOVE)
RETRY: NUMBER OF WRITE RETRIES
WTERR: NUMBER OF WRITE ERRORS NOT ASSOCIATED WITH BAD TAPE
REFWD: NUMBER OF READ FORWARD STATUS ERRORS
REREV: NUMBER OF READ REVERSE STATUS ERRORS
SOFT: NUMBER OF RECOVERED READ ERRORS
HARD: NUMBER OF UNRECOVERED READ ERRORS
DEFWD: NUMBER OF FORWARD DATA ERRORS WITH NO ASSOCIATED STATUS ERROR
DEREV: NUMBER OF REVERSE DATA ERRORS WITH NO ASSOCIATED STATUS ERROR

1187
1188
1189
1190
1191
1192
1193
1194
1195
1196
1197
1198
1199
1200
1201
1202
1203
1204
1205
1206
1207
1208
1209
1210
1211
1212
1213
1214
1215
1216
1217
1218
1219
1220
1221
1222
1223
1224
1225
1226
1227
1228

11. AUTO SEQUENCE

THE AUTO SEQUENCE (START AT ADDRESS 240) WILL EXECUTE A PREDETERMINED TEST PLAN ON ALL AVAILABLE SLAVES ON EACH AVAILABLE TMO3. THE ONLY OPERATOR RESPONSE IS TO THE TYPED REQUESTS FOR THE RH ADDRESS, VECTOR, CONTINUOUS OR SINGLE CYCLE, AND NRZ ONLY. ALL SWITCHES REMAIN ACTIVE AND MAY BE USED NORMALLY; HOWEVER THE IDEA IS TO LEAVE ALL SWITCHES DOWN AND ALLOW FULL EXECUTION OF THE TEST PLAN FOR SYSTEM CHECKOUT.

SAMPLE START AT 240(8): AUTO SEQUENCE.

LOAD ADDRESS 240(8), SET SWITCHES TO ZERO, PRESS START:

TE16 AUTO SEQUENCE TEST
ENTER CONDITIONS IN OCTAL

REGISTER START = 172400(172440)
VECTOR ADDRESS = 224(CR)
NRZ ONLY: (0)
AUTO CONT: (1)

THIS EXAMPLE SHOWS AN AUTO SEQUENCE START WITH THE RH AT BUS ADDRESS 172440 AND A VECTOR OF 224. ALL AVAILABLE HARDWARE WILL BE TESTED CONTINUOUSLY IN BOTH NRZ AND PE MODE.

AS EACH TMO3 AND ITS SLAVES ARE FOUND, A DIVIDER LINE OF ASTERICKS WILL BE PRINTED FOLLOWED BY A PRINTOUT OF THE TMO3 AND ITS SLAVES BEING TESTED. AS EACH TMO3 AND ITS SLAVES ARE FINISHED, ANOTHER DIVIDER IS PRINTED BEFORE TESTING IS RESUMED ON THE NEXT AVAILABLE DRIVE.

WHEN ALL AVAILABLE HARDWARE HAS BEEN TESTED, A PRINTOUT OF END OF SEQUENCE WILL BE DONE AND THE PROGRAM WILL EITHER HALT (AUTO CONT = 0) OR RESTART WITH THE FIRST AVAILABLE UNIT (AUTO CONT = 1).

1229
1230
1231
1232
1233
1234
1235
1236
1237
1238
1239
1240
1241
1242
1243
1244
1245
1246
1247
1248
1249
1250
1251
1252
1253
1254
1255
1256
1257
1258
1259
1260
1261

AUTO SEQUENCE TEST PLAN:

THE AUTO SEQUENCE WILL EXECUTE BOTH AN NRZ AND A PE CYCLE. EACH CYCLE WILL BE STARTED FROM BOT AND CONSIST OF VARIOUS DATA PATTERNS INTENDED TO BE WORST CASE FOR THAT PARTICULAR MODE.

1. NRZ CYCLE:

SIX (6) BLOCKS OF ONE HUNDRED (100) RECORDS OF FOUR THOUSAND (4000) CHARACTERS FOR EACH OF THE FOUR DATA PATTERNS.

PATTERN 1: ALL ONES DATA IN ALL BYTES
PATTERN 10: WALKING ONE/ALL ONE
PATTERN 14: WALKING ZERO/ALL ZERO
RANDOM DATA: RANDOM

2. PE CYCLE: (IF NRZ ONLY = 0)

SIX BLOCKS OF ONE HUNDRED (100) RECORDS OF FOUR THOUSAND (4000) CHARACTERS EACH FOR EACH OF THREE DATA PATTERNS, THEN RANDOM DATA BLOCKS TO END OF TAPE.

PATTERN 10: WALKING ONE/ALL ONE
PATTERN 14: WALKING ZERO/ALL ZERO
PATTERN 15: THREE (3) 0 CHARACTERS, TWO (2) ALL CHARACTERS, THREE 0
THEN COMPLIMENT PATTERN. REPEATED FOR A FULL BUFFER
RANDOM DATA: RANDOM

1262
1263
1264
1265
1266
1267
1268
1269
1270
1271
1272
1273
1274
1275
1276
1277
1278
1279
1280
1281
1282
1283
1284
1285
1286
1287
1288
1289
1290
1291
1292
1293
1294
1295
1296
1297
1298
1299
1300
1301
1302
1303
1304
1305
1306
1307
1308
1309

12. TESTING PROCEDURES

AS PREVIOUSLY STATED THIS PROGRAM CONTAINS NO FIXED TESTS. THE ENTIRE TEST CYCLE TO BE EXECUTED IS DESCRIBED BY THE OPERATOR THROUGH RESPONSES TO TELETYPE REQUESTS FOR PARAMETERS AND CONSOLE SWITCH SETTINGS FOR OPERATION. THE OPERATION SELECTED WILL BE EXECUTED WITH THE PARAMETERS ENTERED CONTINUOUSLY ON EACH AVAILABLE UNIT, ONE BLOCK AT A TIME, UNTIL STOPPED BY THE OPERATOR. THE OPERATION MAY BE CHANGED DYNAMICALLY BY CHANGING THE CONSOLE SWITCHES AT ANY TIME. THE PROGRAM WILL ATTEMPT TO PERFORM ANY OPERATION SET AND THEREFORE CAUTION SHOULD BE TAKEN TO ASSURE THAT THE UNIT IS CAPABLE OF PERFORMING AS REQUESTED. FOR INSTANCE, ONE SHOULD NOT ATTEMPT TO PERFORM READ OPERATIONS ON A TAPE WHICH HAS NOT BEEN WRITTEN AS THE DATA, IF ANY, IS UNPREDICTABLE. HOWEVER, IF A TAPE HAS BEEN WRITTEN WITH THIS PROGRAM, IT CAN BE READ AS OFTEN AS DESIRED WITHOUT BEING REWRITTEN. THIS IS A GOOD PROCEDURE TO USE FOR TESTING TAPE COMPATABILITY. SCOPING OF TAPE UNITS BECOMES SIMPLE; BY SETTING THE DESIRED OPERATION AND ITS PARAMETER, A UNIT MAY BE CONTINUOUSLY EXERCISED IN ANY MANNER DESIRED. BY USING THE VARIOUS ERROR CONTROL SWITCHES AND ENTERING THE NEEDED STALL, ANY FUNCTION CAN BE SCOPED RATHER EASILY. RELIABILITY TESTING CAN BE PERFORMED BY USE OF THE RANDOMIZATION CAPABILITY. PERHAPS A CYCLE OF RANDOM TESTING MIGHT BE SET UP AND ALLOWED TO RUN FOR SOME PERIOD OF TIME, THE STATISTICAL COLLECTION OF DROPS AND PICKS IS THEN SIGNIFICANT. INTERMITTANT PROBLEMS CAN BE FOUND BY SETTING THE DESIRED OPERATION IN MOTION AND DISALLOWING ERROR PRINTOUTS WHILE ALLOWING A HALT ON ERROR. THE ERROR THAT CAUSED THE HALT CAN BE PRINTED BY RESETTING CONSOLE SWITCH TEN AND PRESSING CONTINUE. IF SOME PARTICULAR DATA PATTERN SHOULD BE CAUSING DATA ERROR, USE OF THE YOZZLE SWITCH AND ITS ASSOCIATED STALL WILL TO ALLOW SCOPING OF THIS PARTICULAR RECORD.

AS YOU SEE, THERE ARE MYRIAD TESTING PROCEDURES WHICH COULD BE PERFORMED. THE PARAMETERS, TAPE OPERATIONS, ERROR EXAMINATION AND REPORTING ARE ALL AT YOUR DISCRETION.

TRY IT, YOU'LL LIKE IT.

%

1310
1311
1312
1313
1314
1315
1316
1317
1318
1319
1320
1321
1322
1323
1324
1325
1326
1327
1328
1329
1330
1331
1332
1333
1334
1335
1336
1337
1338
1339
1340
1341
1342
1343
1344
1345
1346
1347
1348
1349
1350
1351
1352
1353
1354
1355
1356
1357
1358
1359
1360
1361
1362

```
.LIST BIN,LOC,SEQ  
.TITLE CZTEDBO TMO3-TE16/TU77 DRT  
:DATA RELIABILITY TEST  
:AC-A800B-MC  
:21 FEB 1977  
:J.G.ADAMS  
:REVISED (++) J.G.ADAMS MAY 1978  
:++B  
:++B  
:++B  
:++B  
:++B
```

1)INCORRECT RECORD COUNT
STORED WHEN EOT REACHED ON WRITE
2)ADJUST STACK PTR ON BAD TAPE OVFLW
3)ADDED TU77 TEST CAPABILITY
4)DOES NOT GENERATE LRC/CRC ON FIRST
RECORD IN AUTO ACCEPT MODE

```
.MCALL .SACT11,.$EOP,$SAVE,$RESTORE,$CHAIN  
.NLIST MC  
.LIST ME  
.ENABLE ABS,AMA
```

:CONSOLE SWITCHES*****

```
:SW15: 1=STOP ON ERROR  
: 0=CONTINUE ON ERROR  
:SW14: 1=PRINT READ/WRITE STATS  
: 0=DO NOT PRINT STATS  
:SW13: 1=DO NOT CHECK DATA  
: 0=CHECK DATA  
:SW12: 1=DO NOT CHECK WRITE ERRORS  
: 0=CHECK WRITE ERRORS  
:SW11: 1=DO NOT CHECK READ ERRORS  
: 0=CHECK READ ERRORS  
:SW10: 1=DO NOT PRINT ERRORS  
: 0=PRINT ERRORS  
:SW9: 1=REWIND TAPE  
: 0=DO NOT REWIND  
:SW8: 1=USE RANDOM DATA  
: 0=USE FIXED DATA PATTERN  
:SW7: 1=USE RANDOM CHARACTER COUNT  
: 0=USE FIXED CHAR COUNT  
:SW6: 1=USE RANDOM RECORD COUNT  
: 0=USE FIXED RECORD COUNT  
:SW5: 1=YOZZLE ON CURRENT RECORD  
: 0=DO NOT YOZZLE  
:SW4: 1=DO BOTH READ AND WRITE RETRIES  
: 0=INHIBIT RETRIES  
:SW3: 1=DO NOT READ FORWARD  
: 0=READ FORWARD  
:SW2: 1=DO NOT READ REVERSE  
: 0=READ REVERSE  
:SW1: 1=READ FORWARD FIRST  
: 0=READ REVERSE FIRST  
:SW0: 1=DO NOT WRITE  
: 0=WRITE
```

;IF SWR <15::00> = 177777 OR NOT AVAILABLE USE SOFTWARE SWITCH REGISTER


```
1409
1410
1411
1412      000020
1413 000020 022772
1414 000022 000340
1415
1416      000004
1417      000034
1418 000034 023124
1419 000036 000340
1420      104400
1421
1422
1423      000040
1424      000046
1425 000046 004676
1426      000052
1427 000052 000000
1428      000040
1429
1430
1431      000060
1432 000060 020734
1433 000062 000340
1434
1435
1436
1437      000176
1438 000176 000000
1439
1440
1441      000200
1442 000200 000137 003022
1443
1444      000204
1445 000204 000137 003136
1446
1447      000210
1448 000210 005037 014404
1449 000214 000137 003144
1450
1451
1452
1453      000224
1454 000224 021160
1455 000226 000340
1456
1457
1458
1459      000240
1460 000240 005237 000734
1461 000244 000137 003122

;TRAP CATCHERS*****
.=20
.WORD TTOUT ;SET IOT TRAP TO TTOUT ROUTINE
.WORD 340 ;PRIORITY LEVEL 7

TYPE=IOT ;EQUATE TYPE TO AN IOT INSTRUCTION
.=34
.WORD OCTP ;SET TRAP TRAP TO OCTP ROUTINE
.WORD 340
TYPOCT=TRAP ;EQUATE TYPOCT TO TRAP INSTRUCTION

;ACT11 HOOK *****
$SVPC= ;SAVE CURRENT LOCATION CTR
.=46
.WORD $ENDAD ;SET LOCATION 46
.=52
.WORD 0 ;SET LOCATION 52 = 0
.= $SVPC ;RESTORE LOCATION CTR

;TTY INTERRUPT VECTOR*****
.=60
.WORD TTINT ;TTY INTERRUPT HANDLER ADDRESS
.WORD 340 ;PRIORITY LEVEL 7

;SOFTWARE SWITCH REGISTER*****
;INVOKED IF SWR <15::00> = 177777 OR NOT AVAILABLE
.=176
SWREG: .WORD 0

;START ADDRESS*****
.=200
JMP START ;ENTER PARAMETERS VIA TTY

.=204
JMP STARTC ;USE FIXED PARAMETERS; HOLD DATA

.=210
CLR RDFL
JMP STARTA ;USE FIXED PARAMETERS; NEW DATA

;MAG TAPE INTERRUPT VECTOR*****
.=224
MTINT ;MAG TAPE INTERRUPT HANDLER ADDRESS
340

;AUTO SEQUENCE START*****
.=240
INC ASEQF ;SET AUTO SEQUENCE FLAG
JMP STAUT ;GO TO START OF AUTO SEQUENCE
```



```
1462 ;SHORT CONVERSATION RESTART*****
1463
1464 . =300
1465 000300 005237 013444 INC SCVFL ;SET SHORT CONVERSATION FLAG
1466 000304 000137 003022 JMP START ;ENTER SHORT PARAMETER LIST
1467
1468 . =510
1469 ;TU16 REGISTER EQUIVS*****
1470
1471 000510 172440 C1: 172440
1472 000512 172442 WC: 172442
1473 000514 172444 BA: 172444
1474 000516 172446 FC: 172446
1475 000520 172450 CS: 172450
1476 000522 172452 DS: 172452
1477 000524 172454 ER: 172454
1478 000526 172456 AS: 172456
1479 000530 172460 CC: 172460
1480 000532 172462 DB: 172462
1481 000534 172464 MR: 172464
1482 000536 172466 DI: 172466
1483 000540 172470 SN: 172470
1484 000542 172472 TC: 172472
1485
1486 ;CONSTANTS*****
1487
1488 000544 172440 REGS: 172440 ;STARTING REGISTER ADDRESS (CS1)
1489 000546 000224 VECT: 224 ;VECTOR ADDRESS (RH INTERRUPT)
1490 000550 000000 DVN: 0 ;DRIVE NUMBER
1491 000552 000000 UDES: 0 ;UNIT DESCRIPTION (PARITY,DENSITY,UNIT,FORMAT)
1492 000554 000100 RCNT: 100 ;RECORD COUNTER
1493 000556 177400 FMCNT: 177400 ;NUMBER OF CHAR (4 - 4000) OCTAL IN TWOS COMPLEMENT
1494 000560 000001 PATRN: 1 ;DATA PATTERN SELECTOR (0 - 15) OCTAL
1495 000562 000000 RDCMD: 0 ;READ COMMAND
1496 000564 000001 TMEX: 1 ;TAPE MARK FLAG: 1=TM 0=NO TM
1497 000566 000000 CRCC: 0 ;CRC CORRECTION FLAG (YES=1,NO=0)
1498 000570 000000 INTRF: 0 ;INTERCHANGE READ 1=YES 0=NO
1499 000572 000000 SPFLG: 0 ;SINGLE PASS 1=YES 0=NO
1500 000574 000001 RSTAL: 1 ;READ STALL
1501 000576 000001 WSTAL: 1 ;WRITE STALL
1502 000600 000001 TSTAL: 1 ;TURN AROUND STAL
1503 000602 002000 YSTAL: 2000 ;YOZZLE STAL
1504 000604 000010 RETRY: 10 ;READ RETRY NUMBER
1505 000606 177776 PSW: 177776 ;PROCESSOR STATUS
1506 000610 177570 SWR: 177570 ;CONSOLE SWITCHES
1507 000612 177560 TKS: 177560 ;TTY READ STATUS REGISTER
1508 000614 177562 TKB: 177562 ;TTY READ BUFFER
1509 000616 177564 TPS: 177564 ;TTY PUNCH STATUS REGISTER
1510 000620 177566 TPB: 177566 ;TTY PUNCH OUTPUT REGISTER
1511 000622 177550 PRS: 177550 ;H/S READER STATUS REGISTER
1512 000624 177552 PRB: 177552 ;H/S READER BUFFER
1513 000626 153624 RANBAS: 153624 ;RANDOM NUMBER GENERATOR BASE
1514 000630 032561 RANSAV: 032561 ;RANDOM NUMBER BUFFER
1515 000632 000100 RCSAV: 100 ;RECORD COUNT SAVE
1516 000634 177400 FCSAV: 177400 ;FRAME COUNT SAVE
1517
```



```
1518                                     ;FLAGS AND COUNTERS*****
1519
1520 000636 000000          TINF: 0          ;TTY ENTRY FLAG
1521 000640                STFLG:          ;
1522 000640 000000          TOB: 0          ;TTY OUTPUT BUFFER
1523 000642 000000          TIB: 0          ;TTY INPUT BUFFER
1524 000644 000000          TEMP1: 0         ;TEMP STORAGE
1525 000646 000000          TEMP2: 0         ;TEMP STORAGE
1526 000650 000000          TEMP3: 0         ;TEMP STORAGE
1527 000652 000000          EMADDR: 0        ;ERROR MSG ADDRESS STORAGE
1528 000654 000000          BLCNTR: 0        ;BLOCK COUNTER
1529 000656 000000          BBC: 0          ;BAD RECORD COUNTER
1530 000660 000000          EOTREC: 0        ;EOT FLAG
1531 000662 000000          RTRN: 0          ;INTERRUPT RETURN STORAGE
1532 000664 000000          HDRFL: 0        ;HEADER FLAG
1533 000666 000000          STAL: 0        ;DELAY STORAGE
1534 000670 000000          PFLG: 0        ;PRINT FLAG
1535 000672 000000          MTC1: 0        ;MAG TAPE CONT REGISTER BUFFER
1536 000674 000000          UNP: 0          ;UNIT TABLE POINTER
1537 000676 000000          TMFLG: 0        ;TAPE MARK FLAG
1538 000700 000000          RPCNT: 0        ;REPEAT COUNTER
1539 000702 000000          RTCNT: 0        ;RETRY COUNTER
1540 000704 000000          DERFL: 0        ;DATA ERROR FLAG
1541 000706 000000          SERFL: 0        ;STATUS ERROR FLAG
1542 000710 000000          BCNT: 0         ;BIT COUNTER
1543 000712 000000          RTYFL: 0        ;RETRY FLAG
1544 000714 000000          UPS: 0         ;UNIT POINTER SAVE
1545 000716 000000          BDPP: 0        ;BITS DROPPED POINTER
1546 000720 000000          BPKP: 0        ;BITS PICKED POINTER
1547 000722 000000          ERSV: 0        ;ERROR SAVE LOC
1548 000724 000000          BTFLG: 0        ;BAD TAPE FLAG
1549 000726 000000          BTSTF: 0       ;STATISTIC PRINT FLAG
1550 000730 000000          BTPT: 0        ;BAD TAPE POINTER
1551 000732 000000          ERTFL: 0        ;ERASE FLAG
1552 000734                ENDFLG:          ;
1553 000734 000000          ASEQF: 0        ;AUTO SEQ FLAG
1554 000736 000000          ABLCNT: 0       ;AUTO BLOCK COUNTER
1555 000740 000000          ASEQCF: 0       ;AUTO SEQ CONTINUOUS FLAG
```


1556
1557
1558
1559 000742 000000
1560 000744 000000
1561 000746 000000
1562 000750 000000
1563 000752 000000
1564 000754 000000
1565 000756 000000
1566 000760 000000
1567 000762 177777
1568
1569
1570
1571 000764 001204
1572 000766 001224
1573 000770 001244
1574 000772 001264
1575 000774 001304
1576 000776 001324
1577 001000 001344
1578 001002 001364
1579 001004 001404
1580 001006 001424
1581 001010 001444
1582 001012 001464
1583 001014 001504
1584 001016 001524
1585 001020 001544
1586 001022 001564
1587
1588
1589
1590 001024 001604
1591 001026 001710
1592 001030 002014
1593 001032 002120
1594 001034 002224
1595 001036 002330
1596 001040 002434
1597 001042 002540
1598
1599
1600
1601
1602 001044
1603 001044 000000
1604 001046 000000
1605 001050 000000
1606 001052 000000
1607 001054 000000
1608 001056 000000
1609 001060 000000
1610 001062 000000
1611

;UNIT ORDER AND DESCRIPTION TABLE *****

UN1: 0 ;THIS TABLE IS LOADED
UN2: 0 ;WITH UNIT NUMBERS AND
UN3: 0 ;THEIR DESCRIPTIONS IN
UN4: 0 ;THE ORDER THAT THEY
UN5: 0 ;WILL BE TESTED
UN6: 0
UN7: 0
UN8: 0
UNX: -1

;UNIT DROPS AND PICKS POINTERS*****

PIK1: BP00
PIK2: BP10
PIK3: BP20
PIK4: BP30
PIK5: BP40
PIK6: BP50
PIK7: BP60
PIK8: BP70
DRP1: BD00
DRP2: BD10
DRP3: BD20
DRP4: BD30
DRP5: BD40
DRP6: BD50
DRP7: BD60
DRP8: BD70

;UNIT BAD TAPE POINTERS*****

BTADDR: BT00
BT01
BT02
BT03
BT04
BT05
BT06
BT07

;UNIT WRITE RETRY COUNTER*****

;SET START OF STATISTICS TABLE

STTBL:
RTY1: 0
RTY2: 0
RTY3: 0
RTY4: 0
RTY5: 0
RTY6: 0
RTY7: 0
RTY8: 0


```
1612                                     ;UNIT WRITE ERRORS*****
1613
1614 001064 000000      WTER1: 0
1615 001066 000000      WTER2: 0
1616 001070 000000      WTER3: 0
1617 001072 000000      WTER4: 0
1618 001074 000000      WTER5: 0
1619 001076 000000      WTER6: 0
1620 001100 000000      WTER7: 0
1621 001102 000000      WTER8: 0
1622
1623                                     ;UNIT READ FORWARD ERRORS*****
1624
1625 001104 000000      RDER1: 0
1626 001106 000000      RDER2: 0
1627 001110 000000      RDER3: 0
1628 001112 000000      RDER4: 0
1629 001114 000000      RDER5: 0
1630 001116 000000      RDER6: 0
1631 001120 000000      RDER7: 0
1632 001122 000000      RDER8: 0
1633
1634                                     ;UNIT DATA ERRORS FORWARD*****
1635
1636 001124 000000      DATER1: 0
1637 001126 000000      0
1638 001130 000000      0
1639 001132 000000      0
1640 001134 000000      0
1641 001136 000000      0
1642 001140 000000      0
1643 001142 000000      0
1644
1645                                     ;UNIT READ REVERSE ERRORS*****
1646
1647 001144 000000      RDERR1: 0
1648 001146 000000      0
1649 001150 000000      0
1650 001152 000000      0
1651 001154 000000      0
1652 001156 000000      0
1653 001160 000000      0
1654 001162 000000      0
1655
1656                                     ;UNIT DATA ERRORS REVERSE*****
1657
1658 001164 000000      DEREV1: 0
1659 001166 000000      0
1660 001170 000000      0
1661 001172 000000      0
1662 001174 000000      0
1663 001176 000000      0
1664 001200 000000      0
1665 001202 000000      0
```


			:DROPS + PICKS PER CHANNEL PER UNIT*****
1666			
1667			
1668	001204	000000	BP00: 0
1669		001224	.=.+16
1670	001224	000000	BP10: 0
1671		001244	.=.+16
1672	001244	000000	BP20: 0
1673		001264	.=.+16
1674	001264	000000	BP30: 0
1675		001304	.=.+16
1676	001304	000000	BP40: 0
1677		001324	.=.+16
1678	001324	000000	BP50: 0
1679		001344	.=.+16
1680	001344	000000	BP60: 0
1681		001364	.=.+16
1682	001364	000000	BP70: 0
1683		001404	.=.+16
1684	001404	000000	BD00: 0
1685		001424	.=.+16
1686	001424	000000	BD10: 0
1687		001444	.=.+16
1688	001444	000000	BD20: 0
1689		001464	.=.+16
1690	001464	000000	BD30: 0
1691		001504	.=.+16
1692	001504	000000	BD40: 0
1693		001524	.=.+16
1694	001524	000000	BD50: 0
1695		001544	.=.+16
1696	001544	000000	BD60: 0
1697		001564	.=.+16
1698	001564	000000	BD70: 0
1699		001604	.=.+16
1700			
1701			

1702				
1703				;UNIT BAD TAPE COUNTER:16 PER SLAVE*****
1704				
1705	001604	000000	BT00:	0
1706		001710		.=.+102
1707	001710	000000	BT01:	0
1708		002014		.=.+102
1709	002014	000000	BT02:	0
1710		002120		.=.+102
1711	002120	000000	BT03:	0
1712		002224		.=.+102
1713	002224	000000	BT04:	0
1714		002330		.=.+102
1715	002330	000000	BT05:	0
1716		002434		.=.+102
1717	002434	000000	BT06:	0
1718		002540		.=.+102
1719	002540	000000	BT07:	0
1720		002644		.=.+102
1721				
1722				;UNIT END OF TAPE COUNTERS 1 PER SLAVE*****
1723				
1724	002644	000000	EOTC0:	0
1725	002646	000000		0
1726	002650	000000		0
1727	002652	000000		0
1728	002654	000000		0
1729	002656	000000		0
1730	002660	000000		0
1731	002662	000000		0
1732				
1733				;UNIT READ FORWARD SOFT ERROR*****
1734				
1735	002664	000000	RF SOFT:	0
1736	002666	000000		0
1737	002670	000000		0
1738	002672	000000		0
1739	002674	000000		0
1740	002676	000000		0
1741	002700	000000		0
1742	002702	000000		0
1743				
1744				;UNIT READ REVERSE SOFT ERROR*****
1745				
1746	002704	000000	RRSOFT:	0
1747	002706	000000		0
1748	002710	000000		0
1749	002712	000000		0
1750	002714	000000		0
1751	002716	000000		0
1752	002720	000000		0
1753	002722	000000		0
1754				


```
1755
1756                ;UNIT READ FORWARD HARD ERROR*****
1757
1758 002724 000000    RFHARD: 0
1759 002726 000000    0
1760 002730 000000    0
1761 002732 000000    0
1762 002734 000000    0
1763 002736 000000    0
1764 002740 000000    0
1765 002742 000000    0
1766
1767                ;UNIT READ REVERSE HARD ERROR*****
1768
1769 002744 000000    RRHARD: 0
1770 002746 000000    0
1771 002750 000000    0
1772 002752 000000    0
1773 002754 000000    0
1774 002756 000000    0
1775 002760 000000    0
1776 002762 000000    0
1777                ;SET END OF STATISTICS TABLE
1778 002764    ENDTBL:
1779
1780                ;DATA PATTERN GENERATORS*****
1781
1782 002764 002764    DATBL: .           ;ENTRY TABLE
1783 002766 013656    DATA0: DAT0       ;EXTERNAL INPUT FROM H/S READER(SEE MAINDEC-11-DZTUF)
1784 002770 014016    DATA1: DAT1       ;ALL ONES
1785 002772 014036    DATA2: DAT2       ;ALL ZEROS
1786 002774 014042    DATA3: DAT3       ;WALKING ONE
1787 002776 014066    DATA4: DAT4       ;WALKING ZERO
1788 003000 014076    DATA5: DAT5       ;ALTERNATING ONE/ZERO
1789 003002 014104    DATA6: DAT6       ;ALTERNATING ZERO/ONE
1790 003004 014112    DATA7: DAT7       ;ALTERNATING ONE/ZERO IN ALTERNATING CHARACTERS
1791 003006 014140    DATA10: DAT10     ;WALKING ONE/ALL ONE IN ALTERNATING CHARACTERS
1792 003010 014170    DATA11: DAT11     ;ALL BITS 0-377
1793 003012 014210    DATA12: DAT12     ;ALL BITS 377-0
1794 003014 014232    DATA13: DAT13     ;ALTERNATING CHARACTERS 0 AND 377
1795 003016 014242    DATA14: DAT14     ;WALKING ZERO/ALL ZERO IN ALTERNATING CHARACTERS
1796 003020 014272    DATA15: DAT15     ;AUTO SEQUENCE PATTERN 0,0,-1,-1,-1,0,0
1797
```


1798
1799
1800
1801
1802
1803
1804
1805
1806
1807
1808
1809
1810
1811
1812
1813
1814
1815
1816
1817
1818
1819
1820
1821
1822
1823
1824
1825
1826
1827
1828
1829
1830
1831
1832
1833
1834
1835
1836
1837
1838
1839
1840
1841
1842
1843
1844
1845
1846
1847
1848
1849
1850
1851
1852
1853

.EVEN
:*****
:PROGRAM START AND SEQUENCE FORMATTER:
:
:THIS ROUTINE IS USED TO PERFORM ALL HOUSEKEEPING,
:DECIDE WHICH TRANSPORT TO TEST AND ITS AVAILABILITY,
:LOAD THE WRITE BUFFER WITH THE SELECTED DATA PATTERN,
:GENERATE ANY RANDOM NUMBER AND THEN EXECUTE
:THE TEST CYCLE REQUESTED BY THE SWITCH SETTING.
:AT THE END OF THE TEST CYCLE THE NEXT UNIT IS SELECTED
:AND CHECKED FOR AVAILABILITY AND THE TEST CYCLE IS
:EXECUTED ON IT.
:THE READ WRITE STATS MAY BE PRINTED AT THE END OF
:EACH TEST CYCLE VIA CONSOLE SWITCH FOURTEEN (14).
:*****

:START 200, & 300*****
START: MOV #500,SP ;SET STACK PTR
CLR ASEQF ;CLEAR AUTO SEQUENCE FLAG
CLR (PC)+ ;CLEAR CHAIN INDICATOR
CHNFLG: .WORD 0 ;CHAIN MODE INDICATOR
;1/0 = CHAIN/NOT CHAIN MODE
;BRANCH IF IN DUMP MODE
TST @#42
BEQ 50\$
MOV #SWREG,SWR ;: INVOKE SOFTWARE SWR
INC CHNFLG ;:SET CHNFLG = CHAIN MODE
JMP 3\$;:GO TO CHAIN ADDRESS
50\$:
3\$: CMPB #6,@#41 ;:BRANCH IF LOADED VIA TMDP
BNE 4\$
TYPE,MSG120 ;ADVISE USER TO REMOVE TMDP FROM SLAVE
HALT
4\$: TST CHNFLG ;:SEE IF IN CHAIN MODE
BEQ STAUT
INC ASEQF ;:SET AUTO SEQUENCE FLAG
TYPE,MSG30 ;:TYPE TITLE
JMP ASEQ0 ;:GO TO AUTO SEQUENCER
:START 240*****
STAUT: MOV #1,TINF ;:SET TTY ENTRY FLAG
CLR RDFL ;:CLEAR RANDOM DATA FLAG
BR STARTB
:START 204*****
STARTC: CLR TINF ;:CLEAR TTY INPUT FLAG
BR STARTD
:START 210*****
STARTA: CLR TINF ;:CLEAR TTY ENTRY FLAG
STARTB: MOV #STFLG,R0 ;:GET STARTING ADDRESS OF FLAGS
MOV #ENDFLG-STFLG,R1
1\$: CLRB (R0)+ ;:CLEAR FLAGS AND COUNTERS
DEC R1
BNE 1\$
MOV #500,SP ;:SET STACK POINTER

1854	003172	004737	004122			JSR	PC,RANSET	:GO RESET RANDOM BASE
1855	003176	012700	001044			MOV	#STTBL,R0	:GET STARTING ADDRESS OF STAT TABLE
1856	003202	012701	001720			MOV	#ENDTBL-STTBL,R1	:AND # OF BYTES IN TABLE
1857	003206	105020			2\$:	CLRB	(R0)+	:CLEAR STATISTIC COUNTERS
1858	003210	005301				DEC	R1	
1859	003212	001375				BNE	2\$	
1860	003214	012700	000742			MOV	#UN1,R0	:SET ALL SLAVES ON-LINE
1861	003220	022710	177777		3\$:	CMP	#-1,(R0)	:BRANCH IF AT END OF TABLE
1862	003224	001403				BEQ	4\$	
1863	003226	042720	040000			BIC	#40000,(R0)+	:MARK SLAVE ON-LINE
1864	003232	000772				BR	3\$	
1865	003234	012737	177777	013652	4\$:	MOV	#-1,PATS	:PRESET PATTERN
1866	003242	012737	000001	000654	STARTE:	MOV	#1,BLCNTR	:PRESET BLOCK COUNTER
1867	003250	013746	000004		STARTD:	MOV	@#4,-(SP)	:SAVE ERROR TRAP VECTOR
1868	003254	013746	000006			MOV	@#6,-(SP)	
1869	003260	022737	000176	000610		CMP	#SWREG,SWR	:BRANCH IF SOFTWARE SWR
1870	003266	001413				BEQ	2\$:ALREADY SELECTED
1871	003270	012737	003314	000004		MOV	#1\$,@#4	:SET TIMEOUT TRAP TO 1\$ BELOW
1872	003276	005037	000006			CLR	@#6	
1873	003302	022777	177777	175300		CMP	#177777,@SWR	:BRANCH IF SWR = 177777 TRAP
1874	003310	001402				BEQ	2\$:IF NOT AVAIL (1\$) OTHERWISE
1875	003312	000404				BR	3\$:GO TO 3\$
1876	003314	022626			1\$:	CMP	(SP)+,(SP)+	:RESET STACK
1877	003316	012737	000176	000610	2\$:	MOV	#SWREG,SWR	:SET SWR = SOFTWARE SWR
1878	003324	012637	000006		3\$:	MOV	(SP)+,@#6	:RESTORE ERROR TRAP
1879	003330	012637	000004			MOV	(SP)+,@#4	
1880	003334	012706	000500			MOV	#500,SP	
1881	003340	004737	011750			JSR	PC,TINP	:GO GET PARAMETERS FROM TTY
1882	003344	012777	000040	175146		MOV	#40,@CS	:INITIALIZE
1883	003352	005000			STAUTO:	CLR	R0	:POINT TO FIRST ENTRY
1884	003354	022760	177777	000742	1\$:	CMP	#-1,UN1(R0)	:BRANCH IF LAST ENTRY
1885	003362	001406				BEQ	2\$	
1886	003364	042760	100000	000742		BIC	#100000,UN1(R0)	:CLEAR EOT FLAG
1887	003372	062700	000002			ADD	#2,R0	:POINT TO NEXT UNIT ENTRY
1888	003376	000766				BR	1\$:CONTINUE CLEARING
1889	003400	113737	004731	004730	2\$:	MOVB	REOTC+1,REOTC	:RESTORE EOT COUNTER
1890	003406	012777	000100	175176	START1:	MOV	#100,@TKS	:SET KEYBOARD IE BIT
1891	003414	013700	000674			MOV	UNP,R0	:R0 = UNIT TABLE POINTER
1892	003420	022760	177777	000742	STAR1A:	CMP	#-1,UN1(R0)	:BRANCH IF LAST ENTRY
1893	003426	001404				BEQ	STAR1B	
1894	003430	016037	000742	000552		MOV	UN1(R0),UDES	:LOAD NEXT UNIT DESCRIPTION
1895	003436	000445				BR	START4	
1896	003440	005237	000654		STAR1B:	INC	BLCNTR	:BUMP BLOCK COUNTER
1897	003444	005737	000734			TST	ASEQF	:SEE IF AUTO SEQ
1898	003450	001411				BEQ	STAR1C	:IF NOT: BR
1899	003452	023737	000654	000736		CMP	BLCNTR,ABLCNT	:SEE IF DONE SEQ
1900	003460	001005				BNE	STAR1C	:IF NOT: BR
1901	003462	005037	000654			CLR	BLCNTR	:RESET BLOCK CNTR
1902	003466	005037	000674			CLR	UNP	:RESET UNIT POINTER
1903	003472	000207				RTS	PC	:RETURN TO AUTO SEQ
1904	003474	005037	000674		STAR1C:	CLR	UNP	
1905	003500	005000				CLR	R0	
1906	003502	016037	000742	000552		MOV	UN1(R0),UDES	:LOAD FIRST UNIT DESCRIPTION
1907	003510	105777	175074			TSTB	@SWR	:SEE IF RANDOM RECORD SIZE
1908	003514	100002				BPL	START2	:IF NOT: BR
1909	003516	004737	011664			JSR	PC,CCNTR	:GO GENERATE RANDOM RECORD SIZE

1910	003522	032777	000400	175060	START2:	BIT	#400,@SWR	:SEE IF RANDOM DATA
1911	003530	001402				BEQ	START3	:IF NOT: BR
1912	003532	004737	014342			JSR	PC,DATR	:GO GENERATE RANDOM DATA
1913	003536	032777	000100	175044	START3:	BIT	#100,@SWR	:SEE IF RANDOM RECORD COUNT
1914	003544	001402				BEQ	START4	:IF NOT: BR
1915	003546	004737	011724			JSR	PC,RCNTR	:GO GENERATE RANDOM RECORD COUNT
1916	003552	032760	140000	000742	START4:	BIT	#140000,UN1(R0)	:BRANCH IF UNIT AT EOT
1917	003560	001365				BNE	START7	:OR MARKED OFF-LINE
1918	003562	012777	000040	174730		MOV	#40,@CS	:DO A MASSBUS CLEAR
1919	003570	013777	000550	174722		MOV	DVN,@CS	:SET DRIVE NUMBER
1920	003576	013777	000552	174736		MOV	UDES,@TC	:SET SLAVE NUMBER
1921	003604	105777	174712		1\$:	TSTB	@DS	:SEE IF SLAVE AVAIL
1922	003610	100405				BMI	2\$:IF SO: BR
1923	003612	005337	000666			DEC	STAL	
1924	003616	001372				BNE	1\$:AWAIT TUR
1925	003620	000137	020312			JMP	OFFLINE	:GO MARK DRIVE OFF-LINE
1926	003624	004737	013472		2\$:	JSR	PC,DSUP	:GO SET UP WRITE DATA
1927	003630	004737	005236			JSR	PC,INIT	:INIT SLAVE
1928	003634	004737	004732			JSR	PC,RWIND	:REWIND
1929	003640	004737	005352			JSR	PC,WRITE	:WRITE
1930	003644	013737	000600	000666		MOV	TSTAL,STAL	:SET TURN AROUND DELAY
1931	003652	004737	011654			JSR	PC,STALL	:DELAY
1932	003656	004737	007210			JSR	PC,RSEQ	:GO TO READ SEQUENCER
1933	003662	013737	000600	000666		MOV	TSTAL,STAL	:SET TURN AROUND DELAY
1934	003670	004737	011654			JSR	PC,STALL	:DELAY
1935	003674	032777	040000	174706		BIT	#40000,@SWR	:SEE IF SHOULD PRINT STATISTICS
1936	003702	001414				BEQ	START7	:IF NOT: BR
1937	003704	012700	000001			MOV	#1,R0	:SET RECORD COUNTER TO 1
1938	003710	004737	022012			JSR	PC,PAPRT	:PRINT CYCLE NUMBER
1939	003714	004737	003744			JSR	PC,STP	:GO PRINT STATS
1940	003720	005237	000726			INC	BTSTF	:SET STAT ONLY PRINT
1941	003724	004737	007126			JSR	PC,BTPRT	:PRINT BAD TAPE STATS
1942	003730	005037	000726			CLR	BTSTF	:CLEAR FLAG
1943	003734	062737	000002	000674	START7:	ADD	#2,JNP	:POINT TO NEXT UNIT
1944	003742	000621			START8:	BR	START1	:CONTINUE


```
1945 ;***** SUBROUTINE TO PRINT STATISTICS *****
1946
1947 003744 004737 016370 STP: JSR PC,DPPRT ;PRINT DROPS AND PICKS
1948 003750 000004 025121 TYPE,MSG65 ;TYPE MSG
1949 003754 013700 000674 MOV UNP,R0
1950 003760 016003 001044 MOV RTY1(R0),R3
1951 003764 104400 TYPOCT ;PRINT RETRIES
1952 003766 000004 025232 TYPE,MSG73 ;TYPE MSG
1953 003772 016003 001064 MOV WTER1(R0),R3
1954 003776 104400 TYPOCT ;PRINT WRITE ERRORS
1955 004000 000004 025221 TYPE,MSG72 ;TYPE MSG
1956 004004 016003 001104 MOV RDER1(R0),R3
1957 004010 104400 TYPOCT ;PRINT READ FORWARD ERRORS
1958 004012 000004 025777 TYPE,MSG113 ;TYPE MSG
1959 004016 016003 002664 MOV RFSOFT(R0),R3
1960 004022 104400 TYPOCT ;PRINT FORWARD SOFT ERRORS
1961 004024 000004 026010 TYPE,MSG114 ;TYPE MSG
1962 004030 016003 002724 MOV RFHARD(R0),R3
1963 004034 104400 TYPOCT ;PRINT HARD FORWARE ERRORS
1964 004036 000004 025316 TYPE,MSG77 ;TYPE MSG
1965 004042 016003 001124 MOV DATER1(R0),R3
1966 004046 104400 TYPOCT ;PRINT DATA ERROR FORWARD NUMBER
1967 004050 000004 025153 TYPE,MSG68 ;TYPE MSG
1968 004054 016003 001144 MOV RDERR1(R0),R3
1969 004060 104400 TYPOCT ;PRINT REVESE ERROR NUMBER
1970 004062 000004 025777 TYPE,MSG113 ;TYPE MSG
1971 004066 016003 002704 MOV RRSOFT(R0),R3
1972 004072 104400 TYPOCT ;PRINT REVERSE SOFT ERROR
1973 004074 000004 026010 TYPE,MSG114 ;TYPE MSG
1974 004100 016003 002744 MOV RRHARD(R0),R3
1975 004104 104400 TYPOCT
1976 004106 000004 025305 TYPE,MSG76 ;TYPE MSG
1977 004112 016003 001164 MOV DEREV1(R0),R3
1978 004116 104400 TYPOCT ;PRINT DATA REVERSE ERROR NUMBER
1979 004120 000207 RTS PC ;RETURN
1980
1981 ;RANDOM BASE RESET*****
1982
1983 004122 012737 153624 000626 RANSET: MOV #153624,RANBAS ;RESET BASE
1984 004130 012737 032561 000630 MOV #32561,RANSAV ;RESET BUFFER
1985 004136 013737 000632 000554 MOV RCSAV,RCNT ;RESET RECORD COUNT
1986 004144 013737 000634 000556 MOV FCSAV,FMCNT ;RESET FRAME COUNT
1987 004152 000207 RTS PC
1988
```


1989
1990
1991
1992
1993
1994
1995
1996
1997
1998
1999
2000

```
*****  
:REWIND FROM EOT:  
:  
:WHEN ANY TRANSPORT BEING TESTED REACHES END OF TAPE  
:DURING A READ OR WRITE OPERATION, IT WILL BE REWOUND  
:AND FLAGGED AS UNAVAILABLE UNTIL ALL AVAILABLE UNITS  
:HAVE REACHED EOT AT WHICH TIME ALL TESTING WILL BE RESUMED  
:AT A BLOCK COUNT OF ONE (1). A MESSAGE WILL BE  
:PRINTED ON THE SUPERVISORS CONSOLE AS EACH UNIT REACHES  
:EOT AND IS REWOUND.  
*****
```

```
2001 004154 013777 000552 174360 REOT: MOV UDES,@TC ;LOAD TAPE CONTROL REGISTER  
2002 004162 013700 000674 MOV UNP,R0 ;GET UNIT POINTER  
2003 004166 032760 040000 000742 BIT #40000,UN1(R0) ;BRANCH IF UNIT MARKED OFF-LINE  
2004 004174 001014 BNE 2$  
2005 004176 012777 000011 174304 MOV #11,@C1 ;DRIVE CLEAR  
2006 004204 105777 174312 1$: TSTB @DS ;WAIT FOR DRY  
2007 004210 100375 BPL 1$  
2008 004212 012777 000007 174270 MOV #7,@C1 ;START REWIND  
2009 004220 005737 000724 TST BTFLG ;SEE IF BAD TAPE OVERFLOW REWIND  
2010 004224 001004 BNE 3$ ;IF SO: BR  
2011 004226 013700 000660 2$: MOV EOTREC,R0  
2012 004232 042700 100000 BIC #100000,R0 ;SET RECORD NUMBER OF EOT  
2013 004236 005037 000660 3$: CLR EOTREC ;CLEAR EOT INDICATOR & REC COUNT  
2014 004242 004737 022012 JSR PC,PAPRT ;PRINT HEADER  
2015 004246 022737 000002 000724 CMP #2,BTFLG ;SEE IF POSITION ERROR  
2016 004254 001004 BNE 4$ ;IF NOT: BR  
2017 004256 012737 025672 004306 MOV #MSG109,6$ ;SET POSITION ERROR MSG  
2018 004264 000407 BR 5$  
2019 004266 022737 000001 000724 4$: CMP #1,BTFLG ;SEE IF BAD TAPE OVERFLOW  
2020 004274 001006 BNE REOT1C ;IF NOT: BR  
2021 004276 012737 025525 004306 MOV #MSG106,6$ ;SET BAD TAPE OVERFLOW MSG  
2022 004304 000004 5$: TYPE ;TYPE MSG  
2023 004306 000000 6$: .WORD 0 ;WILL CONTAIN MESSAGE ADDRESS  
2024 004310 000411 BR REOT1E  
2025 004312 000004 023656 REOT1C: TYPE,MSG20 ;TYPE EOT MSG  
2026 004316 013704 000674 MOV UNP,R4  
2027 004322 005264 002644 INC EOTC(R4) ;BUMP CNTR  
2028 004326 016403 002644 MOV EOTC(R4),R3  
2029 004332 104400 TYPOCT ;PRINT EOT CNTR  
2030 004334 000004 025550 REOT1E: TYPE,MSG16A ;TYPE MSG  
2031 004340 005037 000724 CLR BTFLG ;CLEAR BAD TAPE FLAG  
2032 004344 004737 003744 JSR PC,STP ;PRINT STATS  
2033 004350 004737 007126 JSR PC,BTPRT ;PRINT BAD TAPE STATS  
2034 004354 013700 000674 REOT2: MOV UNP,R0 ;GET UNIT POINTER  
2035 004360 032760 040000 000742 BIT #40000,UN1(R0) ;BRANCH IF UNIT MARKED OFF-LINE  
2036 004366 001010 BNE REOT2A  
2037 004370 105777 174126 TSTB @DS ;BRANCH IF DRY SET  
2038 004374 100405 BMI REOT2A  
2039 004376 005337 000666 DEC STAL  
2040 004402 001364 BNE REOT2 ;WAIT DRY  
2041 004404 000137 020312 JMP OFFLINE ;GO MARK SLAVE OFFLINE  
2042  
2043 004410 105337 004730 REOT2A: DECB REOTC ;SEE IF LAST UNIT TO REACH EOT  
2044 004414 001410 BEQ REOT3 ;IF SO: BR
```


2045	004416	013700	000674			MOV	UNP,RO	
2046	004422	052760	100000	000742		BIS	#100000,UN1(RO)	:SET EOT FLAG
2047	004430	005726				TST	(SP)+	:RESET STACK POINTER
2048	004432	000137	003734			JMP	START7	:GO TO NEXT UNIT
2049	004436	113737	004731	004730	REOT3:	MOVB	REOTC+1,REOTC	:RESTORE UNITS EOT COUNTER
2050	004444	005037	000674			CLR	UNP	
2051	004450	005000				CLR	RO	:POINT TO FIRST UNIT
2052	004452	016037	000742	000552	REOT4:	MOV	UN1(RO),UDES	:LOAD UNIT DESCRIPTION
2053	004460	013777	000552	174054		MOV	UDES,@TC	:SELECT SLAVE
2054	004466	032760	040000	000742		BIT	#40000,UN1(RO)	:BRANCH IF UNIT NOT MARKED OFF-LINE
2055	004474	001412				BEQ	1\$	
2056	004476	032777	010000	174016		BIT	#10000,@DS	:BRANCH IF MEDIUM NOT ON LINE
2057	004504	001427				BEQ	10\$	
2058	004506	062737	000401	004730		ADD	#401,REOTC	:INCREMENT # OF UNITS UNDER TEST
2059	004514	042760	140000	000742		BIC	#140000,UN1(RO)	:MARK UNIT BACK ON-LINE
2060	004522	012777	000011	173760	1\$:	MOV	#11,@C1	:DRIVE CLEAR
2061	004530	105777	173766		2\$:	TSTB	@DS	:WAIT FOR DRIVE READY
2062	004534	100375				BPL	2\$	
2063	004536	012777	000007	173744		MOV	#7,@C1	:REWIND UNIT
2064	004544	032777	000002	173750	3\$:	BIT	#2,@DS	:WAIT FOR BOT TO SET
2065	004552	001774				BEQ	3\$	
2066	004554	032777	020000	173740	4\$:	BIT	#20000,@DS	:WAIT FOR PIP TO CLEAR
2067	004562	001374				BNE	4\$:AWAIT PIP RESET
2068								
2069	004564	042760	100000	000742	10\$:	BIC	#100000,UN1(RO)	:CLEAR EOT FLAG
2070	004572	062737	000002	000674		ADD	#2,UNP	
2071	004600	013700	000674			MOV	UNP,RO	:POINT TO NEXT UNIT
2072	004604	022760	177777	000742		CMP	#-1,UN1(RO)	:BRANCH IF NOT LAST UNIT
2073	004612	001317				BNE	REOT4	
2074	004614	005037	000674		REOT7:	CLR	UNP	:CLEAR UNIT POINTER
2075	004620	005037	000636			CLR	TINF	:CLEAR TTY INPUT FLAG
2076	004624	005737	000734			TST	ASEQF	:SEE IF AUTO SEQ
2077	004630	001402				BEQ	REOTX	:IF NOT: BR
2078	004632	005726				TST	(SP)+	:RESET STACK POINTER
2079	004634	000207				RTS	PC	:RETURN TO AUTO SEQ
2080	004636	004737	004122		REOTX:	JSR	PC,RANSET	:GO RESET RANDOM BASE
2081	004642	012737	177777	013652		MOV	#-1,PATS	:PRESET PATTERN
2082	004650	005037	014404			CLR	RDFL	:CLEAR RANDOM FLAG
2083	004654	005737	000572			TST	SPFLG	:SEE IF SINGLE PASS
2084	004660	001421				BEQ	REOTXX	:IF NOT: BR
2085	004662	000004	025426		TEND:	TYPE,MSG100		:TYPE MSG
2086	004666	013700	000042			MOV	@#42,RO	:GET ACT11 RETURN ADDRESS
2087	004672	001405				BEQ	HERE	:BRANCH IF NOT ACT11
2088	004674	000005				RESET		
2089	004676	004710			\$ENDAD:	JSR	PC,(RO)	
2090	004700	000240				NOP		
2091	004702	000240				NOP		
2092	004704	000240				NOP		
2093	004706	000240			HERE:	NOP		
2094	004710	005737	003034			TST	CHNFLG	:BRANCH IF NOT CHAIN MODE
2095	004714	001402				BEQ	1\$	
2096	004716	000137	021226			JMP	ASEQ0	:RETURN TO AUTO SEQUENCER
2097	004722	000000			1\$:	HALT		
2098	004724	000137	003242		REOTXX:	JMP	STARTE	:RESTART AT BLOCK NUMBER ONE
2099	004730	000000			REOTC:	0		:EOT UNIT COUNTER


```
2100 :*****
2101 :REWIND ALL AVAIL TAPES:
2102 :
2103 :THIS ROUTINE; ENTERED VIA CONSOLE SWITCH NINE (9),
2104 :WILL REWIND ALL AVAILABLE TAPES TO BOT NO MATTER
2105 :WHERE THEY ARE CURRENTLY POSITIONED AND RESUME TESTING
2106 :ON THE CURRENTLY SELECTED UNIT.
2107 :*****
2108
2109 004732 032777 001000 173650 RWND: BIT #1000,@SWR ;SEE IF SHOULD REWIND
2110 004740 001001 BNE RWNDA ;IF SO: BR
2111 004742 000207 RTS PC ;ELSE EXIT
2112 004744 013737 000674 000714 RWNDA: MOV UNP,UPS ;SAVE UNIT POINTER
2113 004752 005037 000674 CLR UNP ;CLEAR POINTER
2114 004756 005037 000660 CLR EOTREC ;CLEAR EOT FLAG
2115 004762 113737 004731 004730 MOV# REOTC+1,REOTC ;++B RESTORE UNIT CTR
2116 004770 013700 000674 RWND0: MOV UNP,RO ;POINT TO UNIT ENTRY
2117 004774 022760 177777 000742 CMP #-1,UN1(RO) ;BRANCH IF LAST ENTRY
2118 005002 001437 BEQ RWND2
2119 005004 032760 140000 000742 BIT #140000,UN1(RO) ;BRANCH IF ALREADY REWINDING
2120 005012 001024 BNE RWND1A ;OR MARKED OFF LINE
2121 005014 016037 000742 000552 MOV UN1(RO),UDES ;SET UNIT DESCRIPTION
2122 005022 013777 000552 173512 MOV UDES,@TC ;LOAD COMMAND REGISTER
2123 005030 012777 000011 173452 MOV #11,@C1 ;DRIVE CLEAR
2124 005036 012777 000007 173444 MOV #7,@C1 ;START REWIND
2125 005044 105777 173452 1$: TSTB @DS
2126 005050 100405 BMI RWND1A ;IF DRY: BR
2127 005052 005337 000666 DEC STAL
2128 005056 001372 BNE 1$ ;AWAIT DRY
2129 005060 000137 020312 JMP OFFLINE ;GO MARK UNIT OFF LINE
2130 005064 042760 100000 000742 RWND1A: BIC #100000,UN1(RO) ;CLEAR EOT FLAG
2131 005072 062737 000002 000674 ADD #2,UNP ;BUMP POINTER
2132 005100 000733 BR RWND0 ;DO NEXT UNIT
2133 005102 005037 000674 RWND2: CLR UNP ;CLEAR POINTER
2134 005106 013700 000674 RWND3: MOV UNP,RO ;POINT TO UNIT ENTRY
2135 005112 022760 177777 000742 CMP #-1,UN1(RO) ;BRANCH IF LAST ENTRY
2136 005120 001433 BEQ RWNDX
2137 005122 016037 000742 000552 MOV UN1(RO),UDES ;SET UNIT DESCRIPTION
2138 005130 032760 040000 000742 BIT #40000,UN1(RO) ;BRANCH IF UNIT MARKED OFF LINE
2139 005136 001015 BNE RWND5
2140 005140 013777 000552 173374 MOV UDES,@TC ;LOAD UNIT DESCRIPTION
2141 005146 032777 020000 173346 1$: BIT #20000,@DS
2142 005154 001374 BNE 1$ ;AWAIT PIP RESET
2143 005156 032777 000002 173336 BIT #2,@DS ;BRANCH IF SLAVE AT BOT
2144 005164 001002 BNE RWND5
2145 005166 000137 020312 JMP OFFLINE ;PRINT OFFLINE MESSAGE
2146 005172 062737 000002 000674 RWND5: ADD #2,UNP ;BUMP POINTER
2147 005200 012777 000011 173302 MOV #11,@C1 ;DRIVE CLEAR
2148 005206 000737 BR RWND3 ;DO NEXT UNIT
2149
2150 005210 013700 000714 RWNDX: MOV UPS,RO ;RESTORE UNIT POINTER
2151 005214 010037 000674 MOV RO,UNP
2152 005220 016037 000742 000552 MOV UN1(RO),UDES ;RESET UNIT DESCRIPTION
2153 005226 013777 000552 173306 MOV UDES,@TC
2154 005234 000207 RTS PC ;RETURN TO TEST
2155
```



```

2156
2157
2158
2159
2160
2161
2162
2163
2164 005236 013746 000552          INIT:  MOV    UDES,-(SP)      ;GET UNIT DESCRIPTION
2165 005242 012777 000040 173250  MOV    #40,@CS      ;DO A MASSBUS CLEAR
2166 005250 013777 000550 173242  MOV    DVN,@CS      ;LOAD DRIVE #
2167 005256 011677 173260          MOV    (SP),@TC     ;LOAD SLAVE # & SLAVE DESCRIPTION
2168 005262 042716 174377          BIC    #174377,(SP) ;CLEAR ALL BUT DENSITY BITS
2169 005266 022726 001400          CMP    #1400,(SP)+  ;BRANCH IF NOT NRZ
2170 005272 001005
2171 005274 032777 000040 173220  BNE    1$           ;BRANCH IF SLAVE IS IN PE MODE
2172 005302 001422
2173 005304 000404
2174 005306 032777 000040 173206 1$:  BIT    #40,@DS      ;BRANCH IF SLAVE IS IN PE MODE
2175 005314 001015
2176 005316 012777 000007 173164 2$:  BNE    4$           ;PES = 1
2177 005324 105777 173172 20$: MOV    #7,@C1       ;LOAD REWIND COMMAND
2178 005330 100375
2179 005332 032777 020000 173162 3$:  TSTB  @DS          ;WAIT FOR READY
2180 005340 001374
2181 005342 012777 000011 173140 3$:  BPL    20$         ;WAIT FOR PIP = 0
2182 005350 000207          BIT    #20000,@DS
2182 005350 000207          BNE    3$
2182 005350 000207          MOV    #11,@C1     ;CLEAR DRIVE
2182 005350 000207          RTS    PC
  
```


2183
2184
2185
2186
2187
2188
2189
2190
2191
2192
2193
2194
2195
2196
2197
2198
2199
2200
2201
2202
2203
2204
2205
2206
2207
2208
2209
2210
2211
2212
2213
2214
2215

```
*****  
:WRITE ROUTINE:  
:  
:THIS ROUTINE IS USED TO WRITE ONTO TAPE THE BLOCK  
:OF DATA DESCRIBED BY THE OPERATOR AND SET UP  
:IN THE SEQUENCE FORMATTER. THE TAPE UNIT TO BE USED  
:HAS BEEN ASSIGNED BY THE SEQUENCE FORMATTER AND  
:ITS PARAMETERS SET IN A UNIT DESCRIPTION WORD.  
:AS EACH RECORD OF THE BLOCK IS WRITTEN, IT IS CHECKED  
:FOR STATUS ERRORS, WORD COUNT ZERO, AND CORRECT CURRENT  
:MEMORY ADDRESS. IF THE WRITE OPERATION RESULTS IN  
:ANY ERROR CONDITION, A WRITE RETRY OF THAT OPERATION  
:MAY BE DONE BY SETTING SWITCH FOUR (4) TO A ONE (1).  
:THE RETRY CONSISTS OF A BACKSPACE, ERASE FORWARD, AND  
:REWRITE OF THE RECORD. (SEE WRITE RETRY SUBROUTINE)  
:AFTER ALL DATA RECORDS IN THE BLOCK HAVE BEEN  
:WRITTEN, THE WRITE ROUTINE WILL EXECUTE A WRITE  
:TAPE MARK COMMAND IF THE TTY RESPONSE TM=1 WAS  
:MADE AT INITIAL START. THE TM IS COUNTED AS TOTAL  
:DATA RECORDS PLUS ONE (IE: IF 100 DATA RECORDS; TM=RECORD 101)  
:IF THE WRITE OPERATION (DATA OR TM) CAUSES THE SELECTED SLAVE  
:TO REACH END OF TAPE (EOT) AND THERE IS TO BE NO READING DONE,  
:(SW2 AND SW3 SET TO A 1) THEN THE SLAVE IS REWOUND AND  
:FLAGGED AS UNAVAILABLE FOR TESTING UNTIL ALL SLAVES HAVE  
:REACHED EOT AND BEEN REWOUND AT WHICH TIME TESTING IS  
:RESUMED ON ALL AVAILABLE SLAVES.  
:WRITE RETRY MAY BE ALLOWED VIA CONSOLE SWITCH FOUR (4).  
:ERROR CHECKING MAY BE DISALLOWED VIA CONSOLE SWITCH  
:TWELVE (12).  
:WRITING TO TAPE MAY BE DISALLOWED VIA CONSOLE SWITCH  
:ZERO (0).  
*****
```

```
2216 005352 032777 000001 173230 WRITE: BIT #1,@SWR ;SEE IF SHOULD WRITE  
2217 005360 001402 BEQ WRTE  
2218 005362 000137 006132 JMP WEX ;IF NOT: BR  
2219 005366 013700 000554 WRTE: MOV RCNT,R0 ;RO=RECORD COUNT  
2220 005372 012737 023544 000652 WO: MOV #MSG5,EMADDR ;SET ERROR MSG ADDRESS  
2221 005400 013777 000556 173110 MOV FMCNT,@FC ;LOAD CHAR COUNT  
2222 005406 012777 026342 173100 MOV #WDATA,@BA ;SET DATA ADDR  
2223 005414 112737 000060 000672 MOVB #60,MTC1 ;SET WRITE OP COMMAND  
2224 005422 012737 005434 000662 MOV #W1,RTRN ;SET RETURN ADDRESS  
2225 005430 000137 020372 JMP TAPG ;GO EXECUTE COMMAND  
2226 005434 032777 002000 173060 W1: BIT #2000,@DS ;SEE IF EOT  
2227 005442 001412 BEQ 1$ ;IF NOT AT EOT: BR  
2228 005444 005737 000660 TST EOTREC ;BRANCH IF WRITTEN PAST EOT  
2229 005450 100407 BMI 1$  
2230 005452 005300 DEC R0 ;ADJUST # OF RECORDS WRITTEN  
2231 005454 052700 100000 BIS #100000,R0 ;SET EOT INDICATOR  
2232 005460 010037 000660 MOV R0,EOTREC ;SAVE RECORD COUNT  
2233 005464 012700 000002 MOV #2,R0 ;SET TO WRITE 1 LAST RECORD  
2234 005470 032777 010000 173112 1$: BIT #10000,@SWR ;SEE IF SHOULD CHECK ERRORS  
2235 005476 001002 BNE 2$ ;IF NOT: BR  
2236 005500 004737 016522 JSR PC,ERCHK ;GO CHECK ERRORS  
2237 005504 013737 000576 000666 2$: MOV WCTAL,STAL ;SET DELAY  
2238 005512 004737 011654 JSR PC,STALL ;DELAY
```


2239	005516	005737	000712		TST	RTYFL		:SEE IF RETRY TIME
2240	005522	001401			BEQ	3\$:IF NOT: BR
2241	005524	000207			RTS	PC		:ELSE RETURN
2242	005526	005737	000706	3\$:	TST	SERFL		:SEE IF WRITE ERROR
2243	005532	001446			BEQ	W5		:IF NOT: BR
2244	005534	013704	000674		MOV	UNP,R4		
2245	005540	005264	001064		INC	WTER1(R4)		:BUMP WRITE ERROR
2246	005544	005037	000706		CLR	SERFL		:CLEAR STATUS ERROR FLAG
2247	005550	032777	000020	173032	BIT	#20,@SWR		:SEE IF RETRY
2248	005556	001434			BEQ	W5		:IF NOT: BR
2249	005560	013703	000722		MOV	ERSAV,R3		
2250	005564	042703	102700		BIC	#102700,R3		:MASK UNRECOVERABLE ERROR
2251	005570	001407			BEQ	W4		:IF SO: BR
2252	005572	004737	022012		JSR	PC,PAPRT		:PRINT HEADER
2253	005576	000004	025327		TYPE,MSG78			:TYPE MSG
2254	005602	004737	010754		JSR	PC,NRTP		:PRINT ER FOR NON-RETRYABLE
2255	005606	000420			BR	W5		
2256	005610	013704	000674	W4:	MOV	UNP,R4		
2257	005614	005264	001044		INC	RTY1(R4)		:BUMP RETRY CNTR
2258	005620	032777	002000	172762	BIT	#2000,@SWR		:SEE IF PRINT ERRORS
2259	005626	001002			BNE	W4A		:IF NOT: BR
2260	005630	000004	025077		TYPE,MSG64			:TYPE MSG
2261	005634	005037	000702	W4A:	CLR	RTCNT		:CLEAR RETRY NUMBER
2262	005640	005037	000700		CLR	RPCNT		:CLEAR REPEAT COUNTER
2263	005644	004737	006166		JSR	PC,WRTY		:GO RETRY WRITE ERROR
2264	005650	005037	000712	W5:	CLR	RTYFL		:CLEAR RETRY COUNTER
2265	005654	005300			DEC	R0		:SEE IF DONE ALL
2266	005656	001245			BNE	W0		:IF NOT: BR
2267	005660	005737	000564	W6:	TST	TMEX		:SEE IF TM
2268	005664	001522			BEQ	WEX		:IF NOT: BR
2269	005666	005237	000676		INC	TMFLG		:SET TM FLAG
2270	005672	012737	025010	000652	WTM:	MOV	#MSG54,EMADDR	:POINT TO TM ERROR MSG
2271	005700	012737	000026	000672	MOV	#26,MTC1		:SET TM OP CODE
2272	005706	005077	172604		CLR	@FC		:LOAD FRAME COUNTER
2273	005712	012777	026342	172574	MOV	#WDATA,@BA		:LOAD BUS ADDRESS
2274	005720	012737	005732	000662	MOV	#WTMO,RTRN		:SAVE RETURN ADDRESS
2275	005726	000137	020377		JMP	TAPG		:WRITE TM
2276	005732	032777	010000	172650	WTMO:	BIT	#10000,@SWR	:SEE IF SHOULD CHECK ERRORS
2277	005740	001074			BNE	WEX		
2278	005742	032777	000004	172552	BIT	#4,@DS		:SEE IF TM STATUS
2279	005750	001011			BNE	WTM1		:IF SO: BR
2280	005752	012737	026342	020224	MOV	#WDATA,CADER		:SET EXPT BUS ADDRESS
2281	005760	012737	000001	020232	MOV	#1,DRVER		:INDICATE ERROR
2282	005766	004737	017352		JSR	PC,ERPT		:PRINT TM ERROR
2283	005772	000404			BR	WTM2		
2284	005774	012703	026342	WTM1:	MOV	#WDATA,R3		:SET EXPT ADDRESS
2285	006000	004737	016614		JSR	PC,ER2		:GO CHECK FOR OTHER ERRORS
2286	006004	005737	000712	WTM2:	TST	RTYFL		:SEE IF RETRY
2287	006010	001401			BEQ	WTM3		:IF NOT: BR
2288	006012	000207			RTS	PC		:ELSE RETURN TO RETRY ROUTINE
2289	006014	005737	000706	WTM3:	TST	SERFL		:SEE IF WRITE ERROR
2290	006020	001444			BEQ	WEX		:IF NOT: BR
2291	006022	013704	000674		MOV	UNP,R4		
2292	006026	005264	001064		INC	WTER1(R4)		:BUMP WRITE ERROR
2293	006032	032777	000020	172550	BIT	#20,@SWR		:SEE IF SHOULD RETRY
2294	006040	001434			BEQ	WEX		:IF NOT: BR

2295	006042	013703	000722		MOV	ERSAV,R3		
2296	006046	042703	102700		BIC	#102700,R3	:MASK UNRECOVERABLE ERROR	
2297	006052	001407			BEQ	WTM4	:IF SO: BR	
2298	006054	004737	022012		JSR	PC,PAPRT	:PRINT HEADER	
2299	006060	000004	025327		TYPE,MSG78		:TYPE MSG	
2300	006064	004737	010754		JSR	PC,NRTP	:PRINT ER FOR NON-RETRYABLE	
2301	006070	000420			BR	WEX		
2302	006072	005037	000700	WTM4:	CLR	RPCNT	:CLEAR REPEAT CNTR	
2303	006076	013704	000674		MOV	UNP,R4		
2304	006102	005264	001044		INC	RTY1(R4)	:BUMP RETRY CNTR	
2305	006106	005037	000702		CLR	RTCNT	:CLEAR RETRY CNTR	
2306	006112	032777	002000	172470	BIT	#2000,@SWR	:SEE IF PRINT ERRORS	
2307	006120	001002			BNE	WTM4A	:IF NOT: BR	
2308	006122	000004	025077		TYPE,MSG64		:TYPE MSG	
2309	006126	004737	006166	WTM4A:	JSR	PC,WRTY	:GO DO RETRY	
2310	006132	005037	000712	WEX:	CLR	RTYFL	:CLEAR RETRY FLAG	
2311	006136	005037	000676		CLR	TMFLG	:CLEAR TAPE MARK FLAG	
2312	006142	005737	000660		TST	EOTREC	:BRANCH IF NOT AT EOT	
2313	006146	100006			BPL	WRWX		
2314	006150	032777	000014	172432	WRW:	BIT	#14,@SWR	:BRANCH IF EITHER READ ENABLED
2315	006156	001002			BNE	WRWX		
2316	006160	000137	004154		JMP	REOT	:ELSE REWIND	
2317	006164	000207		WRWX:	RTS	PC	:EXIT	


```
2318  
2319  
2320  
2321  
2322  
2323 006166 012737 000001 000712 WRTY: MOV #1,RTYFL ;SET RETRY FLAG  
2324 006174 004737 006554 WRTY0: JSR PC,WRTSB ;GO SPACE REVERSE FOR REPEAT  
2325 006200 005737 000676 TST TMFLG ;SEE IF TAPE MARK TIME  
2326 006204 001003 BNE WRTYTM ;IF SO: BR  
2327 006206 004737 005372 JSR PC,W0 ;REWRITE RECORD  
2328 006212 000402 BR WRTYR ;GO ON  
2329 006214 004737 005672 WRTYTM: JSR PC,WTM ;GO WRITE TAPE MARK AGAIN  
2330 006220 005737 000706 WRTYR: TST SERFL ;REWRITE GOOD  
2331 006224 001022 BNE WRTY2 ;IF NOT: BR  
2332 006226 005237 000700 INC RPCNT ;BUMP REPEAT COUNTER  
2333 006232 022737 000004 000700 CMP #4,RPCNT ;SEE IF FOUR GOOD REPEATS  
2334 006240 001355 BNE WRTY0 ;IF NOT: REPEAT  
2335 006242 032777 002000 172340 BIT #2000,@SWR ;SEE IF PRINT  
2336 006250 001007 BNE WRTY1 ;IF NOT: BR  
2337 006252 000004 025512 TYPE,MSG105 ;TYPE MSG  
2338 006256 000004 025121 TYPE,MSG65 ;TYPE MSG  
2339 006262 013703 000702 MOV RTCNT,R3  
2340 006266 104400 TYPOCT ;PRINT RETRY NUMBER  
2341 006270 000207 WRTY1: RTS PC ;RESUME TESTING  
2342 006272 013703 000722 WRTY2: MOV ERSAV,R3 ;GET ER  
2343 006276 005037 000650 CLR TEMP3 ;CLEAR RECOVERABLE ERROR INDICATOR  
2344 006302 042703 102700 BIC #102700,R3 ;MASK RECOVERABLE BITS  
2345 006306 001412 BEQ WRTY2A ;IF RECOVERABLE: BR  
2346 006310 004737 022012 JSR PC,PAPRT ;PRINT HEADER  
2347 006314 000004 025327 TYPE,MSG78 ;TYPE MSG  
2348 006320 004737 010754 JSR PC,NRTP ;PRINT ER  
2349 006324 012737 000001 000650 MOV #1,TEMP3 ;SET FLAG  
2350 006332 000406 BR WRTY2B  
2351 006334 032777 002000 172246 WRTY2A: BIT #2000,@SWR ;SEE IF PRINT  
2352 006342 001022 BNE WRTY3 ;IF NOT: BR  
2353 006344 000004 025722 TYPE,MSG110 ;TYPE MSG  
2354 006350 000004 025121 WRTY2B: TYPE,MSG65 ;TYPE MSG  
2355 006354 013703 000702 MOV RTCNT,R3  
2356 006360 104400 TYPOCT ;PRINT RETRY NUMBER  
2357 006362 000004 025744 TYPE,MSG111 ;TYPE MSG  
2358 006366 013703 000700 MOV RPCNT,R3  
2359 006372 104400 TYPOCT ;PRINT REPEAT NUMBER  
2360 006374 005737 000650 TST TEMP3 ;SEE IF DID NON-RECOVERABLE  
2361 006400 001403 BEQ WRTY3 ;IF NOT: BR  
2362 006402 005037 000650 CLR TEMP3 ;CLEAR FLAG  
2363 006406 000207 RTS PC ;EXIT  
2364 006410 005737 000702 WRTY3: TST RTCNT ;SEE IF FIRST RETRY  
2365 006414 001004 BNE WRTY3A ;IF NOT: BR  
2366 006416 013704 000674 MOV UNP,R4  
2367 006422 005364 001064 DEC WTER1(R4) ;DECREMENT WRITE ERROR CNTR  
2368 006426 013704 000674 WRTY3A: MOV UNP,R4 ;GET UNIT NUMBER  
2369 006432 016437 001024 000730 MOV BTADDR(R4),BTPT ;GET ADDRESS OF UNIT BAD TAPE CNTR  
2370 006440 017704 172264 MOV @BTPT,R4 ;GET COUNTER  
2371 006444 005724 TST (R4)+ ;SET POINTER OFFSET  
2372 006446 010477 172256 MOV R4,@BTPT  
2373 006452 013703 000730 MOV BTPT,R3
```



```
2464
2465
2466
2467 007126 000004 024153
2468 007132 013704 000674
2469 007136 016437 001024
2470 007144 017703 171560
2471 007150 000241
2472 007152 006003
2473 007154 104400
2474 007156 000004 025756
2475 007162 005777 171542
2476 007166 001001
2477 007170 000207
2478 007172 000137 007012
2479
2480
2481
2482 007176 004737 022012
2483 007202 000004 025611
2484 007206 000207
2485

;BAD TAPE STATISTIC PRINT*****
BTPRT: TYPE,MSG28 ;TYPE '<CR><LF>'
MOV UNP,R4
MOV BTADDR(R4),BTPT ;SET TABLE POINTER
MOV @BTPT,R3
CLC
ROR R3 ;CORRECT NUMBER
TYPOCT ;PRINT NUMBER OF BAD SPOTS
TYPE,MSG112 ;TYPE MSG
TST @BTPT ;SEE IF ANY BAD SPOTS
BNE BTPRT1 ;IF SO: BR
RTS PC ;ELSE RETURN
BTPRT1: JMP BTOV0 ;PRINT STATS

;BAD TAPE UNRECOVERABLE SUBROUTINE*****
BTUR: JSR PC,PAPRT ;PRINT HEADER
TYPE,MSG107 ;TYPE MSG
RTS PC ;RESUME TESTING
```



```
2486  
2487  
2488  
2489  
2490  
2491  
2492  
2493  
2494  
2495  
2496  
2497  
2498  
2499  
2500  
2501  
2502 007210 005037 000562 RSEQ: CLR RDCMD  
2503 007214 017704 171370 MOV @SWR,R4 ;READ SWITCHES  
2504 007220 042704 177763 BIC #177763,R4 ;MASK READ BITS & SEE IF BOTH READS  
2505 007224 001004 BNE RSR ;IF NOT: BR  
2506 007226 032777 000002 171354 BIT #2,@SWR ;SEE IF READ REVERSE FIRST  
2507 007234 001041 BNE RSFR ;IF NOT: BR  
2508 007236 032777 000004 171344 RSR: BIT #4,@SWR ;SEE IF SHOULD READ REVERSE  
2509 007244 001005 BNE RSF ;IF NOT: BR  
2510 007246 012737 000001 000562 MOV #1,RDCMD ;LOAD READ REVERSE COMMAND  
2511 007254 004737 007464 JSR PC,READ ;GO READ REVERSE  
2512 007260 032777 000010 171322 RSF: BIT #10,@SWR ;SEE IF SHOULD READ FORWARD  
2513 007266 001066 BNE RSEX ;IF NOT: BR  
2514 007270 005737 000562 TST RDCMD ;SEE IF HAVE READ REVERSE  
2515 007274 001406 BEQ RSF0 ;IF NOT: BR  
2516 007276 013737 000600 000666 MOV TSTAL,STAL  
2517 007304 004737 011654 JSR PC,STALL ;DO READ STALL  
2518 007310 000406 BR RSF1  
2519 007312 032777 000001 171270 RSF0: BIT #1,@SWR ;SEE IF WRITE  
2520 007320 001002 BNE RSF1 ;IF NOT: BR  
2521 007322 004737 011402 JSR PC,BKSP ;GO BACKSPACE  
2522 007326 005037 000562 RSF1: CLR RDCMD ;LOAD READ FORWARD COMMAND  
2523 007332 004737 007464 JSR PC,READ ;GO READ  
2524 007336 000442 BR RSEX ;GO TO EXIT  
2525  
2526 007340 012737 000001 000562 RSFR: MOV #1,RDCMD  
2527 007346 032777 000010 171234 BIT #10,@SWR ;SEE IF SHOULD READ FORWARD  
2528 007354 001012 BNE RSFR1 ;IF NOT: BR  
2529 007356 032777 000001 171224 BIT #1,@SWR ;SEE IF WRITE  
2530 007364 001002 BNE RSFR0 ;IF NOT: BR  
2531 007366 004737 011402 JSR PC,BKSP ;GO BACKSPACE TO START  
2532 007372 005037 000562 RSFR0: CLR RDCMD ;LOAD READ FORWARD COMMAND  
2533 007376 004737 007464 JSR PC,READ ;GO READ FORWARD  
2534 007402 032777 000004 171200 RSFR1: BIT #4,@SWR ;SEE IF SHOULD READ REVERSE  
2535 007410 001015 BNE RSEX ;IF NOT: BR  
2536 007412 005737 000562 TST RDCMD  
2537 007416 001005 BNE RSFR2 ;IF READ REVERSE: BR  
2538 007420 013737 000600 000666 MOV TSTAL,STAL ;DO READ STALL  
2539 007426 004737 011654 JSR PC,STALL  
2540 007432 012737 000001 000562 RSFR2: MOV #1,RDCMD ;LOAD READ REVERSE  
2541 007440 004737 007464 JSR PC,READ ;GO READ REVERSE
```


CZTEDBO TMO3-TE16/TU77 DRT
CZTEDB.P11 15-NOV-78 13:19

MACY11 30A(1052) 21-DEC-78 13:17 ^{E 5} PAGE 56

SEQ 0056

2542	007444	005037	000562
2543	007450	005737	000660
2544	007454	001402	
2545	007456	000137	004154
2546	007462	000207	
2547			

RSEX:	CLR	RDCMD
	TST	EOTREC
	BEQ	RSFRX
	JMP	REOT
RSFRX:	RTS	PC

:BRANCH IF EOT NOT REACHED
:REWIND AND REPORT STATS
:EXIT

2548
2549
2550
2551
2552
2553
2554
2555
2556
2557
2558
2559
2560
2561
2562
2563
2564
2565
2566
2567
2568
2569
2570
2571
2572
2573
2574
2575
2576
2577
2578
2579
2580
2581
2582
2583
2584
2585
2586
2587
2588
2589
2590
2591
2592
2593
2594
2595
2596
2597
2598
2599
2600
2601
2602
2603

```
*****  
:READ ROUTINE:  
:THIS ROUTINE PERFORMS THE READ OPERATION DETERMINED  
:BY THE READ SEQUENCE ROUTINE ONE RECORD AT A TIME.  
:AT THE END OF EACH READ OPERATION THE STATUS REGISTER  
:IS SCANNED FOR EITHER END OF TAPE OR BEGINNING OF TAPE.  
:IF EOT WAS REACHED, CONTROL WILL BE PASSED TO  
:THE EOT SUBROUTINE TO REWIND THE UNIT AND FLAG IT  
:UNAVAILABLE UNTIL ALL UNITS HAVE REACHED EOT.  
:IF BOT WAS REACHED AN ERROR IS PRINTED AND THE  
:PROGRAM WILL HALT. TESTING MAY BE RESUMED BY PRESSING  
:THE CONTINUE SWITCH.  
:IF A TAPE MARK IS EXPECTED (TM=1) THEN THE  
:READ ROUTINE EXPECTS THE FIRST RECORD OF A  
:READ REVERSE TO BE A TM, AND THE LAST RECORD  
:OF A READ FORWARD TO BE A TM. REMEMBER  
:THAT THE TM ADDS ONE (1) TO THE TOTAL NUMBER  
:OF RECORDS IN A BLOCK.  
:CONSOLE SWITCHES ELEVEN (11) AND THIRTEEN (13) DETERMINE WHETHER  
:OR NOT TO CHECK FOR STATUS ERRORS (11) OR DATA ERRORS (13),  
:CONSOLE SWITCH FIVE (5) IS USED TO CAUSE A CONTINUOUS  
:READ AND SPACE (FORWARD OR REVERSE) OF THE CURRENT  
:RECORD ON TAPE (YOZZLE).  
*****
```

```
READ:  MOV    RCNT,R0      ;LOAD REC CNTR  
       TST    EOTREC    ;SEE IF EOT  
       BPL    RDA       ;IF NOT: BR  
       TST    RDCMD     ;SEE IF READ FORWARD  
       BEQ    RDA       ;IF SO: BR  
       BIC    #100000,EOTREC ;CLEAR FLAG  
       MOV    EOTREC,R3 ;GET MODIFIED RECORD COUNT  
       SUB    R3,R0     ;SET RECORD AT  
       INC    R0       ;SET TO PROPER NUMBER OF RECORDS  
RDA:   MOV    #MSG6,EMADDR ;SET ERROR MSG ADDRESS  
       CLR    TMFLG  
       TST    RDCMD  
       BEQ    RDO      ;IF READ FORWARD: BR  
       TST    TMEX     ;SEE IF TM  
       BEQ    RDO      ;IF NOT: BR  
       INC    TMFLG    ;SET TM FLAG  
       INC    R0  
RDO:   MOV    FMCNT,@FC  ;LOAD CHAR CNTR  
       MOV    #RDATA,@BA ;LOAD DATA ADDR  
       TST    RDCMD     ;SEE IF READ REVERSE  
       BEQ    RD1A     ;IF NOT: BR  
       MOV    FMCNT,R3  
       COM    R3  
       BIT    #20,UDES  ;SEE IF CORE DUMP  
       BEQ    RD1      ;IF NOT: BR  
       CLC  
RD1:   ROR    R3        ;R3 = FC/2  
       ADD    R3,@BA    ;SET REVERSE BUS ADDRESS  
       MOV    #7,MTC1  ;SET READ REVERSE  
       BR    RD1B
```


2604	007636	012737	000070	000672	RD1A:	MOV	#70,MTC1	:SET READ FORWARD
2605	007644	012737	007656	000662	RD1B:	MOV	#RD2,RTRN	:SET INTERRUPT RETURN ADDRESS
2606	007652	000137	020372			JMP	TAPG	:GO EXECUTE TAPE COMMAND
2607	007656	005737	000562		RD2:	TST	RDCMD	:IGNORE EOT IF READ REVERSE
2608	007662	001014				BNE	RD3	
2609	007664	032777	002000	170630		BIT	#2000,@DS	:SEE IF EOT
2610	007672	001410				BEQ	RD3	:IF NOT: BR
2611	007674	005737	000676			TST	TMFLG	:SEE IF TM
2612	007700	001005				BNE	RD3	:IF SO: BR
2613	007702	010037	000660			MOV	R0,EOTREC	:GET # OF RECORDS LEFT IN BLOCK TO READ
2614	007706	052737	100000	000660		BIS	#100000,EOTREC	:SET EOT FLAG
2615	007714	032777	000002	170600	RD3:	BIT	#2,@DS	:SEE IF AT LOAD POINT
2616	007722	001407				BEQ	RD4	:IF NOT: BR
2617	007724	004737	022012			JSR	PC,PAPRT	:PRINT CYCLE NUMBER
2618	007730	000004	023711			TYPE	MSG22	:TYPE MSG
2619	007734	000000				HALT		
2620	007736	000137	003144			JMP	STARTA	:RESTART
2621	007742	032777	004000	170640	RD4:	BIT	#4000,@SWR	:SEE IF SHOULD CHECK ERRORS
2622	007750	001116				BNE	RD5	:IF NOT: BR
2623	007752	005737	000676			TST	TMFLG	
2624	007756	001470				BEQ	RD4B	:IF NO TM EXPT: BR
2625	007760	032777	000004	170534		BIT	#4,@DS	
2626	007766	001023				BNE	RD4A	:IF TM RECVD: BR
2627	007770	012737	032350	020224		MOV	#RDATA,CADER	:SAVE EXPT BUS ADDRESS
2628	007776	012737	000002	020232		MOV	#2,DRVER	:SET TM STATUS ERROR FLAG
2629	010004	004737	017352			JSR	PC,ERPT	:GO PRINT TM ERROR
2630	010010	013704	000674			MOV	UNP,R4	
2631	010014	005737	000562			TST	RDCMD	:SEE IF READ REVERSE
2632	010020	001403				BEQ	1\$:IF NOT: BR
2633	010022	005264	001144			INC	RDERR1(R4)	:BUMP READ REVERSE ERROR
2634	010026	000500				BR	RD6	
2635	010030	005264	001104		1\$:	INC	RDER1(R4)	:BUMP READ FORWARD ERROR
2636	010034	000475				BR	RD6	
2637	010036	012703	032350		RD4A:	MOV	#RDATA,R3	
2638	010042	005737	000562			TST	RDCMD	:SEE IF READ REVERSE
2639	010046	001007				BNE	RD4A0	:IF SO: BR
2640	010050	032737	002000	000552		BIT	#2000,UDES	:SEE IF IN PE
2641	010056	001025				BNE	RD4A2	:IF SO: BR
2642	010060	062703	000002			ADD	#2,R3	
2643	010064	000422				BR	RD4A2	
2644	010066	013704	000556		RD4A0:	MOV	FMCNT,R4	
2645	010072	005104				COM	R4	
2646	010074	032737	000020	000552		BIT	#20,UDES	:SEE IF CORE DUMP
2647	010102	001402				BEQ	RD4A1	:IF NOT: BR
2648	010104	000241				CLC		
2649	010106	006004				ROR	R4	:SET TO FC/2
2650	010110	060403			RD4A1:	ADD	R4,R3	:SET EXPT BUS ADDRESS
2651	010112	042703	000001			BIC	#1,R3	:MAKE EXPT ADDRESS EVEN
2652	010116	032737	002000	000552		BIT	#2000,UDES	:SEE IF IN PE
2653	010124	001002				BNE	RD4A2	:IF SO: BR
2654	010126	162703	000002			SUB	#2,R3	
2655	010132	004737	016614		RD4A2:	JSR	PC,ER2	
2656	010136	000402				BR	RD4C	
2657	010140	004737	016522		RD4B:	JSR	PC,ERCHK	:GO CHECK ERRORS
2658	010144	005737	000706		RD4C:	TST	SE ² FL	
2659	010150	001416				BEQ	RD5	:IF NO ERROR: BR

2660	010152	013704	000674		MOV	UNP,R4	
2661	010156	005737	000562		TST	RDCMD	;SEE IF READ REVERSE
2662	010162	001003			BNE	RD4D	;IF SO: BR
2663	010164	005264	001104		INC	RDER1(R4)	;BUMP READ FORWARD ERROR
2664	010170	000402			BR	RD4E	
2665	010172	005264	001144	RD4D:	INC	RDERR1(R4)	;BUMP READ REVERSE ERROR
2666	010176	004737	010374	RD4E:	JSR	PC,RDRTY	;GO RETRY
2667	010202	005037	000712		CLR	RTYFL	;CLEAR RETRY FLAG
2668	010206	032777	020000	170374	RD5:	BIT	#20000,@SWR
2669	010214	001005			BNE	RD6	;SEE IF SHOULD DO DATA CHECK
2670	010216	005737	000676		TST	TMFLG	;IF NOT: BR
2671	010222	001002			BNE	RD6	
2672	010224	004737	014750		JSR	PC,DCHK	;GO CHECK DATA
2673	010230	005037	000706	RD6:	CLR	SERFL	;CLEAR STATUS ERROR FLAG
2674	010234	004737	013614		JSR	PC,DS3	;CLEAR BUFFER
2675	010240	032777	000040	170342	BIT	#40,@SWR	;SEE IF SHOULD YOZZLE
2676	010246	001402			BEQ	RD7	;IF NOT: BR
2677	010250	004737	010770		JSR	PC,YOZ	;ELSE GO YOZZLE
2678	010254	013737	000574	000666	RD7:	MOV	RSTAL,STAL
2679	010262	004737	011654		JSR	PC,STALL	;SET DELAY
2680	010266	005737	000562		TST	RDCMD	;STALL
2681	010272	001403			BEQ	RD7A	;SEE IF READ REVERSE
2682	010274	005037	000676		CLR	TMFLG	;IF NOT: BR
2683	010300	000405			BR	RD10	;CLEAR TAPE MARK FLAG
2684	010302	005737	000660	RD7A:	TST	EOTREC	;SEE IF EOT FOUND
2685	010306	100002			BPL	RD10	;IF NOT: BR
2686	010310	012700	000001		MOV	#1,RO	;SET TO EOT
2687	010314	005300		RD10:	DEC	RO	
2688	010316	001402			BEQ	RD11	;IF DONE ALL: BR
2689	010320	000137	007556		JMP	RDO	
2690	010324	005737	000562	RD11:	TST	RDCMD	;SEE IF READ REVERSE
2691	010330	001016			BNE	RDEX	;IF SO: BR
2692	010332	005737	000660		TST	EOTREC	;SEE IF FOUND EOT
2693	010336	100413			BMI	RDEX	;IF SO: BR
2694	010340	005737	000564		TST	TMEX	;SEE IF TM EXPECTED
2695	010344	001410			BEQ	RDEX	;IF NOT: BR
2696	010346	005737	000676		TST	TMFLG	;SEE IF TM FOUND
2697	010352	001005			BNE	RDEX	;IF SO: BR
2698	010354	005237	000676		INC	TMFLG	;ELSE SET FLAG
2699	010360	005200			INC	RO	;SET RECORD COUNT TO ONE
2700	010362	000137	007556		JMP	RDO	;GO READ TM
2701	010366	005037	000676	RDEX:	CLR	TMFLG	
2702	010372	000207		RDX:	RTS	PC	;EXIT


```
2703 :*****  
2704 :READ ERROR RETRY SUBROUTINE:  
2705 :  
2706 :THIS SUBROUTINE WILL RETRY ALL DATA RELATED  
2707 :READ ERRORS UP TO EIGHT (8) TIMES. IF ALL  
2708 :FOUR RETRIES ARE BAD, IT IS CONSIDERED  
2709 :A HARD ERROR. IF ANY ARE GOOD, IT IS A  
2710 :SOFT ERROR. RETRIES MAY BE INHIBITED  
2711 :VIA SWITCH FOUR (SW4=0: INHIBIT RETRIES)  
2712 :*****  
2713 :  
2714 010374 032777 000020 170206 RDRTY: BIT #20,@SWR ;SEE IF RETRY INHIBITED  
2715 010402 001001 BNE RDRT0 ;IF NOT: BR  
2716 010404 000207 RTS PC ;ELSE RETURN  
2717 :  
2718 010406 013703 000722 RDRT0: MOV ERSAV,R3  
2719 010412 022703 100000 CMP #100000,R3 ;++B BRANCH IF OTHER THAN CORRECTED READ ERROR  
2720 010416 001011 BNE 1$ ;++B  
2721 010420 032777 000040 170074 BIT #40,@DS ;++B BRANCH IF NRZ  
2722 010426 001405 BEQ 1$ ;++B  
2723 010430 005037 000706 CLR SERFL ;++B CLEAR ERROR FLAG  
2724 010434 000004 026302 TYPE,MSG124 ;++B TYPE 'CORRECTED PE DATA ERROR'  
2725 010440 000447 BR RDRT2 ;++B INC SOFT COUNTS  
2726 010442 042703 102700 1$: BIC #102700,R3 ;MARK NON-RECOVERABLE ERROR BITS  
2727 010446 001407 BEQ RDRT1 ;IF NOT: BR  
2728 010450 004737 022012 JSR PC,PAPRT ;PRINT HEADER  
2729 010454 000004 025367 TYPE,MSG79 ;TYPE MSG  
2730 010460 004737 010754 JSR PC,NRTP ;PRINT ER FOR NON-RETRYABLE ERROR  
2731 010464 000207 RTS PC ;RETURN  
2732 010466 032777 002000 170114 RDRT1: BIT #2000,@SWR ;SEE IF PRINT INHIBITED  
2733 010474 001002 BNE RDRT1B ;IF SO: BR  
2734 010476 000004 025077 TYPE,MSG64 ;TYPE MSG  
2735 010502 005037 000702 RDRT1B: CLR RTCNT ;CLEAR RETRY COUNTER  
2736 010506 005037 000706 RDRTG: CLR SERFL ;CLEAR STATUS ERROR FLAG  
2737 010512 012737 000002 000712 MOV #2,RTYFL ;SET READ RETRY FLAG  
2738 010520 004737 010770 JSR PC,YOZ ;GO TO YOZZLE TO RETRY READ  
2739 010524 005737 000706 TST SERFL ;SEE IF RETRY ERROR  
2740 010530 001026 BNE RDRT5 ;IF SO: BR  
2741 010532 032777 002000 170050 BIT #2000,@SWR  
2742 010540 001007 BNE RDRT2  
2743 010542 000004 025512 TYPE,MSG105 ;TYPE MSG  
2744 010546 000004 025121 TYPE,MSG65 ;TYPE MSG  
2745 010552 013703 000702 MOV RTCNT,R3  
2746 010556 104400 TYPOCT ;PRINT RETRY NUMBER  
2747 010560 013704 000674 RDRT2: MOV UNP,R4  
2748 010564 005737 000562 TST RDCMD ;SEE IF READ REVERSE  
2749 010570 001003 BNE RDRT3 ;IF SO: BR  
2750 010572 005264 002664 INC RFSOFT(R4) ;ELSO BUMP FORWARD SOFT ERROR COUNTER  
2751 010576 000402 BR RDRT4  
2752 010600 005264 002704 RDRT3: INC RRSOFT(R4) ;BUMP ERRORS SOFT CNTR  
2753 010604 000207 RDRT4: RTS PC ;RETURN  
2754 010606 013703 000722 RDRT5: MOV ERSAV,R3 ;GET ER  
2755 010612 005037 000650 CLR TEMP3 ;CLEAR RECOVERABLE ERROR INDICATOR  
2756 010616 042703 102700 BIC #102700,R3 ;MASK RECOVERABLE BITS  
2757 010622 001412 BEQ RDRT5A ;IF RECOVERABLE: BR  
2758 010624 004737 022012 JSR PC,PAPRT ;PRINT HEADER
```



```
2759 010630 000004 025367          TYPE,MSG79          ;TYPE MSG
2760 010634 004737 010754          JSR      PC,NRTP    ;PRINT ER
2761 010640 012737 000001 000650  MOV      #1,TEMP3   ;SET FLAG
2762 010646 000404          BR       RDRT5B
2763 010650 032777 002000 167732 RDRT5A: BIT      #2000,@SWR ;SEE IF PRINT INHIBITED
2764 010656 001013          BNE     RDRT6       ;IF SO: BR
2765 010660 000004 025121          RDRT5B: TYPE,MSG65  ;TYPE MSG
2766 010664 013703 000702          MOV      RTCNT,R3
2767 010670 104400          TYPOCT          ;PRINT RETRY NUMBER
2768 010672 005737 000650          TST     TEMP3      ;SEE IF DID NON-RECOVERABLE
2769 010676 001403          BEQ     RDRT6       ;IF NOT: BR
2770 010700 005037 000650          CLR     TEMP3      ;CLEAR FLAG
2771 010704 000207          RTS     PC          ;EXIT
2772 010706 005237 000702          RDRT6: INC     RTCNT
2773 010712 023737 000702 000604  CMP     RTCNT,RETRY ;SEE IF DONE 8 RETRIES
2774 010720 001272          BNE     RDRTG       ;IF NOT: BR
2775 010722 000004 026021          TYPE,MSG115       ;TYPE MSG
2776 010726 013704 000674          MOV     UNP,R4
2777 010732 005737 000562          TST     RDCMD      ;SEE IF READ REVERSE
2778 010736 001003          BNE     RDRT7       ;IF SO: BR
2779 010740 005264 002724          INC     RFHARD(R4) ;BUMP FORWARD HARD ERROR CNTR
2780 010744 000402          BR      RDRTX
2781 010746 005264 002744          RDRT7: INC     RRHARD(R4) ;BUMP REVERSE HARD ERROR CNTR
2782 010752 000207          RDRTX: RTS     PC          ;RETURN
2783
2784 010754 013703 000722          NRTP:  MOV     ERSAV,R3 ;GET ER REGISTER
2785 010760 104400          TYPOCT          ;PRINT ER
2786 010762 004737 020250          JSR     PC,FRPRT   ;PRINT F OR R
2787 010766 000207          RTS     PC          ;RETURN
2788
2789          ;*****
2790          ;YOZZLE SUBROUTINE:
2791          ;
2792          ;THIS SUBROUTINE, ENTERED VIA SWITCH FIVE (5), IS USED TO PERFORM
2793          ;A CONTINUOUS READ AND SPACE OVER OF THE CURRENT RECORD ON TAPE.
2794          ;FULL STATUS AND DATA CHECKING MAY BE PERFORMED
2795          ;OR NOT VIA CONSOLE SWITCHES ELEVEN (11) AND THIRTEEN (13).
2796          ;A SOFTWARE DELAY IS PERFORMED BETWEEN EACH READ
2797          ;AND SPACE OPERATION AND MAY BE VARIED BY TYPING
2798          ;CNTRL C ON THE TTY AND ENTERING A VALUE IN RESPONSE
2799          ;TO THE PRINTED REQUEST.
2800          ;*****
2801 010770 013737 000602 000666 YOZ:  MOV     YSTAL,STAL
2802 010776 004737 011654          JSR     PC,STALL   ;DO YOZZLE STALL
2803 011002 012777 177777 167506 YOZO: MOV     #-1,@FC     ;SET TO 1 RECORD SPACING
2804 011010 005737 000562          TST     RDCMD      ;SEE IF READ REVERSE
2805 011014 001404          BEQ     YOZA       ;IF NOT: BR
2806 011016 112737 000030 000672  MOVVB  #30,MTC1    ;SET TO SPACE FORWARD
2807 011024 000403          BR     YOZB
2808 011026 112737 000032 000672 YOZA: MOVVB  #32,MTC1    ;SET TO SPACE REVERSE
2809 011034 012737 011054 000662 YOZB: MOV     #YOZC,RTRN ;SET RETURN ADDRESS
2810 011042 012737 177775 000666  MOV     #177775,STAL ;SET TIME MULTIPLIER
2811 011050 000137 020372          JMP     TAPG        ;GO YOZZLE
2812 011054 005737 000676          YOZC: TST     TMFLG     ;SEE IF TM
2813 011060 001404          BEQ     1;         ;IF NOT: BR
2814 011062 012737 040000 000666  MOV     #40000,STAL ;SET TM STALL
```


2815	011070	000403				BR	2\$		
2816	011072	013737	000602	000666	1\$:	MOV	YSTAL,STAL		
2817	011100	004737	011654		2\$:	JSR	PC,STALL		:DO YOZZLE STALL
2818	011104	012777	032350	167402		MOV	#RDATA,@BA		:SET BUS ADDRESS
2819	011112	005737	000562			TST	RDCMD		:SEE IF READ REVERSE
2820	011116	001416				BEQ	YOZC1		:IF NOT: BR
2821	011120	013703	000556			MOV	FMCNT,R3		
2822	011124	005103				COM	R3		
2823	011126	032737	000020	000552		BIT	#20,UDES		:SEE IF CORE DUMP
2824	011134	001401				BEQ	YOZC0		:IF NOT: BR
2825	011136	006203				ASR	R3		:R3 = FC/2
2826	011140	060377	167350		YOZC0:	ADD	R3,@BA		:SET REVERSE BUS ADDRESS
2827	011144	012737	000076	000672		MOV	#76,MTC1		:SET READ REVERSE
2828	011152	000403				BR	YOZC2		
2829	011154	012737	000070	000672	YOZC1:	MOV	#70,MTC1		:SET READ FORWARD
2830	011162	013777	000556	167326	YOZC2:	MOV	FMCNT,@FC		:SET CHARACTER COUNT
2831	011170	012737	011202	000662		MOV	#YOZD,RTRN		:SET RETURN ADDRESS
2832	011176	000137	020372			JMP	TAPG		:GO READ
2833	011202	032777	004000	167400	YOZD:	BIT	#4000,@SWR		:SEE IF SHOULD CHECK ERRORS
2834	011210	001047				BNE	YOZE		:IF NOT: BR
2835	011212	005737	000676			TST	TMFLG		:SEE IF TAPE MARK TIME
2836	011216	001442				BEQ	YOZD1		:IF NOT: BR
2837	011220	005737	000562			TST	RDCMD		:SEE IF READ REVERSE
2838	011224	001425				BEQ	YOZD0		:IF NOT: BR
2839	011226	012703	032350			MOV	#RDATA,R3		
2840	011232	013704	000556			MOV	FMCNT,R4		
2841	011236	005104				COM	R4		
2842	011240	032737	000020	000552		BIT	#20,UDES		:SEE IF CORE DUMP
2843	011246	001401				BEQ	YOZD4		:IF NOT: BR
2844	011250	006204				ASR	R4		:SET TO FC/2
2845	011252	060403			YOZD4:	ADD	R4,R3		:SET EXPT BUS ADDRESS
2846	011254	042703	000001			BIC	#1,R3		:MAKE EXPT ADDRESS EVEN
2847	011260	032737	002000	000552		BIT	#2000,UDES		:SEE IF PE
2848	011266	001001				BNE	YOZD2		:IF SO: BR
2849	011270	005743				TST	-(R3)		:SET EXPT BA
2850	011272	004737	016614		YOZD2:	JSR	PC,ER2		:GO CHECK ERRORS
2851	011276	000430				BR	YOZF		
2852	011300	012703	032350		YOZD0:	MOV	#RDATA,R3		
2853	011304	032737	002000	000552		BIT	#2000,UDES		:SEE IF PE
2854	011312	001001				BNE	YOZD3		:IF SO: BR
2855	011314	005723				TST	(R3)+		:SET EXPT BA
2856	011316	004737	016614		YOZD3:	JSR	PC,ER2		:GO CHECK ERRORS
2857	011322	000416				BR	YOZF		
2858	011324	004737	016522		YOZD1:	JSR	PC,ERCHK		:ELSE GO CHECK ERRORS
2859	011330	005737	000712		YOZE:	TST	RTYFL		:SEE IF RETRY
2860	011334	001013				BNE	YOZG		:IF SO: BR
2861	011336	032777	020000	167244		BIT	#20000,@SWR		:SEE IF SHOULD CHECK DATA
2862	011344	001005				BNE	YOZF		:IF NOT: BR
2863	011346	005737	000676			TST	TMFLG		:SEE IF TAPE MARK
2864	011352	001002				BNE	YOZF		:IF SO: BR
2865	011354	004737	014750			JSR	PC,DCHK		:ELSE GO CHECK DATA
2866	011360	004737	013614		YOZF:	JSR	PC,DS3		:GO CLEAR DATA AREA
2867	011364	032777	000040	167216	YOZG:	BIT	#40,@SWR		:SEE IF SHOULD CONTINUE YOZZLE
2868	011372	001402				BEQ	YOZH		:IF NOT: BR
2869	011374	000137	011002			JMP	YOZ0		
2870	011400	000207			YOZH:	RTS	PC		:EXIT


```
2871  
2872  
2873  
2874  
2875  
2876  
2877  
2878  
2879  
2880  
2881  
2882  
2883  
2884  
2885  
2886  
2887 011402 013737 000600 000666 BK.SP: MOV TSTAL,STAL  
2888 011410 004737 011654 JSR PC,STALL ;DO TURN AROUND STALL  
2889 011414 012737 023600 000652 MOV #MSG10,EMADDR  
2890 011422 012703 032350 MOV #RDATA,R3 ;SET EXPECTED BA  
2891 011426 010377 167062 MOV R3,@BA  
2892 011432 005737 000564 TST TMEX ;SEE IF TM  
2893 011436 001436 BEQ BO ;IF NOT: BR  
2894 011440 012777 177777 167050 MOV #-1,@FC  
2895 011446 012737 000032 000672 MOV #32,MTC1  
2896 011454 012737 011466 000662 MOV #1$,RTRN  
2897 011462 000137 020372 JMP TAPG ;SPACE TO TM  
2898 011466 032777 010000 167114 1$: BIT #10000,@SWR ;SEE IF SHOULD CHECK ERROR  
2899 011474 001017 BNE BO ;IF NOT: BR  
2900 011476 012737 025017 000652 MOV #MSG55,EMADDR  
2901 011504 032777 000004 167010 BIT #4,@DS ;SEE IF TM  
2902 011512 001006 BNE 2$ ;IF SO: BR  
2903 011514 012737 032350 020224 MOV #RDATA,CADER  
2904 011522 004737 017352 JSR PC,ERPT ;PRINT ERROR  
2905 011526 000402 BR BO  
2906 011530 004737 016614 2$: JSR PC,ER2  
2907 011534 013700 000554 BU: MOV RCNT,R0  
2908 011540 005737 000660 TST EOTREC ;BRANCH IF EOT NOT DETECTED  
2909 011544 100007 BPL 1$  
2910 011546 042737 100000 000660 BIC #100000,EOTREC ;CLEAR EOT INDICATOR  
2911 011554 013703 000660 MOV EOTREC,R3 ;GET # OF RECORDS LEFT IN BLOCK  
2912 011560 160300 SUB R3,R0 ;FORM # OF RECORDS TO BACK SPACE  
2913 011562 005200 INC R0  
2914 011564 012737 023600 000652 1$: MOV #MSG10,EMADDR ;SET ERROR MESH ADDRESS  
2915 011572 012737 011630 000662 MOV #2$,RTRN ;SET RETURN PC  
2916 011600 012777 177777 166710 MOV #-1,@FC ;SET TO BACKSPACE 1 RECORD  
2917 011606 012703 032350 MOV #RDATA,R3 ;SET EXPECTED BA  
2918 011612 010377 166676 MOV R3,@BA  
2919 011616 012737 000032 000672 MOV #32,MTC1 ;SET SPACE REVERSE  
2920 011624 000137 020372 JMP TAPG ;GO DO SPACE  
2921 011630 004737 016614 2$: JSR PC,ER2  
2922 011634 013737 000600 000666 MOV TSTAL,STAL ;DO STALL  
2923 011642 004737 011654 JSR PC,STALL ;STALL  
2924 011646 005300 DEC R0 ;DECREMENT # OF RECORD TO BACKSPACE  
2925 011650 001345 BNE 1$  
2926 011652 000207 RTS PC ;EXIT
```


2927
2928
2929
2930
2931
2932
2933
2934
2935
2936
2937
2938
2939
2940
2941
2942
2943
2944
2945
2946
2947

011654 005337 000666
011660 001375
011662 000207

STALL: DEC STAL :DELAY
BNE STALL :EXIT
RTS PC

```
*****  
:STALL ROUTINE:  
:  
:THIS ROUTINE IS USED TO PROVIDE SOFTWARE DELAYS  
:DURING READ, WRITE, TURN AROUND, AND YOZZLE.  
:THE DELAY TIMES MAY BE SET BY THE OPERATOR AT  
:INITIAL START FROM 200(8) OR MAY BE MODIFIED  
:AT ANY TIME BY ENTERING CNTRL C ON THE TTY AND  
:INSERTING NEW VALUES IN RESPONSE TO THE REQUEST.  
:THE READ STALL AND THE WRITE STALL ARE DELAYS  
:EXECUTED BETWEEN EACH RECORD OF THE DATA BLOCK.  
:THE TURN AROUND STALL IS EXECUTED EACH TIME  
:THE DIRECTION OF TAPE MOVEMENT IS CHANGED AND  
:ALSO EACH TIME THE TAPE OPERATION CHANGES FROM  
:WRITE TO READ OR READ TO WRITE. THE YOZZLE  
:STALL IS EXECUTED ONLY DURING THE YOZZLE ROUTINE.  
:*****
```


2948
2949
2950
2951
2952
2953
2954
2955
2956
2957
2958
2959
2960
2961
2962
2963
2964
2965
2966
2967
2968
2969
2970
2971
2972
2973
2974
2975
2976
2977
2978
2979
2980
2981
2982
2983
2984
2985

011664	012701	177760		
011670	012702	175000		
011674	004737	022312		
011700	042737	000001	000630	
011706	013737	000630	000556	
011714	012737	177777	013652	
011722	000207			
011724	012702	000001		
011730	012701	000500		
011734	004737	022312		
011740	013737	000630	000554	
011746	000207			

```
CCNTR:  MOV    #-20,R1      ;SET HIGH LIMIT
        MOV    #-3000,R2   ;SET LOW LIMIT
        JSR    PC,RANG     ;GO GENERATE NUMBER
        BIC    #1,RANSAV
        MOV    RANSAV,FMCNT ;SET CHAR COUNT
        MOV    #-1,PAIS    ;PRESET DATA PATTERN
        RTS    PC          ;EXIT
```

```
*****
:RANDOM CHARACTER COUNT GENERATOR:
:
:THIS ROUTINE ENTERED VIA CONSOLE SWITCH
:SEVEN (7) IS USED TO GENERATE A RANDOM
:CHARACTER COUNT FOR EACH DATA BLOCK.
:ALL RECORDS WITHIN A GIVEN BLOCK WILL BE
:THE SAME, BUT EACH BLOCK WILL VARY.
:THE LIMITS ARE TWENTY (20) TO FOUR THOUSAND
:(4000) OCTAL CHARACTERS PER RECORD.
*****
```

```
*****
:RANDOM RECORD COUNT GENERATOR:
:
:THIS ROUTINE ENTERED VIA CONSOLE SWITCH SIX (6)
:IS USED TO GENERATE A RANDOM NUMBER OF RECORDS
:FOR EACH BLOCK OF DATA.
:THE LIMITS ARE ONE (1) TO FIVE HUNDRED (500) OCTAL
:RECORDS PER BLOCK.
*****
```

```
RCNTR:  MOV    #1,R2       ;SET LOW LIMIT
        MOV    #500,R1    ;SET HIGH LIMIT
        JSR    PC,RANG     ;GO GENERATE NUMBER
        MOV    RANSAV,RCNT ;SET RECORD COUNT
        RTS    PC          ;EXIT
```


2986
2987
2988
2989
2990
2991
2992
2993
2994
2995
2996
2997
2998
2999
3000
3001
3002
3003
3004
3005
3006
3007
3008
3009
3010
3011
3012
3013
3014
3015
3016
3017
3018
3019
3020
3021
3022
3023
3024
3025
3026
3027
3028
3029
3030
3031
3032
3033
3034
3035
3036
3037
3038
3039
3040
3041

```
*****  
:TEST CONDITION ENTRY ROUTINE:  
:  
:THIS ROUTINE IS USED TO ALLOW THE OPERATOR  
:TO ENTER, AT THE TTY, THE NECESSARY PARAMETERS  
:TO RUN THE PROGRAM AS HE WISHES. THE  
:ROUTINE IS ONLY ENTERED UPON INITIAL STARTING  
:FROM LOCATION 200(8).  
:THE MAIN PURPOSE OF THIS ROUTINE IS TO ESTABLISH  
:A TABLE OF DEVICES TO BE TESTED. THIS TABLE  
:CONSISTS OF AN ENTRY FOR EACH OF ONE (1) TO  
:EIGHT (8) DEVICES. EACH ENTRY CONTAINS THE  
:SLAVE NUMBER, DENSITY, PARITY, AND  
:FORMAT. THE INFORMATION IS ENTERED  
:IN RESPONSE TO PRINTED REQUESTS AT THE TTY.  
:SLAVES MAY BE ENTERED IN ANY ORDER. EACH  
:PARAMETER IS CHECKED FOR LEGALITY BEFORE BEING  
:SET INTO THE TABLE.  
:THE DRIVE NUMBER REQUEST WILL ALSO CHECK THE MASSBUS  
:FOR THE PRESENCE OF THE REQUESTED DRIVE. IF IT IS NOT FOUND,  
:A NON-EXIST DRIVE MESSAGE WILL BE PRINTED AND ANOTHER DRIVE  
:REQUEST MADE. WHEN THE DRIVE IS FOUND, THE RESPONSE IS STORED  
:AND CONTROL PASSED TO THE SLAVE SELECT ROUTINE.  
:THE SLAVE SELECT ROUTINE ALSO CHECKS FOR THE PRESENCE OF THE  
:SLAVE. IF IT IS NOT PRESENT, A MESSAGE IS PRINTED AND ANOTHER  
:REQUEST IS ISSUED. WHEN THE SELECTED SLAVE IS FOUND TO BE  
:PRESENT, A MESSAGE IS PRINTED IF IT IS A 7 CHANNEL DRIVE  
:TO ASSIST IN SELECTING DENSITY, PARITY, AND FORMAT.  
:UPON COMPLETION OF THE DEVICE TABLE, REQUESTS  
:ARE PRINTED FOR ENTRY OF THE NUMBER OF CHARACTERS  
:PER RECORD AND THE NUMBER OF RECORDS PER BLOCK. THE  
:NEXT REQUEST IS FOR A PATTERN NUMBER TO BE USED  
:FOR WRITING AND CHECKING OF READ DATA.  
:FOLLOWING THE PATTERN REQUEST IS THE TAPE MARK OPTION.  
:RESPONDING TO THE REQUEST (TM=) WITH A ONE (1)  
:WILL CAUSE THE PROGRAM TO WRITE A TM AT THE  
:END OF EACH DATA BLOCK AND TO EXPECT THE  
:TM TO BE DETECTED IN EITHER READ FORWARD AND REVERSE  
:OR DURING SPACE OPERATION. A RESPONSE OF ZERO (TM=0)  
:DISALLOWS WRITING OF THE TM AND CAUSES THE READ  
:AND SPACE ROUTINES TO EXPECT NO TM TO BE PRESENT.  
:THE LAST REQUESTS ARE FOR ENTRY OF THE DESIRED  
:WRITE, READ, AND TURN AROUND STALLS.  
*****
```

011750	005737	000636	TINP:	TST	TINF	:SEE IF SHOULD INPUT FROM TTY
011754	001002			BNE	1\$:IF SO: BR
011756	000137	013270		JMP	TINP4	:GET SWITCHES
011762	005037	000674	1\$:	CLR	UNP	:CLEAR TABLE POINTER
011766	005037	004730		CLR	REOTC	:CLEAR EOT UNIT COUNTER
011772	012737	024227	012016	MOV	#MSG31,41\$:GET TITLE MSG
012000	005737	000734		TST	ASEQF	:SEE IF AUTO SEQ
012004	001403			BEQ	4\$:IF NOT: BR
012006	012737	024155	012016	MOV	#MSG30,41\$:SET AUTO SEQ HDR
012014	000004		4\$:	TYPE		:TYPE MSG

3042	012016	000000		41\$:	.WORD 0	:ADDRESS OF APPROPRIATE TITLE MSG	
3043	012020	105077	177772		CLRB @41\$:DO NOT TYPE TITLE ON RESTART	
3044	012024	000004	024311		TYPE,MSG31A	:TYPE INSTRUCTIONS	
3045	012030	105037	024311		CLRB MSG31A	:DO NOT TYPE STARTUP INSTRUCTIONS ON RESTART	
3046	012034	005737	013444		TST SCVFL	:SEE IF SHORT CONVERSATION	
3047	012040	001065			BNE 6\$:IF SO: BR	
3048	012042	000004	025243		TYPE,MSG74	:REQUEST REGISTER START	
3049	012046	013703	000544		MOV REGS,R3		
3050	012052	104400			TYPOCT	:PRINT CURRENT REG START	
3051	012054	012705	000544		MOV #REGS,R5	:SAVE ADDRESS LOCATION	
3052	012060	012701	000007		MOV #7,R1	:SET SIZE OF ENTRY	
3053	012064	012702	176400		MOV #176400,R2	:SET UPPER LIMIT	
3054	012070	012703	172300		MOV #172300,R3	:SET LOWER LIMIT	
3055	012074	004737	022474		JSR PC,TTR	:GO GET RESPONSE	
3056							
3057	012100	000004	025266		TYPE,MSG75	:REQUEST INTERRUPT VECTOR ADDRESS	
3058	012104	013703	000546		MOV VECT,R3		
3059	012110	104400			TYPOCT	:PRINT CURRENT VECTOR	
3060	012112	012705	000546		MOV #VECT,R5	:SET SAVE LOCATION	
3061	012116	012701	000004		MOV #4,R1	:SET SIZE OF ENTRY	
3062	012122	012702	000224		MOV #224,R2	:SET UPPER LIMIT	
3063	012126	012703	000150		MOV #150,R3	:SET LOWER LIMIT	
3064	012132	004737	022474		JSR PC,TTR	:GO GET RESPONSE	
3065	012136	013700	000546		MOV VECT,R0	:GET VECTOR ADDRESS	
3066	012142	012720	021160		MOV #MTINT,(R0)+	:LOAD VECTOR WITH HANDLER ADDRESS	
3067	012146	012710	000340		MOV #340,(R0)	:LOAD PRIORITY LEVEL	
3068	012152	013700	000544		MOV REGS,R0	:GET STARTING REGISTER ADDRESS	
3069	012156	012701	000016		MOV #16,R1	:SET NUMBER OF REGISTERS	
3070	012162	012702	000510		MOV #C1,R2	:GET FIRST ADDRESS LOCATION	
3071	012166	010022		5\$:	MOV R0,(R2)+	:BUILD TABLE OF ADDRESSES	
3072	012170	062700	000002		ADD #2,R0	:BUMP ADDRESS	
3073	012174	005301			DEC R1	:SEE IF DONE	
3074	012176	001373			BNE 5\$:IF NOT: BR	
3075	012200	005737	000734		TST ASEQF	:SEE IF AUTO SEQ	
3076	012204	001403			BEQ 6\$:IF NOT: BR	
3077	012206	005726			TST (SP)+	:RESET STACK POINTER	
3078	012210	000137	021176		JMP ASEQ	:GO TO AUTO SEQUENCE	
3079							
3080	012214	012777	000040	166276	6\$:	MOV #40,@CS	:INITIALIZE
3081	012222	000004	024754		TYPE,MSG52A	:REQUEST DRIVE (TM03) #	
3082	012226	012705	000550		MOV #DVN,R5	:GET ADDRESS	
3083	012232	012701	000002		MOV #2,R1	:SET SIZE OF RESPONSE	
3084	012236	012702	000007		MOV #7,R2	:SET UPPER LIMIT	
3085	012242	012703	000000		MOV #0,R3	:SET LOWER LIMIT	
3086	012246	004737	022474		JSR PC,TTR	:GO GET DRIVE NUMBER	
3087	012252	013777	000550	166240	MOV DVN,@CS		
3088	012260	005777	166224		TST @C1	:ACCESS DRIVE	
3089	012264	032777	010000	166226	BIT #10000,@CS	:SEE IF NED	
3090	012272	001403			BEQ TINPO	:IF NOT: BR	
3091	012274	000004	025200		TYPE,MSG71	:TYPE 'NON-EXISTANT DRIVE'	
3092	012300	000745			BR 6\$:RETRY DVN	
3093							
3094	012302	012705	000646	TINPO:	MOV #TEMP2,R5	:SET ADDRESS FOR RESPONSE	
3095	012306	000004	024376		TYPE,MSG32	:REQUEST SLAVE (TE16,TU77) #	
3096	012312	005037	000646		CLR TEMP2	:CLEAR BUFFER	
3097	012316	012701	000002		MOV #2,R1	:SET NUMBER OF CHARACTERS TO INPUT	

3098	012322	012702	000007		MOV	#7,R2	:SET MAXIMUM LIMIT
3099	012326	012703	000000		MOV	#0,R3	:SET MINIMUM LIMIT
3100	012332	004737	022474		JSR	PC,TTR	:GO GET UNIT NUMBER
3101	012336	005737	000644		TST	TEMP1	:SEE IF HAVE NEW PARAMETER
3102	012342	001010			BNE	TINPOB	:IF SO: BR
3103	012344	013700	000674		MOV	UNP,R0	
3104	012350	001754			BEQ	TINP0	:BRANCH IF FIRST ENTRY
3105	012352	012760	177777	000742	MOV	#-1,UN1(R0)	:SET END UNIT TABLE
3106	012360	000137	012700		JMP	TINP2C	:GO GET RECORD COUNT
3107	012364	013700	000674		TINPOB: MOV	UNP,R0	
3108	012370	011560	000742		MOV	(R5),UN1(R0)	:SET NEW SLAVE #
3109	012374	012777	000040	166116	MOV	#40,@CS	:DO A MASS BUS CLEAR
3110	012402	013777	000550	166110	MOV	DVN,@CS	:LOAD DRIVE #
3111	012410	016077	000742	166124	MOV	UN1(R0),@TC	:LOAD SLAVE NUMBER
3112	012416	032777	002000	166112	BIT	#2000,@DT	:SEE IF SLAVE PRESENT
3113	012424	001003			BNE	TINPOD	:IF SO: BR
3114	012426	000004	025032		TYPE,MSG57		:TYPE NON-EXISTANT SLAVE'
3115	012432	000723			BR	TIN'0	:REDO
3116	012434	017703	166076		TINPOD: MOV	@DT,R3	:GET CONTENTS OF DT REG
3117	012440	042703	000007		BIC	#7,R3	:CLEAR DRIVE TYPE #
3118	012444	022703	142050		CMP	#142050,R3	:SEE IF 9TRK TMO3
3119	012450	001407			BEQ	TINPOE	:IF SO: BR
3120	012452	000004	024725		TYPE,MSG50		:TYPE 'ILLEGAL DRIVE TYPE'
3121	012456	017703	166054		MOV	@DT,R3	
3122	012462	042703	000007		BIC	#7,R3	:CLEAR SLAVE #
3123	012466	104400			TYPOCT		:PRINT DRIVE TYPE REGISTER
3124	012470	004737	023360		TINPOE: JSR	PC,SNPT	:PRINT SERIAL NUMBER
3125							
3126	012474	000004	024411		TINP1: TYPE,MSG33		:REQUEST DENSITY
3127	012500	005037	000646		CLR	TEMP2	:CLEAR BUFFER
3128	012504	012701	000002		MOV	#2,R1	:SET NUMBER OF CHARACTERS TO INPUT
3129	012510	012702	000004		MOV	#4,R2	:SET MAXIMUM LIMIT
3130	012514	012703	000003		MOV	#3,R3	:SET MINIMUM LIMIT
3131	012520	004737	022474		JSR	PC,TTR	:GO GET DENSITY
3132	012524	012703	000010		MOV	#10,R3	:SET POSITION FACTOR
3133	012530	004737	013446		JSR	PC,TPOS	:GO LOAD DENSITY INTO PROPER POSITION
3134							
3135	012534	000315			TINP2: SWAB	(R5)	:IF DENSITY
3136	012536	022715	000004		CMP	#4,(R5)	:IS 1600BPI
3137	012542	001415			BEQ	1\$:THEN SKIP PARITY REQUEST
3138	012544	000004	024424		TYPE,MSG34		:REQUEST PARITY
3139	012550	005037	000646		CLR	TEMP2	:CLR BFR
3140	012554	012701	000002		MOV	#2,R1	:SET NUMBER OF CHAR. TO INPUT
3141	012560	012702	000001		MOV	#1,R2	:SET HIGH LIMIT
3142	012564	012703	000000		MOV	#0,R3	:SET LOW LIMIT
3143	012570	004737	022474		JSR	PC,TTR	:GO INPUT PARITY
3144	012574	000402			BR	2\$:SKIP 1600 BPI PAROTY SETTING
3145	012576	012715	000000		1\$: MOV	#0,(R5)	:SET ODD PARITY FOR 1600 BPI
3146	012602	012703	000003		2\$: MOV	#3,R3	:SET POSITION FACTOR
3147	012606	004737	013446		JSR	PC,TPOS	:GO POSITION PARITY
3148							
3149	012612	000004	024776		TINP2A: TYPE,MSG53		:REQUEST FORMAT
3150	012616	005037	000646		CLR	TEMP2	
3151	012622	012701	000003		MOV	#3,R1	
3152	012626	012702	000017		MOV	#7,R2	
3153	012632	012703	000000		MOV	#0,R3	

3154	012636	004737	022474		JSR	PC,TTR	:GO GET FORMAT
3155	012642	012703	000004		MOV	#4,R3	
3156	012646	004737	013446		JSR	PC,TPOS	
3157	012652	005237	004730		TINP2B: INC	REOTC	:BUMP EOT UNIT COUNTER
3158	012656	022737	000016	000674	CMP	#16,UNP	:SEE IF DONE UNITS
3159	012664	001405			BEQ	TINP2C	:IF SO: BR
3160	012666	062737	000002	000674	ADD	#2,UNP	:POINT TO NEXT UNIT
3161	012674	000137	012302		JMP	TINP0	:ELSE LOOK FOR NEXT UNIT
3162							
3163							
3164	012700	005037	000674		TINP2C: CLR	UNP	:CLEAR UNIT POINTER
3165	012704	113737	004730	004731	MOVB	REOTC,REOTC+1	:SET # OF UNITS TO TEST
3166							
3167	012712	000004	024436		TINP3: TYPE,MSG35		:REQUEST RECORDS PER BLOCK
3168	012716	013703	000554		MOV	RCNT,R3	
3169	012722	104400			TYPOCT		:PRINT RECORD COUNT
3170	012724	012705	000554		MOV	#RCNT,R5	:SET RECORD COUNT ADDRESS
3171	012730	012701	000007		MOV	#7,R1	:SET NUMBER OF CHARACTERS TO INPUT
3172	012734	012702	177777		MOV	#177777,R2	:SET MAXIMUM LIMIT
3173	012740	012703	000001		MOV	#1,R3	:SET MINIMUM LIMIT
3174	012744	004737	022474		JSR	PC,TTR	:GO GET RECORD COUNT
3175	012750	013737	000554	000632	MOV	RCNT,RCSAV	:SAVE RECORD COUNT
3176							
3177	012756	000004	024456		TYPE,MSG36		:REQUEST CHARACTERS PER RECORD
3178	012762	005437	000556		NEG	FMCNT	
3179	012766	013703	000556		MOV	FMCNT,R3	
3180	012772	104400			TYPOCT		:PRINT CHAR COUNT
3181	012774	012705	000556		MOV	#FMCNT,R5	:SET CHARACTER COUNT ADDRESS
3182	013000	012701	000007		MOV	#7,R1	:SET NUMBER OF CHARACTERS TO INPUT
3183	013004	012702	004000		MOV	#4000,R2	:SET MAXIMUM LIMIT
3184	013010	012703	000004		MOV	#4,R3	:SET MINIMUM LIMIT
3185	013014	004737	022474		JSR	PC,TTR	:GO GET CHARACTER COUNT
3186	013020	005437	000556		NEG	FMCNT	:SET TO TWO'S COMPLIMENT
3187	013024	013737	000556	000634	MOV	FMCNT,FCSAV	:SAVE FRAME COUNT
3188							
3189	013032	000004	024474		TYPE,MSG37		:REQUEST PATTERN #
3190	013036	013703	000560		MOV	PATRN,R3	
3191	013042	104400			TYPOCT		:PRINT PATTERN
3192	013044	005037	014014		CLR	DOFL	:CLEAR EXTERNAL DATA FLAG
3193	013050	012705	000560		MOV	#PATRN,R5	:SET PATTERN NUMBER ADDRESS
3194	013054	012701	000003		MOV	#3,R1	:SET NUMBER OF CHARACTERS TO INPUT
3195	013060	012702	000015		MOV	#15,R2	:SET MAXIMUM LIMIT
3196	013064	012703	000000		MOV	#0,R3	:SET MINIMUM LIMIT
3197	013070	004737	022474		JSR	PC,TTR	:GO GET PATTERN NUMBER
3198							
3199	013074	000004	025164		TYPE,MSG69		:REQUEST TAPE MARK
3200	013100	013703	000564		MOV	TMEX,R3	
3201	013104	104400			TYPOCT		:PRINT CURRENT TM FLAG SETTING
3202	013106	012705	000564		MOV	#TMEX,R5	:GET TM FLAG ADDRESS
3203	013112	012701	000002		MOV	#2,R1	:SET SIZE OF RESPONSE
3204	013116	012702	000001		MOV	#1,R2	:SET UPPER LIMIT
3205	013122	012703	000000		MOV	#0,R3	:SET LOWER LIMIT
3206	013126	004737	022474		JSR	PC,TTR	:TM 1=YES
3207							
3208	013132	000004	023666		TYPE,MSG21		:REQUEST INTERCHANGE READ
3209	013136	013703	000570		MOV	INTRF,R3	

3210	013142	104400		TYPOCT		:PRINT CURRENT SETTING
3211	013144	012705	000570	MOV #INTRF,R5		:GET FLAG ADDRESS
3212	013150	012701	000002	MOV #2,R1		:SET SIZE OF RESPONSE
3213	013154	012702	000001	MOV #1,R2		:SET UPPER LIMIT
3214	013160	012703	000000	MOV #0,R3		:SET LOWER LIMIT
3215	013164	004737	022474	JSR PC,TTR		:GO GET RESPONSE
3216						
3217	013170	000004	024511	TYPE,MSG38		:REQUEST SINGLE PASS
3218	013174	013703	000572	MOV SPFLG,R3		
3219	013200	104400		TYPOCT		:PRINT CURRENT SETTING
3220	013202	012705	000572	MOV #SPFLG,R5		:SET ADDRESS OF FLAG
3221	013206	012701	000002	MOV #2,R1		:SET SIZE OF RESPONSE
3222	013212	012702	000001	MOV #1,R2		:SET UPPER LIMIT
3223	013216	012703	000000	MOV #0,R3		:SET LOWER LIMIT
3224	013222	004737	022474	JSR PC,TTR		:GO GET RESPONSE
3225						
3226	013226	000004	024527	TINP3A: TYPE,MSG39		:REQUEST CRC CORRECTION
3227	013232	013703	000566	MOV CRCC,R3		
3228	013236	104400		TYPOCT		
3229	013240	012705	000566	MOV #CRCC,R5		
3230	013244	012701	000002	MOV #2,R1		
3231	013250	012702	000001	MOV #1,R2		
3232	013254	012703	000000	MOV #0,R3		
3233	013260	004737	022474	JSR PC,TTR		
3234	013264	004737	022344	JSR PC,GTSWR		:GET SWITCHES
3235	013270	005737	013444	TINP4: TST SCVFL		:BRANCH IF SHORT CONVERSATION
3236	013274	001060		BNE TINPX		
3237	013276	005737	000636	1\$: TST TINF		:BRANCH IF NO TTY INPUT
3238	013302	001455		BEQ TINPX		
3239	013304	000004	024565	TYPE,MSG40		:REQUEST READ STALL
3240	013310	013703	000574	MOV RSTAL,R3		
3241	013314	104400		TYPOCT		:PRINT READ STALL
3242	013316	012705	000574	MOV #RSTAL,R5		:SET READ STALL ADDRESS
3243	013322	012701	000007	MOV #7,R1		:SET NUMBER OF CHARACTERS TO INPUT
3244	013326	012702	177777	MOV #-1,R2		:SET MAXIMUM LIMIT
3245	013332	012703	000001	MOV #1,R3		:SET MINIMUM LIMIT
3246	013336	004737	022474	JSR PC,TTR		:GO GET READ STALL
3247						
3248	013342	000004	024614	TYPE,MSG41		:REQUEST WRITE STALL
3249	013346	013703	000576	MOV WSTAL,R3		
3250	013352	104400		TYPOCT		:PRINT READ STALL
3251	013354	012705	000576	MOV #WSTAL,R5		:SET WRITE STALL ADDRESS
3252	013360	012701	000007	MOV #7,R1		:SET NUMBER OF CHARACTERS TO INPUT
3253	013364	012702	177777	MOV #-1,R2		:SET MAXIMUM LIMIT
3254	013370	012703	000001	MOV #1,R3		:SET MINIMUM LIMIT
3255	013374	004737	022474	JSR PC,TTR		:GO GET WRITE STALL
3256						
3257	013400	000004	024625	TYPE,MSG42		:REQUEST TURN AROUND STALL
3258	013404	013703	000600	MOV TSTAL,R3		
3259	013410	104400		TYPOCT		:PRINT TA STALL
3260	013412	012705	000600	MOV #TSTAL,R5		:SET TURN AROUND STALL ADDRESS
3261	013416	012701	000007	MOV #7,R1		:SET NUMBER OF CHARACTERS TO INPUT
3262	013422	012702	177777	MOV #-1,R2		:SET MAXIMUM LIMIT
3263	013426	012703	000001	MOV #1,R3		:SET MINIMUM LIMIT
3264	013432	004737	022474	JSR PC,TTR		:GO GET TURN AROUND STALL
3265	013436	005037	013444	TINPX: CLR SCVFL		:CLEAR SHORT CONVERSATION FLAG

3278
 3279
 3280
 3281
 3282
 3283
 3284
 3285
 3286
 3287
 3288
 3289
 3290
 3291
 3292
 3293
 3294
 3295
 3296
 3297
 3298
 3299
 3300
 3301
 3302
 3303
 3304
 3305
 3306
 3307
 3308
 3309
 3310
 3311
 3312
 3313
 3314
 3315
 3316
 3317
 3318
 3319
 3320
 3321
 3322
 3323
 3324
 3325
 3326
 3327
 3328
 3329
 3330
 3331
 3332
 3333

```

:*****
:DATA SETUP ROUTINE:
:
:THIS ROUTINE IS USED TO GENERATE INTO THE ENTIRE
:WRITE BUFFER (4000 OCTAL CHARACTERS) THE DATA PATTERN
:SELECTED BY THE OPERATOR. THERE ARE 15 (8) FIXED
:DATA PATTERNS AVAILABLE AND ONE SELECTION (DATA PATTERN 0)
:WHICH WILL READ ANY PATTERN PRESENTED AT THE
:HIGH SPEED PAPER TAPE READER. THIS TAPE MUST BE PREPARED
:BY USING THE PROGRAM CALLED DTC. (MAINDEC-11-DZTUF-A-D)
:RANDOM DATA MAY ALSO BE USED VIA CONSOLE
:SWITCH EIGHT (8).
:THIS ROUTINE IS ALSO USED TO CLEAR OUT THE
:READ BUFFER (4000 OCTAL CHARACTERS) BEFORE EACH
:RECORD IS READ.
:*****
  
```

```

3296 013472 005737 014404          DSUP:  TST      RDFL          ;SEE IF DID RANDOM DATA
3297 013476 001044                    BNE      DS2A          ;IF NOT: BR
3298 013500 005737 000734          DSO:    TST      ASEQF         ;SEE IF AUTO SEQ
3299 013504 001406                    BEQ      DSOC          ;IF NOT: BR
3300 013506 005737 000560          TST      PATRN         ;SEE IF AUTO RANDOM
3301 013512 100003                    BPL      DSOC          ;IF NOT: BR
3302 013514 004737 014342          JSR      PC,DATR       ;ELSE GO GENERATE RANDOM DATA
3303                                ;      RTS      PC      ;++B DELETED
3304 013520 000433                    BR       DS2A          ;++B GENERATE EXPECTED LRC/CRC & CLEAR READ BFR
3305 013522 023737 000560 013652  DSOC:    CMP      PATRN,PATS    ;SEE IF NEW PATTERN
3306 013530 001014                    BNE      DSOA          ;IF SO: BR
3307 013532 013703 000552          MOV      UDES,R3      ;GET UNIT DESCRIPTION
3308 013536 042703 177767          BIC      #177767,R3   ;MASK EVEN PARITY
3309 013542 023703 013654          CMP      PARS,R3      ;SEE IF SAME AS LAST TIME
3310 013546 001404                    BEQ      DSOB          ;IF SO: BR
3311 013550 010337 013654          MOV      R3,PARS      ;SAVE PARITY
3312 013554 004737 014406          JSR      PC,CRCLRC    ;GO GENERATE EXPT CRC/LRC
3313 013560 000207                    DS0B:   RTS      PC
3314 013562 012703 026342          DSOA:   MOV      #WDATA,R3 ;R3 = ADDRS OF WRITE BUFFER
3315 013566 013701 000560          MOV      PATRN,R1     ;R1 = PATTERN SELECTOR
3316 013572 010137 013652          MOV      R1,PATS
3317 013576 062701 000001          ADD      #1,R1
3318 013602 006301                    ASL      R1            ;BUMP POINTER
3319 013604 004771 002764          JSR      PC,@DATBL(R1);MAKE PATTERN SELECTOR EVEN
3320 013610 004737 014406          DS2A:   JSR      PC,CRCLRC ;GO GENERATE PATTERN
3321 013614 013702 000556          DS3:    MOV      FMCNT,R2    ;GO GENERATE EXPT CRC/LRC
3322 013620 006202                    ASR      R2            ;R2=BUFFER SIZE
3323 013622 012701 032350          MOV      #RDATA,R1   ;R2=FRAME CMT/2
3324 013626 005021                    DS4:    CLR      (R1)+        ;R1=READ DATA START
3325 013630 005202                    INC      R2            ;CLEAR BUFFER
3326 013632 001375                    BNE      DS4          ;SEE IF DONE ALL
3327 013634 013737 000552 013654  MOV      UDES,PARS    ;IF NOT: BR
3328 013642 042737 177767 013654  BIC      #177767,PARS ;GET UNIT DESCRIPTION
3329 013650 000207                    RTS      PC           ;MASK PARITY
3330 013652 177777                    PATS:   -1            ;EXIT
3331 013654 000000                    PARS:   0            ;PATTERN NUMBER SAVE
  
```



```
3334  
3335  
3336  
3337 013656 005737 014014 DATO: TST DOFL ;BRANCH IF SHOULD DO EXTERNAL INPUT  
3338 013662 001401 BEQ 1$  
3339 013664 000207 RTS PC ;++B RETURN  
3340 013666 012737 000001 014014 1$: MOV #1,DOFL ;SET EXTERNAL FLAG  
3341 013674 005077 164722 CLR @PRS ;CLEAR READER STATUS  
3342 013700 005037 000644 CLR TEMP1 ;CLEAR FOR USE AS CHARACTER FLAG  
3343 013704 052777 000001 164710 DATOA: BIS #1,@PRS ;START READER  
3344 013712 105777 164704 DATOB: TSTB @PRS ;SEE IF DONE  
3345 013716 100375 BPL DATOB ;IF NOT : BR  
3346 013720 005001 CLR R1 ;CLEAR SAVE LOCATION  
3347 013722 117701 164676 MOVB @PRB,R1 ;SAVE CHARACTER  
3348 013726 005737 000644 TST TEMP1 ;SEE IF HAVE FOUND START CHARACTER  
3349 013732 001011 BNE DATOC ;IF SO : BR  
3350 013734 105701 TSTB R1 ;SEE IF CHARACTER IS 0  
3351 013736 001762 BEQ DATOA ;IF SO : BR  
3352 013740 012737 000001 000644 MOV #1,TEMP1 ;ELSE SET CHARACTER FOUND FLAG  
3353 013746 010137 000646 MOV R1,TEMP2 ;SAVE DATA SIZE  
3354 013752 010102 MOV R1,R2 ;SAVE DATA SIZE  
3355 013754 000753 BR DATOA ;GO GET FIRST DATA CHAR  
3356 013756 110123 DATOC: MOVB R1,(R3)+ ;LOAD BUFFER  
3357 013760 005302 DEC R2 ;SEE IF READ ALL  
3358 013762 001350 BNE DATOA ;IF NOT : BR  
3359 013764 012701 026342 DATOD: MOV #WDATA,R1 ;R1 = START OF WRITE BUFFER  
3360 013770 013702 000646 MOV TEMP2,R2 ;R2 = SIZE OF DATA FIELD  
3361 013774 112123 DATOE: MOVB (R1)+,(R3)+ ;REPEAT LOAD OF DATA FIELD  
3362 013776 022703 032350 CMP #RDATA,R3 ;SEE IF DONE  
3363 014002 003001 BGT DATOF ;IF NOT: BR  
3364 014004 000207 RTS PC ;++B RETURN  
3365 014006 005302 DATOF: DEC R2 ;SEE IF AT END OF DATA FIELD  
3366 014010 001371 BNE DATOE ;IF NOT : BR  
3367 014012 000764 BR DATOD ;ELSE RESTART FILL  
3368 014014 000000 DOFL: 0 ;EXTERNAL DATA FLAG=1 IF ALREADY DONE  
3369
```



```

3370                                     ;ALL ONES*****
3371
3372 014016 012701 177777  DAT1:  MOV    #-1,R1          ;R1=DATA
3373 014022 012702 002002  DAT1A: MOV    #2002,R2        ;R2=WORD COUNT +2
3374 014026 010123          1$:  MOV    R1,(R3)+        ;LOAD BUFFER
3375 014030 005302          DEC    R2          ;SEE IF DONE
3376 014032 001375          BNE   1$          ;IF NOT: BR
3377 014034 000207          RTS    PC
3378
3379                                     ;ALL ZEROS*****
3380
3381 014036 005001  DAT2:  CLR    R1          ;R1=DATA
3382 014040 000770          BR     DAT1A        ;LOAD BUFFER
3383
3384                                     ;WALKING ONE*****
3385
3386 014042 012701 000001  DAT3:  MOV    #1,R1          ;R1=DATA
3387 014046 000241          CLC
3388 014050 012702 004004  DAT3A: MOV    #4004,R2        ;R2=CHARACTER COUNT+4
3389 014054 110123          1$:  MOVB   R1,(R3)+        ;LOAD BUFFER
3390 014056 106101          ROLB   R1          ;SET NEXT CHARACTER
3391 014060 005302          DEC    R2          ;SEE IF DONE
3392 014062 001374          BNE   1$          ;IF NOT: BR
3393 014064 000207          RTS    PC
3394
3395                                     ;WALKING ZERO*****
3396
3397 014066 012701 000376  DAT4:  MOV    #376,R1        ;R1=START OF DATA
3398 014072 000261          SEC
3399 014074 000765          BR     DAT3A        ;LOAD BUFFER
3400
3401                                     ;ALTERNATING ONE/ZERO*****
3402
3403
3404 014076 012701 052525  DAT5:  MOV    #52525,R1       ;R1=DATA
3405 014102 000747          BR     DAT1A        ;LOAD BUFFER
3406
3407                                     ;ALTERNATING ZERO/ONE*****
3408
3409 014104 012701 125252  DAT6:  MOV    #125252,R1      ;R1=DATA
3410 014110 000744          BR     DAT1A        ;LOAD BUFFER
3411
3412                                     ;ONE/ZERO IN ALTERNATING WORDS*****
3413
3414 014112 012701 125252  DAT7:  MOV    #125252,R1      ;SET WORD 1
3415 014116 012702 052525  MOV    #52525,R2          ;SET WORD 2
3416 014122 012704 001002  MOV    #1002,R4          ;SET NUMBER OF ENTRIES
3417 014126 010123          1$:  MOV    R1,(R3)+        ;LOAD WORD 1
3418 014130 010223          MOV    R2,(R3)+        ;LOAD WORD 2
3419 014132 005304          DEC    R4          ;SEE IF DONE
3420 014134 001374          BNE   1$          ;IF NOT: BR
3421 014136 000207          RTS    PC
3422

```



```

3423                                     ;WALKING ONE/ALL ONE IN ALTERNATING CHARS****
3424
3425 014140 012702 002002      DAT10: MOV      #2002,R2      ;SET BUFFER SIZE
3426 014144 012701 000001      MOV      #1,R1          ;SET WALK BASE
3427 014150 000241
3428 014152 012713 177400      1$:  MOV      #177400,(R3) ;LOAD ALL ONE BYTE
3429 014156 050123              BIS      R1,(R3)+      ;LOAD WALK BYTE
3430 014160 106101              ROLB    R1             ;WALK ONE
3431 014162 005302              DEC     R2
3432 014164 001372              BNE    1$             ;DO FULL BUFFER
3433 014166 000207              RTS     PC
3434
3435                                     ;ALL BITS 0-377*****
3436
3437 014170 005001      DAT11: CLR     R1          ;R1=STARTING DATA
3438 014172 012702 004004      MOV     #4004,R2       ;R2=CHARACTER COUNT+4
3439 014176 110123      1$:  MOVB   R1,(R3)+    ;LOAD BUFFER
3440 014200 105201      INCB   R1             ;BUMP DATA
3441 014202 005302      DEC    R2             ;SEE IF DONE
3442 014204 001374      BNE    1$             ;IF NOT: BR
3443 014206 000207      RTS    PC             ;RETURN
3444
3445                                     ;ALL BITS 377-0*****
3446
3447 014210 012701 000377      DAT12: MOV     #377,R1   ;R1=STARTING DATA
3448 014214 012702 004004      MOV     #4004,R2       ;R2=CHARACTER COUNT+4
3449 014220 110123      1$:  MOVB   R1,(R3)+    ;LOAD BUFFER
3450 014222 105301      DECB   R1             ;BUMP DATA
3451 014224 005302      DEC    R2             ;SEE IF DONE
3452 014226 001374      BNE    1$             ;IF NOT: BR
3453 014230 000207      RTS    PC             ;RETURN
3454
3455                                     ;ALTERNATING CHARACTERS 0 AND 377*****
3456
3457 014232 012701 000377      DAT13: MOV     #377,R1   ;R1 = DATA
3458 014236 000137 014022      JMP     DAT1A          ;LOAD BUFFER
3459
3460                                     ;WALKING ZERO/ALL ZERO IN ALTERNATING CHARS*****
3461
3462 014242 012702 002002      DAT14: MOV     #2002,R2   ;SET BUFFER SIZE
3463 014246 012701 000376      MOV     #376,R1        ;SET WALK BASE
3464 014252 000261
3465 014254 010113      1$:  MOV     R1,(R3)     ;LOAD WALK BYTE
3466 014256 042723 177400      BIC    #177400,(R3)+  ;CLEAR HIGH BYTE
3467 014262 106101      ROLB   R1             ;WALK ZERO BIT
3468 014264 005302      DEC    R2
3469 014266 001372      BNE    1$             ;FILL BUFFER
3470 014270 000207      RTS    PC             ;RETURN
3471

```



```
3472                                     ;AUTO SEQUENCE PATTERN*****
3473
3474 014272 012702 000200          DAT15: MOV      #200,R2          ;SET NUMBER OF ENTRIES
3475 014276 012701 014322          1$:  MOV      #APATS,R1        ;SET START OF PATTERN
3476 014302 012704 000010          MOV      #10,R4              ;SET SIZE OF PATTERN
3477 014306 012123                2$:  MOV      (R1)+,(R3)+     ;FILL BUFFER
3478 014310 005304                DEC      R4                  ;SEE IF DONE PATTERN
3479 014312 001375                BNE     2$                   ;IF NOT: BR
3480 014314 005302                DEC      R2                  ;SEE IF DONE BUFER
3481 014316 001367                BNE     1$                   ;IF NOT: BR
3482 014320 000207                RTS      PC                  ;RETURN
3483
3484 014322 000000          APATS: 0
3485 014324 177400          177400
3486 014326 000377          377
3487 014330 000000          0
3488 014332 177777          -1
3489 014334 000377          377
3490 014336 177400          177400
3491 014340 177777          -1
3492
3493                                     ;RANDOM DATA GENERATOR SUBROUTINE*****
3494
3495 014342 013704 000556          DATR: MOV      FMCNT,R4        ;SET NUMBER OF FRAMES
3496 014346 012703 026342          MOV      #WDATA,R3         ;SET ADDRESS OF START OF BUFFER
3497 014352 012701 177777          MOV      #-1,R1            ;SET HIGH LIMIT
3498 014356 005002                CLR      R2                 ;SET LOW LIMIT
3499 014360 004737 022312          1$:  JSR      PC,RANG         ;GO GENERATE NUMBER
3500 014364 013723 000630          MOV      RANSV,(R3)+      ;LOAD BUFFER
3501 014370 005204                INC      R4                 ;SEE IF DONE WHOLE BUFFER
3502 014372 001372                BNE     1$                   ;IF NOT: BR
3503 014374 012737 000001 014404  MOV      #1,RDFL           ;SET RANDOM DATA FLAG
3504 014402 000207                RTS      PC                  ;EXIT
3505 014404 000000          RDFL: 0                    ;RANDOM DATA SELECT FLAG
```



```

3506
3507
3508
3509
3510
3511
3512
3513
3514
3515 014406 013700 000556
3516 014412 005400
3517 014414 012701 026342
3518 014420 005037 014742
3519 014424 111104
3520 014426 004737 014614
3521 014432 004737 014716
3522 014436 000241
3523 014440 006004
3524 014442 103014
3525 014444 052704 000400
3526 014450 000241
3527 014452 010405
3528 014454 042705 177703
3529 014460 005105
3530 014462 042705 177703
3531 014466 042704 000074
3532 014472 050504
3533 014474 010437 014742
3534 014500 005300
3535 014502 001350
3536 014504 013704 014742
3537 014510 005137 014742
3538 014514 042737 177050 014742
3539 014522 042704 177727
3540 014526 050437 014742
3541 014532 013737 014742 014744
3542 014540 013700 000556
3543 014544 005400
3544 014546 012701 026342
3545 014552 005037 014742
3546 014556 111104
3547 014560 004737 014614
3548 014564 004737 014716
3549 014570 005300
3550 014572 001371
3551 014574 013704 014744
3552 014600 004737 014716
3553 014604 013737 014742 014746
3554 014612 000207
3555 014614 005704
3556 014616 001010
3557 014620 032737 000010 000552
3558 014626 001404
3559 014630 012704 000420
3560 014634 005201
3561 014636 000207

:*****
:CRC/LRC CHARACTER BUILD;
:
:THIS ROUTINE WILL CONSTRUCT AND SAVE THE EXPECTED
:CRC AND LRC CHARACTERS ACCORDING TO DATA AND
:RECORD SIZE IF OPERATING IN NRZ MODE
:*****

CRCLRC: MOV FMCNT,R0 ;SET RECORD SIZE
        NEG R0
        MOV #WDATA,R1 ;SET START OF BUFFER
        CLR XORS
CL0:    MOV (R1),R4 ;GET CHARACTER
        JSR PC,CLP ;GO GET PARITY OF CHARACTER
        JSR PC,XOR ;XOR CHARACTER
        CLC
        ROR R4 ;ROTATE 1 RIGHT
        BCC CL2 ;IF NO CARRY: BR
        BIS #400,R4 ;SET BIT NINE
        CLC
CL1:    MOV R4,R5 ;SAVE CHARACTER
        BIC #177703,R5
        COM R5
        BIC #177703,R5
        BIC #74,R4
        BIS R5,R4 ;COMPLIMENT BITS 2,3,4,5
CL2:    MOV R4,XORS
        DEC R0
        BNE CLO ;BRANCH IF NOT LAST CHAR
CLLAST: MOV XORS,R4
        COM XORS
        BIC #177050,XORS
        BIC #177727,R4 ;COMPLIMENT ALL BUT BITS 3&5
        BIS R4,XORS
        MOV XORS,EXCRC ;SAVE EXPECTED CRC
        MOV FMCNT,R0
        NEG R0
        MOV #WDATA,R1 ;DO EXPT LRC
        CLR XORS
CL3:    MOV (R1),R4
        JSR PC,CLP ;GET PARITY
        JSR PC,XOR ;XOR CHARACTER
        DEC R0
        BNE CL3 ;DO ALL FOR LRC
        MOV EXCRC,R4
        JSR PC,XOR ;XOR CRC TO DATA
        MOV XORS,EXLRC ;SAVE EXPT LRC
        RTS PC ;RETURN
CLP:    TST R4 ;SEE IF 0 CHAR
        BNE CLPE ;IF NOT: BR
        BIT #10,UDES ;SEE IF EVEN PARITY
        BEQ CLPE ;IF NOT: BR
        MOV #420,R4 ;SET 0 CHAR EVEN PARITY
        INC R1 ;BUMP POINTER
        RTS PC ;RETURN
  
```


3562	014640	005046		CLPE:	CLR	-(SP)		:CLEAR WORLING SPACE ON STACK
3563	014642	106304		1\$:	ASLB	R4		:SHIFT DATA
3564	014644	005516			ADC	(SP)		:ADDUP # OF 1 BITS
3565	014646	105704			TSTB	R4		:BRANCH IF ALL 0'S LEFT
3566	014650	001374			BNE	1\$		
3567	014652	112104			MOVB	(R1)+,R4		
3568	014654	042704	177400		BIC	#177400,R4		
3569	014660	106026			RORB	(SP)+		:BRANCH IF ODD # OF 1 BITS
3570	014662	103405			BCS	CLP2		
3571	014664	032737	000010 000552		BIT	#10,UDES		:SEE IF SHOULD BE EVEN PARITY
3572	014672	001406			BEQ	CLP3		:IF NOT: BR
3573	014674	000207			RTS	PC		:ELSE EXIT
3574	014676	032737	000010 000552	CLP2:	BIT	#10,UDES		:SEE IF SHOULD BE ODD PARITY
3575	014704	001001			BNE	CLP3		:IF NOT: BR
3576	014706	000207			RTS	PC		:ELSE EXIT
3577	014710	052704	000400	CLP3:	BIS	#400,R4		:SET PARITY BIT
3578	014714	000207			RTS	PC		
3579								
3580	014716	010446		XOR:	MOV	R4,-(SP)		
3581	014720	043716	014742		BIC	XORS,(SP)		
3582	014724	040437	014742		BIC	R4,XORS		:XOR SUBROUTINE: R4 WITH XORS
3583	014730	052637	014742		BIS	(SP)+,XORS		
3584	014734	013704	014742		MOV	XORS,R4		
3585	014740	000207			RTS	PC		
3586								
3587	014742	000000		XORS:	0			:XOR SAVE
3588	014744	000000		EXCRC:	0			:EXPECTED CRC
3589	014746	000000		EXLRC:	0			:EXPECTED LRC
3590								

3591
 3592
 3593
 3594
 3595
 3596
 3597
 3598
 3599
 3600
 3601
 3602
 3603
 3604
 3605
 3606
 3607
 3608
 3609
 3610
 3611
 3612
 3613
 3614
 3615
 3616
 3617
 3618
 3619
 3620
 3621
 3622
 3623
 3624
 3625
 3626
 3627
 3628
 3629
 3630
 3631
 3632
 3633
 3634
 3635
 3636
 3637
 3638
 3639
 3640
 3641
 3642
 3643
 3644
 3645
 3646

014750 005037 000656
 014754 005037 000704
 014760 013705 000556
 014764 032737 000020 000552
 014772 001401
 014774 006205
 014776 012701 026342
 015002 012702 032350
 015006 032737 000010 000552
 015014 001430
 015016 032737 000020 000552
 015024 001024
 015026 032737 002000 000552
 015034 001020
 015036 105711
 015040 001404
 015042 005201
 015044 005205
 015046 001373
 015050 000406
 015052 112721 000020
 015056 012737 177777 013652
 015064 000767
 015066 013705 000556
 015072 012701 026342
 015076 005737 000562
 015102 001462
 015104 013704 000556
 015110 005404
 015112 032737 000020 000552
 015120 001402
 015122 000241
 015124 006004
 015126 060401
 015130 060402
 015132 032737 000001 000556
 015140 001401
 015142 105722
 015144 032737 000020 000552
 015152 001431
 015154 000241

```

DCHK: CLR BBC ;CLEAR BAD RECORD CNTR
      CLR DERFL ;CLEAR DATA ERROR FLAG
      MOV FMCNT,R5 ;LOAD CHAR COUNT
      BIT #20,UDES ;SEE IF CORE DUMP
      BEQ DCHK0 ;IF NOT: BR
      ASR R5 ;R5 = FC/2
DCHK0: MOV #WDATA,R1 ;SET WRITE DATA ADDR
      MOV #RDATA,R2 ;SET READ DATA ADDR
      BIT #10,UDES ;SEE IF EVEN PARITY
      BEQ DFOC0 ;IF NOT: BR
      BIT #20,UDES ;SEE IF CORE DUMP PARITY
      BNE DFOC0 ;IF SO: BR
      BIT #2000,UDES ;SEE IF PE MODE
      BNE DFOC0 ;IF SO: BR
DFOF: TSTB (R1) ;SEE IF 0 CHAR
      BEQ DFOD ;IF SO: BR
      INC R1 ;BUMP POINTER
DFOE: INC R5 ;SEE IF DONE
      BNE DFOF ;IF NOT: BR
      BR DFOC ;ELSE CONTINUE
DFOD: MOVB #20,(R1)+ ;SET 20 IN PLACE OF 0
      MOV #-1,PATS ;SET PATTERN GENERATE FLAG
      BR DFOE
DFOC: MOV FMCNT,R5 ;RESET CHAR CNT
      MOV #WDATA,R1 ;RESET DATA ADDRESS
DFOCO: TST RDCMD ;SEE IF READ REVERSE
      BEQ DFO ;IF NOT: BR
      MOV FMCNT,R4 ;GET FRAME COUNT
      NEG R4 ;SET TO WHOLE NUMBER
      BIT #20,UDES ;SEE IF CORE DUMP
      BEQ DFOB0 ;IF NOT: BR
      CLC
      ROR R4 ;SET TO FC/2
DFOB0: ADD R4,R1 ;POINT TO START OF WRITE DATA
      ADD R4,R2 ;POINT TO START OF READ DATA
      BIT #1,FMCNT ;SEE IF ODD FRAME COUNT
      BEQ DFOA ;IF NOT: BR
      TSTB (R2)+ ;BUMP POINTER
DFOA: BIT #20,UDES ;SEE IF CORE DUMP
      BEQ DFOA4 ;IF NOT: BR
      CLC
  
```

```

*****
:DATA CHECK SUBROUTINE:
:
:THIS SUBROUTINE IS USED TO COMPARE EACH CHARACTER
:OF DATA READ FROM TAPE WITH THE EXPECTED CHARACTER.
:ANY ERROR DETECTED WILL CAUSE CONTROL TO BE
:PASSED TO AN ERROR PRINT SUBROUTINE AND A
:SUBROUTINE TO ACCUMULATE THE NUMBER OF BITS
:DROPPED AND PICKED UP FROM EACH CHARACTER.
:THE NUMBER OF READ ERRORS IS ALSO ACCUMULATED.
:DATA CHECKING MAY BE TERMINATED BY USE OF
:CONSOLE SWITCH THIRTEEN (13).
*****
  
```


3647	015156	132742	000001		BITB	#1, -(R2)	:SEE IF BIT 0 = 1
3648	015162	001401			BEQ	DF0A0	:IF NOT: BR
3649	015164	000261			SEC		
3650	015166	106012		DF0A0:	RORB	(R2)	
3651	015170	000241			CLC		
3652	015172	132712	000001		BITB	#1, (R2)	
3653	015176	001401			BEQ	DF0A1	
3654	015200	000261			SEC		
3655	015202	106012		DF0A1:	RORB	(R2)	:POSITION BITS FOR REVERSE CORE DUMP
3656	015204	000241			CLC		
3657	015206	132712	000001		BITB	#1, (R2)	
3658	015212	001401			BEQ	DF0A2	
3659	015214	000261			SEC		
3660	015216	106012		DF0A2:	RORB	(R2)	
3661	015220	000241			CLC		
3662	015222	132712	000001		BITB	#1, (R2)	
3663	015226	001401			BEQ	DF0A3	
3664	015230	000261			SEC		
3665	015232	106012		DF0A3:	RORB	(R2)	
3666	015234	005202			INC	R2	:RESET POINTER
3667	015236	124142		DF0A4:	CMPB	-(R1), -(R2)	:TEST DATA CHARACTER
3668	015240	001010			BNE	DF1	:IF NOT GOOD: BR
3669	015242	105037	000656		CLRB	BBC	:CLEAR BAD RECORD COUNTER
3670	015246	000411			BR	DF2	
3671	015250	122122		DF0:	CMPB	(R1)+, (R2)+	:CHECK DATA
3672	015252	001003			BNE	DF1	:IF BAD: BR
3673	015254	105037	000656		CLRB	BBC	:CLEAR BAD RECORD CNTR
3674	015260	000404			BR	DF2	
3675	015262	004737	016020	DF1:	JSR	PC, DRPKF	:GO GET DROPS AND PICKS
3676	015266	004737	015354		JSR	PC, DERR	:GO DO PRINT
3677	015272	005205		DF2:	INC	R5	:BUMP CHAR CNTR
3678	015274	001404			BEQ	DF3	:IF DONE ALL: BR
3679	015276	005737	000562		TST	RDCMD	:SEE IF READ REVERSE
3680	015302	001762			BEQ	DF0	:IF NOT: BR
3681	015304	000717			BR	DF0A	:ELSE CONTINUE READ REV
3682	015306	005037	000664	DF3:	CLR	HDRFL	:CLEAR HEADER FLAG
3683	015312	005737	000704		TST	DERFL	:SEE IF HAD DATA ERROR
3684	015316	001415			BEQ	DFX	:IF NOT: BR
3685	015320	005737	000706		TST	SERFL	
3686	015324	001012			BNE	DFX	:IF NOT DATA ERROR ONLY: BR
3687	015326	013704	000674		MOV	UNP, R4	
3688	015332	005737	000562		TST	RDCMD	:SEE IF READ REVERSE
3689	015336	001003			BNE	DF4	:IF SO: BR
3690	015340	005264	001124		INC	DATER1(R4)	:BUMP DATA ERROR FORWARD COUNTER
3691	015344	000402			BR	DFX	
3692	015346	005264	001164	DF4:	INC	DEREV1(R4)	:BUMP REVERSE DATA ERROR
3693	015352	000207		DFX:	RTS	PC	:EXIT
3694							

3695
3696
3697
3698
3699
3700
3701
3702
3703
3704
3705
3706
3707
3708
3709
3710
3711
3712
3713
3714
3715
3716
3717
3718
3719
3720
3721
3722
3723
3724
3725
3726
3727
3728
3729
3730
3731
3732
3733
3734
3735
3736
3737
3738
3739
3740
3741
3742
3743
3744
3745
3746
3747
3748
3749
3750

015354 032777 002000 163226
015362 001057
015364 005237 000670
015370 005737 000664
015374 001006
015376 004737 022012
015402 000004 023520
015406 004737 020250
015412 000004 023537
015416 010203
015420 162703 032350
015424 005303
015426 005737 000562
015432 001402
015434 010503
015436 005103
015440 104400
015442 000004 023525
015446 005737 000562
015452 001402
015454 111103
015456 000401
015460 114103
015462 004737 023302
015466 000004 023532
015472 005737 000562
015476 001402
015500 111203

```
*****  
:DATA ERROR SUBROUTINE:  
:  
:THIS SUBROUTINE IS USED TO PRINT OUT ANY  
:ERRORS FOUND DURING THE DATA CHECK.  
:EACH CHARACTER FOUND BAD WILL BE PRINTED  
:IN BIT FORMAT ALONG WITH ITS EXPECTED CHARACTER.  
:AN ERROR HEADER CONSISTING OF THE UNIT NUMBER,  
:BLOCK NUMBER, RECORD NUMBER, SIZE OF RECORD, AND  
:ERROR TYPE (READ FORWARD, READ REVERSE, WRITE, ETC)  
:IS PRINTED ONLY ONCE FOR EACH RECORD FOUND BAD.  
:A COUNT IS MADE OF THE NUMBER OF SUCCESSIVE BAD  
:CHARACTERS, AND IF TEN (10) SUCCESSIVE BAD CHARACTERS  
:ARE FOUND IN A SINGLE RECORD, A MESSAGE INDICATING  
:A BAD RECORD CONDITION IS PRINTED AND THE NEXT  
:TWENTY (20) CHARACTERS ARE SKIPPED BEFORE CHECKING  
:IS RESUMED. IF THE BAD RECORD CONDITION IS FOUND  
:THREE TIMES IN A RECORD, ALL REMAINING DATA IS  
:SKIPPED EXCEPT THE FINAL TEN (10) CHARACTERS.  
:THIS SKIPPING IS OF COURSE ONLY POSSIBLE IN  
:RECORDS WHICH CONTAIN A SUFFICIENT NUMBER OF CHARACTERS.  
:PRINTING OF ERRORS MAY BE DISALLOWED AT ANY TIME  
:BY SETTING CONSOLE SWITCH TEN (10) TO A ONE.  
:THE OPERATOR MAY CAUSE THE PROGRAM TO HALT ON ANY ERROR  
:BY SETTING CONSOLE SWITCH FIFTEEN (15) TO A ONE.  
:*****  
DERR: BIT #2000,@SWR ;BRANCH IF NO ERROR  
BNE DERR4 ;PRINTOUT DESIRED  
DERR0: INC PFLG ;SET PRINT FLAG  
TST HDRFL ;SEE IF HAVE PRINTED HEADER  
BNE DERR0A ;IF SO: BR  
JSR PC,PAPRT ;PRINT CYCLE NUMBER  
TYPE,MSG1 ;TYPE DATA ERROR TAG '*DE'  
DERR0A: JSR PC,FRPRT ;PRINT F OR R  
TYPE,MSG4 ;TYPE CHAR # TAG 'CN'  
MOV R2,R3  
SUB #RDATA,R3 ;POINT TO CHAR  
DEC R3  
TST RDCMD ;SEE IF READ REVERSE  
BEQ DERR0B ;IF NOT: BR  
MOV R5,R3 ;GET CHAR NUMBER  
COM R3  
DERR0B: TYPOCT ;PRINT CHAR NUMBER  
TYPE,MSG2 ;TYPE GOOD CHAR TAG 'G'  
TST RDCMD ;SEE IF READ REVERSE  
BEQ DERR0C ;IF NOT: BR  
MOVB (R1),R3 ;GET CHAR  
BR DERR0D  
DERR0C: MOVB -(R1),R3 ;LOAD EXPECTED DATA  
DERR0D: JSR PC,DOUT ;GO PRINT CHAR  
TYPE,MSG3 ;TYPE BAD CHARACTER TAG 'B'  
TST RDCMD ;SEE IF READ REVERSE  
BEQ DERR1 ;IF NOT: BR  
MOVB (R2),R3 ;GET CHAR
```


3751	015502	000401				BR	DERR2	
3752	015504	114203				DERR1: MOV	-(R2),R3	
3753	015506	004737	023302			DERR2: JSR	PC,DOUT	:PRINT BAD CHAR
3754	015512	005737	000562			TST	RDCMD	:BRANCH IF READ
3755	015516	001001				BNE	DERR4	:REVERSE
3756	015520	122122				DERR3: CMPB	(R1)+,(R2)+	:RESET POINTERS
3757	015522	105237	000656			DERR4: INCB	BBC	:BUMP BAD RECORD CNTR
3758	015526	122737	000010	000656		CMPB	#10,BBC	:SEE IF BLD BTH
3759	015534	001107				BNE	DEREX	:IF NOT: BR
3760	015536	032777	002000	163044		BIT	#2000,@SWR	:SEE IF PRINT INHIBIT
3761	015544	001002				BNE	1\$:IF SO: BR
3762	015546	000004	023620			TYPE,MSG15		:TYPE 'BAD RECORD'
3763	015552	105037	000656			1\$: CLRB	BBC	:RESET BAD RECORD CNTR
3764	015556	105237	000657			INCB	BBC+1	:BUMP AMOUNT
3765	015562	122737	000003	000657		CMPB	#3,BBC+1	:SEE IF HAD 3 BLD BTHS
3766	015570	101047				BHI	DERR4B	:IF NOT: BR
3767	015572	022705	177767			CMP	#177767,R5	:SEE IF ON LAST EIGHT CHARS
3768	015576	101464				BLOS	DERR6	:IF SO: BR
3769	015600	012705	177767			MOV	#177767,R5	:SET CHAR CNTR TO 8
3770	015604	005737	000562			TST	RDCMD	:SEE IF READ REVERSE
3771	015610	001416				BEQ	DERR4A	:IF NOT: BR
3772	015612	012701	026342			MOV	#WDATA,R1	:GET START OF BUFFER
3773	015616	012702	032350			MOV	#RDATA,R2	:GET START OF BUFFER
3774	015622	062701	000010			ADD	#10,R1	
3775	015626	062702	000010			ADD	#10,R2	:POINT TO START +10
3776	015632	032737	000001	000556		BIT	#1,FMCNT	:SEE IF ODD FRAME COUNT
3777	015640	001445				BEQ	DEREX	:IF NOT: BR
3778	015642	105722				TSTB	(R2)+	:BUMP POINTER
3779	015644	000443				BR	DEREX	
3780	015646	013737	000556	000644		DERR4A: MOV	FMCNT,TEMP1	:LOAD CHAR COUNT
3781	015654	005437	000644			NEG	TEMP1	:++B
3782	015660	162737	000010	000644		SUB	#10,TEMP1	:POINT TO BUFFER -8
3783	015666	013701	000644			MOV	TEMP1,R1	:POINT TO NEXT CHAR
3784	015672	062701	026342			ADD	#WDATA,R1	:POINT TO NEXT WRITE CHAR
3785	015676	013702	000644			MOV	TEMP1,R2	:POINT TO END OF READ DATA -8 FORWARD
3786	015702	062702	032350			ADD	#RDATA,R2	:POINT TO NEXT CHAR
3787	015706	000422				BR	DEREX	:EXIT
3788	015710	062705	000024			DERR4B: ADD	#24,R5	:SKIP 20 CHARS
3789	015714	103415				BCS	DERR6	:IF EXCEED RECORD SIZE: BR
3790	015716	005737	000562			TST	RDCMD	:SEE IF READ REVERSE
3791	015722	001405				BEQ	DERR5	:IF NOT: BR
3792	015724	162701	000024			SUB	#24,R1	
3793	015730	162702	000024			SUB	#24,R2	:RESET POINTERS
3794	015734	000407				BR	DEREX	
3795	015736	062701	000024			DERR5: ADD	#24,R1	:SKIP 20 CHARS
3796	015742	062702	000024			ADD	#24,R2	:SKIP FORWARD 20 CHARS
3797	015746	000402				BR	DEREX	
3798	015750	012705	177777			DERR6: MOV	#-1,R5	:SET TO EOR
3799	015754	005777	162630			DEREX: TST	@SWR	:BRANCH IF NOT HALT ON ERROR
3800	015760	100012				BPL	DEREX1	
3801	015762	000000				HALT		
3802	015764	005737	000670			TST	PFLG	:SEE IF PRINTED
3803	015770	001006				BNE	DEREX1	:IF SO: BR
3804	015772	032777	002000	162610		BIT	#2000,@SWR	:SEE IF SHOULD PRINT
3805	016000	001002				BNE	DEREX1	:IF NOT: BR
3806	016002	000137	015364			JMP	DERRO	:ELSE PRINT

CZTEDBO TM03-TE16/TU77 DRT
CZTEDB.P11 15-NOV-78 13:19

MACY11 30A(1052) 21-DEC-78 13:17 ^{F 7} PAGE 83

SEQ 0083

3807 016006 005037 000670
3808 016012 005237 000704
3809 016016 000207
3810

DEREX1: CLR
INC
RTS

PFLG
DERFL
PC

:CLEAR FLAG
:BUMP DATA ERROR FLAG
:RETURN

3811
3812
3813
3814
3815
3816
3817
3818
3819
3820
3821
3822
3823
3824
3825
3826
3827
3828
3829
3830
3831
3832
3833
3834
3835
3836
3837
3838
3839
3840
3841
3842
3843
3844
3845
3846
3847
3848
3849
3850
3851
3852
3853
3854
3855
3856
3857
3858
3859
3860
3861
3862
3863
3864
3865
3866

```
*****  
:DROPS AND PICKS SUBROUTINE:  
:  
:THIS SUBROUTINE IS USED TO ACCUMULATE FROM  
:EACH BAD DATA CHARACTER FOUND THE NUMBER  
:OF BITS WHICH WERE EITHER DROPPED OR PICKED UP.  
:TWO COUNTERS PER SLAVE ARE USED TO ACCUMULATE THIS  
:INFORMATION AND CAN STORE UP TO 32K DROPS  
:OR PICKS BEFORE OVERFLOWING. IF OVERFLOW IS  
:ABOUT TO OCCUR, THESE ACCUMULATORS ARE  
:PRINTED IN OCTAL AND RESET TO ZERO.  
:THE CONTENTS OF THE ACCUMULATORS MAY BE  
:DISPLAYED AT ANY TIME BY SETTING CONSOLE  
:SWITCH FOURTEEN TO A ONE (1). THE PRINTOUT WILL OCCUR  
:AT THE END OF THE CURRENT BLOCK CYCLE.  
*****
```

```
DRPKF: CLR TEMP1  
CLR TEMP2  
CLR TEMP3  
MOVB (R1),TEMP1 ;LOAD GOOD CHAR  
MOVB (R2),TEMP2 ;LOAD BAD CHAR  
MOV UNP,R4  
MOV PIK1(R4),BPKP  
MOV DRP1(R4),BDPP  
TST RDCMD ;SEE IF READ REVERSE  
BNE DRPK ;IF SO: BR  
CMPB -(R1),-(R2) ;POINT TO CHAR  
MOVB (R1)+,TEMP1 ;LOAD GOOD CHAR  
MOVB (R2)+,TEMP2 ;LOAD BAD CHAR  
DRPK: JSR PC,DROP ;GET DROPS  
JSR PC,PICK ;GET PICKS  
RTS PC ;EXIT  
  
DROP: MOVB TEMP1,R3 ;R3 = GOOD CHAR  
MOVB TEMP2,R4 ;R4 = BAD CHAR  
DPC: BICB R4,R3 ;GET DROPS/PICKS  
BNE DPCG ;IF SOME: BR  
RTS PC ;RETURN  
DPCG: MOV #10,BCNT ;SET NUMBER TO CHECK  
DPC0: BITB #1,R3 ;SEE IF DROPPED OR PICKED THIS BIT  
BEQ DPC2 ;IF NOT: BR  
TSTB TEMP3 ;SEE IF ON PICKS  
BNE DPC1 ;IF SO: BR  
INC @BDPP ;BUMP DROP CNTR  
BPL DPC2 ;IF NO OVERFLOW: BR  
BIT #2000,@SWR ;SEE IF HAVE PRINTED DATA  
BEQ DPC0A ;IF SO: BR  
JSR PC,PAPRT ;PRINT CYCLE NUMBER  
DPC0A: JSR PC,DPPRT ;PRINT DROPS AND PICKS  
BR DPC2A  
DPC1: INC @BPKP ;BUMP PICK CNTR  
BPL DPC2 ;& BR IF NO OVERFLOW  
BIT #2000,@SWR ;SEE IF HAVE PRINTED DATA  
BEQ DPC1A ;IF SO: BR
```


3867	016224	004737	022012			JSR	PC,PAPRT	:PRINT CYCLE NUMBER	
3868	016230	004737	016370			DPC1A: JSR	PC,DPPRT	:PRINT DROPS AND PICKS	
3869	016234	013704	000674			DPC2A: MOV	UNP,R4		
3870	016240	016403	001004				MOV	DRP1(R4),R3	:SET DROP POINTER
3871	016244	016404	000764				MOV	PIK1(R4),R4	:SET PICK POINTER
3872	016250	012737	000010	000710			MOV	#10,BCNT	:SET NUMBER OF BITS
3873	016256	005023				DPC2B: CLR	(R3)+	:CLEAR DROPS	
3874	016260	005024					CLR	(R4)+	:CLEAR PICK
3875	016262	005337	000710				DEC	BCNT	:SEE IF DONE
3876	016266	001373					BNE	DPC2B	:IF NOT: BR
3877	016270	000207					RTS	PC	:EXIT
3878	016272	000241				DPC2: CLC			
3879	016274	106003					RORB	R3	:GET NEXT BIT
3880	016276	005337	000710				DEC	BCNT	:SEE IF DONE
3881	016302	001407					BEQ	DPC3	
3882	016304	062737	000002	000720			ADD	#2,BPKP	
3883	016312	062737	000002	000716			ADD	#2,BDPP	
3884	016320	000710					BR	DPC0	:CONTINUE
3885	016322	000207				DPC3: RTS	PC	:RETURN	
3886	016324	013704	000674			PICK: MOV	UNP,R4	:GET UNIT POINTER	
3887	016330	016437	000764	000720			MOV	PIK1(R4),BPKP	:SET PICK POINTER
3888	016336	016437	001004	000716			MOV	DRP1(R4),BDPP	:SET DROP POINTER
3889	016344	113704	000644				MOVB	TEMP1,R4	:R4 = GOOD CHAR
3890	016350	113703	000646				MOVB	TEMP2,R3	:R3 = BAD CHAR
3891	016354	112737	000001	000650			MOVB	#1,TEMP3	:SET PICK FLAG
3892	016362	004737	016126				JSR	PC,DPC	:GO CHECK PICKS
3893	016366	000207					RTS	PC	:EXIT
3894	016370	000004	024131			DPPRT: TYPE	MSG26	:TYPE 'DROPS'	
3895	016374	013704	000674				MOV	UNP,R4	
3896	016400	016437	001004	000716			MOV	DRP1(R4),BDPP	:SET DROP POINTER
3897	016406	016437	000764	000720			MOV	PIK1(R4),BPKP	:SET PICK POINTER
3898	016414	062737	000016	000716			ADD	#16,BDPP	
3899	016422	062737	000016	000720			ADD	#16,BPKP	
3900	016430	012737	000010	000710			MOV	#10,BCNT	:SET NUMBER TO PRINT
3901	016436	017703	162254			DPPRT0: MOV	@BDPP,R3		
3902	016442	104400					TYPOCT		:PRINT DROPS
3903	016444	005337	000710				DEC	BCNT	:SEE IF DONE
3904	016450	001404					BEQ	DPPRT1	:IF NOT: BR
3905	016452	162737	000002	000716			SUB	#2,BDPP	:BUMP POINTER
3906	016460	000766					BR	DPPRT0	:CONTINUE FOR ALL 8 BITS
3907	016462	012737	000010	000710		DPPRT1: MOV	#10,BCNT	:SET NUMBER TO PRINT	
3908	016470	000004	024142				TYPE	MSG27	:TYPE 'PICKS'
3909	016474	017703	162220			DPPRT2: MOV	@BPKP,R3		
3910	016500	104400					TYPOCT		:PRINT PICKS
3911	016502	005337	000710				DEC	BCNT	:SEE IF DONE
3912	016506	001404					BEQ	DPPRTX	:IF SO: BR
3913	016510	162737	000002	000720			SUB	#2,BPKP	:BUMP POINTER
3914	016516	000766					BR	DPPRT2	:CONTINUE FOR ALL 8 BITS
3915	016520	000207				DPPRTX: RTS	PC	:RETURN	

3916
 3917
 3918
 3919
 3920
 3921
 3922
 3923
 3924
 3925
 3926
 3927
 3928
 3929
 3930
 3931
 3932
 3933
 3934
 3935
 3936
 3937
 3938
 3939
 3940

```

:*****
:STATUS CHECK SUBROUTINE:
:
:THIS SUBROUTINE IS USED TO PERFORM A CHECK OF
:BOTHS THE MASSBUS CONTROLLER (RH11) AND THE TAPE
:CONTROLLER (TM02). THE RH11 IS CHECKED FOR ERRORS
:AS REFLECTED IN REGISTERS CS1 AND CS2 AND ALSO THAT
:THE BUS ADDRESS (BA) AND WORD COUNT (WC) ARE
:CORRECT. THE TM02 IS CHECKED FOR DRIVE STATIS (DS),
:DRIVE ERRORS (ER), AND PROPER FRAME COUNT. THE SPECIAL
:CHECK CHARACTERS (CRC+LRC) ARE ALSO CHECKED WHEN
:APPROPRIATE (IE: NRZ READ OR WRITE). CERTAIN TYPES
:OF DRIVE ERRORS IN PE OPERATION WILL BE ACCOMPANIED
:BY THE DISPLAY OF THE DEAD TRACK REGISTER (CC). THESE
:TYPES ARE ER BITS 15,10,7,6. THE PRINTOUTS OF BAD
:CRC,LRC,FC, AND BA WILL SHOW BOTH THE EXPECTED AND
:RECEIVED VALUES (IE: EXPT-RCVD). ONLY THOSE REGISTERS
:WHICH ARE IN ERROR WILL BE PRINTED AND ALL PRINTOUTS
:ARE IN OCTAL FORMAT WITH NO LEADING ZEROS. AS IN
:DATA ERRORS, STATUS ERRORS ARE PRECEDED BY HEADER
:DESCRIBING THE HARDWARE UNDER TEST, THE BLOCKING
:INFORMATION, AND THE ERROR TYPE.
:*****
  
```

3941	016522	013703	000556		ERCHK:	MOV	FMCNT,R3		:GET FRAME COUNT
3942	016526	032703	000001			BIT	#1,R3		:SEE IF ODD
3943	016532	001401				BEQ	1\$:IF NOT: BR
3944	016534	005303				DEC	R3		:BUMP COUNT
3945	016536	005403			1\$:	NEG	R3		
3946	016540	032737	000020	000552		BIT	#20,UDES		:SEE IF CORE DUMP
3947	016546	001401				BEQ	2\$:IF NOT: BR
3948	016550	006203				ASR	R3		:SET TO FC/2
3949	016552	032737	000010	000672	2\$:	BIT	#10,MTC1		:SEE IF WRITE OP
3950	016560	001413				BEQ	4\$:IF SO: BR
3951	016562	005737	000562			TST	RDCMD		
3952	016566	001405				BEQ	3\$		
3953	016570	012703	032350			MOV	#RDATA,R3		
3954	016574	162703	000002			SUB	#2,R3		:SET POINTER
3955	016600	000405				BR	ER2		
3956	016602	062703	032350		3\$:	ADD	#RDATA,R3		:BUILD EXPT READ ADDRESS
3957	016606	000402				BR	ER2		
3958	016610	062703	026342		4\$:	ADD	#WDATA,R3		:BUILD EXPT WRITE ADDRESS
3959									
3960	016614	032777	040000	161702	ER2:	BIT	#40000,@ER		:BRANCH IF NOT UNSAFE
3961	016622	001403				BEQ	1\$		
3962	016624	005726				TST	(SP)+		:ADJUST STACK
3963	016626	000137	020312			JMP	OFFLINE		:GO MARK UNIT OFFLINE
3964	016632	010337	020224		1\$:	MOV	R3,CADER		:SAVE ADDRESS
3965	016636	012704	000007			MOV	#7,R4		
3966	016642	012701	020226			MOV	#BAER,R1		
3967	016646	005021			2\$:	CLR	(R1)+		:CLEAR FLAGS
3968	016650	005304				DEC	R4		
3969	016652	001375				BNE	2\$		
3970	016654	020377	161634			CMP	R3,@BA		:SEE IF ADDRESS OK
3971	016660	001402				BEQ	3\$:IF SO: BR

3972	016662	005237	020226			INC	BAER	:SET BUS ADDRESS ERROR
3973	016666	032737	000010	000672	3\$:	BIT	#10,MTC1	:SEE IF WRITE OPER
3974	016674	001006				BNE	5\$:IF NOT: BR
3975	016676	005777	161614		4\$:	TST	@FC	:SEE IF FC=0
3976	016702	001440				BEQ	ER3	:IF SO: BR
3977	016704	005237	020234			INC	FCER	:SET FC ERROR
3978	016710	000435				BR	ER3	
3979	016712	032737	000040	000672	5\$:	BIT	#40,MTC1	:SEE IF SPACE OPER
3980	016720	001766				BEQ	4\$:IF SO: BR
3981	016722	005737	000676			TST	TMFLG	:SEE IF TM TIME
3982	016726	001011				BNE	7\$:IF SO: BR
3983	016730	013703	000556			MOV	FMCNT,R3	
3984	016734	005403				NEG	R3	:R3 = EXPT RECORD SIZE
3985	016736	020377	161554		6\$:	CMP	R3,@FC	:SEE IF FC = EXPT
3986	016742	001420				BEQ	ER3	:IF SO: BR
3987	016744	005237	020234			INC	FCER	:SET FC ERROR FLAG
3988	016750	000415				BR	ER3	
3989	016752	032737	002000	000552	7\$:	BIT	#2000,UDES	:SEE IF PE
3990	016760	001346				BNE	4\$:IF SO: BR
3991	016762	005737	000562			TST	RDCMD	:SEE IF READ REVERSE
3992	016766	001003				BNE	8\$:IF SO: BR
3993	016770	012703	000002			MOV	#2,R3	
3994	016774	000760				BR	6\$:LOOK FOR EXPT = 2
3995	016776	012703	000001		8\$:	MOV	#1,R3	
3996	017002	000755				BR	6\$:GO CHECK FC FOR TM
3997								
3998	017004	032777	160000	161476	ER3:	BIT	#160000,@C1	:SEE IF COUNT ERROR
3999	017012	001437				BEQ	ER4	
4000	017014	017703	161500			MOV	@CS,R3	:GET CONT STATUS REG
4001	017020	042703	000307			BIC	#307,R3	:MASK OUT IR,OR,UNIT NO. & SEE IF OTHER ERRORS
4002	017024	001406				BEQ	1\$:IF NOT: BR
4003	017026	005737	000676			TST	TMFLG	:SEE IF TAPE MARK TIME
4004	017032	001425				BEQ	3\$:IF NOT: BR
4005	017034	042703	001000			BIC	#1000,R3	:MASK MISSED TRANS & BR IF OTHER ERRORS
4006	017040	001022				BNE	3\$	
4007	017042	032777	060000	161440	1\$:	BIT	#60000,@C1	:SEE IF EITHER TRE OR MCPE
4008	017050	001420				BEQ	ER4	:IF NOT: BR
4009	017052	005737	000676			TST	TMFLG	:SEE IF TM TIME
4010	017056	001413				BEQ	3\$:IF NOT: BR
4011	017060	017703	161440			MOV	@ER,R3	:GET ERROR REGISTER
4012	017064	032737	000010	000552		BIT	#10,UDES	:SEE IF EVEN PARITY
4013	017072	001402				BEQ	2\$:IF NOT: BR
4014	017074	042703	000100			BIC	#100,R3	:MASK PAR
4015	017100	042703	001000		2\$:	BIC	#1000,R3	:MASK FCE
4016	017104	001402				BEQ	ER4	:IF NO ERRORS EXCEPT FCE: BR
4017	017106	005237	020230		3\$:	INC	CONER	:SET CONT ERROR FLAG
4018								
4019	017112	032777	040000	161402	ER4:	BIT	#40000,@DS	:SEE IF DRIVE ERROR
4020	017120	001420				BEQ	ER6	:IF NOT: BR
4021	017122	005737	000676			TST	TMFLG	:SEE IF TAPE MARK TIME
4022	017126	001413				BEQ	2\$:IF NOT: BR
4023	017130	017703	161370			MOV	@ER,R3	:GET ER
4024	017134	032737	000010	000552		BIT	#10,UDES	:SEE IF EVEN PARITY
4025	017142	001402				BEQ	1\$:IF NOT: BR
4026	017144	042703	000100			BIC	#100,R3	:MASK PAR
4027	017150	042703	001000		1\$:	BIC	#1000,R3	:MASK OUT FCE & BRANCH IF

4028	017154	001402				BEQ	ER6		:NO OTHER ERRORS
4029	017156	005237	G20232		2\$:	INC	DRVER		:SET DRIVER ERROR FLAG
4030									
4031	017162	013737	014744	020246	ER6:	MOV	EXCRC,CRCSV		:SAVE EXPECTED CRC
4032	017170	013737	014746	020244		MOV	EXLRC,LRCV		:AND EXPECTED LRC
4033	017176	032737	002000	000552		BIT	#2000,UDES		
4034	017204	001062				BNE	ERPT		:IF IN PE MODE: BR
4035	017206	032777	020000	161374		BIT	#20000,@SWR		:SEE IF NO DATA CHECK
4036	017214	001056				BNE	ERPT		:IF NOT: BR (ALLOW READ OF UNKNOWN TAPES)
4037	017216	032737	000040	000672		BIT	#40,MTC1		:SEE IF WRITE OR READ OP
4038	017224	001452				BEQ	ERPT		:IF NOT: BR
4039	017226	005737	000676			TST	TMFLG		:SEE IF TAPE MARK TIME
4040	017232	001405				BEQ	1\$:IF NOT: BR
4041	017234	005037	014744			CLR	EXCRC		
4042	017240	012737	000023	014746		MOV	#23,EXLRC		:SET CRC/LRC FOR TM
4043	017246	032737	000060	000552	1\$:	BIT	#60,UDES		:SEE IF FORMAT 14
4044	017254	001036				BNE	ERPT		:IF NOT: BR
4045	017256	017703	161246			MOV	@CC,R3		:GET CRC CHARACTER
4046	017262	042703	177000			BIC	#177000,R3		
4047	017266	023703	014744			CMP	EXCRC,R3		
4048	017272	001402				BEQ	2\$:IF CRC GOOD: BR
4049	017274	005237	020240			INC	CRCER		:SET ERROR FLAG
4050	017300	017703	161230		2\$:	MOV	@MR,R3		:GET LRC
4051	017304	000303				SWAB	R3		
4052	017306	005703				TST	R3		
4053	017310	100002				BPL	3\$		
4054	017312	052703	000400			BIS	#400,R3		
4055	017316	042703	177000		3\$:	BIC	#177000,R3		
4056	017322	023703	014746			CMP	EXLRC,R3		
4057	017326	001411				BEQ	ERPT		:IF LRC GOOD: BR
4058	017330	010337	020242			MOV	R3,ACTLRC		:SAVE ACTUAL LRC
4059	017334	005237	020236			INC	LRCER		:SET LRC ERROR FLAG
4060	017340	005737	000562			TST	RDCMD		:SEE IF READ REVERSE
4061	017344	001402				BEQ	ERPT		:IF NOT: BR
4062	017346	005037	020236			CLR	LRCER		:ELSE CLEAR LRC ERROR
4063	017352	012703	000006		ERPT:	MOV	#6,R3		
4064	017356	005037	000706			CLR	SERFL		:CLEAR ERROR FLAG
4065	017362	005037	000722			CLR	ERSAV		
4066	017366	012704	020226			MOV	#BAER,R4		
4067	017372	005724			ERPTT:	TST	(R4)+		:SEE IF ANY ERROR
4068	017374	001004				BNE	ERPTG		:IF SO: BR
4069	017376	005303				DEC	R3		
4070	017400	001374				BNE	ERPTT		
4071	017402	000137	020170			JMP	ERPX1		
4072	017406	005237	000706		ERPTG:	INC	SERFL		:SET ERROR FLAG
4073	017412	017737	161106	000722		MOV	@ER,ERSAV		:SAVE ERROR REGISTER
4074	017420	032777	002000	161162		BIT	#2000,@SWR		:SEE IF PRINT
4075	017426	001420				BEQ	ERPT0		:IF SO: BR
4076	017430	022737	000002	000712		CMP	#2,RTYFL		:SEE IF READ RETRY
4077	017436	001006				BNE	ERPTG1		:IF NOT: BR
4078	017440	013703	000702			MOV	RTCNT,R3		
4079	017444	005203				INC	R3		:BUMP RETRY COUNT
4080	017446	020337	000604			CMP	R3,RETRY		:SEE IF LAST RETRY
4081	017452	001406				BEQ	ERPT0		:IF SO: BR
4082	017454	022737	000002	020232	ERPTG1:	CMP	#2,DRVER		:SEE IF TM STATUS ERROR
4083	017462	001402				BEQ	ERPT0		:IF SO: BR

4084	017464	000137	020050		JMP	ERPX0	
4085	017470	005237	000670		INC	PFLG	
4086	017474	004737	022012		JSR	PC,PAPRT	:PRINT HEADER
4087	017500	013737	000652	017510	MOV	EMADDR,1\$:GET ADDRESS OF ERROR MSG HEADER
4088	017506	000004			TYPE		
4089	017510	000000			1\$: .WORD	0	:ADDRESS OF ERROR MESSAGE HEADER
4090	017512	004737	020250		JSR	PC,FRPRT	:PRINT F OR R
4091	017516	005737	000676		TST	TMFLG	
4092	017522	001406			BEQ	ERPT1	
4093	017524	022737	025010	000652	CMP	#MSG54,EMADDR	
4094	017532	001402			BEQ	ERPT1	
4095	017534	000004	025026		TYPE,MSG56		:TYPE 'TM'
4096	017540	005737	020230		ERPT1: TST	CONER	
4097	017544	001412			BEQ	ERPT2	:IF NO CONT ERROR: BR
4098	017546	000004	023735		TYPE,MSG23		:TYPE 'CS1'
4099	017552	017703	160732		MOV	@C1,R3	
4100	017556	104400			TYPOCT		:PRINT CONTROL 1
4101	017560	000004	023762		TYPE,MSG23D		:TYPE CS TAG
4102	017564	017703	160730		MOV	@CS,R3	
4103	017570	104400			TYPOCT		:PRINT CONT STATUS
4104	017572	005737	020232		ERPT2: TST	DRVER	
4105	017576	001412			BEQ	ERPT3	:IF SO DRIVE ERROR: BR
4106	017600	000004	023770		TYPE,MSG23E		:TYPE DS TAG
4107	017604	017703	160712		MOV	@DS,R3	
4108	017610	104400			TYPOCT		:PRINT DRIVE STATUS
4109	017612	000004	023775		TYPE,MSG23F		:TYPE ER TAG
4110	017616	017703	160702		MOV	@ER,R3	
4111	017622	104400			TYPOCT		:PRINT DRIVE ERROR
4112	017624	005737	020226		ERPT3: TST	BAER	
4113	017630	001412			BEQ	ERPT4	:IF NO BA ERROR: BR
4114	017632	000004	023750		TYPE,MSG23B		:TYPE BA TAG
4115	017636	017703	160652		MOV	@BA,R3	
4116	017642	104400			TYPOCT		:PRINT BUS ADDRESS
4117	017644	000004	023516		TYPE,DASH		
4118	017650	013703	020224		MOV	CADER,R3	
4119	017654	104400			TYPOCT		:PRINT EXPT BUS ADDRESS
4120	017656	005737	020234		ERPT4: TST	FCER	
4121	017662	001405			BEQ	ERPT5	:IF NO FC ERROR: BR
4122	017664	000004	023755		TYPE,MSG23C		:TYPE FC TAG
4123	017670	017703	160622		MOV	@FC,R3	
4124	017674	104400			TYPOCT		:PRINT FRAME COUNT
4125	017676	000004	023743		ERPT5: TYPE,MSG23A		:TYPE WC TAG
4126	017702	017703	160604		MOV	@WC,R3	
4127	017706	104400			TYPOCT		:PRINT WORD COUNT
4128	017710	005737	020240		TST	CRCER	
4129	017714	001414			BEQ	ERPT5A	:IF NO CRC ERROR: BR
4130	017716	000004	025053		TYPE,MSG58		:TYPE CRC TAG
4131	017722	017703	160602		MOV	@CC,R3	
4132	017726	042703	177000		BIC	#177000,R3	
4133	017732	104400			TYPOCT		:PRINT ACTUAL CRC
4134	017734	000004	023516		TYPE,DASH		
4135	017740	013703	014744		MOV	EXCRC,R3	
4136	017744	104400			TYPOCT		:PRINT EXPECTED CRC
4137	017746	005737	020236		ERPT5A: TST	LRCER	
4138	017752	001412			BEQ	ERPT6	:IF NO LRC ERROR: BR
4139	017754	000004	025061		TYPE,MSG59		:TYPE LRC ERR TAG

4140	017760	013703	020242		MOV	ACTLRC,R3		
4141	017764	104400			TYPOCT		:PRINT ACTUAL LRC	
4142	017766	000004	023516		TYPE,DASH			
4143	017772	013703	014746		MOV	EXLRC,R3		
4144	017776	104400			TYPOCT		:PRINT EXPECTED LRC	
4145	020000	005737	020232	ERPT6:	TST	DRVER		
4146	020004	001420			BEQ	ERPT7	:IF NO DRIVE ERROR: BR	
4147	020006	032737	002000	000552	BIT	#2000,UDES		
4148	020014	001414			BEQ	ERPT7	:IF NO PE: BR	
4149	020016	017704	160502		MOV	@ER,R4		
4150	020022	042704	075477		BIC	#75477,R4	:MASK OUT ALL BUT BITS 15,10,7,6	
4151	020026	001407			BEQ	ERPT7	:IF NO CONDITIONALS SET: BR	
4152	020030	000004	024007		TYPE,MSG23H		:TYPE CC TAG	
4153	020034	017703	160470		MOV	@CC,R3		
4154	020040	042703	177000		BIC	#177000,R3	:MASK CC	
4155	020044	104400			TYPOCT		:PRINT CHECK CHARACTERS	
4156	020046	000240		ERPT7:	NOP			
4157	020050	005777	160534	ERPX0:	TST	@SWR	:BRANCH IF NOT HALT ON ERROR	
4158	020054	100012			BPL	ERPX		
4159	020056	000000			HALT			
4160	020060	005737	000670		TST	PFLG	:SEE IF HAVE PRINTED	
4161	020064	001006			BNE	ERPX	:IF SO: BR	
4162	020066	032777	002000	160514	BIT	#2000,@SWR	:SEE IF SHOULD PRINT	
4163	020074	001002			BNE	ERPX	:IF NOT: BR	
4164	020076	000137	017470		JMP	ERPT0	:PRINT ERROR	
4165	020102	005037	000670	ERPX:	CLR	PFLG		
4166	020106	005737	000566		TST	CRCC	:BRANCH IF CRC ERROR	
4167	020112	001007			BNE	1\$:CORRECTION DESIRED	
4168	020114	012777	000040	160376	MOV	#40,@CS	:ELSE INIT	
4169	020122	013777	000550	160370	MOV	DVN,@CS	:RESET DRIVE NUMBER	
4170	020130	000414			BR	2\$		
4171	020132	012777	000011	160350	1\$:	MOV	#11,@C1	:DRIVE CLEAR
4172	020140	017704	160362		MOV	@AS,R4		
4173	020144	010477	160356		MOV	R4,@AS	:CLEAR AS	
4174	020150	013704	000510		MOV	C1,R4		
4175	020154	005204			INC	R4		
4176	020156	152714	000100		BISB	#100,(R4)	:RESET TRE	
4177	020162	013777	000552	160352	2\$:	MOV	UDES,@TC	:RESET TC
4178	020170	032737	000040	000672	ERPX1:	BIT	#40,MTC1	
4179	020176	001411			BEQ	ERPX2	:IF NOT READ/WRITE OP: BR	
4180	020200	005737	000676		TST	TMFLG		
4181	020204	001406			BEQ	ERPX2	:IF NOT TM TIME: BR	
4182	020206	013737	020246	014744	MOV	CRCSV,EXCRC	:RESTORE CRC	
4183	020214	013737	020244	014746	MOV	LRCSV,EXLRC	:RESTORE LRC	
4184	020222	000207		ERPX2:	RTS	PC	:EXIT	
4185	020224	000000		CADER:	C		:EXPT ADDRESS SAVE	
4186	020226	000000		BAER:	0			
4187	020230	000000		CONER:	0			
4188	020232	000000		DRVER:	0			
4189	020234	000000		FCER:	0			
4190	020236	000000		LRCER:	0			
4191	020240	000000		CR CER:	0			
4192	020242	000000		ACTLRC:	0			
4193	020244	000000		LRCSV:	0			
4194	020246	000000		CRCSV:	0			
4195								

4196
4197
4198
4199
4200
4201
4202
4203
4204
4205
4206
4207
4208
4209
4210
4211
4212
4213
4214
4215
4216
4217
4218
4219
4220
4221
4222
4223
4224
4225
4226
4227
4228
4229
4230
4231

```
*****  
:F FOR FORWARD/R FOR REVERSE PRINT SUBROUTINE:  
:THIS SUBROUTINE IS USED TO PRINT OUT THE  
:TAPE DIRECTION USED WHEN ANY ERROR IS  
:DETECTED IN STATUS OF READ OR WRITE, DATA, OR  
:SPACING OPERATIONS.  
*****
```

```
FRPRT: BIT #10,MTC1 ;SEE IF WRITE COMMAND  
BEQ 3$ ;IF SO: BR  
MOV #MSG17,2$ ;PRSET MESSAGE TO READ REVERSE  
BIT #2,MTC1 ;BRANCH IF REVERSE  
BNE 1$  
MOV #MSG16,2$ ;SET FORWARD MESSAGE  
1$: TYPE ;TYPE MSG  
2$: .WORD 0  
3$: RTS PC ;EXIT
```

:ROUTINE TO MARK UNIT OFF LINE

```
OFFLINE:MOV UNP,R1 ;GET UNIT POINTER  
BIS #40000,UN1(R1) ;MARK UNIT OFF LINE  
TYPE,MSG25 ;TYPE 'SLAVE UNSAFE-NO FURTHER TESTING ON SLAVE  
TST ASEQF ;BRANCH IF NOT IN AUTO SEQUENCE  
BEQ 1$  
TYPE,MSG123 ;TYPE 'AUTO-SEQ TEST WILL RESTART  
MOV #500,SP ;RESET STACK PTR  
JMP ASEQ0 ;RESTART AUTO-SEQ  
1$: DECB REOTC+1 ;DECREMENT UNITS TO TEST CTR  
BNE 2$  
TYPE,MSG122 ;TYPE 'NO UNITS LEFT TO TEST: HALT'  
2$: HALT  
JMP REOT
```


4232
4233
4234
4235
4236
4237
4238
4239
4240
4241
4242
4243
4244
4245
4246
4247
4248
4249
4250
4251
4252
4253
4254
4255
4256
4257
4258
4259
4260

```

:*****
:TAPE COMMAND EXECUTE SUBROUTINE:
:
:THIS SUBROUTINE IS USED TO EXECUTE THE
:MAG TAPE COMMAND DESCRIBED BY THE READ
:OR WRITE ROUTINE. THE FINAL COMMAND IS
:SENT TO THE DEVICE REGISTER ALONG WITH THE
:INTERRUPT ENABLE AND GO BITS.
:ONCE THE COMMAND IS ISSUED, AN INTERRUPT
:TIMER IS STARTED AND IF NO INTERRUPT IS RETURNED
:BEFORE TIME OUT OCCURS, AN ERROR WILL BE
:PRINTED AND THE PROGRAM STOPPED. TESTING MAY
:BE RESUMED BY PRESSING THE CONTINUE SWITCH.
:TWO INTERRUPT HANDLERS ARE USED, ONE FOR MAG TAPE
:AND ANOTHER FOR TELETYPE (TTY).
:UPON RECEIPT OF A MAG TAPE INTERRUPT, HOUSEKEEPING
:IS PERFORMED AND CONTROL RETURNED TO THE CALLING
:ROUTINE (READ,WRITE,ETC).
:RECEIPT OF A TTY INTERRUPT WILL CAUSE THE
:PROGRAM TO CHECK FOR ENTRY OF A CNTRL C CHARACTER.
:IF NOT CNTRL C, THEN CONTINUATION OF WAIT FOR MAG
:TAPE INTERRUPT IS RETURNED. IF, HOWEVER, THE TTY
:INTERRUPT WAS CAUSED BY ENTRY OF A CNTRL C,
:THEN AT THIS TIME REQUESTS FOR NEW STALL VALUES
:ARE PRINTED AND THE RESPONSES ENTERED. RESUMPTION
:OF TAPE INTERRUPT WAIT IS THEN RESUMED.
:*****

```

```

4261 020372 005037 000644 TAPG: CLR TEMP1
4262 020376 013777 000550 160114 MOV DVN,@CS ;SET DRIVE NO.
4263 020404 032777 040000 160112 1$: BIT #40000,@ER ;SEE IF UNIT SAFE
4264 020412 001402 BEQ TAPG3 ;IF SO: BR
4265 020414 000137 020312 JMP OFFLINE ;GO MARK UNIT OFF-LINE
4266 020420 032777 020000 160074 TAPG3: BIT #20000,@DS ;SEE IF PIP RESET
4267 020426 001410 BEQ TAPG3F ;IF SO: BR
4268 020430 004737 022012 JSR PC,PAPRT ;PRINT HEADER
4269 020434 000004 026042 TYPE,MSG116 ;TYPE MSG
4270 020440 032777 020000 160054 1$: BIT #20000,@DS
4271 020446 001374 BNE 1$ ;AWAIT PIP RESET
4272 020450 022737 000026 000672 TAPG3F: CMP #26,MTC1 ;SEE IF WRITE TM
4273 020456 001003 BNE TAPG3A ;IF NOT: BR
4274 020460 012704 177777 MOV #-1,R4 ;ELSE SET FC FOR -1
4275 020464 000406 BR TAPG3B
4276 020466 013704 000556 TAPG3A: MOV FMCNT,R4
4277 020472 032704 000001 BIT #1,R4
4278 020476 001401 BEQ TAPG3B
4279 020500 005304 DEC R4
4280 020502 000261 TAPG3B: SEC
4281 020504 006004 ROR R4 ;SET WC = FC/2 FOR NORMAL FORMAT
4282 020506 032737 000020 000552 BIT #20,UNDES ;SEE IF CORE DUMP FORMAT
4283 020514 001402 BEQ TAPG3C ;IF NOT: BR
4284 020516 000261 SEC
4285 020520 006004 ROR R4 ;SET WC = FC/4 FOR CORE DUMP
4286 020522 010477 157764 TAPG3C: MOV R4,@WC ;SET WORD COUNT
4287 020526 012777 000011 157754 MOV #11,@C1 ;DRIVE CLEAR

```


4288	020534	017777	157756	157754	MOV	@FC,@FC	:RESET FC LOADED
4289	020542	005737	000570		TST	INTRF	:SEE IF INTERCHANGE READ
4290	020546	001407			BEQ	TAPG3D	:IF NOT: BR
4291	020550	032737	000040	000672	BIT	#40,MTC1	:SEE IF READ OP
4292	020556	001403			BEQ	TAPG3D	:IF NOT: BR
4293	020560	012777	000003	157746	MOV	#3,@MR	:SET INTERCHANGE READ MAINT. MODE
4294	020566	013704	000672		TAPG3D: MOV	MTC1,R4	:GET COMMAND
4295	020572	042704	177707		BIC	#177707,R4	:MASK OP CODE
4296	020576	022704	000030		CMP	#30,R4	:SEE IF SPACE OP CODE
4297	020602	001403			BEQ	TAPG3E	:IF SO: BR
4298	020604	012737	177740	000666	MOV	#-40,STAL	:SET INTERRUPT DELAY MULT TO 40
4299	020612	052737	000101	000672	TAPG3E: BIS	#101,MTC1	:SET INTERRUPT ENABLE AND GO
4300	020620	000240			NOP		
4301	020622	013777	000672	157660	MOV	MTC1,@C1	:EXECUTE COMMAND
4302	020630	005077	157752		CLR	@PSW	:CLEAR PRIORITY
4303	020634	005037	000644		CLR	TEMP1	
4304	020640	005237	000644		TAPG4: INC	TEMP1	:SEE IF HAVE TIMED OUT
4305	020644	001375			BNE	TAPG4	:IF NOT: BR
4306	020646	005237	000666		INC	STAL	
4307	020652	001372			BNE	TAPG4	:DO TIME DELAY MULTIPLIER
4308	020654	012777	000340	157724	TAPG5: MOV	#340,@PSW	:RESET PRIORITY
4309	020662	032777	002000	157720	BIT	#2000,@SWR	:SEE IF SHCULD PRINT ERRORS
4310	020670	001013			BNE	TAPG6	:IF NOT: BR
4311	020672	004737	022012		JSR	PC,PAPRT	:PRINT CYCLE NUMBER
4312	020676	013737	000652	020706	MOV	EMADDR,1\$	
4313	020704	000004			TYPE		:TYPE MSG
4314	020706	000000			1\$: .WORD	0	
4315	020710	004737	020250		JSR	PC,FRPRT	:PRINT F OR R
4316	020714	000004	024040		TYPE,MSG24		:TYPE 'NO INTERRUPT'
4317	020720	005777	157664		TAPG6: TST	@SWR	:BRANCH IF NOT HALT ON ERROR
4318	020724	100001			BPL	TAPG7	
4319	020726	000000			HALT		
4320	020730	000137	021162		TAPG7: JMP	MTINTA	:RETURN TO CALLING ROUTINE
4321							


```

4322
4323
4324 020734 017746 157654
4325 020740 042716 000200
4326 020744 122716 000003,
4327 020750 001005
4328 020752 000005
4329 020754 005077 157626
4330 020760 000137 000200
4331 020764 122716 000001
4332 020770 001015
4333 020772 022737 000176 000610
4334 021000 001014
4335 021002 012737 177570 000610
4336 021010 004737 022430
4337 021014 000004 026160
4338 021020 004737 022452
4339 021024 022716 000007
4340 021030 001005
4341 021032 012737 000176 000610
4342 021040 004737 022344
4343 021044 022716 000002
4344 021050 001041
4345 021052 004737 022430
4346 021056 005237 013444
4347 021062 004737 013226
4348 021066 032777 000040 157514
4349 021074 001425
4350 021076 000004 024650
4351 021102 013703 000602
4352 021106 104400
4353 021110 012705 000602
4354 021114 012701 000007
4355 021120 012702 177777
4356 021124 012703 002000
4357 021130 004737 022474
4358 021134 004737 022452
4359 021140 005726
4360 021142 012716 010770
4361 021146 000002
4362 021150 004737 022452
4363 021154 005726
4364 021156 000002
4365
4366
4367 021160 000240
4368 021162 042777 000037 157344
4369 021170 013716 000662
4370 021174 000002

;TTY INTERRUPT HANDLER
TTINT: MOV @TKB,-(SP) ;GET CHARACTER
      BIC #200,(SP) ;STRIP PARITY BIT
      CMPB #3,(SP) ;BRANCH IF NOT ^C
      BNE 1$
      RESET ;RESET ALL I/O
      CLR @PSW ;CLEAR PSW
      JMP @#200 ;RESTART PROGRAM
1$: CMPB #1,(SP) ;BRANCH IF NOT ^A
   BNE 2$
   CMP #SWREG,SWR ;BRANCH IF HARDWARE SWR IS INVOKED
   BNE 3$
   MOV #177570,SWR ;INVOKE HARWARE SWR
   JSR PC,SAVE ;SAVE REGISTERS ON THE STACK
   TYPE,MSG121 ;TYPE 'HARDWARE SWR IN USE'
   JSR PC,RESTORE ;RESTORE REGISTERS
2$: CMP #7,(SP) ;BRANCH IF NOT ^G
   BNE 4$
   MOV #SWREG,SWR ;INVOKE SOFTWARE SWR
   JSR PC,GTSWR ;GET SWITCHES
4$: CMP #2,(SP) ;BRANCH IF NOT ^B
   BNE 6$
   JSR PC,SAVE ;SAVE REGISTERS ON THE STACK
   INC SCVFL ;SET FLAG
   JSR PC,TINP3A ;GO CHECK CRC CORRECTION
   BIT #40,@SWR ;BRANCH IF NOT YOZZLING
   BEQ 5$
   TYPE,MSG44 ;REQUEST NEW YOZZLE STALL
   MOV YSTAL,R3
   TYPOCT ;PRINT PRESENT STALL
   MOV #YSTAL,R5 ;SET ADDRESS OF YSTL
   MOV #7,R1 ;SET NUMBER OF CHAR TO INPUT
   MOV #-1,R2 ;SET MAXIMUM LIMIT
   MOV #2000,R3 ;SET MINIMUM LIMIT
   JSR PC,TTR ;GO GET VALUE
   JSR PC,RESTORE ;RESTORE REGISTERS
   TST (SP)+ ;POP CHARACTER OF THE STACK
   MOV #YOZ,(SP) ;RETURN TO 'YOZ'
   RTI ;RETURN TO YOZ
5$: JSR PC,RESTORE ;POP CHARACTER OFF THE STACK
6$: TST (SP)+ ;RETURN
   RTI

;MAG TAPE INTERRUPT HANDLER
MTINT: NOP
MTINTA: BIC #37,@MR ;CLEAR MAINT MODE
        MOV RTRN,(SP) ;SET RETURN TO (RTRN)
        RTI ;RETURN
  
```


4371
4372
4373
4374
4375
4376
4377
4378
4379
4380 021176 000004 025475
4381 021202 012705 000740
4382 021206 012701 000002
4383 021212 012702 000001
4384 021216 012703 000000
4385 021222 004737 022474
4386 021226 005037 000550
4387 021232 004737 021340
4388 021236 000004 025445
4389 021242 000004 024754
4390 021246 013703 000550
4391 021252 104400
4392 021254 000004 026333
4393 021260 000004 024376
4394 021264 012700 000742
4395 021270 012003
4396 021272 100402
4397 021274 104400
4398 021276 000774
4399 021300 004737 021524
4400 021304 004737 021656
4401 021310 022737 000007 000550
4402 021316 001403
4403 021320 005237 000550
4404 021324 000742
4405 021326 005737 000740
4406 021332 001335
4407 021334 000137 004662

```
*****  
:AUTO SEQUENCE  
:THIS ROUTINE ENTERED VIA STARTING ADDRESS 240  
:WILL EXERCISE ALL AVAILABLE SLAVES ON ALL AVAILABLE  
:DRIVES IN BOTH PE AND NRZ ACCORDING TO THE PRESELECTED  
:TEST PLAN. IF NRZ ONLY, PE TESTING WILL NOT BE ATTEMPTED.  
*****  
ASEQ: TYPE,MSG104 ;REQUEST 'AUTO CONT'  
MOV #ASEQCF,R5 ;SET ADDRESS OF ENTRY  
MOV #2,R1 ;SET SIZE OF ENTRY  
MOV #1,R2 ;SET UPPER LIMIT  
MOV #0,R3 ;SET LOWER LIMIT  
JSR PC,TTR ;GO GET INPUT  
ASEQ0: CLR DVN ;SET DRIVE # 0  
ASEQ1: JSR PC,HRDS ;GO SELECT HARDWARE CONFIGURATION  
TYPE,MSG101 ;TYPE '*****...***'  
TYPE,MSG52A ;TYPE 'DRIVE (TM03) = '  
MOV DVN,R3 ;PRINT DRIVE #  
TYPOCT ;TYPE ' SLAVE # = '  
TYPE,SPACE ;POINT TO START OF SLAVE TABLE  
TYPE,MSG32  
1$: MOV #UN1,R0  
MOV (R0)+,R3  
BMI 2$  
TYPOCT ;PRINT SLAVE TABLE  
BR 1$ ;DO ALL  
2$: JSR PC,AMOD1 ;GO DO MODE 1(NRZ)  
JSR PC,AMOD2 ;GO DO MODE 2(PE)  
000550 ASEQ4: CMP #7,DVN ;SEE IF DONE ALL DRIVES  
BEQ ASEQX ;IF SO: BR  
INC DVN ;BUMP DRIVE NUMBER  
BR ASEQ1 ;CONTINUE  
ASEQX: TST ASEQCF ;SEE IF CONTINUOUS AUTO SEQ  
BNE ASEQ0 ;++B CONTINUE TESTING  
JMP TEND
```



```

4408
4409
4410      ;SUBROUTINE TO SELECT AUTO SEQUENCE HARDWARE*****
4411 021340 005037 004730      HRDS: CLR      REOTC      ;CLEAR EOT UNIT CNTR
4412 021344 012777 000040 157146 MOV      #40,@CS    ;INIT
4413 021352 013777 000550 157140 MOV      DVN,@CS    ;SET DRIVE
4414 021360 005777 157124      TST      @C1        ;ACCESS DRIVE
4415 021364 032777 010000 157126 BIT      #10000,@CS ;TEST FOR NON-EXISTANT DRIVE
4416 021372 001403      BEQ      2$          ;IF DRIVE AVAIL: BR
4417 021374 005726      1$: TST      (SP)+      ;RESET STACK POINTER
4418 021376 000137 021310      JMP      ASEQ4      ;GO SEE IF TRIED ALL DRIVES
4419 021402 017700 157130      2$: MOV      @DT,R0    ;++B GET CONTENTS OF DRIVE TYPE REG
4420 021406 042700 002007      BIC      #2007,R0  ;++B CLEAR SPR AND SPEED BITS
4421 021412 022700 140050      CMP      #140050,R0 ;++B BRANCH IF NOT TM03 MAGTAPE DRIVE
4422 021416 001366      BNE      1$
4423 021420 005000      CLR      R0
4424 021422 012701 000742      MOV      #UN1,R1   ;SET START OF SLAVE TABLE
4425 021426 005737 003034      TST      CHNFLG    ;BRANCH IF NOT IN CHAIN MODE
4426 021432 001410      BEQ      3$
4427 021434 122737 000006 000041 CMPB     #6,@#41    ;BRANCH IF NOT LOADED VIA TMDP
4428 021442 001004      BNE      3$
4429 021444 005737 000550      TST      DVN        ;BRANCH IF NOT DRIVE 0
4430 021450 001001      BNE      3$
4431 021452 005200      INC      R0
4432 021454 010077 157062      3$: MOV      R0,@TC    ;DO NOT TEST SLAVE 0
4433 021460 032777 010000 157034 BIT      #10000,@DS ;SELECT SLAVE
4434 021466 001404      BEQ      4$          ;SEE IF SLAVE AVAIL FOR TEST(MOL)
4435 021470 062737 000401 004730 ADD      #401,REOTC ;IF NOT: BR
4436 021476 010021      MOV      R0,(R1)+   ;INCREMENT UNITS TO TEST COUNT
4437 021500 005200      4$: INC      R0      ;LOAD SLAVE # INTO SLAVE TABLE
4438 021502 022700 000010      CMP      #10,R0    ;STEP TO NEXT SLAVE
4439 021506 001362      BNE      3$          ;BRANCH IF ALL SLAVE NOT DONE
4440 021510 005737 004730      5$: TST      REOTC    ;SEE IF FOUND ANY SLAVES
4441 021514 001727      BEQ      1$          ;IF NOT: BR
4442 021516 012711 177777      MOV      #-1,(R1)  ;TERMINATE SLAVE TABLE
4443 021522 000207      RTS      PC        ;RETURN TO SEQ
  
```



```
4444
4445
4446
4447 021524 005037 000654
4448 021530 012701 000742
4449 021534 052721 001700
4450 021540 022711 177777
4451 021544 001373
4452 021546 004737 004744
4453 021552 012737 000006 000736
4454 021560 012737 174000 000556
4455 021566 012737 000100 000554
4456 021574 012737 000001 000560
4457 021602 005037 000564
4458 021606 005037 000570
4459 021612 004737 003352
4460 021616 012737 000010 000560
4461 021624 004737 003352
4462 021630 012737 000014 000560
4463 021636 004737 003352
4464 021642 012737 177777 000560
4465 021650 004737 003352
4466 021654 000207

;SUBROUTINE TO SELECT NRZ AUTO TEST MODE*****
AMOD1: CLR BLCNTR ;ASSURE BLOCK COUNTER IS 0
MOV #UN1,R1 ;GET START OF SLAVE TABLE
1$: BIS #1700,(R1)+ ;SET ALL SLAVE TO NRZ,NORM,ODD
CMP #-1,(R1) ;LOOP UNTIL REACED END OF TABLE
BNE 1$
JSR PC,RWINDA ;GO REWIND ALL AVAIL SLAVES
MOV #6,ABLCNT ;SET NUMBER OF BLOCKS FOR MODE 1
MOV #-4000,FMCNT ;SET FC = 4000
MOV #100,RCNT ;SET REC CNTR = 100
MOV #1,PATRN ;SELECT PATTERN 1
CLR TMEX ;ASSURE NO TMK
CLR INTRF ;ASSURE NORMAL READ
JSR PC,STAUTO ;GO DO AUTO MODE 1
MOV #10,PATRN ;SELECT PATTERN 10
JSR PC,STAUTO ;GO DO PATTERN 10
MOV #14,PATRN ;SELECT PATTERN 14
JSR PC,STAUTO
3$: MOV #-1,PATRN ;SELECT AUTO RANDOM DATA
JSR PC,STAUTO
RTS ;RETURN TO SEQ
```



```
4467  
4468  
4469  
4470 021656 005037 000654  
4471 021662 012701 000742  
4472 021666 042711 001700  
4473 021672 052721 002300  
4474 021676 022711 177777  
4475 021702 001371  
4476 021704 004737 004744  
4477 021710 012737 000006 000736  
4478 021716 012737 174000 000556  
4479 021724 012737 000100 000554  
4480 021732 012737 000010 000560  
4481 021740 004737 003352  
4482 021744 012737 000014 000560  
4483 021752 004737 003352  
4484 021756 012737 000015 000560  
4485 021764 004737 003352  
4486 021770 012737 177777 000736  
4487 021776 012737 177777 000560  
4488 022004 004737 003352  
4489 022010 000207  
4490  
4491
```

;SUBROUTINE TO SELECT PE AUTO TEST MODE*****

```
AMOD2: CLR BLCNTR ;CLEAR BLOCK CNTR  
MOV #UN1,R1 ;SET START OF SLAVE TABLE  
1$: BIC #1700,(R1) ;CLEAR NRZ  
BIS #2300,(R1)+ ;SET TO PE NORM, ODD  
CMP #-1,(R1) ;LOOP UNTIL END OF TABLE  
BNE 1$  
JSR PC,RWINDA ;REWIND ALL SLAVES  
MOV #6,ABLCNT ;SET AUTO BLOCK COUNT  
MOV #-4000,FMCNT ;SET FC = 4000  
MOV #100,RCNT ;SET REC CNTR TO 100  
MOV #10,PATRN ;SELECT PATTERN 10  
JSR PC,STAUTO ;GO DO AUTO SEQ  
MOV #14,PATRN ;SELECT PATTERN 14  
JSR PC,STAUTO  
MOV #15,PATRN ;SELECT PATTERN 15  
JSR PC,STAUTO  
MOV #-1,ABLCNT ;FORCE TO END OF TAPE  
MOV #-1,PATRN ;SELECT AUTO RANDOM DATA  
JSR PC,STAUTO  
3$: RTS ;RETURN TO SEQ
```


4492
4493
4494
4495
4496
4497
4498
4499
4500
4501
4502
4503
4504
4505
4506
4507
4508
4509
4510
4511
4512
4513
4514
4515
4516
4517
4518
4519
4520
4521
4522
4523
4524
4525
4526
4527
4528
4529
4530
4531
4532
4533
4534
4535
4536
4537
4538
4539
4540
4541
4542
4543
4544
4545
4546
4547

022012 000004 024752
022016 013703 000550
022022 104400
022024 000004 024376
022030 013703 000552
022034 042703 177770
022040 104400
022042 000004 023520
022046 013703 000552
022052 000303
022054 042703 177770
022060 104400
022062 000004 025067
022066 005003
022070 032737 000010 000552
022076 001401
022100 005203
022102 104400
022104 000004 025073
022110 013703 000552
022114 006003
022116 006003
022120 006003
022122 006003
022124 042703 177760
022130 104400
022132 000004 023563
022136 005737 000560
022142 100003
022144 000004 023653
022150 000403
022152 013703 000560
022156 104400
022160 000004 023605
022164 013703 000654
022170 104400
022172 000004 023613
022176 010003
022200 032737 000010 000672
022206 001416

```
*****  
:ERROR HEADER PRINT SUBROUTINE:  
:  
:THIS ROUTINE IS USED TO PRINT OUT A HEADER  
:WITH EACH ERROR MESSAGE. THE PRINT IS IN TWO  
:LINES AND CONTAINS THE FOLLOWING INFORMATION.  
:LINE 1: DRIVE NO. SLAVE NO. DENSITY PARITY FORMAT  
:LINE 2: CURRENT BLOCK NUMBER, RECORD NUMBER IN  
:WHICH THE ERROR OCCURED PLUS THE TOTAL NUMBER  
:OF RECORDS IN THIS BLOCK, THE RECORD SIZE (NUMBER  
:OF CHARACTERS), AND THE ERROR TYPE (READ,WRITE, SPACE, ETC)  
:PLUS THE TAPE DIRECTION (FORWARD OR REVERSE).  
:ALL NUMBERS ARE IN OCTAL.  
:*****  
PAPRT: TYPE,MSG52 ;TYPE 'DRIVE # = '  
MOV DVN,R3 ;PRINT DRIVE NUMBER  
TYPOCT ;TYPE 'SLAVE # = '  
TYPE,MSG32 ;TYPE 'SLAVE # = '  
MOV UDES,R3 ;PRINT SLAVE NUMBER  
BIC #177770,R3 ;TYPE DENSITY TAG '*DE'  
TYPOCT ;PRINT DENSITY  
TYPE,MSG1 ;TYPE PARITY TAG '*P'  
MOV UDES,R3 ;SET PARITY INDICATOR = EVEN  
SWAB R3 ;PRINT PARITY BIT STATE  
BIC #177770,R3 ;TYPE FORMAT TAG '*F'  
TYPOCT ;POSITION FORMAT BITS  
TYPE,MSG61 ;PRINT FORMAT  
CLR R3 ;TYPE PATTERN # TAG '*PATRN'  
BIT #10,UDES ;BRANCH IF NOT RANDOM PATTERN  
BEQ PAPRT0 ;TYPE 'R' FOR RANDOM  
INC R3 ;PRINT PATRN NUMBER  
PAPRT0: TYPOCT ;TYPE SLOCK # TAG '*BN'  
TYPE,MSG62 ;PRINT NUMBER  
MOV UDES,R3 ;TYPE RECORD # TAG '*RN'  
ROR R3 ;GET # OF RECORDS LEFT TO PROCESS  
ROR R3 ;SEE IF WRITE OPERATION  
ROR R3 ;IF SO: BR  
ROR R3  
BIC #177760,R3  
TYPOCT  
TYPE,MSG8  
TST PATRN  
BPL PAPRTC  
PAPRTA: TYPE,MSG17  
BR PAPRTD  
PAPRTC: MOV PATRN,R3  
TYPOCT  
PAPRTD: TYPE,MSG13  
MOV BLCNTR,R3  
TYPOCT  
TYPE,MSG14  
MOV R0,R3  
BIT #1,MTCT  
BEQ PAPRT1
```


4569
 4570
 4571
 4572
 4573
 4574
 4575
 4576
 4577
 4578
 4579
 4580
 4581
 4582
 4583
 4584
 4585
 4586
 4587
 4588
 4589
 4590
 4591
 4592
 4593
 4594
 4595
 4596
 4597
 4598
 4599
 4600
 4601
 4602
 4603
 4604
 4605
 4606
 4607
 4608
 4609
 4610
 4611
 4612
 4613
 4614
 4615
 4616
 4617
 4618
 4619
 4620
 4621
 4622

022312 063737 000630
 022320 063737 000626
 022326 023701 000630
 022332 101367
 022334 020237 000630
 022340 101364
 022342 000207

 022344 022737 000176
 022352 001025 000610
 022354 004737 022430
 022360 000004 023476
 022364 017703 156220
 022370 104400
 022372 000004 023506
 022376 013705 000610
 022402 012701 000007
 022406 012702 177777
 022412 012703 000000
 022416 004737 022474
 022422 004737 022452
 022426 000207

 022430 010546
 022432 010446
 022434 010346
 022436 010246
 022440 010146
 022442 010046
 022444 016646 000014
 022450 000207

 022452 012666 000014
 022456 012600
 022460 012601
 022462 012602
 022464 012603
 022466 012604
 022470 012605
 022472 000207

```

:*****
:RANDOM NUMBER GENERATOR SUBROUTINE:
:
:THIS SUBROUTINE IS USED TO GENERATE THE RANDOM
:NUMBERS REQUIRED FOR USE AS RANDOM DATA,
:RECORD COUNT, AND CHARACTER COUNT.
:*****
RANG:  ADD    RANSAV,RANBAS
      ADD    RANBAS,RANSAV    ;GET NEW NUMBER
      CMP    RANSAV,R1       ;SEE IF NUMBER TOO BIG
      BHI    RANG            ;IF SO: BR
      CMP    R2,RANSAV       ;SEE IF NUMBER TOO SMALL
      BHI    RANG            ;IF SO: BR
      RTS    PC              ;EXIT

;SUBROUTINE TO GET NEW SOFTWARE SWR
GTSWR: CMP    #SWREG,SWR     ;BRANCH IF SOFTWARE SWR
      BNE    1$             ;NOT INVOKED
      JSR    PC,,SAVE       ;SAVE REGISTERS ON THE STACK
      TYPE ,SMSWR
      MOV    @SWR,R3        ;GET CURRENT SWR
      TYPOCT
      TYPE ,SMNEW
      MOV    SWR,R5         ;REQUEST NEW SWR SETTING
      MOV    #7,R1          ;TTR ROUTINE RETURNS VALUE TO (R5)
      MOV    #177777,R2     ;LIMIT RESPONSE TO 7 CHARS
      MOV    #0,R3          ;BETWEEN 0 AND 177777
      JSR    PC,TTR         ;GET RESPONSE
      JSR    PC,,RESTORE    ;RESTORE REGISTERS
      RTS    PC             ;RETURN
1$:

;;ROUTINE TO SAVE REGISTERS ON THE STACK
.SAVE: MOV    %5,-(SP)      ;;R5 IS SAVED AT 12(SP)
      MOV    %4,-(SP)      ;;R4 IS SAVED AT 10(SP)
      MOV    %3,-(SP)      ;;R3 IS SAVED AT 6(SP)
      MOV    %2,-(SP)      ;;R2 IS SAVED AT 4(SP)
      MOV    %1,-(SP)      ;;R1 IS SAVED AT 2(SP)
      MOV    %0,-(SP)      ;;R0 IS SAVED AT (SP)
      MOV    14(SP),-(SP)  ;;PUSH RETURN PC ON THE STACK
      RTS    PC            ;;RETURN TO CALLER

;;ROUTINE TO RESTORE REGISTERS SAVED ON THE STACK
.RESTORE:MOV (SP)+,14(SP)  ;;STORE RETURN PC ON STACK
      MOV    (SP)+,%0
      MOV    (SP)+,%1
      MOV    (SP)+,%2
      MOV    (SP)+,%3
      MOV    (SP)+,%4
      MOV    (SP)+,%5
      RTS    PC            ;;RETURN
  
```



```

4623
4624
4625
4626
4627
4628
4629
4630
4631
4632
4633
4634
4635
4636
4637
4638
4639
4640 022474 010146
4641 022476 011601
4642 022500 005037 000644
4643 022504 005000
4644 022506 004737 022720
4645 022512 122737 000003 000642
4646 022520 001003
4647 022522 000005
4648 022524 000137 000200
4649 022530 122737 000015 000642
4650 022536 001004
4651 022540 005737 000644
4652 022544 001455
4653 022546 000447
4654 022550 122737 000025 000642
4655 022556 001003
4656 022560 000004 024153
4657 022564 000744
4658 022566 122737 000177 000642
4659 022574 001010
4660 022576 000241
4661 022600 006000
4662 022602 006200
4663 022604 006200
4664 022606 000004 026110
4665 022612 005201
4666 022614 000734
4667 022616 122737 000060 000642
4668 022624 101027
4669 022626 122737 000070 000642
4670 022634 101423
4671 022636 005237 000644
4672 022642 006300
4673 022644 006300
4674 022646 006300
4675 022650 042737 177770 000642
4676 022656 053700 000642
4677 022662 005301
4678 022664 001310

```

```

:*****
:TTY ENTRY SUBROUTINE:
:
:THIS SUBROUTINE IS USED BY THE TEST CONDITION
:ENTRY ROUTINE TO READ THE RESPONSE ENTERED
:AT THE TTY AND CHECK THEM FOR LEGALITY AND
:LIMITS. ALL RESPONSE MUST BE TYPED IN OCTAL
:(0-7) AND MUST FALL WITHIN THE LIMITS SET BY
:THE CALLING ROUTINE.
:IF AN ENTRY IS ILLEGAL OR OUTSIDE THE LIMITS,
:A QUESTION MARK IS TYPED (?) AND THE RESPONSE
:MAY BE REENTERED.
:ENTRIES MAY NOT EXCEED SIX (6) CHARACTERS AND
:MAY BE TERMINATED AT LESS THAN SIX BY TYPING A
:CARRIAGE RETURN
:*****

```

```

TTR: MOV R1, -(SP) ;SAVE CHAR COUNT
10$: MOV (SP), R1 ;RESTORE CHAR COUNT (FOR ^U)
CLR TEMP1 ;CLEAR FIRST CHARACTER FLAG
CLR R0
1$: JSR PC, TTIN ;GO READ CHARACTER
CMPB #3, TIB ;BRANCH IF NOT ^C
BNE 11$
RESET
JMP @#200 ;RESTART AT 200
11$: CMPB #15, TIB ;SEE IF CR
BNE 2$ ;IF NOT: BR
TST TEMP1 ;SEE IF FIRST CHARACTER
BEQ 9$ ;IF SO: BR
BR 6$ ;ELSE GO LOAD VALUE
2$: CMPB #25, TIB ;BRANCH IF NOT CONTROL U
BNE 21$
TYPE, MSG28 ;TYPE <CR><LF>
BR 10$
21$: CMPB #177, TIB ;BRANCH IF NOT 'RUBOUT'
BNE 3$
CLC ;REMOVE LAST CHARACTER
ROR R0
ASR R0
ASR R0
TYPE, MSG118 ;TYPE '\ '
INC R1 ;DEC CHAR RECEIVED COUNT
BR 1$ ;GET NEXT CHARACTER
3$: CMPB #60, TIB ;SEE IF CHAR IS LESS THAN 0
BHI T1NER
4$: CMPB #70, TIB ;SEE IF CHAR IS GREATER THAN 7
BLOS T1NER
5$: INC TEMP1 ;SET FIRST CHARACTER FLAG
ASL R0
ASL R0 ;SHIFT 3 LEFT
ASL R0
BIC #177770, TIB ;STRIP ASCII
BIS TIB, R0 ;LOAD CHARACTER
DEC R1 ;SEE IF DONE
BNE 1$ ;IF NOT: BR

```


4679	022666	020002		6\$:	CMP	R0,R2		:SEE IF EXCEEDED MAXIMUM LIMIT
4680	022670	101005			BHI	TINER		
4681	022672	020300		7\$:	CMP	R3,R0		:SEE IF BELOW MINIMUM LIMIT
4682	022674	101003			BHI	TINER		
4683	022676	010015		8\$:	MOV	R0,(R5)		:LOAD VALUE
4684	022700	005726		9\$:	TST	(SP)+		:POP CHAR COUNT OFF STACK
4685	022702	000207			RTS	PC		:EXIT
4686								
4687	022704	000004	024644	TINER:	TYPE,#MSG43			:TYPE '?'
4688	022710	005726			TST	(SP)+		:POP CHAR COUNT OFF STACK
4689	022712	162716	000020		SUB	#20,(SP)		:RESET SP TO START OF VALUE ROUTINE
4690	022716	000207			RTS	PC		:REDO VALUE ENTRY


```

4691
4692
4693
4694 022720 005277 155666
4695 022724 105777 155662
4696 022730 100375
4697 022732 017737 155656 000642
4698 022740 042737 177600 000642
4699 022746 022737 000015 000642
4700 022754 001003
4701 022756 000004 024153
4702 022762 000402
4703 022764 000004 000642
4704 022770 000207
4705
4706
4707
4708 022772 010446
4709 022774 010346
4710 022776 017604 000004
4711 023002 062766 000002 000004
4712 023010 111437 000640
4713 023014 001431
4714 023016 122724 000045
4715 023022 001403
4716 023024 004737 023106
4717 023030 000767
4718
4719 023032 112737 000015 000640
4720 023040 004737 023106
4721 023044 012703 000006
4722 023050 005037 000640
4723 023054 004737 023106
4724 023060 005303
4725 023062 001372
4726 023064 112737 000012 000640
4727 023072 004737 023106
4728 023076 000744
4729 023100 012603
4730 023102 012604
4731 023104 000002
4732
4733 023106 105777 155504
4734 023112 100375
4735 023114 113777 000640 155476
4736 023122 000207
4737

;TTY READ SUBROUTINE*****
TTIN: INC @TKS
1$: TSTB @TKS
BPL 1$
MOV @TKB,TIB
BIC #177600,TIB ;STRIP PARITY BIT
CMP #15,TIB ;BRANCH IF NOT <CR>
BNE 2$
TYPE,MSG28 ;TYPE '<CR><LF>'
BR 3$
2$: TYPE,TIB ;ECHO CHARACTER
3$: RTS PC

;TTY OUTPUT SUBROUTINE*****
TTOUT: MOV R4,-(SP) ;SAVE R4 ON THE STACK
MOV R3,-(SP)
MOV @4(SP),R4 ;GET ADDRESS OF MESSAGE TO TYPE
ADD #2,4(SP) ;ADJUST RETURN PC
10$: MOVB (R4),TOB ;GET A CHARACTER
BEQ 3$ ;AND BRANCH IF END OF MSG
CMPB #45,(R4)+ ;BRANCH IF CRLF CHARACTER (%)
BEQ 1$
JSR PC,TOG ;ECHO CHARACTER
BR 10$
1$: MOVB #15,TOB
JSR PC,TOG
MOV #6,R3
2$: CLR TOB
JSR PC,TOG
DEC R3
BNE 2$ ;DO FILLERS
MOVB #12,TOB
JSR PC,TOG
BR 10$
3$: MOV (SP)+,R3 ;RESTORE REGISTERS
MOV (SP)+,R4
RTI ;RETURN
TOG: TSTB @TPS
BPL TOG
MOVB TOB,@TPB
RTS PC ;RETURN

```



```

4738                                     :OCTAL OUTPUT SUBROUTINE*****
4739
4740 023124 005037 023300      OCTP: CLR      OFL      :CLEAR FLAG FOR LEADING ZERO
4741 023130 010304             MOV      R3,R4      :SEE IF NUMBER IS ZERO
4742 023132 001003             BNE      1$         :IF NOT ZERO: BR
4743 023134 000004 026335     TYPE,DIGIT0
4744 023140 000434             BR        4$         :SPACE AND EXIT
4745 023142 100004             BPL      3$         :BRANCH IF MSD IS A '0'
4746 023144 012704 000001     MOV      #1,R4
4747 023150 004737 023240     JSR     PC,OCTPG    :PRINT 1
4748 023154 006004             3$:  ROR      R4
4749 023156 006004             ROR      R4
4750 023160 006004             ROR      R4         :POSITION DIGIT
4751 023162 006004             ROR      R4
4752 023164 000304             SWAB     R4
4753 023166 004737 023240     JSR     PC,OCTPG    :PRINT DIGIT 2
4754 023172 006004             ROR      R4
4755 023174 000304             SWAB     R4
4756 023176 004737 023240     JSR     PC,OCTPG    :PRINT DIGIT 3
4757 023202 006104             ROL      R4
4758 023204 006104             ROL      R4
4759 023206 000304             SWAB     R4
4760 023210 004737 023240     JSR     PC,OCTPG    :PRINT DIGIT 4
4761 023214 006004             ROR      R4
4762 023216 006004             ROR      R4
4763 023220 006004             ROR      R4
4764 023222 004737 023240     JSR     PC,OCTPG    :PRINT DIGIT 5
4765 023226 004737 023240     JSR     PC,OCTPG    :PRINT DIGIT 6
4766 023232 000004 026333     4$:  TYPE,SPACE    :TYPE A SPACE
4767 023236 000002             RTI         :EXIT
4768
4769 023240 042704 177770     OCTPG: BIC      #177770,R4
4770 023244 001003             BNE      1$
4771 023246 005737 023300     TST     OFL
4772 023252 001410             BEQ     2$
4773 023254 005237 023300     1$:  INC      OFL
4774 023260 052704 000260     BIS     #260,R4
4775 023264 010437 000640     MOV     R4,TOB
4776 023270 004737 023106     JSR     PC,TOG
4777 023274 010304             2$:  MOV     R3,R4
4778 023276 000207             RTS     PC
4779 023300 000000             OFL:  0         :FIRST CHAR FLAG
4780
4781                                     ;DATA CHARACTER OUTPUT SUBROUTINE*****
4782
4783
4784 023302 012704 000010     DOUT: MOV      #10,R4      :SET NUMBER TO PRINT
4785 023306 110346             MOVB     R3,-(SP)    :GET CHAR TO OUTPUT
4786 023310 106316             1$:  ASLB     (SF)     :BRANCH IF BIT IS A ZERO
4787 023312 103003             BCC     2$
4788 023314 000004 026337     TYPE,DIGIT1
4789 023320 000402             BR      3$
4790 023322 000004 026335     2$:  TYPE,DIGIT0
4791 023326 005304             3$:  DEC     R4
4792 023330 001367             BNE     1$
4793 023332 005726             TST     (SP)+      :POP STACK
  
```



```

4794 023334 000207          RTS      PC
4795
4796 023336 113703 000651    DOUTD:  MOVB   TEMP3+1,R3
4797 023342 004737 023302      JSR     PC,DOUT
4798 023346 013703 000650      MOV     TEMP3,R3
4799 023352 004737 023302      JSR     PC,DOUT
4800 023356 000207          RTS      PC
4801
4802
4803          ;TU16 SERIAL NUMBER PRINT SUBROUTINE*****
4804 023360 017703 155154    SNPT:   MOV     @SN,R3          ;GET CONTENTS OF SERIAL # REG
4805 023364 000004 023573      TYPE,MSG9          ;TYPE SN TAG
4806 023370 010304          MOV     R3,R4
4807 023372 000304          SWAB   R4
4808 023374 006004          ROR    R4
4809 023376 006004          ROR    R4
4810 023400 006004          ROR    R4
4811 023402 006004          ROR    R4
4812 023404 004737 023452    JSR     PC,SNPG      ;PRINT FIRST DIGIT
4813 023410 010304          MOV     R3,R4
4814 023412 000304          SWAB   R4
4815 023414 004737 023452    JSR     PC,SNPG      ;PRINT SECOND DIGIT
4816 023420 010304          MOV     R3,R4
4817 023422 006004          ROR    R4
4818 023424 006004          ROR    R4
4819 023426 006004          ROR    R4
4820 023430 006004          ROR    R4
4821 023432 004737 023452    JSR     PC,SNPG      ;PRINT THIRD DIGIT
4822 023436 010304          MOV     R3,R4
4823 023440 004737 023452    JSR     PC,SNPG      ;PRINT FOURTH DIGIT
4824 023444 000004 024153    TYPE,MSG28         ;TYPE <CR><LF>
4825 023450 000207          RTS      PC          ;EXIT
4826 023452 012737 000260 000640 SNPG:  MOV     #260,TOB     ;SET NUMBER BASE
4827 023460 042704 177760      BIC    #177760,R4   ;MASK NUMBER
4828 023464 050437 000640      BIS    R4,TOB       ;BUILD DIGIT
4829 023470 004737 023106    JSR     PC,TOG      ;GO TYPE
4830 023474 000207          RTS      PC          ;RETURN
4831

```


4832
4833
4834
4835
4836
4837
4838
4839
4840
4841
4842
4843
4844
4845
4846
4847
4848
4849
4850
4851
4852
4853
4854
4855
4856
4857
4858
4859
4860
4861
4862
4863
4864
4865
4866
4867
4868
4869
4870
4871
4872
4873
4874
4875
4876
4877
4878
4879
4880
4881
4882
4883
4884
4885
4886
4887

023476 051445 051127 036440
023504 000040
023506 047040 053505 036440
023514 000040
023516 000055
023520 042052 020105 000
023525 045 035507 000040
023532 041045 020073 000
023537 045 047103 000040
023544 053452 020105 000
023551 052 042522 000040
023556 051052 020123 000
023563 052 040520 051124
023570 020116 000
023573 123 035116 000040
023600 051452 020105 000
023605 045 041052 020116
023612 000
023613 052 047122 000040
023620 020045 020040 020040
023626 020040 020040 041040
023634 042101 051040 041505
023642 051117 022504 000045
023650 043040 000
023653 040 000122
023656 042440 052117 021440
023664 000040
023666 047111 042524 041522
023674 040510 043516 020105
023702 042522 042101 020077
023710 000
023711 045 046111 042514
023716 040507 020114 047502
023724 035124 044040 046101
023732 022524 000
023735 045 051503 020061
023742 000
023743 045 041527 000040
023750 041045 020101 000
023755 045 041506 000040
023762 041445 031123 000040
023770 042045 020123 000
023775 045 051105 000040
024002 040445 020123 000
024007 045 045503 000040
024014 042045 020102 000
024021 045 051115 000040
024026 042045 020124 000
024033 045 041524 000040
024040 047045 020117 047111
024046 042524 051122 050125
024054 022524 000
024057 045 046123 053101

:ERROR MESSAGES*****
\$MSWR: .ASCIZ /%SWR = /
\$MNEW: .ASCIZ / NEW = /
DASH: .ASCIZ /-/
MSG1: .ASCIZ /*DE /
MSG2: .ASCIZ /%G; /
MSG3: .ASCIZ /%B; /
MSG4: .ASCIZ /%CN /
MSG5: .ASCIZ /*WE /
MSG6: .ASCIZ /*RE /
MSG7: .ASCIZ /*RS /
MSG8: .ASCIZ /*PATRN /
MSG9: .ASCIZ /SN: /
MSG10: .ASCIZ /*SE /
MSG13: .ASCIZ /%*BN /
MSG14: .ASCIZ /*RN /
MSG15: .ASCIZ /% BAD RECORD%/

MSG16: .ASCIZ / F/
MSG17: .ASCIZ / R/
MSG20: .ASCIZ / EOT # /
MSG21: .ASCIZ /INTERCHANGE READ? /
MSG22: .ASCIZ /%ILLEGAL BOT: HALT%/

MSG23: .ASCIZ /%CS1 /
MSG23A: .ASCIZ /%WC /
MSG23B: .ASCIZ /%BA /
MSG23C: .ASCIZ /%FC /
MSG23D: .ASCIZ /%CS2 /
MSG23E: .ASCIZ /%DS /
MSG23F: .ASCIZ /%ER /
MSG23G: .ASCIZ /%AS /
MSG23H: .ASCIZ /%CK /
MSG23I: .ASCIZ /%DB /
MSG23J: .ASCIZ /%MR /
MSG23K: .ASCIZ /%DT /
MSG23L: .ASCIZ /%TC /
MSG24: .ASCIZ /%NO INTERRUPT%/

MSG25: .ASCIZ /%SLAVE UNSAFE-TEST DISCONTINUED ON SLAVE%/

4888	024064	020105	047125	040523	
4889	024072	042506	052055	051505	
4890	024100	020124	044504	041523	
4891	024106	047117	044524	052516	
4892	024114	042105	047440	020116	
4893	024122	046123	053101	022505	
4894	024130	000			
4895	024131	045	051104	050117	MSG26: .ASCIZ /%DROPS: /
4896	024136	035123	000040		
4897	024142	050045	041511	051513	MSG27: .ASCIZ /%PICKS: /
4898	024150	020072	000		
4899	024153	045	000		MSG28: .ASCIZ /%/
4900	024155	045	052045	047515	MSG30: .ASCIZ '%TMO3-TE16/TU77 AUTO SEQUENCE (CZTEDBO)%';++B
4901	024162	026463	042524	033061	
4902	024170	052057	033525	020067	
4903	024176	052501	047524	051440	
4904	024204	050505	042525	041516	
4905	024212	020105	041450	052132	
4906	024220	042105	030102	022451	
4907	024226	000			
4908	024227	045	052045	030115	MSG31: .ASCIZ '%TMO3-TE16/TU77 DATA RELIABILITY TEST (CZTEDBO)%';++B
4909	024234	026463	042524	033061	
4910	024242	052057	033525	020067	
4911	024250	040504	040524	051040	
4912	024256	046105	040511	044502	
4913	024264	044514	054524	052040	
4914	024272	051505	020124	041450	
4915	024300	052132	042105	030102	
4916	024306	022451	000		
4917	024311	124	050131	020105	MSG31A: .ASCIZ /TYPE <CR> TO TERMINATE ALL REQUESTS & ^C TO RESTART%/
4918	024316	041474	037122	052040	
4919	024324	020117	042524	046522	
4920	024332	047111	052101	020105	
4921	024340	046101	020114	042522	
4922	024346	052521	051505	051524	
4923	024354	023040	057040	020103	
4924	024362	047524	051040	051505	
4925	024370	040524	052122	000045	
4926	024376	046123	053101	020105	MSG32: .ASCIZ /SLAVE # = /
4927	024404	020043	020075	000	
4928	024411	104	047105	044523	MSG33: .ASCIZ /DENSITY = /
4929	024416	054524	036440	000040	
4930	024424	040520	044522	054524	MSG34: .ASCIZ /PARITY = /
4931	024432	036440	000040		
4932	024436	042522	047503	042122	MSG35: .ASCIZ /RECORD COUNT = /
4933	024444	041440	052517	052116	
4934	024452	036440	000040		
4935	024456	044103	051101	041440	MSG36: .ASCIZ /CHAR COUNT = /
4936	024464	052517	052116	036440	
4937	024472	000040			
4938	024474	040520	052124	051105	MSG37: .ASCIZ /PATTERN # = /
4939	024502	020116	020043	020075	
4940	024510	000			
4941	024511	123	047111	046107	MSG38: .ASCIZ /SINGLE PASS? /
4942	024516	020105	040520	051523	
4943	024524	020077	000		

4944	024527	103	041522	041440	MSG39:	.ASCIZ	/CRC CORRECTION (YES=1,NO=0)? /
4945	024534	051117	042522	052103			
4946	024542	047511	020116	054450			
4947	024550	051505	030475	047054			
4948	024556	036517	024460	020077			
4949	024564	000					
4950	024565	045	042445	052116	MSG40:	.ASCIZ	/%ENTER STALLS%READ = /
4951	024572	051105	051440	040524			
4952	024600	046114	022523	042522			
4953	024606	042101	036440	000040			
4954	024614	051127	052111	020105	MSG41:	.ASCIZ	/WRITE = /
4955	024622	020075	000				
4956	024625	124	051125	020116	MSG42:	.ASCIZ	/TURN AROUND = /
4957	024632	051101	052517	042116			
4958	024640	036440	000040				
4959	024644	037445	000045		MSG43:	.ASCIZ	/%?%/
4960	024650	042445	052116	051105	MSG44:	.ASCIZ	/%ENTER YOZZLE STALL = /
4961	024656	054440	055117	046132			
4962	024664	020105	052123	046101			
4963	024672	020114	020075	000			
4964	024677	045	051105	020122	MSG45:	.ASCIZ	/%ERR AMT /
4965	024704	046501	020124	000			
4966	024711	045	047516	020124	MSG49:	.ASCIZ	/%NOT AVAIL /
4967	024716	053101	044501	020114			
4968	024724	000					
4969	024725	045	046111	042514	MSG50:	.ASCIZ	/%ILLEGAL DRIVE TYPE /
4970	024732	040507	020114	051104			
4971	024740	053111	020105	054524			
4972	024746	042520	000040				
4973	024752	022445			MSG52:	.ASCII	/%%/
4974	024754	051104	053111	020105	MSG52A:	.ASCIZ	/DRIVE (TM03) # = /
4975	024762	052050	030115	024463			
4976	024770	021440	036440	000040			
4977	024776	047506	046522	052101	MSG53:	.ASCIZ	/FORMAT = /
4978	025004	036440	000040				
4979	025010	053452	020105	046524	MSG54:	.ASCIZ	/*WE TM/
4980	025016	000					
4981	025017	052	042523	052040	MSG55:	.ASCIZ	/*SE TM/
4982	025024	000115					
4983	025026	052040	000115		MSG56:	.ASCIZ	/ TM/
4984	025032	047045	047117	042455	MSG57:	.ASCIZ	/%NON-EXIST SLAVE/
4985	025040	044530	052123	051440			
4986	025046	040514	042526	000			
4987	025053	045	051103	020103	MSG58:	.ASCIZ	/%CRC /
4988	025060	000					
4989	025061	045	051114	020103	MSG59:	.ASCIZ	/%LRC /
4990	025066	000					
4991	025067	052	020120	000	MSG61:	.ASCIZ	/*P /
4992	025073	052	020106	000	MSG62:	.ASCIZ	/*f /
4993	025077	045	047452	044522	MSG64:	.ASCIZ	/%*ORIGINAL ERROR*/
4994	025104	044507	040516	020114			
4995	025112	051105	047522	025122			
4996	025120	000					
4997	025121	045	042522	051124	MSG65:	.ASCIZ	/%RETRY: /
4998	025126	035131	000040				
4999	025132	051452	020105	052122	MSG66:	.ASCIZ	/*SE RTRY /

5000	025140	054522	000040						
5001	025144	042452	040522	042523	MSG67:	.ASCIZ	/*ERASE/		
5002	025152	000							
5003	025153	045	042522	042522	MSG68:	.ASCIZ	/%REREV: /		
5004	025160	035126	000040						
5005	025164	040524	042520	046440	MSG69:	.ASCIZ	/TAPE MARK? /		
5006	025172	051101	037513	000040					
5007	025200	047045	047117	042455	MSG71:	.ASCIZ	/%NON-EXIST DRIVE/		
5008	025206	044530	052123	042040					
5009	025214	044522	042526	000					
5010	025221	045	042522	053506	MSG72:	.ASCIZ	/%REFWD: /		
5011	025226	035104	000040						
5012	025232	053445	042524	051122	MSG73:	.ASCIZ	/%WTERR: /		
5013	025240	020072	000						
5014	025243	045	042522	044507	MSG74:	.ASCIZ	/%REGISTER START = /		
5015	025250	052123	051105	051440					
5016	025256	040524	052122	036440					
5017	025264	000040							
5018	025266	042526	052103	051117	MSG75:	.ASCIZ	/%VECTOR ADRS = /		
5019	025274	040440	051104	020123					
5020	025302	020075	000						
5021	025305	045	042504	042522	MSG76:	.ASCIZ	/%DEREV: /		
5022	025312	035126	000040						
5023	025316	042045	043105	042127	MSG77:	.ASCIZ	/%DEFWD: /		
5024	025324	020072	000						
5025	025327	045	047516	026516	MSG78:	.ASCIZ	/%NON-RETRYABLE WRITE ERROR: ER /		
5026	025334	042522	051124	040531					
5027	025342	046102	020105	051127					
5028	025350	052111	020105	051105					
5029	025356	047522	035122	042440					
5030	025364	020122	000						
5031	025367	045	047516	026516	MSG79:	.ASCIZ	/%NON-RETRYABLE READ ERROR: ER /		
5032	025374	042522	051124	040531					
5033	025402	046102	020105	042522					
5034	025410	042101	042440	051122					
5035	025416	051117	020072	051105					
5036	025424	000040							
5037	025426	042445	042116	047440	MSG100:	.ASCIZ	/%END OF PASS %/		
5038	025434	020106	040520	051523					
5039	025442	022440	000						
5040	025445	045	025045	025052	MSG101:	.ASCIZ	/%*****%/		
5041	025452	025052	025052	025052					
5042	025460	025052	025052	025052					
5043	025466	025052	025052	022452					
5044	025474	000							
5045	025475	101	052125	020117	MSG104:	.ASCIZ	/AUTO CONT.? /		
5046	025502	047503	052116	037456					
5047	025510	000040							
5048	025512	051045	041505	053117	MSG105:	.ASCIZ	/%RECOVERED/		
5049	025520	051105	042105	000					
5050	025525	052	040502	020104	MSG106:	.ASCIZ	/*BAD TAPE OVERFLOW/		
5051	025532	040524	042520	047440					
5052	025540	042526	043122	047514					
5053	025546	000127							
5054	025550	051045	053505	047111	MSG16A:	.ASCIZ	/%REWIND TAPE; RESTART AT BLOCK 1/		
5055	025556	020104	040524	042520					

5056	025564	020073	042522	052123	
5057	025572	051101	020124	052101	
5058	025600	041040	047514	045503	
5059	025606	030440	000		
5060	025611	045	047125	042522	MSG107: .ASCII /%UNRECOVERABLE BAD SPOT/
5061	025616	047503	042526	040522	
5062	025624	046102	020105	040502	
5063	025632	020104	050123	052117	
5064	025640	041045	042101	051040	.ASCIZ /%BAD RECORD LEFT ON TAPE%/
5065	025646	041505	051117	020104	
5066	025654	042514	052106	047440	
5067	025662	020116	040524	042520	
5068	025670	000045			
5069	025672	050052	051517	052111	MSG109: .ASCIZ /*POSITION LOST IN RETRY/
5070	025700	047511	020116	047514	
5071	025706	052123	044440	020116	
5072	025714	042522	051124	000131	
5073	025722	051445	051525	042520	MSG110: .ASCIZ /%SUSPECT BAD TAPE/
5074	025730	052103	041040	042101	
5075	025736	052040	050101	000105	
5076	025744	051045	050105	040505	MSG111: .ASCIZ /%REPEAT: /
5077	025752	035124	000040		
5078	025756	041040	042101	052040	MSG112: .ASCIZ / BAD TAPE SPOTS%/
5079	025764	050101	020105	050123	
5080	025772	052117	022523	000	
5081					
5082	025777	045	051440	043117	MSG113: .ASCIZ /% SOFT: /
5083	026004	035124	000040		
5084					
5085	026010	020045	040510	042122	MSG114: .ASCIZ /% HARD: /
5086	026016	020072	000		
5087					
5088	026021	045	040510	042122	MSG115: .ASCIZ /%HARD READ ERROR/
5089	026026	051040	040505	020104	
5090	026034	051105	047522	000122	
5091	026042	051445	040514	042526	MSG116: .ASCIZ /%SLAVE REWINDING: WILL RESTART AT BOT/
5092	026050	051040	053505	047111	
5093	026056	044504	043516	020072	
5094	026064	044527	046114	051040	
5095	026072	051505	040524	052122	
5096	026100	040440	020124	047502	
5097	026106	000124			
5098	026110	000134			
5099	026112	051045	046505	053117	MSG118: .ASCIZ /\
5100	026120	020105	046524	050104	MSG120: .ASCIZ /%REMOVE TMDP FROM SLAVE TO BE TESTED%/
5101	026126	043040	047522	020115	
5102	026134	046123	053101	020105	
5103	026142	047524	041040	020105	
5104	026150	042524	052123	042105	
5105	026156	000045			
5106	026160	044045	051101	053504	MSG121: .ASCIZ /%HARDWARE SWR IN USE%/
5107	026166	051101	020105	053523	
5108	026174	020122	047111	052440	
5109	026202	042523	000045		
5110	026206	047516	051440	040514	MSG122: .ASCIZ /%NO SLAVES LEFT TO TEST: HALT%/
5111	026214	042526	020123	042514	

5112 026222 052106 052040 020117
5113 026230 042524 052123 020072
5114 026236 040510 052114 000045
5115 026244 040445 052125 026517
5116 026252 042523 035121 052040
5117 026260 051505 020124 044527
5118 026266 046114 051040 051505
5119 026274 040524 052122 000045
5120 026302 041445 051117 042522
5121 026310 052103 042105 050040
5122 026316 020105 040504 040524
5123 026324 042440 051122 051117
5124 026332 000
5125 026333 040 000
5126 026335 060 000
5127 026337 061 000
5128
5129 026342 000000
5130 026342 000000
5131
5132 032350
5133 032350 000000
5134
5135 000001

MSG123: .ASCIZ /%AUTO-SEQ: TEST WILL RESTART%/

MSG124: .ASCIZ /%CORRECTED PE DATA ERROR/

SPACE: .ASCIZ ' '
DIGIT0: .ASCIZ '0'
DIGIT1: .ASCIZ '1'

WDATA: 0 .EVEN ;WRITE BUFFER

RDATA: 0 .+.4004 ;READ BUFFER

.END

DAT14	014242	1795	3462#						
DAT15	014272	1796	3474#						
DAT2	014036	1785	3381#						
DAT3	014042	1786	3386#						
DAT3A	014050	3388#	3399						
DAT4	014066	1787	3397#						
DAT5	014076	1788	3404#						
DAT6	014104	1789	3409#						
DAT7	014112	1790	3414#						
DB	000532	1480#							
DCHK	014750	2672	2865	3606#					
DCHKO	014776	3610	3612#						
DEREV1	001164	1658#	1977	3692*					
DEREX	015754	3759	3777	3779	3787	3794	3797	3799#	
DEREX1	016006	3800	3803	3805	3807#				
DERFL	000704	1540#	3607*	3683	3808*				
DERR	015354	3676	3723#						
DERRO	015364	3725#	3806						
DERROA	015412	3727	3731#						
DERROB	015440	3736	3739#						
DERROC	015460	3742	3745#						
DERROD	015462	3744	3746#						
DERR1	015504	3749	3752#						
DERR2	015506	3751	3753#						
DERR3	015520	3756#							
DERR4	015522	3724	3755	3757#					
DERR4A	015646	3771	3780#						
DERR4B	015710	3766	3788#						
DERR5	015736	3791	3795#						
DERR6	015750	3768	3789	3798#					
DFX	015352	3684	3686	3691	3693#				
DF0	015250	3632	3671#	3680					
DF0A	015144	3642	3644#	3681					
DF0A0	015166	3648	3650#						
DF0A1	015202	3653	3655#						
DF0A2	015216	3658	3660#						
DF0A3	015232	3663	3665#						
DF0A4	015236	3645	3667#						
DF0B	015104	3633#							
DF0B0	015126	3636	3639#						
DF0C	015066	3625	3629#						
DF0C0	015076	3615	3617	3619	3631#				
DF0D	015052	3621	3626#						
DF0E	015044	3623#	3628						
DF0F	015036	3620#	3624						
DF1	015262	3668	3672	3675#					
DF2	015272	3670	3674	3677#					
DF3	015306	3678	3682#						
DF4	015346	3689	3692#						
DIGIT0	026335	4743	4790	5126#					
DIGIT1	026337	4788	5127#						
DOUT	023302	3746	3753	4784#	4797	4799			
DOUTD	023336	4796#							
DPC	016126	3848#	3892						
DPCG	016134	3849	3851#						
DPCO	016142	3852#	3884						

MSG14	023613	2445	4544	4853#				
MSG15	023620	3762	4854#					
MSG16	023650	4210	4858#					
MSG16A	025550	2030	5054#					
MSG17	023653	4207	4537	4859#				
MSG2	023525	3740	4841#					
MSG20	023656	2025	4860#					
MSG21	023666	3208	4862#					
MSG22	023711	2618	4866#					
MSG23	023735	4098	4870#					
MSG23A	023743	4125	4872#					
MSG23B	023750	4114	4873#					
MSG23C	023755	4122	4874#					
MSG23D	023762	4101	4875#					
MSG23E	023770	4106	4876#					
MSG23F	023775	4109	4877#					
MSG23G	024002	4878#						
MSG23H	024007	4152	4879#					
MSG23I	024014	4880#						
MSG23J	024021	4881#						
MSG23K	024026	4882#						
MSG23L	024033	4883#						
MSG24	024040	4316	4884#					
MSG25	024057	4220	4887#					
MSG26	024131	3894	4895#					
MSG27	024142	3908	4897#					
MSG28	024153	2453	2467	4656	4701	4824	4899#	
MSG3	023532	3747	4842#					
MSG30	024155	1834	3040	4900#				
MSG31	024227	3037	4908#					
MSG31A	024311	3044	3045*	4917#				
MSG32	024376	3095	4393	4511	4926#			
MSG33	024411	3126	4928#					
MSG34	024424	3138	4930#					
MSG35	024436	3167	4932#					
MSG36	024456	3177	4935#					
MSG37	024474	3189	4938#					
MSG38	024511	3217	4941#					
MSG39	024527	3226	4944#					
MSG4	023537	3731	4843#					
MSG40	024565	3239	4950#					
MSG41	024614	3248	4954#					
MSG42	024625	3257	4956#					
MSG43	024644	4687	4959#					
MSG44	024650	4350	4960#					
MSG45	024677	4964#						
MSG49	024711	4966#						
MSG5	023544	2220	4844#					
MSG50	024725	3120	4969#					
MSG52	024752	4508	4973#					
MSG52A	024754	3081	4389	4974#				
MSG53	024776	3149	4977#					
MSG54	025010	2270	4093	4979#				
MSG55	025017	2900	4981#					
MSG56	025026	4095	4983#					
MSG57	025032	3114	4984#					

RWIND1A	005064	2120	2126	2130#										
RWIND2	005102	2118	2133#											
RWIND3	005106	2134#	2148											
RWIND5	005172	2139	2144	2146#										
SCVFL	013444	1465*	3046	3235	3265*	3267#	4346*							
SERFL	000706	1541#	2242	2246*	2289	2330	2394*	2406	2416*	2425	2658	2673*	2723*	2736*
		2739	3685	4064*	4072*									
SN	000540	1483#	4804											
SNPG	023452	4812	4815	4821	4823	4826#								
SNPT	023360	3124	4804#											
SPACE	026333	4392	4766	5125#										
SPFLG	000572	1499#	2083	3218	3220									
STAL	000666	1533#	1923*	1930*	1933*	2039*	2127*	2237*	2395*	2516*	2538*	2678*	2801*	2810*
		2814*	2816*	2887*	2922*	2945*	4298*	4306*						
STALL	011654	1931	1934	2238	2396	2405	2517	2539	2679	2802	2817	2888	2923	2945#
		2946												
START	003022	1442	1466	1816#										
STARTA	003144	1449	1847#	2620										
STARTB	003150	1840	1848#											
STARTC	003136	1445	1843#											
STARTD	003250	1844	1867#											
STARTE	003242	1866#	2098											
START1	003406	1890#	1944											
START2	003522	1908	1910#											
START3	003536	1911	1913#											
START4	003552	1895	1914	1916#										
START7	003734	1917	1936	1943#	2048									
START8	003742	1944#												
STAR1A	003420	1892#												
STAR1B	003440	1893	1896#											
STAR1C	003474	1898	1900	1904#										
STAUT	003122	1461	1832	1838#										
STAUTO	003352	1883#	4459	4461	4463	4465	4481	4483	4485	4488				
STFLG	000640	1521#	1848	1849										
STP	003744	1939	1947#	2032										
STTBL	001044	1602#	1855	1856										
SWR	000610	1506#	1823*	1869	1873	1877*	1907	1910	1913	1935	2109	2216	2234	2247
		2258	2276	2293	2306	2314	2335	2351	2503	2506	2508	2512	2519	2527
		2529	2534	2621	2668	2675	2714	2732	2741	2763	2833	2861	2867	2898
		3723	3760	3799	3804	3858	3865	4035	4074	4157	4162	4309	4317	4333
		4335*	4341*	4348	4588	4592	4595							
SWREG	000176	1438#	1823	1869	1877	4333	4341	4588						
TAPG	020372	2225	2275	2403	2423	2606	2811	2832	2897	2920	4261#			
TAPG3	020420	4264	4266#											
TAPG3A	020466	4273	4276#											
TAPG3B	020502	4275	4278	4280#										
TAPG3C	020522	4283	4286#											
TAPG3D	020566	4290	4292	4294#										
TAPG3E	020612	4297	4299#											
TAPG3F	020450	4267	4272#											
TAPG4	020640	4304#	4305	4307										
TAPG5	020654	4308#												
TAPG6	020720	4310	4317#											
TAPG7	020730	4318	4320#											
TC	000542	1484#	1920*	2001*	2053*	2122*	2140*	2153*	2167*	3111*	4177*	4432*		
TEMP1	000644	1524#	3101	3342*	3348	3352*	3780*	3781*	3782*	3783	3785	3829*	3832*	3840*

TEMP2	000646	3846	3889	4261*	4303*	4304*	4642*	4651	4671*	3353*	3360	3830*	3833*	3841*
		1525#	3094	3096*	3127*	3139*	3150*	3271*	3275					
TEMP3	000650	3847	3890											
		1526#	2343*	2349*	2360	2362*	2755*	2761*	2768	2770*	3831*	3854	3891*	4796
		4798												
TEND	004662	2085#	4407											
TIB	000642	1523#	4645	4649	4654	4658	4667	4669	4675*	4676	4697*	4698*	4699	4703
TINER	022704	4668	4670	4680	4682	4687#								
TINF	000636	1520#	1838*	1843*	1847*	2075*	3032	3237						
TINP	011750	1881	3032#											
TINPX	013436	3236	3238	3265#										
TINPO	012302	3090	3094#	3104	3115	3161								
TINPOB	012364	3102	3107#											
TINPOD	012434	3113	3116#											
TINPOE	012470	3119	3124#											
TINP1	012474	3126#												
TINP2	012534	3135#												
TINP2A	012612	3149#												
TINP2B	012652	3157#												
TINP2C	012700	3106	3159	3164#										
TINP3	012712	3167#												
TINP3A	013226	3226#	4347											
TINP4	013270	3034	3235#											
TKB	000614	1508#	4324	4697										
TKS	000612	1507#	1890*	4694*	4695									
TMEX	000564	1496#	2267	2587	2694	2892	3200	3202	4457*					
TMFLG	000676	1537#	2269*	2311*	2325	2584*	2589*	2611	2623	2670	2682*	2696	2698*	2701*
		2812	2835	2863	3981	4003	4009	4021	4039	4091	4180			
TOB	000640	1522#	4712*	4719*	4722*	4726*	4735	4775*	4826*	4828*				
TOG	023106	4716	4720	4723	4727	4733#	4734	4776	4829					
TPB	000620	1510#	4735*											
TPOS	013446	3133	3147	3156	3271#	3273								
TPS	000616	1509#	4733											
TSTAL	000600	1502#	1930	1933	2395	2516	2538	2887	2922	3258	3260			
TTIN	022720	4644	4694#											
TTINT	020734	1432	4324#											
TTOUT	022772	1413	4708#											
TTR	022474	3055	3064	3086	3100	3131	3143	3154	3174	3185	3197	3206	3215	3224
		3233	3246	3255	3264	4357	4385	4599	4640#					
TYPE =	000004	1416#	1829	1834	1948	1952	1955	1958	1961	1964	1967	1970	1973	1976
		2022	2025	2030	2085	2253	2260	2299	2308	2337	2338	2347	2353	2354
		2357	2442	2445	2453	2467	2474	2483	2618	2724	2729	2734	2743	2744
		2759	2765	2775	3041	3044	3048	3057	3081	3091	3095	3114	3120	3126
		3138	3149	3167	3177	3189	3199	3208	3217	3226	3239	3248	3257	3279
		3731	3740	3747	3762	3894	3908	4088	4095	4098	4101	4106	4109	4114
		4117	4122	4125	4130	4134	4139	4142	4152	4211	4220	4223	4228	4269
		4313	4316	4337	4350	4380	4388	4389	4392	4393	4508	4511	4515	4520
		4526	4534	4537	4541	4544	4559	4562	4591	4594	4656	4664	4687	4701
		4703	4743	4766	4788	4790	4805	4824						
TYPOCT=	104400	1420#	1951	1954	1957	1960	1963	1966	1969	1972	1975	1978	2029	2340
		2356	2359	2441	2444	2448	2473	2746	2767	2785	3050	3059	3123	3169
		3180	3191	3201	3210	3219	3228	3241	3250	3259	3739	3902	3910	4100
		4103	4108	4111	4116	4119	4124	4127	4133	4136	4141	4144	4155	4352
		4391	4397	4510	4514	4519	4525	4533	4540	4543	4558	4561	4565	4593
UDES	000552	1491#	1894*	1906*	1920	2001	2052*	2053	2121*	2122	2137*	2140	2152*	2153
		2164	2597	2640	2646	2652	2823	2842	2847	2853	3307	3327	3557	3571

\$CHAIN	1323#	1818
\$RESTO	1323#	4613
\$SAVE	1323#	4603
.\$ACT1	1323#	1421
.\$EOP	1323#	2086

. ABS. 032352 000

ERRORS DETECTED: 0

DSKZ:CZTEDB,DSKZ:CZTEDB,SEQ/CRF/SOL=CZTEAB.SML/ML,CZTEDB.P11
RUN-TIME: 6 12 1 SECONDS
RUN-TIME RATIO: 129/20=6.1
CORE USED: 11K (21 PAGES)