

TM03

TM03 TE16 TU77 DRT
CZTEDCO

AH A801C MC
COPYRIGHT 77 79
FICHE 1 OF 1

NOV 1979
digital
MADE IN USA

This microfiche card contains a grid of frames. The first 12 columns of frames contain data, while the remaining 12 columns are blank. The data in the frames is organized into columns and rows, with some frames containing headers and footers. The data appears to be a list of records, possibly related to the 'TM03' project mentioned in the header. The frames are arranged in a 12x12 grid, with the first 12 columns containing data and the last 12 columns being blank.

.REM %

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

IDENTIFICATION

PRODUCT CODE: AC-A800C-MC
PRODUCT NAME: CZTEDCO TM03-TE16/TU77 DATA RELIBILITY PROGRAM
DATE CREATED: 10 MAY 1979
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 ,1979 BY DIGITAL EQUIPMENT CORPORATION

TABLE OF CONTENTS

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

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 PROCESSOR
- 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(TM03 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 REENTERED. 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 THEREFORE 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 TMO3) 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) CHARATERS 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 TMO3 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 MAINTENENCE 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 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.

z

```
1310 .LIST BIN,LOC,SEQ
1311 .TITLE CZTEDCO TMO3-TE16/TU77 DRT
1312 :DATA RELIABILITY TEST
1313 :AC-A800C-MC
1314 :21 FEB 1977
1315 :J.G.ADAMS
1316 :REVISED (++) J.G.ADAMS MAY 1978
1317 :++B
1318 :++B
1319 :++B
1320 :++B
1321 :++B
1322
1323 :(++C) M.PAGE FEB 79
1324 :++C
1325 :
1326 :
1327
1328 .MCALL .SACT11,.$EOP,$SAVE,$RESTORE,$CHAIN
1329 .NLIST MC
1330 .LIST ME
1331 .ENABLE ABS,AMA
1332
1333 :CONSOLE SWITCHES*****
1334
1335 :SW15: 1=STOP ON ERROR
1336 :      0=CONTINUE ON ERROR
1337 :SW14: 1=PRINT READ/WRITE STATS
1338 :      0=DO NOT PRINT STATS
1339 :SW13: 1=DO NOT CHECK DATA
1340 :      0=CHECK DATA
1341 :SW12: 1=DO NOT CHECK WRITE ERRORS
1342 :      0=CHECK WRITE ERRORS
1343 :SW11: 1=DO NOT CHECK READ ERRORS
1344 :      0=CHECK READ ERRORS
1345 :SW10: 1=DO NOT PRINT ERRORS
1346 :      0=PRINT ERRORS
1347 :SW9:  1=REWIND TAPE
1348 :      0=DO NOT REWIND
1349 :SW8:  1=USE RANDOM DATA
1350 :      0=USE FIXED DATA PATTERN
1351 :SW7:  1=USE RANDOM CHARACTER COUNT
1352 :      0=USE FIXED CHAR COUNT
1353 :SW6:  1=USE RANDOM RECORD COUNT
1354 :      0=USE FIXED RECORD COUNT
1355 :SW5:  1=YOZZLE ON CURRENT RECORD
1356 :      0=DO NOT YOZZLE
1357 :SW4:  1=DO BOTH READ AND WRITE RETRIES
1358 :      0=INHIBIT RETRIES
1359 :SW3:  1=DO NOT READ FORWARD
1360 :      0=READ FORWARD
1361 :SW2:  1=DO NOT READ REVERSE
1362 :      0=READ REVERSE
1363 :SW1:  1=READ FORWARD FIRST
1364 :      0=READ REVERSE FIRST
1365 :SW0:  1=DO NOT WRITE
```

```
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
```

```
RECORD NUMBERING SYSTEM NOT CONSISTENT
BETWEEN FORWARD AND REVERSE TAPE MOVEMEN
FORMAT ERROR (BIT 4) MADE RETRYABLE
```

CZTEDCO TM03-TE16/TU77 DRT
CZTEDC.P11 10-MAY-79 10:32

MACY11 30A(1052) 10-MAY-79 10:46 F 3 PAGE 31

SEQ 0031

1366
1367

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


```

1414
1415
1416
1417
1418 000020 000020
1419 000022 000340
1420
1421 000004
1422 000034 000034
1423 000034 023126
1424 000036 000340
1425 104400
1426
1427
1428 000040
1429 000042
1430 000042 000000
1431 000046 000046
1432 000046 004676
1433 000052 000052
1434 000052 000000
1435 000040
1436
1437
1438 000060
1439 000060 020734
1440 000062 000340
1441
1442
1443
1444 000176 000176
1445 000176 000000
1446
1447
1448 000200 000200
1449 000200 000137 003022
1450
1451 000204 000204
1452 000204 000137 003136
1453
1454 000210 000210
1455 000210 005037 014404
1456 000214 000137 003144
1457
1458
1459
1460 000224 000224
1461 000224 021160
1462 000226 000340
1463
1464
1465
1466 000240 000240
1467 000240 005237 000734
1468 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
.=42
.WORD 0
.=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

```

```
1469 ;SHORT CONVERSATION RESTART*****
1470
1471 000300 000300 . =300
1472 000300 005237 013444 INC SCVFL ;SET SHORT CONVERSATION FLAG
1473 000304 000137 003022 JMP START ;ENTER SHORT PARAMETER LIST
1474
1475 000510 . =510
1476 ;TU16 REGISTER EQUIVS*****
1477
1478 000510 172440 C1: 172440
1479 000512 172442 WC: 172442
1480 000514 172444 BA: 172444
1481 000516 172446 FC: 172446
1482 000520 172450 CS: 172450
1483 000522 172452 DS: 172452
1484 000524 172454 ER: 172454
1485 000526 172456 AS: 172456
1486 000530 172460 CC: 172460
1487 000532 172462 DB: 172462
1488 000534 172464 MR: 172464
1489 000536 172466 DT: 172466
1490 000540 172470 SN: 172470
1491 000542 172472 TC: 172472
1492
1493 ;CONSTANTS*****
1494
1495 000544 172440 REGS: 172440 ;STARTING REGISTER ADDRESS (CS1)
1496 000546 000224 VECT: 224 ;VECTOR ADDRESS (RH INTERRUPT)
1497 000550 000000 DVN: 0 ;DRIVE NUMBER
1498 000552 000000 UDES: 0 ;UNIT DESCRIPTION (PARITY,DENSITY,UNIT,FORMAT)
1499 000554 000100 RCNT: 100 ;RECORD COUNTER
1500 000556 177400 FMCNT: 177400 ;NUMBER OF CHAR (4 - 4000) OCTAL IN TWOS COMPLEMENT
1501 000560 000001 PATRN: 1 ;DATA PATTERN SELECTOR (0 - 15) OCTAL
1502 000562 000000 RDCMD: 0 ;READ COMMAND
1503 000564 000001 TMEX: 1 ;TAPE MARK FLAG: 1=TM 0=NO TM
1504 000566 000000 CRCC: 0 ;CRC CORRECTION FLAG (YES=1,NO=0)
1505 000570 000000 INTRF: 0 ;INTERCHANGE READ 1=YES 0=NO
1506 000572 000000 SPFLG: 0 ;SINGLE PASS 1=YES 0=NO
1507 000574 000001 RSTAL: 1 ;READ STALL
1508 000576 000001 WSTAL: 1 ;WRITE STALL
1509 000600 000001 TSTAL: 1 ;TURN AROUND STAL
1510 000602 002000 YSTAL: 2000 ;YOZZLE STAL
1511 000604 000010 RETRY: 10 ;READ RETRY NUMBER
1512 000606 177776 PSW: 177776 ;PROCESSOR STATUS
1513 000610 177570 SWR: 177570 ;CONSOLE SWITCHES
1514 000612 177560 TKS: 177560 ;TTY READ STATUS REGISTER
1515 000614 177562 TKB: 177562 ;TTY READ BUFFER
1516 000616 177564 TPS: 177564 ;TTY PUNCH STATUS REGISTER
1517 000620 177566 TPB: 177566 ;TTY PUNCH OUTPUT REGISTER
1518 000622 177550 PRS: 177550 ;H/S READER STATUS REGISTER
1519 000624 177552 PRB: 177552 ;H/S READER BUFFER
1520 000626 153624 RANBAS: 153624 ;RANDOM NUMBER GENERATOR BASE
1521 000630 032561 RANSAV: 032561 ;RANDOM NUMBER BUFFER
1522 000632 000100 RCSAV: 100 ;RECORD COUNT SAVE
1523 000634 177400 FCSAV: 177400 ;FRAME COUNT SAVE
1524
```

```
1525 ;FLAGS AND COUNTERS*****
1526
1527 000636 000000 TINF: 0 ;TTY ENTRY FLAG
1528 000640 STFLG:
1529 000640 000000 TOB: 0 ;TTY OUTPUT BUFFER
1530 000642 000000 TIB: 0 ;TTY INPUT BUFFER
1531 000644 000000 TEMP1: 0 ;TEMP STORAGE
1532 000646 000000 TEMP2: 0 ;TEMP STORAGE
1533 000650 000000 TEMP3: 0 ;TEMP STORAGE
1534 000652 000000 EMADDR: 0 ;ERROR MSG ADDRESS STORAGE
1535 000654 000000 BLCNTR: 0 ;BLOCK COUNTER
1536 000656 000000 BBC: 0 ;BAD RECORD COUNTER
1537 000660 000000 EOTREC: 0 ;EOT FLAG
1538 000662 000000 RTRN: 0 ;INTERRUPT RETURN STORAGE
1539 000664 000000 HDRFL: 0 ;HEADER FLAG
1540 000666 000000 STAL: 0 ;DELAY STORAGE
1541 000670 000000 PFLG: 0 ;PRINT FLAG
1542 000672 000000 MTC1: 0 ;MAG TAPE CONT REGISTER BUFFER
1543 000674 000000 UNP: 0 ;UNIT TABLE POINTER
1544 000676 000000 TMFLG: 0 ;TAPE MARK FLAG
1545 000700 000000 RPCNT: 0 ;REPEAT COUNTER
1546 000702 000000 RTCNT: 0 ;RETRY COUNTER
1547 000704 000000 DERFL: 0 ;DATA ERROR FLAG
1548 000706 000000 SERFL: 0 ;STATUS ERROR FLAG
1549 000710 000000 BCNT: 0 ;BIT COUNTER
1550 000712 000000 RTYFL: 0 ;RETRY FLAG
1551 000714 000000 UPS: 0 ;UNIT POINTER SAVE
1552 000716 000000 BDPP: 0 ;BITS DROPPED POINTER
1553 000720 000000 BPKP: 0 ;BITS PICKED POINTER
1554 000722 000000 ERSV: 0 ;ERROR SAVE LOC
1555 000724 000000 BTFLG: 0 ;BAD TAPE FLAG
1556 000726 000000 BTSTF: 0 ;STATISTIC PRINT FLAG
1557 000730 000000 BTPT: 0 ;BAD TAPE POINTER
1558 000732 000000 ERTFL: 0 ;ERASE FLAG
1559 000734 ENDFLG:
1560 000734 000000 ASEQF: 0 ;AUTO SEQ FLAG
1561 000736 000000 ABLCNT: 0 ;AUTO BLCCK COUNTER
1562 000740 000000 ASEQCF: 0 ;AUTO SEQ CONTINUOUS FLAG
```

1563
1564
1565
1566 000742 000000
1567 000744 000000
1568 000746 000000
1569 000750 000000
1570 000752 000000
1571 000754 000000
1572 000756 000000
1573 000760 000000
1574 000762 177777
1575
1576
1577
1578 000764 001204
1579 000766 001224
1580 000770 001244
1581 000772 001264
1582 000774 001304
1583 000776 001324
1584 001000 001344
1585 001002 001364
1586 001004 001404
1587 001006 001424
1588 001010 001444
1589 001012 001464
1590 001014 001504
1591 001016 001524
1592 001020 001544
1593 001022 001564
1594
1595
1596
1597 001024 001604
1598 001026 001710
1599 001030 002014
1600 001032 002120
1601 001034 002224
1602 001036 002330
1603 001040 002434
1604 001042 002540
1605
1606
1607
1608
1609 001044
1610 001044 000000
1611 001046 000000
1612 001050 000000
1613 001052 000000
1614 001054 000000
1615 001056 000000
1616 001060 000000
1617 001062 000000
1618

;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

```
1619 ;UNIT WRITE ERRORS*****
1620
1621 001064 000000 WTER1: 0
1622 001066 000000 WTER2: 0
1623 001070 000000 WTER3: 0
1624 001072 000000 WTER4: 0
1625 001074 000000 WTER5: 0
1626 001076 000000 WTER6: 0
1627 001100 000000 WTER7: 0
1628 001102 000000 WTER8: 0
1629
1630 ;UNIT READ FORWARD ERRORS*****
1631
1632 001104 000000 RDER1: 0
1633 001106 000000 RDER2: 0
1634 001110 000000 RDER3: 0
1635 001112 000000 RDER4: 0
1636 001114 000000 RDER5: 0
1637 001116 000000 RDER6: 0
1638 001120 000000 RDER7: 0
1639 001122 000000 RDER8: 0
1640
1641 ;UNIT DATA ERRORS FORWARD*****
1642
1643 001124 000000 DATER1: 0
1644 001126 000000 0
1645 001130 000000 0
1646 001132 000000 0
1647 001134 000000 0
1648 001136 000000 0
1649 001140 000000 0
1650 001142 000000 0
1651
1652 ;UNIT READ REVERSE ERRORS*****
1653
1654 001144 000000 RDERR1: 0
1655 001146 000000 0
1656 001150 000000 0
1657 001152 000000 0
1658 001154 000000 0
1659 001156 000000 0
1660 001160 000000 0
1661 001162 000000 0
1662
1663 ;UNIT DATA ERRORS REVERSE*****
1664
1665 001164 000000 DEREV1: 0
1666 001166 000000 0
1667 001170 000000 0
1668 001172 000000 0
1669 001174 000000 0
1670 001176 000000 0
1671 001200 000000 0
1672 001202 000000 0
```

```
1673 ;DROPS + PICKS PER CHANNEL PER UNIT*****
1674
1675 001204 000000 BP00: 0
1676 001224 001224 .=.+16
1677 001224 000000 BP10: 0
1678 001244 001244 .=.+16
1679 001244 000000 BP20: 0
1680 001264 001264 .=.+16
1681 001264 000000 BP30: 0
1682 001304 001304 .=.+16
1683 001304 000000 BP40: 0
1684 001324 001324 .=.+16
1685 001324 000000 BP50: 0
1686 001344 001344 .=.+16
1687 001344 000000 BP60: 0
1688 001364 001364 .=.+16
1689 001364 000000 BP70: 0
1690 001404 001404 .=.+16
1691 001404 000000 BD00: 0
1692 001424 001424 .=.+16
1693 001424 000000 BD10: 0
1694 001444 001444 .=.+16
1695 001444 000000 BD20: 0
1696 001464 001464 .=.+16
1697 001464 000000 BD30: 0
1698 001504 001504 .=.+16
1699 001504 000000 BD40: 0
1700 001524 001524 .=.+16
1701 001524 000000 BD50: 0
1702 001544 001544 .=.+16
1703 001544 000000 BD60: 0
1704 001564 001564 .=.+16
1705 001564 000000 BD70: 0
1706 001604 001604 .=.+16
1707
1708
```

```
1709
1710 ;UNIT BAD TAPE COUNTER:16 PER SLAVE*****
1711
1712 001604 000000 BT00: 0
1713 001710 001710 .=.+102
1714 001710 000000 BT01: 0
1715 002014 002014 .=.+102
1716 002014 000000 BT02: 0
1717 002120 002120 .=.+102
1718 002120 000000 BT03: 0
1719 002224 002224 .=.+102
1720 002224 000000 BT04: 0
1721 002330 002330 .=.+102
1722 002330 000000 BT05: 0
1723 002434 002434 .=.+102
1724 002434 000000 BT06: 0
1725 002540 002540 .=.+102
1726 002540 000000 BT07: 0
1727 002644 002644 .=.+102
1728
1729 ;UNIT END OF TAPE COUNTERS 1 PER SLAVE*****
1730
1731 002644 000000 EOTCO: 0
1732 002646 000000 0
1733 002650 000000 0
1734 002652 000000 0
1735 002654 000000 0
1736 002656 000000 0
1737 002660 000000 0
1738 002662 000000 0
1739
1740 ;UNIT READ FORWARD SOFT ERROR*****
1741
1742 002664 000000 RFSOFT: 0
1743 002666 000000 0
1744 002670 000000 0
1745 002672 000000 0
1746 002674 000000 0
1747 002676 000000 0
1748 002700 000000 0
1749 002702 000000 0
1750
1751 ;UNIT READ REVERSE SOFT ERROR*****
1752
1753 002704 000000 RRSOFT: 0
1754 002706 000000 0
1755 002710 000000 0
1756 002712 000000 0
1757 002714 000000 0
1758 002716 000000 0
1759 002720 000000 0
1760 002722 000000 0
1761
```



```
1762
1763                                     ;UNIT READ FORWARD HARD ERROR*****
1764
1765 002724 000000 RFHARD: 0
1766 002726 000000 0
1767 002730 000000 0
1768 002732 000000 0
1769 002734 000000 0
1770 002736 000000 0
1771 002740 000000 0
1772 002742 000000 0
1773
1774                                     ;UNIT READ REVERSE HARD ERROR*****
1775
1776 002744 000000 RRHARD: 0
1777 002746 000000 0
1778 002750 000000 0
1779 002752 000000 0
1780 002754 000000 0
1781 002756 000000 0
1782 002760 000000 0
1783 002762 000000 0
1784                                     ;SET END OF STATISTICS TABLE
1785 002764 ENDTBL:
1786
1787                                     ;DATA PATTERN GENERATORS*****
1788
1789 002764 002764 DATBL: . ;ENTRY TABLE
1790 002766 013656 DATA0: DAT0 ;EXTERNAL INPUT FROM H/S READER(SEE MAINDEC-11-DZTUF)
1791 002770 014016 DATA1: DAT1 ;ALL ONES
1792 002772 014036 DATA2: DAT2 ;ALL ZEROS
1793 002774 014042 DATA3: DAT3 ;WALKING ONE
1794 002776 014066 DATA4: DAT4 ;WALKING ZERO
1795 003000 014076 DATA5: DAT5 ;ALTERNATING ONE/ZERO
1796 003002 014104 DATA6: DAT6 ;ALTERNATING ZERO/ONE
1797 003004 014112 DATA7: DAT7 ;ALTERNATING ONE/ZERO IN ALTERNATING CHARACTERS
1798 003006 014140 DATA10: DAT10 ;WALKING ONE/ALL ONE IN ALTERNATING CHARACTERS
1799 003010 014170 DATA11: DAT11 ;ALL BITS 0-377
1800 003012 014210 DATA12: DAT12 ;ALL BITS 377-0
1801 003014 014232 DATA13: DAT13 ;ALTERNATING CHARACTERS 0 AND 377
1802 003016 014242 DATA14: DAT14 ;WALKING ZERO/ALL ZERO IN ALTERNATING CHARACTERS
1803 003020 014272 DATA15: DAT15 ;AUTO SEQUENCE PATTERN 0,0,-1,-1,-1,0,0
1804
```

```
1805 .EVEN
1806 :*****
1807 :PROGRAM START AND SEQUENCE FORMATTER:
1808 :
1809 :THIS ROUTINE IS USED TO PERFORM ALL HOUSEKEEPING,
1810 :DECIDE WHICH TRANSPORT TO TEST AND ITS AVAILABILITY,
1811 :LOAD THE WRITE BUFFER WITH THE SELECTED DATA PATTERN,
1812 :GENERATE ANY RANDOM NUMBER AND THEN EXECUTE
1813 :THE TEST CYCLE REQUESTED BY THE SWITCH SETTING.
1814 :AT THE END OF THE TEST CYCLE THE NEXT UNIT IS SELECTED
1815 :AND CHECKED FOR AVAILABILITY AND THE TEST CYCLE IS
1816 :EXECUTED ON IT.
1817 :THE READ WRITE STATS MAY BE PRINTED AT THE END OF
1818 :EACH TEST CYCLE VIA CONSOLE SWITCH FOURTEEN (14).
1819 :*****
1820
1821
1822 ;START 200, & 300*****
1823 003022 012706 000500 START: MOV #500,SP ;SET STACK PTR
1824 003026 005037 000734 CLR ASEQF ;CLEAR AUTO SEQUENCE FLAG
1825 003032 005027 CLR (PC)+ ;:CLEAR CHAIN INDICATOR
1826 003034 000000 CHNFLG: .WORD 0 ;:CHAIN MODE INDICATOR
1827 ;:1/0 = CHAIN/NOT CHAIN MODE
1828 003036 005737 000042 TST @#42 ;:BRANCH IF IN DUMP MODE
1829 003042 001407 BEQ 50$
1830 003044 012737 000176 000610 MOV #SWREG,SWR ;:INVOKE SOFTWARE SWR
1831 003052 005237 003034 INC CHNFLG ;:SET CHNFLG = CHAIN MODE
1832 003056 000137 003062 JMP 3$ ;:GO TO CHAIN ADDRESS
1833 003062 50$:
1834 003062 122737 000006 000041 3$: CMPB #6,@#41 ;:BRANCH IF LOADED VIA TMDP
1835 003070 001003 BNE 4$
1836 003072 000004 026114 TYPE,MSG120 ;ADVISE USER TO REMOVE TMDP FROM SLAVE
1837 003076 000000 HALT
1838 003100 005737 003034 4$: TST CHNFLG ;SEE IF IN CHAIN MODE
1839 003104 001406 BEQ STAUT
1840 003106 005237 000734 INC ASEQF ;SET AUTO SEQUENCE FLAG
1841 003112 000004 024157 TYPE,MSG30 ;TYPE TITLE
1842 003116 000137 021226 JMP ASEQ0 ;GO TO AUTO SEQUENCER
1843
1844 ;START 240*****
1845 003122 012737 000001 000636 STAUT: MOV #1,TINF ;SET TTY ENTRY FLAG
1846 003130 005037 014404 CLR RDFL ;CLEAR RANDOM DATA FLAG
1847 003134 000405 BR STARTB
1848
1849 ;START 204*****
1850 003136 005037 000636 STARTC: CLR TINF ;CLEAR TTY INPUT FLAG
1851 003142 000442 BR STARTD
1852
1853 ;START 210*****
1854 003144 005037 000636 STARTA: CLR TINF ;CLEAR TTY ENTRY FLAG
1855 003150 012700 000640 STARTB: MOV #STFLG,R0 ;GET STARTING ADDRESS OF FLAGS
1856 003154 012701 000074 MOV #ENDFLG-STFLG,R1
1857 003160 105020 1$: CLR (R0)+ ;CLEAR FLAGS AND COUNTERS
1858 003162 005301 DEC R1
1859 003164 001375 BNE 1$
1860 003166 012706 000500 MOV #500,SP ;SET STACK POINTER
```

1861	003172	004737	004122			JSR	PC,RANSET		;GO RESET RANDOM BASE
1862	003176	012700	001044			MOV	#STTBL,R0		;GET STARTING ADDRESS OF STAT TABLE
1863	003202	012701	001720			MOV	#ENDTBL-STTBL,R1		;AND # OF BYTES IN TABLE
1864	003206	105020			2\$:	CLRB	(R0)+		;CLEAR STATISTIC COUNTERS
1865	003210	005301				DEC	R1		
1866	003212	001375				BNE	2\$		
1867	003214	012700	000742			MOV	#UN1,R0		;SET ALL SLAVES ON-LINE
1868	003220	022710	177777		3\$:	CMP	#-1,(R0)		;BRANCH IF AT END OF TABLE
1869	003224	001403				BEQ	4\$		
1870	003226	042720	040000			BIC	#40000,(R0)+		;MARK SLAVE ON-LINE
1871	003232	000772				BR	3\$		
1872	003234	012737	177777	013652	4\$:	MOV	#-1,PATS		;PRESET PATTERN
1873	003242	012737	000001	000654	STARTE:	MOV	#1,BLCNTR		;PRESET BLOCK COUNTER
1874	003250	013746	000004		STARTD:	MOV	@#4,-(SP)		;SAVE ERROR TRAP VECTOR
1875	003254	013746	000006			MOV	@#6,-(SP)		
1876	003260	022737	000176	000610		CMP	#SWREG,SWR		;BRANCH IF SOFTWARE SWR
1877	003266	001413				BEQ	2\$;ALREADY SELECTED
1878	003270	012737	003314	000004		MOV	#1\$,@#4		;SET TIMEOUT TRAP TO 1\$ BELOW
1879	003276	005037	000006			CLR	@#6		
1880	003302	022777	177777	175300		CMP	#177777,@SWR		;BRANCH IF SWR = 177777 TRAP
1881	003310	001402				BEQ	2\$;IF NOT AVAIL (1\$) OTHERWISE
1882	003312	000404				BR	3\$;GO TO 3\$
1883	003314	022626			1\$:	CMP	(SP)+,(SP)+		;RESET STACK
1884	003316	012737	000176	000610	2\$:	MOV	#SWREG,SWR		;SET SWR = SOFTWARE SWR
1885	003324	012637	000006		3\$:	MOV	(SP)+,@#6		;RESTORE ERROR TRAP
1886	003330	012637	000004			MOV	(SP)+,@#4		
1887	003334	012706	000500			MOV	#500,SP		
1888	003340	004737	011750			JSR	PC,TINP		;GO GET PARAMETERS FROM TTY
1889	003344	012777	000040	175146		MOV	#40,@CS		;INITIALIZE
1890	003352	005000			STAUTO:	CLR	R0		;POINT TO FIRST ENTRY
1891	003354	022760	177777	000742	1\$:	CMP	#-1,UN1(R0)		;BRANCH IF LAST ENTRY
1892	003362	001406				BEQ	2\$		
1893	003364	042760	100000	000742		BIC	#100000,UN1(R0)		;CLEAR EOT FLAG
1894	003372	062700	000002			ADD	#2,R0		;POINT TO NEXT UNIT ENTRY
1895	003376	000766				BR	1\$;CONTINUE CLEARING
1896	003400	113737	004731	004730	2\$:	MOVB	REOTC+1,REOTC		;RESTORE EOT COUNTER
1897	003406	012777	000100	175176	START1:	MOV	#100,@TKS		;SET KEYBOARD IE BIT
1898	003414	013700	000674			MOV	UNP,R0		;R0 = UNIT TABLE POINTER
1899	003420	022760	177777	000742	STAR1A:	CMP	#-1,UN1(R0)		;BRANCH IF LAST ENTRY
1900	003426	001404				BEQ	STAR1B		
1901	003430	016037	000742	000552		MOV	UN1(R0),UDES		;LOAD NEXT UNIT DESCRIPTION
1902	003436	000445				BR	START4		
1903	003440	005237	000654		STAR1B:	INC	BLCNTR		;BUMP BLOCK COUNTER
1904	003444	005737	000734			TST	ASEQF		;SEE IF AUTO SEQ
1905	003450	001411				BEQ	STAR1C		;IF NOT: BR
1906	003452	023737	000654	000736		CMP	BLCNTR,ABL CNT		;SEE IF DONE SEQ
1907	003460	001005				BNE	STAR1C		;IF NOT: BR
1908	003462	005037	000654			CLR	BLCNTR		;RESET BLOCK CNTR
1909	003466	005037	000674			CLR	UNP		;RESET UNIT POINTER
1910	003472	000207				RTS	PC		;RETURN TO AUTO SEQ
1911	003474	005037	000674		STAR1C:	CLR	UNP		
1912	003500	005000				CLR	R0		
1913	003502	016037	000742	000552		MOV	UN1(R0),UDES		;LOAD FIRST UNIT DESCRIPTION
1914	003510	105777	175074			TSTB	@SWR		;SEE IF RANDOM RECORD SIZE
1915	003514	100002				BPL	START2		;IF NOT: BR
1916	003516	004737	011664			JSR	PC,CCNTR		;GO GENERATE RANDOM RECORD SIZE

1917	003522	032777	000400	175060	START2:	BIT	#400,@SWR	:SEE IF RANDOM DATA
1918	003530	001402				BEQ	START3	:IF NOT: BR
1919	003532	004737	014342			JSR	PC,DATR	:GO GENERATE RANDOM DATA
1920	003536	032777	000100	175044	START3:	BIT	#100,@SWR	:SEE IF RANDOM RECORD COUNT
1921	003544	001402				BEQ	START4	:IF NOT: BR
1922	003546	004737	011724			JSR	PC,RCNTR	:GO GENERATE RANDOM RECORD COUNT
1923	003552	032760	140000	000742	START4:	BIT	#140000,UN1(R0)	:BRANCH IF UNIT AT EOT
1924	003560	001065				BNE	START7	:OR MARKED OFF-LINE
1925	003562	012777	000040	174730		MOV	#40,@CS	:DO A MASSBUS CLEAR
1926	003570	013777	000550	174722		MOV	DVN,@CS	:SET DRIVE NUMBER
1927	003576	013777	000552	174736		MOV	UDES,@TC	:SET SLAVE NUMBER
1928	003604	105777	174712		1\$:	TSTB	@DS	:SEE IF SLAVE AVAIL
1929	003610	100405				BMI	2\$:IF SO: BR
1930	003612	005337	000666			DEC	STAL	
1931	003616	001372				BNE	1\$:AWAIT TUR
1932	003620	000137	020312			JMP	OFFLINE	:GO MARK DRIVE OFF-LINE
1933	003624	004737	013472		2\$:	JSR	PC,DSUP	:GO SET UP WRITE DATA
1934	003630	004737	005236			JSR	PC,INIT	:INIT SLAVE
1935	003634	004737	004732			JSR	PC,RWND	:REWIND
1936	003640	004737	005352			JSR	PC,WRITE	:WRITE
1937	003644	013737	000600	000666		MOV	TSTAL,STAL	:SET TURN AROUND DELAY
1938	003652	004737	011654			JSR	PC,STALL	:DELAY
1939	003656	004737	007210			JSR	PC,RSEQ	:GO TO READ SEQUENCER
1940	003662	013737	000600	000666		MOV	TSTAL,STAL	:SET TURN AROUND DELAY
1941	003670	004737	011654			JSR	PC,STALL	:DELAY
1942	003674	032777	040000	174706		BIT	#40000,@SWR	:SEE IF SHOULD PRINT STATISTICS
1943	003702	001414				BEQ	START7	:IF NOT: BR
1944	003704	012700	000001			MOV	#1,R0	:SET RECORD COUNTER TO 1
1945	003710	004737	022012			JSR	PC,PAPRT	:PRINT CYCLE NUMBER
1946	003714	004737	003744			JSR	PC,STP	:GO PRINT STATS
1947	003720	005237	000726			INC	BTSTF	:SET STAT ONLY PRINT
1948	003724	004737	007126			JSR	PC,BTPRT	:PRINT BAD TAPE STATS
1949	003730	005037	000726			CLR	BTSTF	:CLEAR FLAG
1950	003734	062737	000002	000674	START7:	ADD	#2,UNP	:POINT TO NEXT UNIT
1951	003742	000621			START8:	BR	START1	:CONTINUE

1952
1953
1954 003744 004737 016370
1955 003750 000004 025123
1956 003754 013700 000674
1957 003760 016003 001044
1958 003764 104400
1959 003766 000004 025234
1960 003772 016003 001064
1961 003776 104400
1962 004000 000004 025223
1963 004004 016003 001104
1964 004010 104400
1965 004012 000004 026001
1966 004016 016003 002664
1967 004022 104400
1968 004024 000004 026012
1969 004030 016003 002724
1970 004034 104400
1971 004036 000004 025320
1972 004042 016003 001124
1973 004046 104400
1974 004050 000004 025155
1975 004054 016003 001144
1976 004060 104400
1977 004062 000004 026001
1978 004066 016003 002704
1979 004072 104400
1980 004074 000004 026012
1981 004100 016003 002744
1982 004104 104400
1983 004106 000004 025307
1984 004112 016003 001164
1985 004116 104400
1986 004120 000207
1987
1988
1989
1990 004122 012737 153624 000626
1991 004130 012737 032561 000630
1992 004136 013737 000632 000554
1993 004144 013737 000634 000556
1994 004152 000207
1995

***** SUBROUTINE TO PRINT STATISTICS *****

STP: JSR PC,DPPRT ;PRINT DROPS AND PICKS
TYPE,MSG65 ;TYPE MSG
MOV UNP,R0
MOV RTY1(R0),R3
TYPOCT ;PRINT RETRIES
TYPE,MSG73 ;TYPE MSG
MOV WTER1(R0),R3
TYPOCT ;PRINT WRITE ERRORS
TYPE,MSG72 ;TYPE MSG
MOV RDER1(R0),R3
TYPOCT ;PRINT READ FORWARD ERRORS
TYPE,MSG113 ;TYPE MSG
MOV RFSOFT(R0),R3
TYPOCT ;PRINT FORWARD SOFT ERRORS
TYPE,MSG114 ;TYPE MSG
MOV RFHARD(R0),R3
TYPOCT ;PRINT HARD FORWARE ERRORS
TYPE,MSG77 ;TYPE MSG
MOV DATER1(R0),R3
TYPOCT ;PRINT DATA ERROR FORWARD NUMBER
TYPE,MSG68 ;TYPE MSG
MOV RDERR1(R0),R3
TYPOCT ;PRINT REVESE ERROR NUMBER
TYPE,MSG113 ;TYPE MSG
MOV RRSOFT(R0),R3
TYPOCT ;PRINT REVERSE SOFT ERROR
TYPE,MSG114 ;TYPE MSG
MOV RRHARD(R0),R3
TYPOCT
TYPE,MSG76 ;TYPE MSG
MOV DEREV1(R0),R3
TYPOCT ;PRINT DATA REVERSE ERROR NUMBER
RTS PC ;RETURN

***** RANDOM BASE RESET*****

RANSET: MOV #153624,RANBAS ;RESET BASE
MOV #32561,RANSAV ;RESET BUFFER
MOV RCSAV,RCNT ;RESET RECORD COUNT
MOV FCSAV,FMCNT ;RESET FRAME COUNT
RTS PC

```
1996
1997
1998
1999
2000
2001
2002
2003
2004
2005
2006
2007
2008 004154 013777 000552 174360 REOT: MOV UDES,@TC ;LOAD TAPE CONTROL REGISTER
2009 004162 013700 000674 MOV UNP,R0 ;GET UNIT POINTER
2010 004166 032760 040000 000742 BIT #40000,UN1(R0) ;BRANCH IF UNIT MARKED OFF-LINE
2011 004174 001014 BNE 2$
2012 004176 012777 000011 174304 MOV #11,@C1 ;DRIVE CLEAR
2013 004204 105777 174312 1$: TSTB @DS ;WAIT FOR DRY
2014 004210 100375 RPL 1$
2015 004212 012777 000007 174270 MOV #7,@C1 ;START REWIND
2016 004220 005737 000724 TST BTFLG ;SEE IF BAD TAPE OVERFLOW REWIND
2017 004224 001004 BNE 3$ ;IF SO: BR
2018 004226 013700 000660 2$: MOV EOTREC,R0
2019 004232 042700 100000 BIC #100000,R0 ;SET RECORD NUMBER OF EOT
2020 004236 005037 000660 3$: CLR EOTREC ;CLEAR EOT INDICATOR & REC COUNT
2021 004242 004737 022012 JSR PC,PAPRT ;PRINT HEADER
2022 004246 022737 000002 000724 CMP #2,@TFLG ;SEE IF POSITION ERROR
2023 004254 001004 BNE 4$ ;IF NOT: BR
2024 004256 012737 025674 004306 MOV #MSG109,6$ ;SET POSITION ERROR MSG
2025 004264 000407 BR 5$
2026 004266 022737 000001 000724 4$: CMP #1,BTFLG ;SEE IF BAD TAPE OVERFLOW
2027 004274 001006 BNE REOT1C ;IF NOT: BR
2028 004276 012737 025527 004306 MOV #MSG106,6$ ;SET BAD TAPE OVERFLOW MSG
2029 004304 000004 5$: TYPE ;TYPE MSG
2030 004306 000000 6$: .WORD 0 ;WILL CONTAIN MESSAGE ADDRESS
2031 004310 000411 BR REOT1E
2032 004312 000004 023660 REOT1C: TYPE,MSG20 ;TYPE EOT MSG
2033 004316 013704 000674 MOV UNP,R4
2034 004322 005264 002644 INC EOTC0(R4) ;BUMP CNTR
2035 004326 016403 002644 MOV EOTC0(R4),R3
2036 004332 104400 TYPOCT ;PRINT EOT CNTR
2037 004334 000004 025552 REOT1E: TYPE,MSG16A ;TYPE MSG
2038 004340 005037 000724 CLR BTFLG ;CLEAR BAD TAPE FLAG
2039 004344 004737 003744 JSR PC,STP ;PRINT STATS
2040 004350 004737 007126 JSR PC,BTPRT ;PRINT BAD TAPE STATS
2041 004354 013700 000674 REOT2: MOV UNP,R0 ;GET UNIT POINTER
2042 004360 032760 040000 000742 BIT #40000,UN1(R0) ;BRANCH IF UNIT MARKED OFF-LINE
2043 004366 001010 BNE REOT2A
2044 004370 105777 174126 TSTB @DS ;BRANCH IF DRY SET
2045 004374 100405 BMI REOT2A
2046 004376 005337 000666 DEC STAL
2047 004402 001364 BNE REOT2 ;WAIT DRY
2048 004404 000137 020312 JMP OFFLINE ;GO MARK SLAVE OFFLINE
2049
2050 004410 105337 004730 REOT2A: DECB REOTC ;SEE IF LAST UNIT TO REACH EOT
2051 004414 001410 BEQ REOT3 ;IF SO: BR
```

```

2052 004416 013700 000674      MOV      UNP,RO
2053 004422 052760 100000 000742      BIS      #100000,UN1(RO) ;SET EOT FLAG
2054 004430 005726      TST      (SP)+          ;RESET STACK POINTER
2055 004432 000137 003734      JMP      START7        ;GO TO NEXT UNIT
2056 004436 113737 004731 004730 REOT3:  MOVB    REOTC+1,REOTC   ;RESTORE UNITS EOT COUNTER
2057 004444 005037 000674      CLR      UNP
2058 004450 005000      CLR      RO            ;POINT TO FIRST UNIT
2059 004452 016037 000742 000552 REOT4:  MOV      UN1(RO),UDES   ;LOAD UNIT DESCRIPTION
2060 004460 013777 000552 174054      MOV      UDES,@TC      ;SELECT SLAVE
2061 004466 032760 040000 000742      BIT      #40000,UN1(RO) ;BRANCH IF UNIT NOT MARKED OFF-LINE
2062 004474 001412      BEQ      1$
2063 004476 032777 010000 174016      BIT      #10000,@DS    ;BRANCH IF MEDIUM NOT ON LINE
2064 004504 001427      BEQ      10$
2065 004506 062737 000401 004730      ADD      #401,REOTC    ;INCREMENT # OF UNITS UNDER TEST
2066 004514 042760 140000 000742      BIC      #140000,UN1(RO) ;MARK UNIT BACK ON-LINE
2067 004522 012777 000011 173760 1$:      MOV      #11,@C1      ;DRIVE CLEAR
2068 004530 105777 173766 2$:      TSTB    @DS          ;WAIT FOR DRIVE READY
2069 004534 100375      BPL      2$
2070 004536 012777 000007 173744      MOV      #7,@C1       ;REWIND UNIT
2071 004544 032777 000002 173750 3$:      BIT      #2,@DS       ;WAIT FOR BOT TO SET
2072 004552 001774      BEQ      3$
2073 004554 032777 020000 173740 4$:      BIT      #20000,@DS   ;WAIT FOR PIP TO CLEAR
2074 004562 001374      BNE      4$           ;AWAIT PIP RESET
2075
2076 004564 042760 100000 000742 10$:     BIC      #100000,UN1(RO) ;CLEAR EOT FLAG
2077 004572 062737 000002 000674      ADD      #2,UNP
2078 004600 013700 000674      MOV      UNP,RO       ;POINT TO NEXT UNIT
2079 004604 022760 177777 000742      CMP      #-1,UN1(RO)  ;BRANCH IF NOT LAST UNIT
2080 004612 001317      BNE      REOT4
2081 004614 005037 000674      REOT7:  CLR      UNP          ;CLEAR UNIT POINTER
2082 004620 005037 000636      CLR      TINF         ;CLEAR TTY INPUT FLAG
2083 004624 005737 000734      TST      ASEQF        ;SEE IF AUTO SEQ
2084 004630 001402      BEQ      REOTX        ;IF NOT: BR
2085 004632 005726      TST      (SP)+        ;RESET STACK POINTER
2086 004634 000207      RTS      PC           ;RETURN TO AUTO SEQ
2087 004636 004737 004122 013652 REOTX:  JSR      PC,RANSET   ;GO RESET RANDOM BASE
2088 004642 012737 177777 013652      MOV      #-1,PATS     ;PRESET PATTERN
2089 004650 005037 014404      CLR      RDFL        ;CLEAR RANDOM FLAG
2090 004654 005737 000572      TST      SPFLG       ;SEE IF SINGLE PASS
2091 004660 001421      BEQ      REOTXX      ;IF NOT: BR
2092 004662 000004 025430 TEND:   TYPE    MSG100   ;TYPE MSG
2093 004666 013700 000042      MOV      @#42,RO     ;GET ACT11 RETURN ADDRESS
2094 004672 001405      BEQ      HERE        ;BRANCH IF NOT ACT11
2095 004674 000005      RESET
2096 004676 004710  SENDAD: JSR      PC,(RO)
2097 004700 000240      NOP
2098 004702 000240      NOP
2099 004704 000240      NOP
2100 004706 000240  HERE:   NOP
2101 004710 005737 003034      TST      CHNFLG      ;BRANCH IF NOT CHAIN MODE
2102 004714 001402      BEQ      1$
2103 004716 000137 021226      JMP      ASEQO       ;RETURN TO AUTO SEQUENCER
2104 004722 000000 1$:      HALT
2105 004724 000137 003242 REOTXX: JMP      STARTE      ;RESTART AT BLOCK NUMBER ONE
2106 004730 000000 REOTC:  0            ;EOT UNIT COUNTER
  
```

```
2107 ;*****
2108 ;REWIND ALL AVAIL TAPES:
2109 ;
2110 ;THIS ROUTINE; ENTERED VIA CONSOLE SWITCH NINE (9),
2111 ;WILL REWIND ALL AVAILABLE TAPES TO BOT NO MATTER
2112 ;WHERE THEY ARE CURRENTLY POSITIONED AND RESUME TESTING
2113 ;ON THE CURRENTLY SELECTED UNIT.
2114 ;*****
2115
2116 004732 032777 001000 173650 RWND: BIT #1000,@SWR ;SEE IF SHOULD REWIND
2117 004740 001001 BNE RWNDA ;IF SO: BR
2118 004742 000207 RTS PC ;ELSE EXIT
2119 004744 013737 000674 000714 RWNDA: MOV UNP,UPS ;SAVE UNIT POINTER
2120 004752 005037 000674 CLR UNP ;CLEAR POINTER
2121 004756 005037 000660 CLR EOTREC ;CLEAR EOT FLAG
2122 004762 113737 004731 004730 MOVB REOTC+1,REOTC ;++B RESTORE UNIT CTR
2123 004770 013700 000674 RWND0: MOV UNP,RO ;POINT TO UNIT ENTRY
2124 004774 022760 177777 000742 CMP #-1,UN1(RO) ;BRANCH IF LAST ENTRY
2125 005002 001437 BEQ RWND2
2126 005004 032760 140000 000742 BIT #140000,UN1(RO) ;BRANCH IF ALREADY REWINDING
2127 005012 001024 BNE RWND1A ;OR MARKED OFF LINE
2128 005014 016037 000742 000552 MOV UN1(RO),UDES ;SET UNIT DESCRIPTION
2129 005022 013777 000552 173512 MOV UDES,@TC ;LOAD COMMAND REGISTER
2130 005030 012777 000011 173452 MOV #11,@C1 ;DRIVE CLEAR
2131 005036 012777 000007 173444 MOV #7,@C1 ;START REWIND
2132 005044 105777 173452 1$: TSTB @DS
2133 005050 100405 BMI RWND1A ;IF DRY: BR
2134 005052 005337 000666 DEC STAL
2135 005056 001372 BNE 1$ ;AWAIT DRY
2136 005060 000137 020312 JMP OFFLINE ;GO MARK UNIT OFF LINE
2137 005064 042760 100000 000742 RWND1A: BIC #100000,UN1(RO) ;CLEAR EOT FLAG
2138 005072 062737 000002 000674 ADD #2,UNP ;BUMP POINTER
2139 005100 000733 BR RWND0 ;DO NEXT UNIT
2140 005102 005037 000674 RWND2: CLR UNP ;CLEAR POINTER
2141 005106 013700 000674 RWND3: MOV UNP,RO ;POINT TO UNIT ENTRY
2142 005112 022760 177777 000742 CMP #-1,UN1(RO) ;BRANCH IF LAST ENTRY
2143 005120 001433 BEQ RWNDX
2144 005122 016037 000742 000552 MOV UN1(RO),UDES ;SET UNIT DESCRIPTION
2145 005130 032760 040000 000742 BIT #40000,UN1(RO) ;BRANCH IF UNIT MARKED OFF LINE
2146 005136 001015 BNE RWND5
2147 005140 013777 000552 173374 MOV UDES,@TC ;LOAD UNIT DESCRIPTION
2148 005146 032777 020000 173346 1$: BIT #20000,@DS
2149 005154 001374 BNE 1$ ;AWAIT PIP RESET
2150 005156 032777 000002 173336 BIT #2,@DS ;BRANCH IF SLAVE AT BOT
2151 005164 001002 BNE RWND5
2152 005166 000137 020312 JMP OFFLINE ;PRINT OFFLINE MESSAGE
2153 005172 062737 000002 000674 RWND5: ADD #2,UNP ;BUMP POINTER
2154 005200 012777 000011 173302 MOV #11,@C1 ;DRIVE CLEAR
2155 005206 000737 BR RWND3 ;DO NEXT UNIT
2156
2157 005210 013700 000714 RWNDX: MOV UPS,RO ;RESTORE UNIT POINTER
2158 005214 010037 000674 MOV RO,UNP
2159 005220 016037 000742 000552 MOV UN1(RO),UDES ;RESET UNIT DESCRIPTION
2160 005226 013777 000552 173306 MOV UDES,@TC
2161 005234 000207 RTS PC ;RETURN TO TEST
2162
```


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
2216
2217
2218
2219
2220
2221
2222

```
*****  
: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).  
*****
```

```
2223 005352 032777 000001 173230 WRITE: BIT #1,@SWR ;SEE IF SHOULD WRITE  
2224 005360 001402 BEQ WRTE  
2225 005362 000137 006132 JMP WEX ;IF NOT: BR  
2226 005366 013700 000554 WRTE: MOV RCNT,RO ;RO=RECORD COUNT  
2227 005372 012737 023546 000652 WO: MOV #MSG5,EMADDR ;SET ERROR MSG ADDRESS  
2228 005400 013777 000556 173110 MOV FMCNT,@FC ;LOAD CHAR COUNT  
2229 005406 012777 026344 173100 MOV #WDATA,@BA ;SET DATA ADDR  
2230 005414 112737 000060 000672 MOVB #60,MTC1 ;SET WRITE OP COMMAND  
2231 005422 012737 005434 000662 MOV #W1,RTRN ;SET RETURN ADDRESS  
2232 005430 000137 020372 JMP TAPG ;GO EXECUTE COMMAND  
2233 005434 032777 002000 173060 W1: BIT #2000,@DS ;SEE IF EOT  
2234 005442 001412 BEQ 1$ ;IF NOT AT EOT: BR  
2235 005444 005737 000660 TST EOTREC ;BRANCH IF WRITTEN PAST EOT  
2236 005450 100407 BMI 1$  
2237 005452 005300 DEC RO ;ADJUST # OF RECORDS WRITTEN  
2238 005454 052700 100000 BIS #100000,RO ;SET EOT INDICATOR  
2239 005460 010037 000660 MOV RO,EOTREC ;SAVE RECORD COUNT  
2240 005464 012700 000002 MOV #2,RO ;SET TO WRITE 1 LAST RECORD  
2241 005470 032777 010000 173112 1$: BIT #10000,@SWR ;SEE IF SHOULD CHECK ERRORS  
2242 005476 001002 BNE 2$ ;IF NOT: BR  
2243 005500 004737 016522 JSR PC,ERCHK ;GO CHECK ERRORS  
2244 005504 013737 000576 000666 2$: MOV WSTAL,STAL ;SET DELAY  
2245 005512 004737 011654 JSR PC,STALL ;DELAY
```

2246	005516	005737	000712		TST	RTYFL		:SEE IF RETRY TIME
2247	005522	001401			BEQ	3\$:IF NOT: BR
2248	005524	000207			RTS	PC		:ELSE RETURN
2249	005526	005737	000706	3\$:	TST	SERFL		:SEE IF WRITE ERROR
2250	005532	001446			BEQ	W5		:IF NOT: BR
2251	005534	013704	000674		MOV	UNP,R4		
2252	005540	005264	001064		INC	WTER1(R4)		:BUMP WRITE ERROR
2253	005544	005037	000706		CLR	SERFL		:CLEAR STATUS ERROR FLAG
2254	005550	032777	000020	173032	BIT	#20,@SWR		:SEE IF RETRY
2255	005556	001434			BEQ	W5		:IF NOT: BR
2256	005560	013703	000722		MOV	ERSAV,R3		
2257	005564	042703	102720		BIC	#102720,R3		:MASK UNRECOVERABLE ERROR
2258	005570	001407			BEQ	W4		:IF SO: BR
2259	005572	004737	022012		JSR	PC,PAPRT		:PRINT HEADER
2260	005576	000004	025331		TYPE,MSG78			:TYPE MSG
2261	005602	004737	010754		JSR	PC,NRTP		:PRINT ER FOR NON-RETRYABLE
2262	005606	000420			BR	W5		
2263	005610	013704	000674	W4:	MOV	UNP,R4		
2264	005614	005264	001044		INC	RTY1(R4)		:BUMP RETRY CNTR
2265	005620	032777	002000	172762	BIT	#2000,@SWR		:SEE IF PRINT ERRORS
2266	005626	001002			BNE	W4A		:IF NOT: BR
2267	005630	000004	025101		TYPE,MSG64			:TYPE MSG
2268	005634	005037	000702	W4A:	CLR	RTCNT		:CLEAR RETRY NUMBER
2269	005640	005037	000700		CLR	RPCNT		:CLEAR REPEAT COUNTER
2270	005644	004737	006166		JSR	PC,WRTY		:GO RETRY WRITE ERROR
2271	005650	005037	000712	W5:	CLR	RTYFL		:CLEAR RETRY COUNTER
2272	005654	005300			DEC	RO		:SEE IF DONE ALL
2273	005656	001245			BNE	W0		:IF NOT: BR
2274	005660	005737	000564	W6:	TST	TMEX		:SEE IF TM
2275	005664	001522			BEQ	WEX		:IF NOT: BR
2276	005666	005237	000676		INC	TMFLG		:SET TM FLAG
2277	005672	012737	025012	000652	MOV	#MSG54,EMADDR		:POINT TO TM ERROR MSG
2278	005700	012737	000026	000672	MOV	#26,MTC1		:SET TM OP CODE
2279	005706	005077	172604		CLR	@FC		:LOAD FRAME COUNTER
2280	005712	012777	026344	172574	MOV	#WDATA,@BA		:LOAD BUS ADDRESS
2281	005720	012737	005732	000662	MOV	#WTMO,RTRN		:SAVE RETURN ADDRESS
2282	005726	000137	020372		JMP	TAPG		:WRITE TM
2283	005732	032777	010000	172650	WTMO:	BIT	#10000,@SWR	:SEE IF SHOULD CHECK ERRORS
2284	005740	001074			BNE	WEX		
2285	005742	032777	000004	172552	BIT	#4,@DS		:SEE IF TM STATUS
2286	005750	001011			BNE	WTM1		:IF SO: BR
2287	005752	012737	026344	020224	MOV	#WDATA,CADER		:SET EXPT BUS ADDRESS
2288	005760	012737	000001	020232	MOV	#1,DRVER		:INDICATE ERROR
2289	005766	004737	017352		JSR	PC,ERPT		:PRINT TM ERROR
2290	005772	000404			BR	WTM2		
2291	005774	012703	026344	WTM1:	MOV	#WDATA,R3		:SET EXPT ADDRESS
2292	006000	004737	016614		JSR	PC,ER2		:GO CHECK FOR OTHER ERRORS
2293	006004	005737	000712	WTM2:	TST	RTYFL		:SEE IF RETRY
2294	006010	001401			BEQ	WTM3		:IF NOT: BR
2295	006012	000207			RTS	PC		:ELSE RETURN TO RETRY ROUTINE
2296	006014	005737	000706	WTM3:	TST	SERFL		:SEE IF WRITE ERROR
2297	006020	001444			BEQ	WEX		:IF NOT: BR
2298	006022	013704	000674		MOV	UNP,R4		
2299	006026	005264	001064		INC	WTER1(R4)		:BUMP WRITE ERROR
2300	006032	032777	000020	172550	BIT	#20,@SWR		:SEE IF SHOULD RETRY
2301	006040	001434			BEQ	WEX		:IF NOT: BR

2302	006042	013703	000722		MOV	ERSAV,R3	
2303	006046	042703	102720		BIC	#102720,R3	;MASK UNRECOVERABLE ERROR
2304	006052	001407			BEQ	WTM4	;IF SO: BR
2305	006054	004737	022012		JSR	PC,PAPRT	;PRINT HEADER
2306	006060	000004	025331		TYPE,MSG78		;TYPE MSG
2307	006064	004737	010754		JSR	PC,NRTP	;PRINT ER FOR NON-RETRYABLE
2308	006070	000420			BR	WEX	
2309	006072	005037	000700	WTM4:	CLR	RPCNT	;CLEAR REPEAT CNTR
2310	006076	013704	000674		MOV	UNP,R4	
2311	006102	005264	001044		INC	RTY1(R4)	;BUMP RETRY CNTR
2312	006106	005037	000702		CLR	RTCNT	;CLEAR RETRY CNTR
2313	006112	032777	002000	172470	BIT	#2000,@SWR	;SEE IF PRINT ERRORS
2314	006120	001002			BNE	WTM4A	;IF NOT: BR
2315	006122	000004	025101		TYPE,MSG64		;TYPE MSG
2316	006126	004737	006166	WTM4A:	JSR	PC,WRTY	;GO DO RETRY
2317	006132	005037	000712	WEX:	CLR	RTYFL	;CLEAR RETRY FLAG
2318	006136	005037	000676		CLR	TMFLG	;CLEAR TAPE MARK FLAG
2319	006142	005737	000660		TST	EOTREC	;BRANCH IF NOT AT EOT
2320	006146	100006			BPL	WRWX	
2321	006150	032777	000014	172432	WRW:	BIT	#14,@SWR
2322	006156	001002			BNE	WRWX	;BRANCH IF EITHER READ ENABLED
2323	006160	000137	004154		JMP	REOT	;ELSE REWIND
2324	006164	000207		WRWX:	RTS	PC	;EXIT

```
2325
2326
2327
2328
2329
2330 006166 012737 000001 000712 WRTY: MOV #1,RTYFL ;SET RETRY FLAG
2331 006174 004737 006554 WRTY0: JSR PC,WRTSB ;GO SPACE REVERSE FOR REPEAT
2332 006200 005737 000676 TST TMFLG ;SEE IF TAPE MARK TIME
2333 006204 001003 BNE WRTYTM ;IF SO: BR
2334 006206 004737 005372 JSR PC,W0 ;REWRITE RECORD
2335 006212 000402 BR WRTYR ;GO ON
2336 006214 004737 005672 WRTYTM: JSR PC,WTM ;GO WRITE TAPE MARK AGAIN
2337 006220 005737 000706 WRTYR: TST SERFL ;REWRITE GOOD
2338 006224 001022 BNE WRTY2 ;IF NOT: BR
2339 006226 005237 000700 INC RPCNT ;BUMP REPEAT COUNTER
2340 006232 022737 000004 000700 CMP #4,RPCNT ;SEE IF FOUR GOOD REPEATS
2341 006240 001355 BNE WRTY0 ;IF NOT: REPEAT
2342 006242 032777 002000 172340 BIT #2000,@SWR ;SEE IF PRINT
2343 006250 001007 BNE WRTY1 ;IF NOT: BR
2344 006252 000004 025514 TYPE,MSG105 ;TYPE MSG
2345 006256 000004 025123 TYPE,MSG65 ;TYPE MSG
2346 006262 013703 000702 MOV RTCNT,R3
2347 006266 104400 TYPOCT ;PRINT RETRY NUMBER
2348 006270 000207 WRTY1: RTS PC ;RESUME TESTING
2349 006272 013703 000722 WRTY2: MOV ERSV,R3 ;GET ER
2350 006276 005037 000650 CLR TEMP3 ;CLEAR RECOVERABLE ERROR INDICATOR
2351 006302 042703 102720 BIC #102720,R3 ;MASK RECOVERABLE BITS
2352 006306 001412 BEQ WRTY2A ;IF RECOVERABLE: BR
2353 006310 004737 022012 JSR PC,PAPRT ;PRINT HEADER
2354 006314 000004 025331 TYPE,MSG78 ;TYPE MSG
2355 006320 004737 010754 JSR PC,NRTP ;PRINT ER
2356 006324 012737 000001 000650 MOV #1,TEMP3 ;SET FLAG
2357 006332 000406 BR WRTY2B
2358 006334 032777 002000 172246 WRTY2A: BIT #2000,@SWR ;SEE IF PRINT
2359 006342 001022 BNE WRTY3 ;IF NOT: BR
2360 006344 000004 025724 TYPE,MSG110 ;TYPE MSG
2361 006350 000004 025123 WRTY2B: TYPE,MSG65 ;TYPE MSG
2362 006354 013703 000702 MOV RTCNT,R3
2363 006360 104400 TYPOCT ;PRINT RETRY NUMBER
2364 006362 000004 025746 TYPE,MSG111 ;TYPE MSG
2365 006366 013703 000700 MOV RPCNT,R3
2366 006372 104400 TYPOCT ;PRINT REPEAT NUMBER
2367 006374 005737 000650 TST TEMP3 ;SEE IF DID NON-RECOVERABLE
2368 006400 001403 BEQ WRTY3 ;IF NOT: BR
2369 006402 005037 000650 CLR TEMP3 ;CLEAR FLAG
2370 006406 000207 RTS PC ;EXIT
2371 006410 005737 000702 WRTY3: TST RTCNT ;SEE IF FIRST RETRY
2372 006414 001004 BNE WRTY3A ;IF NOT: BR
2373 006416 013704 000674 MOV UNP,R4
2374 006422 005364 001064 DEC WTER1(R4) ;DECREMENT WRITE ERROR CNTR
2375 006426 013704 000674 WRTY3A: MOV UNP,R4 ;GET UNIT NUMBER
2376 006432 016437 001024 000730 MOV BTADDR(R4),BTPT ;GET ADDRESS OF UNIT BAD TAPE CNTR
2377 006440 017704 172264 MOV @BTPT,R4 ;GET COUNTER
2378 006444 005724 TST (R4)+ ;SET POINTER OFFSET
2379 006446 010477 172256 MOV R4,@BTPT
2380 006452 013703 000730 MOV BTPT,R3
```



```
2471
2472
2473
2474 007126 000004 024155
2475 007132 013704 000674
2476 007136 016437 001024 000730
2477 007144 017703 171560
2478 007150 000241
2479 007152 006003
2480 007154 104400
2481 007156 000004 025760
2482 007162 005777 171542
2483 007166 001001
2484 007170 000207
2485 007172 000137 007012
2486
2487
2488
2489 007176 004737 022012
2490 007202 000004 025613
2491 007206 000207
2492
```

;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
TYOCT ;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
```



```
2493 :*****
2494 :READ SEQUENCER:
2495 :
2496 :THIS ROUTINE IS USED TO DETERMINE THE SEQUENCE
2497 :IN WHICH READ TAPE OPERATIONS ARE TO BE PERFORMED.
2498 :THIS IS NECESSARY WHEN THE UNIT BEING TESTED IS
2499 :CAPABLE OF READING DATA IN BOTH THE FORWARD AND
2500 :REVERSE DIRECTIONS. CONSOLE SWITCHES ONE (1), TWO (2),
2501 :AND THREE (3) ARE USED TO DETERMINE THE READ SEQUENCE.
2502 :CONSOLE SWITCH ONE (1) DETERMINES WHETHER TO READ
2503 :THE BLOCK OF DATA FORWARD FIRST OR REVERSE FIRST.
2504 :SWITCH TWO (2) DISALLOWS READING IN THE REVERSE
2505 :DIRECTION AND SWITCH THREE (3) DISALLOWS READING IN
2506 :THE FORWARD DIRECTION.
2507 :*****
2508
2509 007210 005037 000562 RSEQ: CLR RDCMD
2510 007214 017704 171370 MOV @SWR,R4 ;READ SWITCHES
2511 007220 042704 177763 BIC #177763,R4 ;MASK READ BITS & SEE IF BOTH READS
2512 007224 001004 BNE RSR ;IF NOT: BR
2513 007226 032777 000002 171354 BIT #2,@SWR ;SEE IF READ REVERSE FIRST
2514 007234 001041 BNE RSFR ;IF NOT: BR
2515 007236 032777 000004 171344 RSR: BIT #4,@SWR ;SEE IF SHOULD READ REVERSE
2516 007244 001005 BNE RSF ;IF NOT: BR
2517 007246 012737 000001 000562 MOV #1,RDCMD ;LOAD READ REVERSE COMMAND
2518 007254 004737 007464 JSR PC,READ ;GO READ REVERSE
2519 007260 032777 000010 171322 RSF: BIT #10,@SWR ;SEE IF SHOULD READ FORWARD
2520 007266 001066 BNE RSEX ;IF NOT: BR
2521 007270 005737 000562 TST RDCMD ;SEE IF HAVE READ REVERSE
2522 007274 001406 BEQ RSFO ;IF NOT: BR
2523 007276 013737 000600 000666 MOV TSTAL,STAL
2524 007304 004737 011654 JSR PC,STALL ;DO READ STALL
2525 007310 000406 BR RSF1
2526 007312 032777 000001 171270 RSFO: BIT #1,@SWR ;SEE IF WRITE
2527 007320 001002 BNE RSF1 ;IF NOT: BR
2528 007322 004737 011402 JSR PC,BKSP ;GO BACKSPACE
2529 007326 005037 000562 RSF1: CLR RDCMD ;LOAD READ FORWARD COMMAND
2530 007332 004737 007464 JSR PC,READ ;GO READ
2531 007336 000442 BR RSEX ;GO TO EXIT
2532
2533 007340 012737 000001 000562 RSFR: MOV #1,RDCMD
2534 007346 032777 000010 171234 BIT #10,@SWR ;SEE IF SHOULD READ FORWARD
2535 007354 001012 BNE RSFR1 ;IF NOT: BR
2536 007356 032777 000001 171224 BIT #1,@SWR ;SEE IF WRITE
2537 007364 001002 BNE RSFR0 ;IF NOT: BR
2538 007366 004737 011402 JSR PC,BKSP ;GO BACKSPACE TO START
2539 007372 005037 000562 RSFR0: CLR RDCMD ;LOAD READ FORWARD COMMAND
2540 007376 004737 007464 JSR PC,READ ;GO READ FORWARD
2541 007402 032777 000004 171200 RSFR1: BIT #4,@SWR ;SEE IF SHOULD READ REVERSE
2542 007410 001015 BNE RSEX ;IF NOT: BR
2543 007412 005737 000562 TST RDCMD
2544 007416 001005 BNE RSFR2 ;IF READ REVERSE: BR
2545 007420 013737 000600 000666 MOV TSTAL,STAL ;DO READ STALL
2546 007426 004737 011654 JSR PC,STALL
2547 007432 012737 000001 000562 RSFR2: MOV #1,RDCMD ;LOAD READ REVERSE
2548 007440 004737 007464 JSR PC,READ ;GO READ REVERSE
```

CZTEDCO TM03-TE16/TU77 DRT
CZTEDC.P11 10-MAY-79 10:32

MACY11 30A(1052) 10-MAY-79 10:46 F 5 PAGE 57

SEQ 0057

2549 007444 005037 000562
2550 007450 005737 000660
2551 007454 001402
2552 007456 000137 004154
2553 007462 000207
2554

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

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

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

```
*****  
: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).  
*****
```

```
2581 007464 013700 000554 READ: MOV RCNT,R0 ;LOAD REC CNTR  
2582 007470 005737 000660 TST EOTREC ;SEE IF EOT  
2583 007474 100012 BPL RDA ;IF NOT: BR  
2584 007476 005737 000562 TST RDCMD ;SEE IF READ FORWARD  
2585 007502 001407 BEQ RDA ;IF SO: BR  
2586 007504 042737 100000 000660 BIC #100000,EOTREC ;CLEAR FLAG  
2587 007512 013703 000660 MOV EOTREC,R3 ;GET MODIFIED RECORD COUNT  
2588 007516 160300 SUB R3,R0 ;SET RECORD AT  
2589 007520 005200 INC R0 ;SET TO PROPER NUMBER OF RECORDS  
2590 007522 012737 023553 000652 RDA: MOV #MSG6,EMADDR ;SET ERROR MSG ADDRESS  
2591 007530 005037 000676 CLR TMFLG  
2592 007534 005737 000562 TST RDCMD  
2593 007540 001406 BEQ RDO ;IF READ FORWARD: BR  
2594 007542 005737 000564 TST TMEX ;SEE IF TM  
2595 007546 001403 BEQ RDO ;IF NOT: BR  
2596 007550 005237 000676 INC TMFLG ;SET TM FLAG  
2597 007554 005200 INC R0  
2598 007556 013777 000556 170732 RDO: MOV FMCNT,@FC ;LOAD CHAR CNTR  
2599 007564 012777 032352 170722 MOV #RDATA,@BA ;LOAD DATA ADDR  
2600 007572 005737 000562 TST RDCMD ;SEE IF READ REVERSE  
2601 007576 001417 BEQ RD1A ;IF NOT: BR  
2602 007600 013703 000556 MOV FMCNT,R3  
2603 007604 005103 COM R3  
2604 007606 032737 000020 000552 BIT #20,UDES ;SEE IF CORE DUMP  
2605 007614 001402 BEQ RD1 ;IF NOT: BR  
2606 007616 000241 CLC  
2607 007620 006003 ROR R3 ;R3 = FC/2  
2608 007622 060377 170666 RD1: ADD R3,@BA ;SET REVERSE BUS ADDRESS  
2609 007626 012737 000076 000672 MOV #76,MTC1 ;SET READ REVERSE  
2610 007634 000403 BR RD1B
```

2611	007636	012737	000070	000672	RD1A:	MOV	#70,MTC1	:SET READ FORWARD
2612	007644	012737	007656	000662	RD1B:	MOV	#RD2,RTRN	:SET INTERRUPT RETURN ADDRESS
2613	007652	000137	020372			JMP	TAPG	:GO EXECUTE TAPE COMMAND
2614	007656	005737	000562		RD2:	TST	RDCMD	:IGNORE EOT IF READ REVERSE
2615	007662	001014				BNE	RD3	
2616	007664	032777	002000	170630		BIT	#2000,@DS	:SEE IF EOT
2617	007672	001410				BEQ	RD3	:IF NOT: BR
2618	007674	005737	000676			TST	TMFLG	:SEE IF TM
2619	007700	001005				BNE	RD3	:IF SO: BR
2620	007702	010037	000660			MOV	R0,EOTREC	:GET # OF RECORDS LEFT IN BLOCK TO READ
2621	007706	052737	100000	000660		BIS	#100000,EOTREC	:SET EOT FLAG
2622	007714	032777	000002	170600	RD3:	BIT	#2,@DS	:SEE IF AT LOAD POINT
2623	007722	001407				BEQ	RD4	:IF NOT: BR
2624	007724	004737	022012			JSR	PC,PAPRT	:PRINT CYCLE NUMBER
2625	007730	000004	023713			TYPE,MSG22		:TYPE MSG
2626	007734	000000				HALT		
2627	007736	000137	003144			JMP	STARTA	:RESTART
2628	007742	032777	004000	170640	RD4:	BIT	#4000,@SWR	:SEE IF SHOULD CHECK ERRORS
2629	007750	001116				BNE	RD5	:IF NOT: BR
2630	007752	005737	000676			TST	TMFLG	
2631	007756	001470				BEQ	RD4B	:IF NO TM EXPT: BR
2632	007760	032777	000004	170534		BIT	#4,@DS	
2633	007766	001023				BNE	RD4A	:IF TM RECVD: BR
2634	007770	012737	032352	020224		MOV	#RDATA,CADER	:SAVE EXPT BUS ADDRESS
2635	007776	012737	000002	020232		MOV	#2,DRVER	:SET TM STATUS ERROR FLAG
2636	010004	004737	017352			JSR	PC,ERPT	:GO PRINT TM ERROR
2637	010010	013704	000674			MOV	UNP,R4	
2638	010014	005737	000562			TST	RDCMD	:SEE IF READ REVERSE
2639	010020	001403				BEQ	1\$:IF NOT: BR
2640	010022	005264	001144			INC	RDERR1(R4)	:BUMP READ REVERSE ERROR
2641	010026	000500				BR	RD6	
2642	010030	005264	001104		1\$:	INC	RDER1(R4)	:BUMP READ FORWARD ERROR
2643	010034	000475				BR	RD6	
2644	010036	012703	032352		RD4A:	MOV	#RDATA,R3	
2645	010042	005737	000562			TST	RDCMD	:SEE IF READ REVERSE
2646	010046	001007				BNE	RD4A0	:IF SO: BR
2647	010050	032737	002000	000552		BIT	#2000,UDES	:SEE IF IN PE
2648	010056	001025				BNE	RD4A2	:IF SO: BR
2649	010060	062703	000002			ADD	#2,R3	
2650	010064	000422				BR	RD4A2	
2651	010066	013704	000556		RD4A0:	MOV	FMCNT,R4	
2652	010072	005104				COM	R4	
2653	010074	032737	000020	000552		BIT	#20,UDES	:SEE IF CORE DUMP
2654	010102	001402				BEQ	RD4A1	:IF NOT: BR
2655	010104	000241				CLC		
2656	010106	006004				ROR	R4	:SET TO FC/2
2657	010110	060403			RD4A1:	ADD	R4,R3	:SET EXPT BUS ADDRESS
2658	010112	042703	000001			BIC	#1,R3	:MAKE EXPT ADDRESS EVEN
2659	010116	032737	002000	000552		BIT	#2000,UDES	:SEE IF IN PE
2660	010124	001002				BNE	RD4A2	:IF SO: BR
2661	010126	162703	000002			SUB	#2,R3	
2662	010132	004737	016614		RD4A2:	JSR	PC,ER2	
2663	010136	000402				BR	RD4C	
2664	010140	004737	016522		RD4B:	JSR	PC,ERCHK	:GO CHECK ERRORS
2665	010144	005737	000706		RD4C:	TST	SERFL	
2666	010150	001416				BEQ	RD5	:IF NO ERROR: BR

2667	010152	013704	000674		MOV	UNP,R4	
2668	010156	005737	000562		TST	RDCMD	:SEE IF READ REVERSE
2669	010162	001003			BNE	RD4D	:IF SO: BR
2670	010164	005264	001104		INC	RDER1(R4)	:BUMP READ FORWARD ERROR
2671	010170	000402			BR	RD4E	
2672	010172	005264	001144	RD4D:	INC	RDERR1(R4)	:BUMP READ REVERSE ERROR
2673	010176	004737	010374	RD4E:	JSR	PC,RDRTY	:GO RETRY
2674	010202	005037	000712		CLR	RTYFL	:CLEAR RETRY FLAG
2675	010206	032777	020000	170374	RD5:	BIT	#20000,@SWR
2676	010214	001005			BNE	RD6	:SEE IF SHOULD DO DATA CHECK
2677	010216	005737	000676		TST	TMFLG	:IF NOT; BR
2678	010222	001002			BNE	RD6	
2679	010224	004737	014750		JSR	PC,DCHK	:GO CHECK DATA
2680	010230	005037	000706	RD6:	CLR	SERFL	:CLEAR STATUS ERROR FLAG
2681	010234	004737	013614		JSR	PC,DS3	:CLEAR BUFFER
2682	010240	032777	000040	170342	BIT	#40,@SWR	:SEE IF SHOULD YOZZLE
2683	010246	001402			BEQ	RD7	:IF NOT: BR
2684	010250	004737	010770		JSR	PC,YOZ	:ELSE GO YOZZLE
2685	010254	013737	000574	000666	RD7:	MOV	RSTAL,STAL
2686	010262	004737	011654		JSR	PC,STALL	:SET DELAY
2687	010266	005737	000562		TST	RDCMD	:STALL
2688	010272	001403			BEQ	RD7A	:SEE IF READ REVERSE
2689	010274	005037	000676		CLR	TMFLG	:IF NOT: BR
2690	010300	000405			BR	RD10	:CLEAR TAPE MARK FLAG
2691	010302	005737	000660	RD7A:	TST	EOTREC	:SEE IF EOT FOUND
2692	010306	100002			BPL	RD10	:IF NOT: BR
2693	010310	012700	000001		MOV	#1,R0	:SET TO EOT
2694	010314	005300		RD10:	DEC	R0	
2695	010316	001402			BEQ	RD11	:IF DONE ALL: BR
2696	010320	000137	007556		JMP	RDO	
2697	010324	005737	000562	RD11:	TST	RDCMD	:SEE IF READ REVERSE
2698	010330	001016			BNE	RDEX	:IF SO: BR
2699	010332	005737	000660		TST	EOTREC	:SEE IF FOUND EOT
2700	010336	100413			BMI	RDEX	:IF SO: BR
2701	010340	005737	000564		TST	TMEX	:SEE IF TM EXPECTED
2702	010344	001410			BEQ	RDEX	:IF NOT: BR
2703	010346	005737	000676		TST	TMFLG	:SEE IF TM FOUND
2704	010352	001005			BNE	RDEX	:IF SO: BR
2705	010354	005237	000676		INC	TMFLG	:ELSE SET FLAG
2706	010360	005200			INC	R0	:SET RECORD COUNT TO ONE
2707	010362	000137	007556		JMP	RDO	:GO READ TM
2708	010366	005037	000676	RDEX:	CLR	TMFLG	
2709	010372	000207		RDX:	RTS	PC	:EXIT

```
2710
2711
2712
2713
2714
2715
2716
2717
2718
2719
2720
2721 010374 032777 000020 170206 RDRTY: BIT #20,@SWR ;SEE IF RETRY INHIBITED
2722 010402 001001 BNE RDRT0 ;IF NOT: BR
2723 010404 000207 RTS PC ;ELSE RETURN
2724
2725 010406 013703 000722 RDRT0: MOV ERSAV,R3
2726 010412 022703 100000 CMP #100000,R3 ;++B BRANCH IF OTHER THAN CORRECTED READ ERROR
2727 010416 001011 BNE 1$ ;++B
2728 010420 032777 000040 170074 BIT #40,@DS ;++B BRANCH IF NRZ
2729 010426 001405 BEQ 1$ ;++B
2730 010430 005037 000706 CLR SERFL ;++B CLEAR ERROR FLAG
2731 010434 000004 026304 TYPE,MSG124 ;++B TYPE 'CORRECTED PE DATA ERROR'
2732 010440 000447 BR RDRT2 ;++B INC SOFT COUNTS
2733 010442 042703 102720 1$: BIC #102720,R3 ;MARK NON-RECOVERABLE ERROR BITS
2734 010446 001407 BEQ RDRT1 ;IF NOT: BR
2735 010450 004737 022012 JSR PC,PAPRT ;PRINT HEADER
2736 010454 000004 025371 TYPE,MSG79 ;TYPE MSG
2737 010460 004737 010754 JSR PC,NRTP ;PRINT ER FOR NON-RETRYABLE ERROR
2738 010464 000207 RTS PC ;RETURN
2739 010466 032777 002000 170114 RDRT1: BIT #2000,@SWR ;SEE IF PRINT INHIBITED
2740 010474 001002 BNE RDRT1B ;IF SO: BR
2741 010476 000004 025101 TYPE,MSG64 ;TYPE MSG
2742 010502 005037 000702 RDRT1B: CLR RTCNT ;CLEAR RETRY COUNTER
2743 010506 005037 000706 RDRTG: CLR SERFL ;CLEAR STATUS ERROR FLAG
2744 010512 012737 000002 000712 MOV #2,RTYFL ;SET READ RETRY FLAG
2745 010520 004737 010770 JSR PC,YOZ ;GO TO YOZZLE TO RETRY READ
2746 010524 005737 000706 TST SERFL ;SEE IF RETRY ERROR
2747 010530 001026 BNE RDRT5 ;IF SO: BR
2748 010532 032777 002000 170050 BIT #2000,@SWR
2749 010540 001007 BNE RDRT2
2750 010542 000004 025514 TYPE,MSG105 ;TYPE MSG
2751 010546 000004 025123 TYPE,MSG65 ;TYPE MSG
2752 010552 013703 000702 MOV RTCNT,R3
2753 010556 104400 TYPOCT ;PRINT RETRY NUMBER
2754 010560 013704 000674 RDRT2: MOV UNP,R4
2755 010564 005737 000562 TST RDCMD ;SEE IF READ REVERSE
2756 010570 001003 BNE RDRT3 ;IF SO: BR
2757 010572 005264 002664 INC RFSOFT(R4) ;ELSO BUMP FORWARD SOFT ERROR COUNTER
2758 010576 000402 BR RDRT4
2759 010600 005264 002704 RDRT3: INC RRSOFT(R4) ;BUMP ERRORS SOFT CNTR
2760 010604 000207 RDRT4: RTS PC ;RETURN
2761 010606 013703 000722 RDRT5: MOV ERSAV,R3 ;GET ER
2762 010612 005037 000650 CLR TEMP3 ;CLEAR RECOVERABLE ERROR INDICATOR
2763 010616 042703 102720 BIC #102720,R3 ;MASK RECOVERABLE BITS
2764 010622 001412 BEQ RDRT5A ;IF RECOVERABLE: BR
2765 010624 004737 022012 JSR PC,FAPRT ;PRINT HEADER
```

```
2766 010630 000004 025371          TYPE,MSG79          ;TYPE MSG
2767 010634 004737 010754          JSR    PC,NRTP      ;PRINT ER
2768 010640 012737 000001 000650  MOV    #1,TEMP3     ;SET FLAG
2769 010646 000404          BR     RDRT5B
2770 010650 032777 002000 167732 RDRT5A: BIT    #2000,@SWR ;SEE IF PRINT INHIBITED
2771 010656 001013          BNE   RDRT6         ;IF SO: BR
2772 010660 000004 025123          RDRT5B: TYPE,MSG65  ;TYPE MSG
2773 010664 013703 000702          MOV    RTCNT,R3
2774 010670 104400          TYPOCT             ;PRINT RETRY NUMBER
2775 010672 005737 000650          TST   TEMP3        ;SEE IF DID NON-RECOVERABLE
2776 010676 001403          BEQ   RDRT6         ;IF NOT: BR
2777 010700 005037 000650          CLR   TEMP3        ;CLEAR FLAG
2778 010704 000207          RTS    PC           ;EXIT
2779 010706 005237 000702          RDRT6: INC   RTCNT
2780 010712 023737 000702 000604  CMP    RTCNT,RETRY ;SEE IF DONE 8 RETRIES
2781 010720 001272          BNE   RDRTG         ;IF NOT: BR
2782 010722 000004 026023          TYPE,MSG115
2783 010726 013704 000674          MOV    UNP,R4
2784 010732 005737 000562          TST   RDCMD        ;SEE IF READ REVERSE
2785 010736 001003          BNE   RDRT7         ;IF SO: BR
2786 010740 005264 002724          INC   RFHARD(R4)   ;BUMP FORWARD HARD ERROR CNTR
2787 010744 000402          BR    RDRTX
2788 010746 005264 002744          RDRT7: INC   RRHARD(R4) ;BUMP REVERSE HARD ERROR CNTR
2789 010752 000207          RDRTX: RTS    PC           ;RETURN
2790
2791 010754 013703 000722          NRTP:  MOV    ERSV,R3 ;GET ER REGISTER
2792 010760 104400          TYPOCT             ;PRINT ER
2793 010762 004737 020250          JSR    PC,FRPRT    ;PRINT F OR R
2794 010766 000207          RTS    PC           ;RETURN
2795
2796          ;*****
2797          ;YOZZLE SUBROUTINE:
2798          ;
2799          ;THIS SUBROUTINE, ENTERED VIA SWITCH FIVE (5), IS USED TO PERFORM
2800          ;A CONTINUOUS READ AND SPACE OVER OF THE CURRENT RECORD ON TAPE.
2801          ;FULL STATUS AND DATA CHECKING MAY BE PERFORMED
2802          ;OR NOT VIA CONSOLE SWITCHES ELEVEN (11) AND THIRTEEN (13).
2803          ;A SOFTWARE DELAY IS PERFORMED BETWEEN EACH READ
2804          ;AND SPACE OPERATION AND MAY BE VARIED BY TYPING
2805          ;CNTRL C ON THE TTY AND ENTERING A VALUE IN RESPONSE
2806          ;TO THE PRINTED REQUEST.
2807          ;*****
2808 010770 013737 000602 000666 YOZ:  MOV    YSTAL,STAL
2809 010776 004737 011654          JSR    PC,STALL    ;DO YOZZLE STALL
2810 011002 012777 177777 167506 YOZO: MOV    #-1,@FC      ;SET TO 1 RECORD SPACING
2811 011010 005737 000562          TST   RDCMD        ;SEE IF READ REVERSE
2812 011014 001404          BEQ   YOZA         ;IF NOT: BR
2813 011016 112737 000030 000672  MOVB  #30,MTC1     ;SET TO SPACE FORWARD
2814 011024 000403          BR    YOZB
2815 011026 112737 000032 000672 YOZA: MOVB  #32,MTC1     ;SET TO SPACE REVERSE
2816 011034 012737 011054 000662 YOZB: MOV    #YOZC,RTRN ;SET RETURN ADDRESS
2817 011042 012737 177775 000666  MOV    #177775,STAL ;SET TIME MULTIPLIER
2818 011050 000137 020372          JMP    TAPG        ;GO YOZZLE
2819 011054 005737 000676          YOZC: TST   TMFLG      ;SEE IF TM
2820 011060 001404          BEQ   1$          ;IF NOT: BR
2821 011062 012737 040000 000666  MOV    #40000,STAL ;SET TM STALL
```

2822	011070	000403				BR	2%		
2823	011072	013737	000602	000666	1%	MOV	YSTAL,STAL		
2824	011100	004737	011654		2%	JSR	PC,STALL		
2825	011104	012777	032352	167402		MOV	#RDATA,@BA		;DO YOZZLE STALL
2826	011112	005737	000562			TST	RDCMD		;SET BUS ADDRESS
2827	011116	001416				BEQ	YOZC1		;SEE IF READ REVERSE
2828	011120	013703	000556			MOV	FMCNT,R3		;IF NOT: BR
2829	011124	005103				COM	R3		
2830	011126	032737	000020	000552		BIT	#20,UDES		;SEE IF CORE DUMP
2831	011134	001401				BEQ	YOZC0		;IF NOT: BR
2832	011136	006203				ASR	R3		;R3 = FC/2
2833	011140	060377	167350			ADD	R3,@BA		;SET REVERSE BUS ADDRESS
2834	011144	012737	000076	000672		MOV	#76,MTC1		;SET READ REVERSE
2835	011152	000403				BR	YOZC2		
2836	011154	012737	000070	000672		MOV	#70,MTC1		;SET READ FORWARD
2837	011162	013777	000556	167326		MOV	FMCNT,@FC		;SET CHARACTER COUNT
2838	011170	012737	011202	000662		MOV	#YOZD,RTRN		;SET RETURN ADDRESS
2839	011176	000137	020372			JMP	TAPG		;GO READ
2840	011202	032777	004000	167400		BIT	#4000,@SWR		;SEE IF SHOULD CHECK ERRORS
2841	011210	001047				BNE	YOZE		;IF NOT: BR
2842	011212	005737	000676			TST	TMFLG		;SEE IF TAPE MARK TIME
2843	011216	001442				BEQ	YOZD1		;IF NOT: BR
2844	011220	005737	000562			TST	RDCMD		;SEE IF READ REVERSE
2845	011224	001425				BEQ	YOZD0		;IF NOT: BR
2846	011226	012703	032352			MOV	#RDATA,R3		
2847	011232	013704	000556			MOV	FMCNT,R4		
2848	011236	005104				COM	R4		
2849	011240	032737	000020	000552		BIT	#20,UDES		;SEE IF CORE DUMP
2850	011246	001401				BEQ	YOZD4		;IF NOT: BR
2851	011250	006204				ASR	R4		;SET TO FC/2
2852	011252	060403				ADD	R4,R3		;SET EXPT BUS ADDRESS
2853	011254	042703	000001			BIC	#1,R3		;MAKE EXPT ADDRESS EVEN
2854	011260	032737	002000	000552		BIT	#2000,UDES		;SEE IF PE
2855	011266	001001				BNE	YOZD2		;IF SO: BR
2856	011270	005743				TST	-(R3)		;SET EXPT BA
2857	011272	004737	016614			JSR	PC,ER2		;GO CHECK ERRORS
2858	011276	000430				BR	YOZF		
2859	011300	012703	032352			MOV	#RDATA,R3		
2860	011304	032737	002000	000552		BIT	#2000,UDES		;SEE IF PE
2861	011312	001001				BNE	YOZD3		;IF SO: BR
2862	011314	005723				TST	(R3)+		;SET EXPT BA
2863	011316	004737	016614			JSR	PC,ER2		;GO CHECK ERRORS
2864	011322	000416				BR	YOZF		
2865	011324	004737	016522			JSR	PC,ERCHK		;ELSE GO CHECK ERRORS
2866	011330	005737	000712			TST	RTYFL		;SEE IF RETRY
2867	011334	001013				BNE	YOZG		;IF SO: BR
2868	011336	032777	020000	167244		BIT	#20000,@SWR		;SEE IF SHOULD CHECK DATA
2869	011344	001005				BNE	YOZF		;IF NOT: BR
2870	011346	005737	000676			TST	TMFLG		;SEE IF TAPE MARK
2871	011352	001002				BNE	YOZF		;IF SO: BR
2872	011354	004737	014750			JSR	PC,DCHK		;ELSE GO CHECK DATA
2873	011360	004737	013614			JSR	PC,DS3		;GO CLEAR DATA AREA
2874	011364	032777	000040	167216		BIT	#40,@SWR		;SEE IF SHOULD CONTINUE YOZZLE
2875	011372	001402				BEQ	YOZH		;IF NOT: BR
2876	011374	000137	011002			JMP	YOZO		
2877	011400	000207				RTS	PC		;EXIT


```

2878
2879
2880
2881
2882
2883
2884
2885
2886
2887
2888
2889
2890
2891
2892
2893
2894 011402 013737 000600 000666 BKSP: MOV TSTAL,STAL
2895 011410 004737 011654 JSR PC,STALL ;DO TURN AROUND STALL
2896 011414 012737 023602 000652 MOV #MSG10,EMADDR
2897 011422 012703 032352 MOV #RDATA,R3 ;SET EXPECTED BA
2898 011426 010377 167062 MOV R3,@BA
2899 011432 005737 000564 TST TMEX ;SEE IF TM
2900 011436 001436 BEQ B0 ;IF NOT: BR
2901 011440 012777 177777 167050 MOV #-1,@FC
2902 011446 012737 000032 000672 MOV #32,MTC1
2903 011454 012737 011466 000662 MOV #1$,RTRN
2904 011462 000137 020372 JMP TAPG ;SPACE TO TM
2905 011466 032777 010000 167114 1$: BIT #10000,@SWR ;SEE IF SHOULD CHECK ERROR
2906 011474 001017 BNE B0 ;IF NOT: BR
2907 011476 012737 025021 000652 MOV #MSG55,EMADDR
2908 011504 032777 000004 167010 BIT #4,@DS ;SEE IF TM
2909 011512 001006 BNE 2$ ;IF SO: BR
2910 011514 012737 032352 020224 MOV #RDATA,CADER
2911 011522 004737 017352 JSR PC,ERPT ;PRINT ERROR
2912 011526 000402 BR B0
2913 011530 004737 016614 2$: JSR PC,ER2
2914 011534 013700 000554 B0: MOV RCNT,R0
2915 011540 005737 000660 TST EOTREC ;BRANCH IF EOT NOT DETECTED
2916 011544 100007 BPL 1$
2917 011546 042737 100000 000660 BIC #100000,EOTREC ;CLEAR EOT INDICATOR
2918 011554 013703 000660 MOV EOTREC,R3 ;GET # OF RECORDS LEFT IN BLOCK
2919 011560 160300 SUB R3,R0 ;FORM # OF RECORDS TO BACK SPACE
2920 011562 005200 INC R0
2921 011564 012737 023602 000652 1$: MOV #MSG10,EMADDR ;SET ERROR MESH ADDRESS
2922 011572 012737 011630 000662 MOV #2$,RTRN ;SET RETURN PC
2923 011600 012777 177777 166710 MOV #-1,@FC ;SET TO BACKSPACE 1 RECORD
2924 011606 012703 032352 MOV #RDATA,R3 ;SET EXPECTED BA
2925 011612 010377 166676 MOV R3,@BA
2926 011616 012737 000032 000672 MOV #32,MTC1 ;SET SPACE REVERSE
2927 011624 000137 020372 JMP TAPG ;GO DO SPACE
2928 011630 004737 016614 2$: JSR PC,ER2
2929 011634 013737 000600 000666 MOV TSTAL,STAL ;DO STALL
2930 011642 004737 011654 JSR PC,STALL ;STALL
2931 011646 005300 DEC R0 ;DECREMENT # OF RECORD TO BACKSPACE
2932 011650 001345 BNE 1$
2933 011652 000207 RTS PC ;EXIT

```

2934
2935
2936
2937
2938
2939
2940
2941
2942
2943
2944
2945
2946
2947
2948
2949
2950
2951
2952
2953
2954

011654 005337 000666
011660 001375
011662 000207

```
*****  
: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.  
*****  
STALL: DEC STAL  
BNE STALL :DELAY  
RTS PC :EXIT
```

2955
2956
2957
2958
2959
2960
2961
2962
2963
2964
2965
2966
2967
2968 011664 012701 177760
2969 011670 012702 175000
2970 011674 004737 022314
2971 011700 042737 000001 000630
2972 011706 013737 000630 000556
2973 011714 012737 177777 013652
2974 011722 000207
2975
2976
2977
2978
2979
2980
2981
2982
2983
2984
2985
2986 011724 012702 000001
2987 011730 012701 000500
2988 011734 004737 022314
2989 011740 013737 000630 000554
2990 011746 000207
2991
2992

```
*****  
: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.  
:*****
```

```
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,PATS   ;PRESET DATA PATTERN  
        RTS    PC       ;EXIT
```

```
*****  
: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
```

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
3042
3043
3044
3045
3046
3047
3048

```

:*****
: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 WRITTING 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$: CLR UNP :CLEAR TABLE POINTER
011766 005037 004730 CLR REOTC :CLEAR EOT UNIT COUNTER
011772 012737 024231 012016 MOV #MSG31,41$ :GET TITLE MSG
012000 005737 000734 TST ASEQF :SEE IF AUTO SEQ
012004 001403 BEQ 4$ :IF NOT: BR
012006 012737 024157 012016 MOV #MSG30,41$ :SET AUTO SEQ HDR
012014 000004 4$: TYPE :TYPE MSG

```

3049	012016	000000		41\$:	.WORD 0	:ADDRESS OF APPROPRIATE TITLE MSG
3050	012020	105077	177772		CLRB @41\$:DO NOT TYPE TITLE ON RESTART
3051	012024	000004	024313		TYPE,MSG31A	:TYPE INSTRUCTIONS
3052	012030	105037	024313		CLRB MSG31A	:DO NOT TYPE STARTUP INSTRUCTIONS ON RESTART
3053	012034	005737	013444		TST SCVFL	:SEE IF SHORT CONVERSATION
3054	012040	001065			BNE 6\$:IF SO: BR
3055	012042	000004	025245		TYPE,MSG74	:REQUEST REGISTER START
3056	012046	013703	000544		MOV REGS,R3	
3057	012052	104400			TYPOCT	:PRINT CURRENT REG START
3058	012054	012705	000544		MOV #REGS,R5	:SAVE ADDRESS LOCATION
3059	012060	012701	000007		MOV #7,R1	:SET SIZE OF ENTRY
3060	012064	012702	176400		MOV #176400,R2	:SET UPPER LIMIT
3061	012070	012703	172300		MOV #172300,R3	:SET LOWER LIMIT
3062	012074	004737	022476		JSR PC,TTR	:GO GET RESPONSE
3063						
3064	012100	000004	025270		TYPE,MSG75	:REQUEST INTERRUPT VECTOR ADDRESS
3065	012104	013703	000546		MOV VECT,R3	
3066	012110	104400			TYPOCT	:PRINT CURRENT VECTOR
3067	012112	012705	000546		MOV #VECT,R5	:SET SAVE LOCATION
3068	012116	012701	000004		MOV #4,R1	:SET SIZE OF ENTRY
3069	012122	012702	000224		MOV #224,R2	:SET UPPER LIMIT
3070	012126	012703	000150		MOV #150,R3	:SET LOWER LIMIT
3071	012132	004737	022476		JSR PC,TTR	:GO GET RESPONSE
3072	012136	013700	000546		MOV VECT,R0	:GET VECTOR ADDRESS
3073	012142	012720	021160		MOV #MTINT,(R0)+	:LOAD VECTOR WITH HANDLER ADDRESS
3074	012146	012710	000340		MOV #340,(R0)	:LOAD PRIORITY LEVEL
3075	012152	013700	000544		MOV REGS,R0	:GET STARTING REGISTER ADDRESS
3076	012156	012701	000016		MOV #16,R1	:SET NUMBER OF REGISTERS
3077	012162	012702	000510		MOV #C1,R2	:GET FIRST ADDRESS LOCATION
3078	012166	010022		5\$:	MOV R0,(R2)+	:BUILD TABLE OF ADDRESSES
3079	012170	062700	000002		ADD #2,R0	:BUMP ADDRESS
3080	012174	005301			DEC R1	:SEE IF DONE
3081	012176	001373			BNE 5\$:IF NOT: BR
3082	012200	005737	000734		TST ASEQF	:SEE IF AUTO SEQ
3083	012204	001403			BEQ 6\$:IF NOT: BR
3084	012206	005726			TST (SP)+	:RESET STACK POINTER
3085	012210	000137	021176		JMP ASEQ	:GO TO AUTO SEQUENCE
3086						
3087	012214	012777	000040	166276	6\$: MOV #40,@CS	:INITIALIZE
3088	012222	000004	024756		TYPE,MSG52A	:REQUEST DRIVE (TM03) #
3089	012226	012705	000550		MOV #DVN,R5	:GET ADDRESS
3090	012232	012701	000002		MOV #2,R1	:SET SIZE OF RESPONSE
3091	012236	012702	000007		MOV #7,R2	:SET UPPER LIMIT
3092	012242	012703	000000		MOV #0,R3	:SET LOWER LIMIT
3093	012246	004737	022476		JSR PC,TTR	:GO GET DRIVE NUMBER
3094	012252	013777	000550	166240	MOV DVN,@CS	
3095	012260	005777	166224		TST @C1	:ACCESS DRIVE
3096	012264	032777	010000	166226	BIT #10000,@CS	:SEE IF NED
3097	012272	001403			BEQ TINPO	:IF NOT: BR
3098	012274	000004	025202		TYPE,MSG71	:TYPE 'NON-EXISTANT DRIVE'
3099	012300	000745			BR 6\$:RETRY DVN
3100						
3101	012302	012705	000646	TINPO:	MOV #TEMP2,R5	:SET ADDRESS FOR RESPONSE
3102	012306	000004	024400		TYPE,MSG32	:REQUEST SLAVE (TE16,TU77) #
3103	012312	005037	000646		CLR TEMP2	:CLEAR BUFFER
3104	012316	012701	000002		MOV #2,R1	:SET NUMBER OF CHARACTERS TO INPUT

3105	012322	012702	000007		MOV	#7,R2	;SET MAXIMUM LIMIT
3106	012326	012703	000000		MOV	#0,R3	;SET MINIMUM LIMIT
3107	012332	004737	022476		JSR	PC,TTR	;GO GET UNIT NUMBER
3108	012336	005737	000644		TST	TEMP1	;SEE IF HAVE NEW PARAMETER
3109	012342	001010			BNE	TINPOB	;IF SO: BR
3110	012344	013700	000674		MOV	UNP,R0	
3111	012350	001754			BEQ	TINPO	;BRANCH IF FIRST ENTRY
3112	012352	012760	177777	000742	MOV	#-1,UN1(R0)	;SET END UNIT TABLE
3113	012360	000137	012700		JMP	TINP2C	;GO GET RECORD COUNT
3114	012364	013700	000674		TINPOB: MOV	UNP,R0	
3115	012370	011560	000742		MOV	(R5),UN1(R0)	;SET NEW SLAVE #
3116	012374	012777	000040	166116	MOV	#40,@CS	;DO A MASS BUS CLEAR
3117	012402	013777	000550	166110	MOV	DVN,@CS	;LOAD DRIVE #
3118	012410	016077	000742	166124	MOV	UN1(R0),@TC	;LOAD SLAVE NUMBER
3119	012416	032777	002000	166112	BIT	#2000,@DT	;SEE IF SLAVE PRESENT
3120	012424	001003			BNE	TINPOD	;IF SO: BR
3121	012426	000004	025034		TYPE,MSG57		;TYPE NON-EXISTANT SLAVE'
3122	012432	000723			BR	TINPO	;REDO
3123	012434	017703	166076		TINPOD: MOV	@DT,R3	;GET CONTENTS OF DT REG
3124	012440	042703	000007		BIC	#7,R3	;CLEAR DRIVE TYPE #
3125	012444	022703	142050		CMP	#142050,R3	;SEE IF 9TRK TM03
3126	012450	001407			BEQ	TINPOE	;IF SO: BR
3127	012452	000004	024727		TYPE,MSG50		;TYPE 'ILLEGAL DRIVE TYPE'
3128	012456	017703	166054		MOV	@DT,R3	
3129	012462	042703	000007		BIC	#7,R3	;CLEAR SLAVE #
3130	012466	104400			TYPOCT		;PRINT DRIVE TYPE REGISTER
3131	012470	004737	023362		TINPOE: JSR	PC,SNPT	;PRINT SERIAL NUMBER
3132							
3133	012474	000004	024413		TINP1: TYPE,MSG33		;REQUEST DENSITY
3134	012500	005037	000646		CLR	TEMP2	;CLEAR BUFFER
3135	012504	012701	000002		MOV	#2,R1	;SET NUMBER OF CHARACTERS TO INPUT
3136	012510	012702	000004		MOV	#4,R2	;SET MAXIMUM LIMIT
3137	012514	012703	000003		MOV	#3,R3	;SET MINIMUM LIMIT
3138	012520	004737	022476		JSR	PC,TTR	;GO GET DENSITY
3139	012524	012703	000010		MOV	#10,R3	;SET POSITION FACTOR
3140	012530	004737	013446		JSR	PC,TPOS	;GO LOAD DENSITY INTO PROPER POSITION
3141							
3142	012534	000315			TINP2: SWAB	(R5)	;IF DENSITY
3143	012536	022715	000004		CMP	#4,(R5)	;IS 1600BPI
3144	012542	001415			BEQ	1\$;THEN SKIP PARITY REQUEST
3145	012544	000004	024426		TYPE,MSG34		;REQUEST PARITY
3146	012550	005037	000646		CLR	TEMP2	;CLR BFR
3147	012554	012701	000002		MOV	#2,R1	;SET NUMBER OF CHAR. TO INPUT
3148	012560	012702	000001		MOV	#1,R2	;SET HIGH LIMIT
3149	012564	012703	000000		MOV	#0,R3	;SET LOW LIMIT
3150	012570	004737	022476		JSR	PC,TTR	;GO INPUT PARITY
3151	012574	000402			BR	2\$;SKIP 1600 BPI PAROTY SETTING
3152	012576	012715	000000		1\$: MOV	#0,(R5)	;SET ODD PARITY FOR 1600 BPI
3153	012602	012703	000003		2\$: MOV	#3,R3	;SET POSITION FACTOR
3154	012606	004737	013446		JSR	PC,TPOS	;GO POSITION PARITY
3155							
3156	012612	000004	025000		TINP2A: TYPE,MSG53		;REQUEST FORMAT
3157	012616	005037	000646		CLR	TEMP2	
3158	012622	012701	000003		MOV	#3,R1	
3159	012626	012702	000017		MOV	#17,R2	
3160	012632	012703	000000		MOV	#0,R3	

3161	012636	004737	022476		JSR	PC,TTR		:GO GET FORMAT
3162	012642	012703	000004		MOV	#4,R3		
3163	012646	004737	013446		JSR	PC,TPOS		
3164	012652	005237	004730		TINP2B: INC	REOTC		:BUMP EOT UNIT COUNTER
3165	012656	022737	000016	000674	CMP	#16,UNP		:SEE IF DONE UNITS
3166	012664	001405			BEQ	TINP2C		:IF SO: BR
3167	012666	062737	000002	000674	ADD	#2,UNP		:POINT TO NEXT UNIT
3168	012674	000137	012302		JMP	TINPO		:ELSE LOOK FOR NEXT UNIT
3169								
3170								
3171	012700	005037	000674		TINP2C: CLR	UNP		:CLEAR UNIT POINTER
3172	012704	113737	004730	004731	MOVB	REOTC,REOTC+1		:SET # OF UNITS TO TEST
3173								
3174	012712	000004	024440		TINP3: TYPE,MSG35			:REQUEST RECORDS PER BLOCK
3175	012716	013703	000554		MOV	RCNT,R3		
3176	012722	104400			TYPOCT			:PRINT RECORD COUNT
3177	012724	012705	000554		MOV	#RCNT,R5		:SET RECORD COUNT ADDRESS
3178	012730	012701	000007		MOV	#7,R1		:SET NUMBER OF CHARACTERS TO INPUT
3179	012734	012702	177777		MOV	#177777,R2		:SET MAXIMUM LIMIT
3180	012740	012703	000001		MOV	#1,R3		:SET MINIMUM LIMIT
3181	012744	004737	022476		JSR	PC,TTR		:GO GET RECORD COUNT
3182	012750	013737	000554	000632	MOV	RCNT,RCSAV		:SAVE RECORD COUNT
3183								
3184	012756	000004	024460		TYPE,MSG36			:REQUEST CHARACTERS PER RECORD
3185	012762	005437	000556		NEG	FMCNT		
3186	012766	013703	000556		MOV	FMCNT,R3		
3187	012772	104400			TYPOCT			:PRINT CHAR COUNT
3188	012774	012705	000556		MOV	#FMCNT,R5		:SET CHARACTER COUNT ADDRESS
3189	013000	012701	000007		MOV	#7,R1		:SET NUMBER OF CHARACTERS TO INPUT
3190	013004	012702	004000		MOV	#4000,R2		:SET MAXIMUM LIMIT
3191	013010	012703	000004		MOV	#4,R3		:SET MINIMUM LIMIT
3192	013014	004737	022476		JSR	PC,TTR		:GO GET CHARACTER COUNT
3193	013020	005437	000556		NEG	FMCNT		:SET TO TWO'S COMPLIMENT
3194	013024	013737	000556	000634	MOV	FMCNT,FCSAV		:SAVE FRAME COUNT
3195								
3196	013032	000004	024476		TYPE,MSG37			:REQUEST PATTERN #
3197	013036	013703	000560		MOV	PATRN,R3		
3198	013042	104400			TYPOCT			:PRINT PATTERN
3199	013044	005037	014014		CLR	DOFL		:CLEAR EXTERNAL DATA FLAG
3200	013050	012705	000560		MOV	#PATRN,R5		:SET PATTERN NUMBER ADDRESS
3201	013054	012701	000003		MOV	#3,R1		:SET NUMBER OF CHARACTERS TO INPUT
3202	013060	012702	000015		MOV	#15,R2		:SET MAXIMUM LIMIT
3203	013064	012703	000000		MOV	#0,R3		:SET MINIMUM LIMIT
3204	013070	004737	022476		JSR	PC,TTR		:GO GET PATTERN NUMBER
3205								
3206	013074	000004	025166		TYPE,MSG69			:REQUEST TAPE MARK
3207	013100	013703	000564		MOV	TMEX,R3		
3208	013104	104400			TYPOCT			:PRINT CURRENT TM FLAG SETTING
3209	013106	012705	000564		MOV	#TMEX,R5		:GET TM FLAG ADDRESS
3210	013112	012701	000002		MOV	#2,R1		:SET SIZE OF RESPONSE
3211	013116	012702	000001		MOV	#1,R2		:SET UPPER LIMIT
3212	013122	012703	000000		MOV	#0,R3		:SET LOWER LIMIT
3213	013126	004737	022476		JSR	PC,TTR		:TM 1=YES
3214								
3215	013132	000004	023670		TYPE,MSG21			:REQUEST INTERCHANGE READ
3216	013136	013703	000570		MOV	INTRF,R3		

3217	013142	104400		TYPOCT		:PRINT CURRENT SETTING
3218	013144	012705	000570	MOV	#INTRF,R5	:GET FLAG ADDRESS
3219	013150	012701	000002	MOV	#2,R1	:SET SIZE OF RESPONSE
3220	013154	012702	000001	MOV	#1,R2	:SET UPPER LIMIT
3221	013160	012703	000000	MOV	#0,R3	:SET LOWER LIMIT
3222	013164	004737	022476	JSR	PC,TTR	:GO GET RESPONSE
3223						
3224	013170	000004	024513	TYPE,MSG38		:REQUEST SINGLE PASS
3225	013174	013703	000572	MOV	SPFLG,R3	
3226	013200	104400		TYPOCT		:PRINT CURRENT SETTING
3227	013202	012705	000572	MOV	#SPFLG,R5	:SET ADDRESS OF FLAG
3228	013206	012701	000002	MOV	#2,R1	:SET SIZE OF RESPONSE
3229	013212	012702	000001	MOV	#1,R2	:SET UPPER LIMIT
3230	013216	012703	000000	MOV	#0,R3	:SET LOWER LIMIT
3231	013222	004737	022476	JSR	PC,TTR	:GO GET RESPONSE
3232						
3233	013226	000004	024531	TINP3A: TYPE,MSG39		:REQUEST CRC CORRECTION
3234	013232	013703	000566	MOV	CRCC,R3	
3235	013236	104400		TYPOCT		
3236	013240	012705	000566	MOV	#CRCC,R5	
3237	013244	012701	000002	MOV	#2,R1	
3238	013250	012702	000001	MOV	#1,R2	
3239	013254	012703	000000	MOV	#0,R3	
3240	013260	004737	022476	JSR	PC,TTR	
3241	013264	004737	022346	JSR	PC,GTSWR	:GET SWITCHES
3242	013270	005737	013444	TINP4: TST	SCVFL	:BRANCH IF SHORT CONVERSATION
3243	013274	001060		BNE	TINPX	
3244	013276	005737	000636	1\$: TST	TINF	:BRANCH IF NO TTY INPUT
3245	013302	001455		BEQ	TINPX	
3246	013304	000004	024567	TYPE,MSG40		:REQUEST READ STALL
3247	013310	013703	000574	MOV	RSTAL,R3	
3248	013314	104400		TYPOCT		:PRINT READ STALL
3249	013316	012705	000574	MOV	#RSTAL,R5	:SET READ STALL ADDRESS
3250	013322	012701	000007	MOV	#7,R1	:SET NUMBER OF CHARACTERS TO INPUT
3251	013326	012702	177777	MOV	#-1,R2	:SET MAXIMUM LIMIT
3252	013332	012703	000001	MOV	#1,R3	:SET MINIMUM LIMIT
3253	013336	004737	022476	JSR	PC,TTR	:GO GET READ STALL
3254						
3255	013342	000004	024616	TYPE,MSG41		:REQUEST WRITE STALL
3256	013346	013703	000576	MOV	WSTAL,R3	
3257	013352	104400		TYPOCT		:PRINT READ STALL
3258	013354	012705	000576	MOV	#WSTAL,R5	:SET WRITE STALL ADDRESS
3259	013360	012701	000007	MOV	#7,R1	:SET NUMBER OF CHARACTERS TO INPUT
3260	013364	012702	177777	MOV	#-1,R2	:SET MAXIMUM LIMIT
3261	013370	012703	000001	MOV	#1,R3	:SET MINIMUM LIMIT
3262	013374	004737	022476	JSR	PC,TTR	:GO GET WRITE STALL
3263						
3264	013400	000004	024627	TYPE,MSG42		:REQUEST TURN AROUND STALL
3265	013404	013703	000600	MOV	TSTAL,R3	
3266	013410	104400		TYPOCT		:PRINT TA STALL
3267	013412	012705	000600	MOV	#TSTAL,R5	:SET TURN AROUND STALL ADDRESS
3268	013416	012701	000007	MOV	#7,R1	:SET NUMBER OF CHARACTERS TO INPUT
3269	013422	012702	177777	MOV	#-1,R2	:SET MAXIMUM LIMIT
3270	013426	012703	000001	MOV	#1,R3	:SET MINIMUM LIMIT
3271	013432	004737	022476	JSR	PC,TTR	:GO GET TURN AROUND STALL
3272	013436	005037	013444	TINPX: CLR	SCVFL	:CLEAR SHORT CONVERSATION FLAG

3273 013442 000207
3274 013444 000000
3275
3276
3277
3278 013446 006337 000646
3279 013452 005303
3280 013454 001374
3281 013456 013700 000674
3282 013462 053760 000646 000742
3283 013470 000207
3284

RTS PC :EXIT
SCVFL: 0 ;SHORT CONVERSATION FLAG
;UNIT DESCRIPTION POSITIONING SUBROUTINE*****
TPOS: ASL TEMP2 :POSITION CHARACTER
DEC R3 :SEE IF DONE
BNE TPOS :IF NOT: BR
MOV UNP,R0 :LOAD UNIT POINTER
BIS TEMP2,UN1(R0) :LOAD CHARACTER INTO UN1(R0)
RTS PC :EXIT

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
3334
3335
3336
3337
3338
3339
3340

013472 005737 014404
013476 001044
013500 005737 000734
013504 001406
013506 005737 000560
013512 100003
013514 004737 014342
013520 000433
013522 023737 000560 013652
013530 001014
013532 013703 000552
013536 042703 177767
013542 023703 013654
013546 001404
013550 010337 013654
013554 004737 014406
013560 000207
013562 012703 026344
013566 013701 000560
013572 010137 013652
013576 062701 000001
013602 006301
013604 004771 002764
013610 004737 014406
013614 013702 000556
013620 006202
013622 012701 032352
013626 005021
013630 005202
013632 001375
013634 013737 000552 013654
013642 042737 177767 013654
013650 000207
013652 177777
013654 000000

DSUP:
DSO:
;
DSOC:
DSOB:
DSOA:
DS2A:
DS3:
DS4:
PATS:
PARS:

TST RDFL
BNE DS2A
TST ASEQF
BEQ DSOC
TST PATRN
BPL DSOC
JSR PC,DATR
RTS PC
BR DS2A
CMP PATRN,PATS
BNE DSOA
MOV UDES,R3
BIC #177767,R3
CMP PARS,R3
BEQ DSOB
MOV R3,PARS
JSR PC,CRCLRC
RTS PC
MOV #WDATA,R3
MOV PATRN,R1
MOV R1,PATS
ADD #1,R1
ASL R1
JSR PC,@DATBL(R1)
DS2A: JSR PC,CRCLRC
DS3: MOV FMCNT,R2
ASR R2
MOV #RDATA,R1
DS4: CLR (R1)+
INC R2
BNE DS4
MOV UDES,PARS
BIC #177767,PARS
RTS PC
-1
0

: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.
:*****
:SEE IF DID RANDOM DATA
:IF NOT: BR
:SEE IF AUTO SEQ
:IF NOT: BR
:SEE IF AUTO RANDOM
:IF NOT: BR
:ELSE GO GENERATE RANDOM DATA
:++B DELETED
:++B GENERATE EXPECTED LRC/CRC & CLEAR READ BFR
:SEE IF NEW PATTERN
:IF SO: BR
:GET UNIT DESCRIPTION
:MASK EVEN PARITY
:SEE IF SAME AS LAST TIME
:IF SO: BR
:SAVE PARITY
:GO GENERATE EXPT CRC/LRC
:R3 = ADDRS OF WRITE BUFFER
:R1 = PATTERN SELECTOR
:BUMP POINTER
:MAKE PATTERN SELECTOR EVEN
:GO GENERATE PATTERN
:GO GENERATE EXPT CRC/LRC
:R2=BUFFER SIZE
:R2=FRAME CMT/2
:R1=READ DATA START
:CLEAR BUFFER
:SEE IF DONE ALL
:IF NOT: BR
:GET UNIT DESCRIPTION
:MASK PARITY
:EXIT
:PATTERN NUMBER SAVE

```

3341
3342                                     ;EXTERNAL DATA INPUT FROM H/S READER (256 CHARACTER MAXIMUM)
3343
3344 013656 005737 014014          DATO:  TST      DOFL          ;BRANCH IF SHOULD DO EXTERNAL INPUT
3345 013662 001401                BEQ      1$
3346 013664 000207                RTS      PC
3347 013666 012737 000001 014014  1$:  MOV      #1,DOFL      ;++B RETURN
3348 013674 005077 164722                CLR      @PRS        ;SET EXTERNAL FLAG
3349 013700 005037 000644                CLR      TEMP1      ;CLEAR READER STATUS
3350 013704 052777 000001 164710  DATOA:  BIS      #1,@PRS    ;CLEAR FOR USE AS CHARACTER FLAG
3351 013712 105777 164704                DATOB:  TSTB     @PRS    ;START READER
3352 013716 100375                BPL      DATOB      ;SEE IF DONE
3353 013720 005001                CLR      R1         ;IF NOT : BR
3354 013722 117701 164676                MOVB     @PRB,R1    ;CLEAR SAVE LOCATION
3355 013726 005737 000644                TST      TEMP1     ;SAVE CHARACTER
3356 013732 001011                BNE      DATOC      ;SEE IF HAVE FOUND START CHARACTER
3357 013734 105701                TSTB     R1         ;IF SO : BR
3358 013736 001762                BEQ      DATOA      ;SEE IF CHARACTER IS 0
3359 013740 012737 000001 000644        MOV      #1,TEMP1   ;IF SO : BR
3360 013746 010137 000646                MOV      R1,TEMP2   ;ELSE SET CHARACTER FOUND FLAG
3361 013752 010102                MOV      R1,R2      ;SAVE DATA SIZE
3362 013754 000753                BR       DATOA      ;SAVE DATA SIZE
3363 013756 110123                DATOC:  MOVB     R1,(R3)+ ;GO GET FIRST DATA CHAR
3364 013760 005302                DEC      R2         ;LOAD BUFFER
3365 013762 001350                BNE      DATOA      ;SEE IF READ ALL
3366 013764 012701 026344                DATOD:  MOV      #WDATA,R1 ;IF NOT : BR
3367 013770 013702 000646                MOV      TEMP2,R2   ;R1 = START OF WRITE BUFFER
3368 013774 112123                DATOE:  MOVB     (R1)+,(R3)+ ;R2 = SIZE OF DATA FIELD
3369 013776 022703 032352                CMP      #RDATA,R3  ;REPEAT LOAD OF DATA FIELD
3370 014002 003001                BGT      DATOF      ;SEE IF DONE
3371 014004 000207                RTS      PC         ;IF NOT: BR
3372 014006 005302                DATOF:  DEC      R2   ;++B RETURN
3373 014010 001371                BNE      DATOE      ;SEE IF AT END OF DATA FIELD
3374 014012 000764                BR       DATOD      ;IF NOT : BR
3375 014014 000000                DOFL:   0           ;ELSE RESTART FILL
3376                                     ;EXTERNAL DATA FLAG=! IF ALREADY DONE

```

```

3377                                     ;ALL ONES*****
3378
3379 014016 012701 177777   DAT1:  MOV    #-1,R1           ;R1=DATA
3380 014022 012702 002002   DAT1A: MOV    #2002,R2        ;R2=WORD COUNT +2
3381 014026 010123         1$:    MOV    R1,(R3)+       ;LOAD BUFFER
3382 014030 005302         DEC    R2                ;SEE IF DONE
3383 014032 001375         BNE    1$                ;IF NOT: BR
3384 014034 000207         RTS    PC
3385
3386                                     ;ALL ZEROS*****
3387
3388 014036 005001   DAT2:  CLR    R1           ;R1=DATA
3389 014040 000770   BR     DAT1A        ;LOAD BUFFER
3390
3391                                     ;WALKING ONE*****
3392
3393 014042 012701 000001   DAT3:  MOV    #1,R1         ;R1=DATA
3394 014046 000241         CLC
3395 014050 012702 004004   DAT3A: MOV    #4004,R2      ;R2=CHARACTER COUNT+4
3396 014054 110123         1$:    MOVB   R1,(R3)+     ;LOAD BUFFER
3397 014056 106101         ROLB   R1                ;SET NEXT CHARACTER
3398 014060 005302         DEC    R2                ;SEE IF DONE
3399 014062 001374         BNE    1$                ;IF NOT: BR
3400 014064 000207         RTS    PC
3401
3402                                     ;WALKING ZERO*****
3403
3404 014066 012701 000376   DAT4:  MOV    #376,R1       ;R1=START OF DATA
3405 014072 000261         SEC
3406 014074 000765         BR     DAT3A            ;LOAD BUFFER
3407
3408                                     ;ALTERNATING ONE/ZERO*****
3409
3410
3411 014076 012701 052525   DAT5:  MOV    #52525,R1     ;R1=DATA
3412 014102 000747         BR     DAT1A            ;LOAD BUFFER
3413
3414                                     ;ALTERNATING ZERO/ONE*****
3415
3416 014104 012701 125252   DAT6:  MOV    #125252,R1    ;R1=DATA
3417 014110 000744         BR     DAT1A            ;LOAD BUFFER
3418
3419                                     ;ONE/ZERO IN ALTERNATING WORDS*****
3420
3421 014112 012701 125252   DAT7:  MOV    #125252,R1    ;SET WORD 1
3422 014116 012702 052525   MOV    #52525,R2          ;SET WORD 2
3423 014122 012704 001002   MOV    #1002,R4           ;SET NUMBER OF ENTRIES
3424 014126 010123         1$:    MOV    R1,(R3)+     ;LOAD WORD 1
3425 014130 010223         MOV    R2,(R3)+         ;LOAD WORD 2
3426 014132 005304         DEC    R4                ;SEE IF DONE
3427 014134 001374         BNE    1$                ;IF NOT: BR
3428 014136 000207         RTS    PC
3429

```

```

3430                                     ;WALKING ONE/ALL ONE IN ALTERNATING CHARS****
3431
3432 014140 012702 002002      DAT10: MOV      #2002,R2      ;SET BUFFER SIZE
3433 014144 012701 000001      MOV      #1,R1        ;SET WALK BASE
3434 014150 000241
3435 014152 012713 177400      1$:  MOV      #177400,(R3) ;LOAD ALL ONE BYTE
3436 014156 050123              BIS      R1,(R3)+    ;LOAD WALK BYTE
3437 014160 106101              ROLB    R1           ;WALK ONE
3438 014162 005302              DEC     R2
3439 014164 001372              BNE    1$           ;DO FULL BUFFER
3440 014166 000207              RTS     PC
3441
3442                                     ;ALL BITS 0-377*****
3443
3444 014170 005001      DAT11: CLR     R1        ;R1=STARTING DATA
3445 014172 012702 004004      MOV     #4004,R2    ;R2=CHARACTER COUNT+4
3446 014176 110123      1$:  MOVB   R1,(R3)+  ;LOAD BUFFER
3447 014200 105201      INCB   R1           ;BUMP DATA
3448 014202 005302      DEC    R2           ;SEE IF DONE
3449 014204 001374      BNE    1$           ;IF NOT: BR
3450 014206 000207      RTS     PC          ;RETURN
3451
3452                                     ;ALL BITS 377-0*****
3453
3454 014210 012701 000377      DAT12: MOV     #377,R1 ;R1=STARTING DATA
3455 014214 012702 004004      MOV     #4004,R2    ;R2=CHARACTER COUNT+4
3456 014220 110123      1$:  MOVB   R1,(R3)+  ;LOAD BUFFER
3457 014222 105301      DECB   R1           ;BUMP DATA
3458 014224 005302      DEC    R2           ;SEE IF DONE
3459 014226 001374      BNE    1$           ;IF NOT: BR
3460 014230 000207      RTS     PC          ;RETURN
3461
3462                                     ;ALTERNATING CHARACTERS 0 AND 377*****
3463
3464 014232 012701 000377      DAT13: MOV     #377,R1 ;R1 = DATA
3465 014236 000137 014022      JMP     DAT1A       ;LOAD BUFFER
3466
3467                                     ;WALKING ZERO/ALL ZERO IN ALTERNATING CHARS*****
3468
3469 014242 012702 002002      DAT14: MOV     #2002,R2 ;SET BUFFER SIZE
3470 014246 012701 000376      MOV     #376,R1     ;SET WALK BASE
3471 014252 000261
3472 014254 010113      1$:  MOV     R1,(R3)   ;LOAD WALK BYTE
3473 014256 042723 177400      BIC    #177400,(R3)+ ;CLEAR HIGH BYTE
3474 014262 106101      ROLB   R1           ;WALK ZERO BIT
3475 014264 005302      DEC    R2
3476 014266 001372      BNE    1$           ;FILL BUFFER
3477 014270 000207      RTS     PC          ;RETURN
3478

```

```

3479                                     ;AUTO SEQUENCE PATTERN*****
3480
3481 014272 012702 000200      DAT15: MOV    #200,R2      ;SET NUMBER OF ENTRIES
3482 014276 012701 014322      1$:   MOV    #APATS,R1     ;SET START OF PATTERN
3483 014302 012704 000010            MOV    #10,R4       ;SET SIZE OF PATTERN
3484 014306 012123      2$:   MOV    (R1)+,(R3)+  ;FILL BUFFER
3485 014310 005304            DEC    R4              ;SEE IF DONE PATTERN
3486 014312 001375            BNE    2$              ;IF NOT: BR
3487 014314 005302            DEC    R2              ;SEE IF DONE BUFER
3488 014316 001367            BNE    1$              ;IF NOT: BR
3489 014320 000207            RTS    PC              ;RETURN
3490
3491 014322 000000      APATS: 0
3492 014324 177400            177400
3493 014326 000377            377
3494 014330 000000            0
3495 014332 177777            -1
3496 014334 000377            377
3497 014336 177400            177400
3498 014340 177777            -1
3499
3500                                     ;RANDOM DATA GENERATOR SUBROUTINE*****
3501
3502 014342 013704 000556      DATR: MOV    FMCNT,R4     ;SET NUMBER OF FRAMES
3503 014346 012703 026344            MOV    #WDATA,R3    ;SET ADDRESS OF START OF BUFFER
3504 014352 012701 177777            MOV    #-1,R1       ;SET HIGH LIMIT
3505 014356 005002            CLR    R2              ;SET LOW LIMIT
3506 014360 004737 022314      1$:   JSR    PC,RANG      ;GO GENERATE NUMBER
3507 014364 013723 000630            MOV    RANSV,(R3)+  ;LOAD BUFFER
3508 014370 005204            INC    R4              ;SEE IF DONE WHOLE BUFFER
3509 014372 001372            BNE    1$              ;IF NOT: BR
3510 014374 012737 000001 014404  MOV    #1,RDFL       ;SET RANDOM DATA FLAG
3511 014402 000207            RTS    PC              ;EXIT
3512 014404 000000      RDFL: 0              ;RANDOM DATA SELECT FLAG

```

```

3513
3514
3515
3516
3517
3518
3519
3520
3521
3522 014406 013700 000556      CRCLRC: MOV      FMCNT,R0      ;SET RECORD SIZE
3523 014412 005400              NEG      R0
3524 014414 012701 026344      MOV      #WDATA,R1      ;SET START OF BUFFER
3525 014420 005037 014742      CLR      XORS
3526 014424 111104              CL0:    MOVVB   (R1),R4      ;GET CHARACTER
3527 014426 004737 014614      JSR      PC,CLP          ;GO GET PARITY OF CHARACTER
3528 014432 004737 014716      JSR      PC,XOR          ;XOR CHARACTER
3529 014436 000241              CLC
3530 014440 006004              ROR      R4              ;ROTATE 1 RIGHT
3531 014442 103014              BCC      CL2              ;IF NO CARRY: BR
3532 014444 052704 000400      BIS      #400,R4         ;SET BIT NINE
3533 014450 000241              CLC
3534 014452 010405              CL1:    MOV      R4,R5      ;SAVE CHARACTER
3535 014454 042705 177703      BIC      #177703,R5
3536 014460 005105              COM      R5
3537 014462 042705 177703      BIC      #177703,R5
3538 014466 042704 000074      BIC      #74,R4
3539 014472 050504              BIS      R5,R4           ;COMPLIMENT BITS 2,3,4,5
3540 014474 010437 014742      CL2:    MOV      R4,XORS
3541 014500 005300              DEC      R0
3542 014502 001350              BNE      CL0              ;BRANCH IF NOT LAST CHAR
3543 014504 013704 014742      CLLAST: MOV      XORS,R4
3544 014510 005137 014742      COM      XORS
3545 014514 042737 177050 014742 BIC      #177050,XORS
3546 014522 042704 177727      BIC      #177727,R4      ;COMPLIMENT ALL BUT BITS 3&5
3547 014526 050437 014742      BIS      R4,XORS
3548 014532 013737 014742 014744 MOV      XORS,EXCRC      ;SAVE EXPECTED CRC
3549 014540 013700 000556      MOV      FMCNT,R0
3550 014544 005400              NEG      R0
3551 014546 012701 026344      MOV      #WDATA,R1      ;DO EXPT LRC
3552 014552 005037 014742      CLR      XORS
3553 014556 111104              CL3:    MOVVB   (R1),R4      ;GET PARITY
3554 014560 004737 014614      JSR      PC,CLP          ;XOR CHARACTER
3555 014564 004737 014716      JSR      PC,XOR
3556 014570 005300              DEC      R0
3557 014572 001371              BNE      CL3              ;DO ALL FOR LRC
3558 014574 013704 014744      MOV      EXCRC,R4
3559 014600 004737 014716      JSR      PC,XOR          ;XOR CRC TO DATA
3560 014604 013737 014742 014746 MOV      XORS,EXLRC      ;SAVE EXPT LRC
3561 014612 000207              RTS      PC              ;RETURN
3562 014614 005704              CLP:    TST      R4        ;SEE IF 0 CHAR
3563 014616 001010              BNE      CLPE            ;IF NOT: BR
3564 014620 032737 000010 000552 BIT      #10,UDES        ;SEE IF EVEN PARITY
3565 014626 001404              BEQ      CLPE            ;IF NOT: BR
3566 014630 012704 000420      MOV      #420,R4        ;SET 0 CHAR EVEN PARITY
3567 014634 005201              INC      R1              ;BUMP POINTER
3568 014636 000207              RTS      PC              ;RETURN
  
```

3569	014640	005046		CLPE:	CLR	-(SP)		:CLEAR WORKING SPACE ON STACK
3570	014642	106304		1\$:	ASLB	R4		:SHIFT DATA
3571	014644	005516			ADC	(SP)		:ADDUP # OF 1 BITS
3572	014646	105704			TSTB	R4		:BRANCH IF ALL 0'S LEFT
3573	014650	001374			BNE	1\$		
3574	014652	112104			MOVB	(R1)+,R4		
3575	014654	042704	177400		BIC	#177400,R4		
3576	014660	106026			RORB	(SP)+		:BRANCH IF ODD # OF 1 BITS
3577	014662	103405			BCS	CLP2		
3578	014664	032737	000010 000552		BIT	#10,UDES		:SEE IF SHOULD BE EVEN PARITY
3579	014672	001406			BEQ	CLP3		:IF NOT: BR
3580	014674	000207			RTS	PC		:ELSE EXIT
3581	014676	032737	000010 000552	CLP2:	BIT	#10,UDES		:SEE IF SHOULD BE ODD PARITY
3582	014704	001001			BNE	CLP3		:IF NOT: BR
3583	014706	000207			RTS	PC		:ELSE EXIT
3584	014710	052704	000400	CLP3:	BIS	#400,R4		:SET PARITY BIT
3585	014714	000207			RTS	PC		
3586								
3587	014716	010446		XOR:	MOV	R4,-(SP)		
3588	014720	043716	014742		BIC	XORS,(SP)		
3589	014724	040437	014742		BIC	R4,XORS		:XOR SUBROUTINE: R4 WITH XORS
3590	014730	052637	014742		BIS	(SP)+,XORS		
3591	014734	013704	014742		MOV	XORS,R4		
3592	014740	000207			RTS	PC		
3593								
3594	014742	000000		XORS:	0			:XOR SAVE
3595	014744	000000		EXCRC:	0			:EXPECTED CRC
3596	014746	000000		EXLRC:	0			:EXPECTED LRC
3597								


```
3598
3599
3600
3601
3602
3603
3604
3605
3606
3607
3608
3609
3610
3611
3612
3613 014750 005037 000656          DCHK: CLR      BBC          ;CLEAR BAD RECORD CNTR
3614 014754 005037 000704          CLR      DERFL        ;CLEAR DATA ERROR FLAG
3615 014760 013705 000556          MOV      FMCNT,R5     ;LOAD CHAR COUNT
3616 014764 032737 000020 000552  BIT      #20,UDES     ;SEE IF CORE DUMP
3617 014772 001401          BEQ      DCHK0        ;IF NOT: BR
3618 014774 006205          ASR      R5           ;R5 = FC/2
3619 014776 012701 026344          DCHK0: MOV      #WDATA,R1 ;SET WRITE DATA ADDR
3620 015002 012702 032352          MOV      #RDATA,R2   ;SET READ DATA ADDR
3621 015006 032737 000010 000552  BIT      #10,UDES     ;SEE IF EVEN PARITY
3622 015014 001430          BEQ      DFOCO        ;IF NOT: BR
3623 015016 032737 000020 000552  BIT      #20,UDES     ;SEE IF CORE DUMP PARITY
3624 015024 001024          BNE     DFOCO        ;IF SO: BR
3625 015026 032737 002000 000552  BIT      #2000,UDES   ;SEE IF PE MODE
3626 015034 001020          BNE     DFOCO        ;IF SO: BR
3627 015036 105711          DFOF:  TSTB      (R1)  ;SEE IF 0 CHAR
3628 015040 001404          BEQ      DFOD        ;IF SO: BR
3629 015042 005201          INC      R1          ;BUMP POINTER
3630 015044 005205          DFOE:  INC      R5          ;SEE IF DONE
3631 015046 001373          BNE     DFOF        ;IF NOT: BR
3632 015050 000406          BR       DFOC        ;ELSE CONTINUE
3633 015052 112721 000020          DFOD:  MOVB     #20,(R1)+ ;SET 20 IN PLACE OF 0
3634 015056 012737 177777 013652  MOV      #-1,PATS    ;SET PATTERN GENERATE FLAG
3635 015064 000767          BR       DFOE
3636 015066 013705 000556          DFOC:  MOV      FMCNT,R5 ;RESET CHAR CNT
3637 015072 012701 026344          MOV      #WDATA,R1   ;RESET DATA ADDRESS
3638 015076 005737 000562          DFOCO: TST      RDCMD   ;SEE IF READ REVERSE
3639 015102 001462          BEQ     DFO          ;IF NOT: BR
3640 015104 013704 000556          DFOB:  MOV      FMCNT,R4 ;GET FRAME COUNT
3641 015110 005404          NEG      R4          ;SET TO WHOLE NUMBER
3642 015112 032737 000020 000552  BIT      #20,UDES     ;SEE IF CORE DUMP
3643 015120 001402          BEQ     DFOB0       ;IF NOT: BR
3644 015122 000241          CLC
3645 015124 006004          ROR      R4          ;SET TO FC/2
3646 015126 060401          DFOB0: ADD      R4,R1      ;POINT TO START OF WRITE DATA
3647 015130 060402          ADD      R4,R2      ;POINT TO START OF READ DATA
3648 015132 032737 000001 000552  BIT      #1,FMCNT    ;SEE IF ODD FRAME COUNT
3649 015140 001401          BEQ     DFOA        ;IF NOT: BR
3650 015142 105722          TSTB    (R2)+       ;BUMP POINTER
3651 015144 032737 000020 000552  DFOA:  BIT      #20,UDES ;SEE IF CORE DUMP
3652 015152 001431          BEQ     DFOA4       ;IF NOT: BR
3653 015154 000241          CLC
```

3654	015156	132742	000001		BITB	#1,-(R2)		:SEE IF BIT 0 = 1
3655	015162	001401			BEQ	DF0A0		:IF NOT: BR
3656	015164	000261			SEC			
3657	015166	106012		DF0A0:	RORB	(R2)		
3658	015170	000241			CLC			
3659	015172	132712	000001		BITB	#1,(R2)		
3660	015176	001401			BEQ	DF0A1		
3661	015200	000261			SEC			
3662	015202	106012		DF0A1:	RORB	(R2)		:POSITION BITS FOR REVERSE CORE DUMP
3663	015204	000241			CLC			
3664	015206	132712	000001		BITB	#1,(R2)		
3665	015212	001401			BEQ	DF0A2		
3666	015214	000261			SEC			
3667	015216	106012		DF0A2:	RORB	(R2)		
3668	015220	000241			CLC			
3669	015222	132712	000001		BITB	#1,(R2)		
3670	015226	001401			BEQ	DF0A3		
3671	015230	000261			SEC			
3672	015232	106012		DF0A3:	RORB	(R2)		
3673	015234	005202			INC	R2		:RESET POINTER
3674	015236	124142		DF0A4:	CMPB	-(R1),-(R2)		:TEST DATA CHARACTER
3675	015240	001010			BNE	DF1		:IF NOT GOOD: BR
3676	015242	105037	000656		CLRB	BBC		:CLEAR BAD RECORD COUNTER
3677	015246	000411			BR	DF2		
3678	015250	122122		DF0:	CMPB	(R1)+,(R2)+		:CHECK DATA
3679	015252	001003			BNE	DF1		:IF BAD: BR
3680	015254	105037	000656		CLRB	BBC		:CLEAR BAD RECORD CNTR
3681	015260	000404			BR	DF2		
3682	015262	004737	016020	DF1:	JSR	PC,DRPKF		:GO GET DROPS AND PICKS
3683	015266	004737	015354		JSR	PC,DERR		:GO DO PRINT
3684	015272	005205		DF2:	INC	R5		:BUMP CHAR CNTR
3685	015274	001404			BEQ	DF3		:IF DONE ALL: BR
3686	015276	005737	000562		TST	RDCMD		:SEE IF READ REVERSE
3687	015302	001762			BEQ	DF0		:IF NOT: BR
3688	015304	000717			BR	DF0A		:ELSE CONTINUE READ REV
3689	015306	005037	000664	DF3:	CLR	HDRFL		:CLEAR HEADER FLAG
3690	015312	005737	000704		TST	DERFL		:SEE IF HAD DATA ERROR
3691	015316	001415			BEQ	DFX		:IF NOT: BR
3692	015320	005737	000706		TST	SERFL		
3693	015324	001012			BNE	DFX		:IF NOT DATA ERROR ONLY: BR
3694	015326	013704	000674		MOV	UNP,R4		
3695	015332	005737	000562		TST	RDCMD		:SEE IF READ REVERSE
3696	015336	001003			BNE	DF4		:IF SO: BR
3697	015340	005264	001124		INC	DATER1(R4)		:BUMP DATA ERROR FORWARD COUNTER
3698	015344	000402			BR	DFX		
3699	015346	005264	001164	DF4:	INC	DEREV1(R4)		:BUMP REVERSE DATA ERROR
3700	015352	000207		DFX:	RTS	PC		:EXIT
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 015354 032777 002000 163226
3731 015362 001057
3732 015364 005237 000670
3733 015370 005737 000664
3734 015374 001006
3735 015376 004737 022012
3736 015402 000004 023522
3737 015406 004737 020250
3738 015412 000004 023541
3739 015416 010203
3740 015420 162703 032352
3741 015424 005303
3742 015426 005737 000562
3743 015432 001402
3744 015434 010503
3745 015436 005103
3746 015440 104400
3747 015442 000004 023527
3748 015446 005737 000562
3749 015452 001402
3750 015454 111103
3751 015456 000401
3752 015460 114103
3753 015462 004737 023304
3754 015466 000004 023534
3755 015472 005737 000562
3756 015476 001402
3757 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  
DERRO: INC PFLG ;SET PRINT FLAG  
TST HDRFL ;SEE IF HAVE PRINTED HEADER  
BNE DERROA ;IF SO: BR  
JSR PC,PAPRT ;PRINT CYCLE NUMBER  
TYPE,MSG1 ;TYPE DATA ERROR TAG '*DE'  
JSR PC,FRPRT ;PRINT F OR R  
DERROA: TYPE,MSG4 ;TYPE CHAR # TAG 'CN'  
MOV R2,R3 ;POINT TO CHAR  
SUB #RDATA,R3  
DEC R3  
TST RDCMD ;SEE IF READ REVERSE  
BEQ DERROB ;IF NOT: BR  
MOV R5,R3 ;GET CHAR NUMBER  
COM R3  
DERROB: TYPOCT ;PRINT CHAR NUMBER  
TYPE,MSG2 ;TYPE GOOD CHAR TAG 'G'  
TST RDCMD ;SEE IF READ REVERSE  
BEQ DERROC ;IF NOT: BR  
MOVB (R1),R3 ;GET CHAR  
BR DERROD  
DERROC: MOVB -(R1),R3 ;LOAD EXPECTED DATA  
DERROD: 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
```

3758	015502	000401			BR	DERR2			
3759	015504	114203			DERR1: MOV	-(R2),R3			
3760	015506	004737	023304		DERR2: JSR	PC,DOUT			;PRINT BAD CHAR
3761	015512	005737	000562			TST	RDCMD		;BRANCH IF READ
3762	015516	001001				BNE	DERR4		;REVERSE
3763	015520	122122			DERR3: CMPB	(R1)+,(R2)+			;RESET POINTERS
3764	015522	105237	000656		DERR4: INCB	BBC			;BUMP BAD RECORD CNTR
3765	015526	122737	000010	000656		CMPB	#10,BBC		;SEE IF BLD BTH
3766	015534	001107				BNE	DEREX		;IF NOT: BR
3767	015536	032777	002000	163044		BIT	#2000,@SWR		;SEE IF PRINT INHIBIT
3768	015544	001002				BNE	1\$;IF SO: BR
3769	015546	000004	023622			TYPE,MSG15			;TYPE 'BAD RECORD'
3770	015552	105037	000656		1\$: CLR	BBC			;RESET BAD RECORD CNTR
3771	015556	105237	000657			INCB	BBC+1		;BUMP AMOUNT
3772	015562	122737	000003	000657		CMPB	#3,BBC+1		;SEE IF HAD 3 BLD BTHS
3773	015570	101047				BHI	DERR4B		;IF NOT: BR
3774	015572	022705	177767			CMP	#177767,R5		;SEE IF ON LAST EIGHT CHARS
3775	015576	101464				BLOS	DERR6		;IF SO: BR
3776	015600	012705	177767			MOV	#177767,R5		;SET CHAR CNTR TO 8
3777	015604	005737	000562			TST	RDCMD		;SEE IF READ REVERSE
3778	015610	001416				BEQ	DERR4A		;IF NOT: BR
3779	015612	012701	026344			MOV	#WDATA,R1		;GET START OF BUFFER
3780	015616	012702	032352			MOV	#RDATA,R2		;GET START OF BUFFER
3781	015622	062701	000010			ADD	#10,R1		
3782	015626	062702	000010			ADD	#10,R2		;POINT TO START +10
3783	015632	032737	000001	000556		BIT	#1,FMCNT		;SEE IF ODD FRAME COUNT
3784	015640	001445				BEQ	DEREX		;IF NOT: BR
3785	015642	105722				TSTB	(R2)+		;BUMP POINTER
3786	015644	000443				BR	DEREX		
3787	015646	013737	000556	000644	DERR4A: MOV	FMCNT,TEMP1			;LOAD CHAR COUNT
3788	015654	005437	000644			NEG	TEMP1		;++B
3789	015660	162737	000010	000644		SUB	#10,TEMP1		;POINT TO BUFFER -8
3790	015666	013701	000644			MOV	TEMP1,R1		;POINT TO NEXT CHAR
3791	015672	062701	026344			ADD	#WDATA,R1		;POINT TO NEXT WRITE CHAR
3792	015676	013702	000644			MOV	TEMP1,R2		;POINT TO END OF READ DATA -8 FORWARD
3793	015702	062702	032352			ADD	#RDATA,R2		;POINT TO NEXT CHAR
3794	015706	000422				BR	DEREX		;EXIT
3795	015710	062705	000024		DERR4B: ADD	#24,R5			;SKIP 20 CHARS
3796	015714	103415				BCS	DERR6		;IF EXCEED RECORD SIZE: BR
3797	015716	005737	000562			TST	RDCMD		;SEE IF READ REVERSE
3798	015722	001405				BEQ	DERR5		;IF NOT: BR
3799	015724	162701	000024			SUB	#24,R1		
3800	015730	162702	000024			SUB	#24,R2		;RESET POINTERS
3801	015734	000407				BR	DEREX		
3802	015736	062701	000024		DERR5: ADD	#24,R1			;SKIP 20 CHARS
3803	015742	062702	000024			ADD	#24,R2		;SKIP FORWARD 20 CHARS
3804	015746	000402				BR	DEREX		
3805	015750	012705	177777		DERR6: MOV	#-1,R5			;SET TO EOR
3806	015754	005777	162630		DEREX: TST	@SWR			;BRANCH IF NOT HALT ON ERROR
3807	015760	100012				BPL	DEREX1		
3808	015762	000000				HALT			
3809	015764	005737	000670			TST	PFLG		;SEE IF PRINTED
3810	015770	001006				BNE	DEREX1		;IF SO: BR
3811	015772	032777	002000	162610		BIT	#2000,@SWR		;SEE IF SHOULD PRINT
3812	016000	001002				BNE	DEREX1		;IF NOT: BR
3813	016002	000137	015364			JMP	DERRO		;ELSE PRINT

CZTEDCO TM03-TE16/TU77 DRT
CZTEDC.P11 10-MAY-79 10:32

MACY11 30A(1052) 10-MAY-79 10:46 ^{6 7} PAGE 84

SEQ 0084

3814 016006 005037 000670
3815 016012 005237 000704
3816 016016 000207
3817

DEREX1: CLR PFLG ;CLEAR FLAG
INC DERFL ;BUMP DATA ERROR FLAG
RTS PC ;RETURN

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
3867
3868
3869
3870
3871
3872
3873

016020 005037 000644
016024 005037 000646
016030 005037 000650
016034 111137 000644
016040 111237 000646
016044 013704 000674
016050 016437 000764
016056 016437 001004
016064 005737 000562
016070 001005
016072 124142
016074 112137 000644
016100 112237 000646
016104 004737 016116
016110 004737 016324
016114 000207
016116 113703 000644
016122 113704 000646
016126 140403
016130 001001
016132 000207
016134 012737 000010
016142 132703 000001
016146 001451
016150 105737 000650
016154 001014
016156 005277 162534
016162 100043
016164 032777 002000
016172 001402
016174 004737 022012
016200 004737 016370
016204 000413
016206 005277 162506
016212 100027
016214 032777 002000
016222 001402

000720
000716
000710
162416
162366

```
*****  
: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  
DPC0A: JSR PC,PAPRT ;PRINT CYCLE NUMBER  
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
```

3874	016224	004737	022012		JSR	PC,PAPRT		:PRINT CYCLE NUMBER
3875	016230	004737	016370		DPC1A: JSR	PC,DPPRT		:PRINT DROPS AND PICKS
3876	016234	013704	000674		DPC2A: MOV	UNP,R4		
3877	016240	016403	001004		MOV	DRP1(R4),R3		:SET DROP POINTER
3878	016244	016404	000764		MOV	PIK1(R4),R4		:SET PICK POINTER
3879	016250	012737	000010	000710	MOV	#10,BCNT		:SET NUMBER OF BITS
3880	016256	005023			DPC2B: CLR	(R3)+		:CLEAR DROPS
3881	016260	005024			CLR	(R4)+		:CLEAR PICK
3882	016262	005337	000710		DEC	BCNT		:SEE IF DONE
3883	016266	001373			BNE	DPC2B		:IF NOT: BR
3884	016270	000207			RTS	PC		:EXIT
3885	016272	000241			DPC2: CLC			
3886	016274	106003			RORB	R3		:GET NEXT BIT
3887	016276	005337	000710		DEC	BCNT		:SEE IF DONE
3888	016302	001407			BEQ	DPC3		
3889	016304	062737	000002	000720	ADD	#2,BPKP		
3890	016312	062737	000002	000716	ADD	#2,BDPP		
3891	016320	000710			BR	DPC0		:CONTINUE
3892	016322	000207			DPC3: RTS	PC		:RETURN
3893	016324	013704	000674		PICK: MOV	UNP,R4		:GET UNIT POINTER
3894	016330	016437	000764	000720	MOV	PIK1(R4),BPKP		:SET PICK POINTER
3895	016336	016437	001004	000716	MOV	DRP1(R4),BDPP		:SET DROP POINTER
3896	016344	113704	000644		MOVB	TEMP1,R4		:R4 = GOOD CHAR
3897	016350	113703	000646		MOVB	TEMP2,R3		:R3 = BAD CHAR
3898	016354	112737	000001	000650	MOVB	#1,TEMP3		:SET PICK FLAG
3899	016362	004737	016126		JSR	PC,DPC		:GO CHECK PICKS
3900	016366	000207			RTS	PC		:EXIT
3901	016370	000004	024133		DPPRT: TYPE,MSG26			:TYPE 'DROPS'
3902	016374	013704	000674		MOV	UNP,R4		
3903	016400	016437	001004	000716	MOV	DRP1(R4),BDPP		:SET DROP POINTER
3904	016406	016437	000764	000720	MOV	PIK1(R4),BPKP		:SET PICK POINTER
3905	016414	062737	000016	000716	ADD	#16,BDPP		
3906	016422	062737	000016	000720	ADD	#16,BPKP		
3907	016430	012737	000010	000710	MOV	#10,BCNT		:SET NUMBER TO PRINT
3908	016436	017703	162254		DPPRT0: MOV	@BDPP,R3		
3909	016442	104400			TYPOCT			:PRINT DROPS
3910	016444	005337	000710		DEC	BCNT		:SEE IF DONE
3911	016450	001404			BEQ	DPPRT1		:IF NOT: BR
3912	016452	162737	000002	000716	SUB	#2,BDPP		:BUMP POINTER
3913	016460	000766			BR	DPPRT0		:CONTINUE FOR ALL 8 BITS
3914	016462	012737	000010	000710	DPPRT1: MOV	#10,BCNT		:SET NUMBER TO PRINT
3915	016470	000004	024144		TYPE,MSG27			:TYPE 'PICKS'
3916	016474	017703	162220		DPPRT2: MOV	@BPKP,R3		
3917	016500	104400			TYPOCT			:PRINT PICKS
3918	016502	005337	000710		DEC	BCNT		:SEE IF DONE
3919	016506	001404			BEQ	DPPRTX		:IF SO: BR
3920	016510	162737	000002	000720	SUB	#2,BPKP		:BUMP POINTER
3921	016516	000766			BR	DPPRT2		:CONTINUE FOR ALL 8 BITS
3922	016520	000207			DPPRTX: RTS	PC		:RETURN

3923
 3924
 3925
 3926
 3927
 3928
 3929
 3930
 3931
 3932
 3933
 3934
 3935
 3936
 3937
 3938
 3939
 3940
 3941
 3942
 3943
 3944
 3945
 3946
 3947

```

*****
: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 STATUS (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.
*****
  
```

```

3948 016522 013703 000556          ERCHK: MOV      FMCNT,R3          ;GET FRAME COUNT
3949 016526 032703 000001          BIT      #1,R3           ;SEE IF ODD
3950 016532 001401                   BEQ      1$              ;IF NOT: BR
3951 016534 005303                   DEC      R3              ;BUMP COUNT
3952 016536 005403                   NEG      R3
3953 016540 032737 000020 000552  1$:  BIT      #20,UDES        ;SEE IF CORE DUMP
3954 016546 001401                   BEQ      2$              ;IF NOT: BR
3955 016550 006203                   ASR      R3              ;SET TO FC/2
3956 016552 032737 000010 000672  2$:  BIT      #10,MTC1       ;SEE IF WRITE OP
3957 016560 001413                   BEQ      4$              ;IF SO: BR
3958 016562 005737 000562          TST      RDCMD
3959 016566 001405                   BEQ      3$
3960 016570 012703 032352          MOV      #RDATA,R3
3961 016574 162703 000002          SUB      #2,R3           ;SET POINTER
3962 016600 000405                   BR       ER2
3963 016602 062703 032352          3$:  ADD      #RDATA,R3    ;BUILD EXPT READ ADDRESS
3964 016606 000402                   BR       ER2
3965 016610 062703 026344          4$:  ADD      #WDATA,R3    ;BUILD EXPT WRITE ADDRESS
3966
3967 016614 032777 040000 161702  ER2:  BIT      #40000,@ER       ;BRANCH IF NOT UNSAFE
3968 016622 001403                   BEQ      1$
3969 016624 005726                   TST      (SP)+           ;ADJUST STACK
3970 016626 000137 020312          JMP      OFFLINE         ;GO MARK UNIT OFFLINE
3971 016632 010337 020224          1$:  MOV      R3,CADER     ;SAVE ADDRESS
3972 016636 012704 000007          MOV      #7,R4
3973 016642 012701 020226          MOV      #BAER,R1
3974 016646 005021                   2$:  CLR      (R1)+        ;CLEAR FLAGS
3975 016650 005304                   DEC      R4
3976 016652 001375                   BNE      2$
3977 016654 020377 161634          CMP      R3,@BA          ;SEE IF ADDRESS OK
3978 016660 001402                   BEQ      3$              ;IF SO: BR
  
```


3979	016662	005237	020226			INC	BAER	:SET BUS ADDRESS ERROR
3980	016666	032737	000010	000672	3\$:	BIT	#10,MTC1	:SEE IF WRITE OPER
3981	016674	001006				BNE	5\$:IF NOT: BR
3982	016676	005777	161614		4\$:	TST	@FC	:SEE IF FC=0
3983	016702	001440				BEQ	ER3	:IF SO: BR
3984	016704	005237	020234			INC	FCER	:SET FC ERROR
3985	016710	000435				BR	ER3	
3986	016712	032737	000040	000672	5\$:	BIT	#40,MTC1	:SEE IF SPACE OPER
3987	016720	001766				BEQ	4\$:IF SO: BR
3988	016722	005737	000676			TST	TMFLG	:SEE IF TM TIME
3989	016726	001011				BNE	7\$:IF SO: BR
3990	016730	013703	000556			MOV	FMCNT,R3	
3991	016734	005403				NEG	R3	:R3 = EXPT RECORD SIZE
3992	016736	020377	161554		6\$:	CMP	R3,@FC	:SEE IF FC = EXPT
3993	016742	001420				BEQ	ER3	:IF SO: BR
3994	016744	005237	020234			INC	FCER	:SET FC ERROR FLAG
3995	016750	000415				BR	ER3	
3996	016752	032737	002000	000552	7\$:	BIT	#2000,UDES	:SEE IF PE
3997	016760	001346				BNE	4\$:IF SO: BR
3998	016762	005737	000562			TST	RDCMD	:SEE IF READ REVERSE
3999	016766	001003				BNE	8\$:IF SO: BR
4000	016770	012703	000002			MOV	#2,R3	
4001	016774	000760				BR	6\$:LOOK FOR EXPT = 2
4002	016776	012703	000001		8\$:	MOV	#1,R3	
4003	017002	000755				BR	6\$:GO CHECK FC FOR TM
4004								
4005	017004	032777	160000	161476	ER3:	BIT	#160000,@C1	:SEE IF COUNT ERROR
4006	017012	001437				BEQ	ER4	
4007	017014	017703	161500			MOV	@CS,R3	:GET CONT STATUS REG
4008	017020	042703	000307			BIC	#307,R3	:MASK OUT IR,OR,UNIT NO. & SEE IF OTHER ERRORS
4009	017024	001406				BEQ	1\$:IF NOT: BR
4010	017026	005737	000676			TST	TMFLG	:SEE IF TAPE MARK TIME
4011	017032	001425				BEQ	3\$:IF NOT: BR
4012	017034	042703	001000			BIC	#1000,R3	:MASK MISSED TRANS & BR IF OTHER ERRORS
4013	017040	001022				BNE	3\$	
4014	017042	032777	060000	161440	1\$:	BIT	#60000,@C1	:SEE IF EITHER TRE OR MCPE
4015	017050	001420				BEQ	ER4	:IF NOT: BR
4016	017052	005737	000676			TST	TMFLG	:SEE IF TM TIME
4017	017056	001413				BEQ	3\$:IF NOT: BR
4018	017060	017703	161440			MOV	@ER,R3	:GET ERROR REGISTER
4019	017064	032737	000010	000552		BIT	#10,UDES	:SEE IF EVEN PARITY
4020	017072	001402				BEQ	2\$:IF NOT: BR
4021	017074	042703	000100			BIC	#100,R3	:MASK PAR
4022	017100	042703	001000		2\$:	BIC	#1000,R3	:MASK FCE
4023	017104	001402				BEQ	ER4	:IF NO ERRORS EXCEPT FCE: BR
4024	017106	005237	020230		3\$:	INC	CONER	:SET CONT ERROR FLAG
4025								
4026	017112	032777	040000	161402	ER4:	BIT	#40000,@DS	:SEE IF DRIVE ERROR
4027	017120	001420				BEQ	ER6	:IF NOT: BR
4028	017122	005737	000676			TST	TMFLG	:SEE IF TAPE MARK TIME
4029	017126	001413				BEQ	2\$:IF NOT: BR
4030	017130	017703	161370			MOV	@ER,R3	:GET ER
4031	017134	032737	000010	000552		BIT	#10,UDES	:SEE IF EVEN PARITY
4032	017142	001402				BEQ	1\$:IF NOT: BR
4033	017144	042703	000100			BIC	#100,R3	:MASK PAR
4034	017150	042703	001000		1\$:	BIC	#1000,R3	:MASK OUT FCE & BRANCH IF

4035	017154	001402				BEQ	ER6		;NO OTHER ERRORS
4036	017156	005237	020232		2\$:	INC	DRVER		;SET DRIVER ERROR FLAG
4037									
4038	017162	013737	014744	020246	ER6:	MOV	EXCRC,CRCSV		;SAVE EXPECTED CRC
4039	017170	013737	014746	020244		MOV	EXLRC,LRC\$V		;AND EXPECTED LRC
4040	017176	032737	002000	000552		BIT	#2000,UDES		
4041	017204	001062				BNE	ERPT		;IF IN PE MODE: BR
4042	017206	032777	020000	161374		BIT	#20000,@SWR		;SEE IF NO DATA CHECK
4043	017214	001056				BNE	ERPT		;IF NOT: BR (ALLOW READ OF UNKNOWN TAPES)
4044	017216	032737	000040	000672		BIT	#40,MTC1		;SEE IF WRITE OR READ OP
4045	017224	001452				BEQ	ERPT		;IF NOT: BR
4046	017226	005737	000676			TST	TMFLG		;SEE IF TAPE MARK TIME
4047	017232	001405				BEQ	1\$;IF NOT: BR
4048	017234	005037	014744			CLR	EXCRC		
4049	017240	012737	000023	014746		MOV	#23,EXLRC		;SET CRC/LRC FOR TM
4050	017246	032737	000060	000552	1\$:	BIT	#60,UDES		;SEE IF FORMAT 14
4051	017254	001036				BNE	ERPT		;IF NOT: BR
4052	017256	017703	161246			MOV	@CC,R3		;GET CRC CHARACTER
4053	017262	042703	177000			BIC	#177000,R3		
4054	017266	023703	014744			CMR	EXCRC,R3		
4055	017272	001402				BEQ	2\$;IF CRC GOOD: BR
4056	017274	005237	020240			INC	CRCER		;SET ERROR FLAG
4057	017300	017703	161230		2\$:	MOV	@MR,R3		;GET LRC
4058	017304	000303				SWAB	R3		
4059	017306	005703				TST	R3		
4060	017310	100002				BPL	3\$		
4061	017312	052703	000400			BIS	#400,R3		
4062	017316	042703	177000		3\$:	BIC	#177000,R3		
4063	017322	023703	014746			CMR	EXLRC,R3		
4064	017326	001411				BEQ	ERPT		;IF LRC GOOD: BR
4065	017330	010337	020242			MOV	R3,ACTLRC		;SAVE ACTUAL LRC
4066	017334	005237	020236			INC	LRCER		;SET LRC ERROR FLAG
4067	017340	005737	000562			TST	RDCMD		;SEE IF READ REVERSE
4068	017344	001402				BEQ	ERPT		;IF NOT: BR
4069	017346	005037	020236			CLR	LRCER		;ELSE CLEAR LRC ERROR
4070	017352	012703	000006		ERPT:	MOV	#6,R3		
4071	017356	005037	000706			CLR	SERFL		;CLEAR ERROR FLAG
4072	017362	005037	000722			CLR	ERSAV		
4073	017366	012704	020226			MOV	#BAER,R4		
4074	017372	005724			ERPTT:	TST	(R4)+		;SEE IF ANY ERROR
4075	017374	001004				BNE	ERPTG		;IF SO: BR
4076	017376	005303				DEC	R3		
4077	017400	001374				BNE	ERPTT		
4078	017402	000137	020170			JMP	ERPX1		
4079	017406	005237	000706		ERPTG:	INC	SERFL		;SET ERROR FLAG
4080	017412	017737	161106	000722		MOV	@ER,ERSAV		;SAVE ERROR REGISTER
4081	017420	032777	002000	161162		BIT	#2000,@SWR		;SEE IF PRINT
4082	017426	001420				BEQ	ERPT0		;IF SO: BR
4083	017430	022737	000002	000712		CMR	#2,RTYFL		;SEE IF READ RETRY
4084	017436	001006				BNE	ERPTG1		;IF NOT: BR
4085	017440	013703	000702			MOV	RTCNT,R3		
4086	017444	005203				INC	R3		;BUMP RETRY COUNT
4087	017446	020337	000604			CMR	R3,RETRY		;SEE IF LAST RETRY
4088	017452	001404				BEQ	ERPT0		;IF SO: BR
4089	017454	022737	000002	020232	ERPTG1:	CMR	#2,DRVER		;SEE IF TM STATUS ERROR
4090	017462	001402				BEQ	ERPT0		;IF SO: BR

4091	017464	000137	020050		JMP	ERPX0	
4092	017470	005237	000670		ERPT0: INC	PFLG	
4093	017474	004737	022012		JSR	PC,PAPRT	;PRINT HEADER
4094	017500	013737	000652	017510	MOV	EMADDR,1\$;GET ADDRESS OF ERROR MSG HEADER
4095	017506	000004			TYPE		
4096	017510	000000			1\$: .WORD	0	;ADDRESS OF ERROR MESSAGE HEADER
4097	017512	004737	020250		JSR	PC,FRPRT	;PRINT F OR R
4098	017516	005737	000676		TST	TMFLG	
4099	017522	001406			BEQ	ERPT1	
4100	017524	022737	025012	000652	CMP	#MSG54,EMADDR	
4101	017532	001402			BEQ	ERPT1	
4102	017534	000004	025030		TYPE,MSG56		;TYPE 'TM'
4103	017540	005737	020230		ERPT1: TST	CONER	
4104	017544	001412			BEQ	ERPT2	;IF NO CONT ERROR: BR
4105	017546	000004	023737		TYPE,MSG23		;TYPE 'CS1'
4106	017552	017703	160732		MOV	@C1,R3	
4107	017556	104400			TYPOCT		;PRINT CONTROL 1
4108	017560	000004	023764		TYPE,MSG23D		;TYPE CS TAG
4109	017564	017703	160730		MOV	@CS,R3	
4110	017570	104400			TYPOCT		;PRINT CONT STATUS
4111	017572	005737	020232		ERPT2: TST	DRVER	
4112	017576	001412			BEQ	ERPT3	;IF SO DRIVE ERROR: BR
4113	017600	000004	023772		TYPE,MSG23E		;TYPE DS TAG
4114	017604	017703	160712		MOV	@DS,R3	
4115	017610	104400			TYPOCT		;PRINT DRIVE STATUS
4116	017612	000004	023777		TYPE,MSG23F		;TYPE ER TAG
4117	017616	017703	160702		MOV	@ER,R3	
4118	017622	104400			TYPOCT		;PRINT DRIVE ERROR
4119	017624	005737	020226		ERPT3: TST	BAER	
4120	017630	001412			BEQ	ERPT4	;IF NO BA ERROR: BR
4121	017632	000004	023752		TYPE,MSG23B		;TYPE BA TAG
4122	017636	017703	160652		MOV	@BA,R3	
4123	017642	104400			TYPOCT		;PRINT BUS ADDRESS
4124	017644	000004	023520		TYPE,DASH		
4125	017650	013703	020224		MOV	CADER,R3	
4126	017654	104400			TYPOCT		;PRINT EXPT BUS ADDRESS
4127	017656	005737	020234		ERPT4: TST	FCER	
4128	017662	001405			BEQ	ERPT5	;IF NO FC ERROR: BR
4129	017664	000004	023757		TYPE,MSG23C		;TYPE FC TAG
4130	017670	017703	160622		MOV	@FC,R3	
4131	017674	104400			TYPOCT		;PRINT FRAME COUNT
4132	017676	000004	023745		ERPT5: TYPE,MSG23A		;TYPE WC TAG
4133	017702	017703	160604		MOV	@WC,R3	
4134	017706	104400			TYPOCT		;PRINT WORD COUNT
4135	017710	005737	020240		TST	CRCER	
4136	017714	001414			BEQ	ERPT5A	;IF NO CRC ERROR: BR
4137	017716	000004	025055		TYPE,MSG58		;TYPE CRC TAG
4138	017722	017703	160602		MOV	@CC,R3	
4139	017726	042703	177000		BIC	#177000,R3	
4140	017732	104400			TYPOCT		;PRINT ACTUAL CRC
4141	017734	000004	023520		TYPE,DASH		
4142	017740	013703	014744		MOV	EXCRC,R3	
4143	017744	104400			TYPOCT		;PRINT EXPECTED CRC
4144	017746	005737	020236		ERPT5A: TST	LRCER	
4145	017752	001412			BEQ	ERPT6	;IF NO LRC ERROR: BR
4146	017754	000004	025063		TYPE,MSG59		;TYPE LRC ERR TAG

4147	017760	013703	020242		MOV	ACTLRC,R3	
4148	017764	104400			TYPOCT		;PRINT ACTUAL LRC
4149	017766	000004	023520		TYPE,DASH		
4150	017772	013703	014746		MOV	EXLRC,R3	
4151	017776	104400			TYPOCT		;PRINT EXPECTED LRC
4152	020000	005737	020232		ERPT6: TST	DRVER	
4153	020004	001420			BEQ	ERPT7	;IF NO DRIVE ERROR: BR
4154	020006	032737	002000	000552	BIT	#2000,UDES	
4155	020014	001414			BEQ	ERPT7	;IF NO PE: BR
4156	020016	017704	160502		MOV	@ER,R4	
4157	020022	042704	075477		BIC	#75477,R4	;MASK OUT ALL BUT BITS 15,10,7,6
4158	020026	001407			BEQ	ERPT7	;IF NO CONDITIONALS SET: BR
4159	020030	000004	024011		TYPE,MSG23H		;TYPE CC TAG
4160	020034	017703	160470		MOV	@CC,R3	
4161	020040	042703	177000		BIC	#177000,R3	;MASK CC
4162	020044	104400			TYPOCT		;PRINT CHECK CHARACTERS
4163	020046	000240			ERPT7: NOP		
4164	020050	005777	160534		ERPX0: TST	@SWR	;BRANCH IF NOT HALT ON ERROR
4165	020054	100012			BPL	ERPX	
4166	020056	000000			HALT		
4167	020060	005737	000670		TST	PFLG	;SEE IF HAVE PRINTED
4168	020064	001006			BNE	ERPX	;IF SO: BR
4169	020066	032777	002000	160514	BIT	#2000,@SWR	;SEE IF SHOULD PRINT
4170	020074	001002			BNE	ERPX	;IF NOT: BR
4171	020076	000137	017470		JMP	ERPT0	;PRINT ERROR
4172	020102	005037	000670		ERPX: CLR	PFLG	
4173	020106	005737	000566		TST	CRCC	;BRANCH IF CRC ERROR
4174	020112	001007			BNE	1\$;CORRECTION DESIRED
4175	020114	012777	000040	160376	MOV	#40,@CS	;ELSE INIT
4176	020122	013777	000550	160370	MOV	DVN,@CS	;RESET DRIVE NUMBER
4177	020130	000414			BR	2\$	
4178	020132	012777	000011	160350	1\$: MOV	#11,@C1	;DRIVE CLEAR
4179	020140	017704	160362		MOV	@AS,R4	
4180	020144	010477	160356		MOV	R4,@AS	;CLEAR AS
4181	020150	013704	000510		MOV	C1,R4	
4182	020154	005204			INC	R4	
4183	020156	152714	000100		BISB	#100,(R4)	;RESET TRE
4184	020162	013777	000552	160352	2\$: MOV	UDES,@TC	;RESET TC
4185	020170	032737	000040	000672	ERPX1: BIT	#40,MTC1	
4186	020176	001411			BEQ	ERPX2	;IF NOT READ/WRITE OP: BR
4187	020200	005737	000676		TST	TMFLG	
4188	020204	001406			BEQ	ERPX2	;IF NOT TM TIME: BR
4189	020206	013737	020246	014744	MOV	CRCSV,EXCRC	;RESTORE CRC
4190	020214	013737	020244	014746	MOV	LRCSV,EXLRC	;RESTORE LRC
4191	020222	000207			ERPX2: RTS	PC	;EXIT
4192	020224	000000			CADER: 0		;EXPT ADDRESS SAVE
4193	020226	000000			BAER: 0		
4194	020230	000000			CONER: 0		
4195	020232	000000			DRVER: 0		
4196	020234	000000			FCER: 0		
4197	020236	000000			LRCER: 0		
4198	020240	000000			CR CER: 0		
4199	020242	000000			ACTLRC: 0		
4200	020244	000000			LRCSV: 0		
4201	020246	000000			CRCSV: 0		
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
 4232
 4233
 4234
 4235
 4236
 4237
 4238

```

:*****
: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$:     JMP REOT
  
```

```
4239  
4240  
4241  
4242  
4243  
4244  
4245  
4246  
4247  
4248  
4249  
4250  
4251  
4252  
4253  
4254  
4255  
4256  
4257  
4258  
4259  
4260  
4261  
4262  
4263  
4264  
4265  
4266  
4267  
*****  
;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.  
*****
```

```
4268 020372 005037 000644 TAPG: CLR TEMP1  
4269 020376 013777 000550 160114 MOV DVN,@CS ;SET DRIVE NO.  
4270 020404 032777 040000 160112 1$: BIT #40000,@ER ;SEE IF UNIT SAFE  
4271 020412 001402 BEQ TAPG3 ;IF SO: BR  
4272 020414 000137 020312 JMP OFFLINE ;GO MARK UNIT OFF-LINE  
4273 020420 032777 020000 160074 TAPG3: BIT #20000,@DS ;SEE IF PIP RESET  
4274 020426 001410 BEQ TAPG3F ;IF SO: BR  
4275 020430 004737 022012 JSR PC,PAPRT ;PRINT HEADER  
4276 020434 000004 026044 TYPE,MSG116 ;TYPE MSG  
4277 020440 032777 020000 160054 1$: BIT #20000,@DS  
4278 020446 001374 BNE 1$ ;AWAIT PIP RESET  
4279 020450 022737 000026 000672 TAPG3F: CMP #26,MTC1 ;SEE IF WRITE TM  
4280 020456 001003 BNE TAPG3A ;IF NOT: BR  
4281 020460 012704 177777 MOV #-1,R4 ;ELSE SET FC FOR -1  
4282 020464 000406 BR TAPG3B  
4283 020466 013704 000556 TAPG3A: MOV FMCNT,R4  
4284 020472 032704 000001 BIT #1,R4  
4285 020476 001401 BEQ TAPG3B  
4286 020500 005304 DEC R4  
4287 020502 000261 TAPG3B: SEC  
4288 020504 006004 ROR R4 ;SET WC = FC/2 FOR NORMAL FORMAT  
4289 020506 032737 000020 000552 BIT #20,UDES ;SEE IF CORE DUMP FORMAT  
4290 020514 001402 BEQ TAPG3C ;IF NOT: BR  
4291 020516 000261 SEC  
4292 020520 006004 ROR R4 ;SET WC = FC/4 FOR CORE DUMP  
4293 020522 010477 157764 TAPG3C: MOV R4,@WC ;SET WORD COUNT  
4294 020526 012777 000011 157754 MOV #11,@C1 ;DRIVE CLEAR
```



```

4329
4330
4331 020734 017746 157654
4332 020740 042716 000200
4333 020744 122716 000003
4334 020750 001005
4335 020752 000005
4336 020754 005077 157626
4337 020760 000137 000200
4338 020764 122716 000001
4339 020770 001015
4340 020772 022737 000176 000610
4341 021000 001014
4342 021002 012737 177570 000610
4343 021010 004737 022432
4344 021014 000004 026162
4345 021020 004737 022454
4346 021024 022716 000007
4347 021030 001005
4348 021032 012737 000176 000610
4349 021040 004737 022346
4350 021044 022716 000002
4351 021050 001041
4352 021052 004737 022432
4353 021056 005237 013444
4354 021062 004737 013226
4355 021066 032777 000040 157514
4356 021074 001425
4357 021076 000004 024652
4358 021102 013703 000602
4359 021106 104400
4360 021110 012705 000602
4361 021114 012701 000007
4362 021120 012702 177777
4363 021124 012703 002000
4364 021130 004737 022476
4365 021134 004737 022454
4366 021140 005726
4367 021142 012716 010770
4368 021146 000002
4369 021150 004737 022454
4370 021154 005726
4371 021156 000002
4372
4373
4374 021160 000240
4375 021162 042777 000037 157344
4376 021170 013716 000662
4377 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$: CMP #7,(SP) ;BRANCH IF NOT ^G
    BNE 4$
3$: MOV #SWREG,SWR ;INVOKE SOFTWARE SWR
    JSR PC,GTSWR ;GET SWITCHES
4$: 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$: JSR PC,.RESTORE
6$: TST (SP)+ ;POP CHARACTER OFF THE STACK
    RTI ;RETURN

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



```
4378
4379
4380
4381
4382
4383
4384
4385
4386
4387 021176 000004 025477
4388 021202 012705 000740
4389 021206 012701 000002
4390 021212 012702 000001
4391 021216 012703 000000
4392 021222 004737 022476
4393 021226 005037 000550
4394 021232 004737 021340
4395 021236 000004 025447
4396 021242 000004 024756
4397 021246 013703 000550
4398 021252 104400
4399 021254 000004 026335
4400 021260 000004 024400
4401 021264 012700 000742
4402 021270 012003
4403 021272 100402
4404 021274 104400
4405 021276 000774
4406 021300 004737 021524
4407 021304 004737 021656
4408 021310 022737 000007 000550
4409 021316 001403
4410 021320 005237 000550
4411 021324 000742
4412 021326 005737 000740
4413 021332 001335
4414 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
      TYPOCT          ;PRINT DRIVE #
      TYPE,SPACE
      TYPE,MSG32
      MOV  #UN1,R0
1$:   MOV  (R0)+,R3
      BMI  2$
      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)
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
```

```

4415
4416
4417
4418 021340 005037 004730 HRDS: CLR REOTC ;CLEAR EOT UNIT CNTR
4419 021344 012777 000040 157146 MOV #40,@CS ;INIT
4420 021352 013777 000550 157140 MOV DVN,@CS ;SET DRIVE
4421 021360 005777 157124 TST @C1 ;ACCESS DRIVE
4422 021364 032777 010000 157126 BIT #10000,@CS ;TEST FOR NON-EXISTANT DRIVE
4423 021372 001403 BEQ 2$ ;IF DRIVE AVAIL: BR
4424 021374 005726 1$: TST (SP)+ ;RESET STACK POINTER
4425 021376 000137 021310 JMP ASEQ4 ;GO SEE IF TRIED ALL DRIVES
4426 021402 017700 157130 2$: MOV @DT,R0 ;++B GET CONTENTS OF DRIVE TYPE REG
4427 021406 042700 002007 BIC #2007,R0 ;++B CLEAR SPR AND SPEED BITS
4428 021412 022700 140050 CMP #140050,R0 ;++B BRANCH IF NOT TMO3 MAGTAPE DRIVE
4429 021416 001366 BNE 1$
4430 021420 005000 CLR R0
4431 021422 012701 000742 MOV #UN1,R1 ;SET START OF SLAVE TABLE
4432 021426 005737 003034 TST CHNFLG ;BRANCH IF NOT IN CHAIN MODE
4433 021432 001410 BEQ 3$
4434 021434 122737 000006 000041 CMPB #6,@#41 ;BRANCH IF NOT LOADED VIA TMDP
4435 021442 001004 BNE 3$
4436 021444 005737 000550 TST DVN ;BRANCH IF NOT DRIVE 0
4437 021450 001001 BNE 3$
4438 021452 005200 INC R0 ;DO NOT TEST SLAVE 0
4439 021454 010077 157062 3$: MOV R0,@TC ;SELECT SLAVE
4440 021460 032777 010000 157034 BIT #10000,@DS ;SEE IF SLAVE AVAIL FOR TEST(MOL)
4441 021466 001404 BEQ 4$ ;IF NOT: BR
4442 021470 062737 000401 004730 ADD #401,REOTC ;INCREMENT UNITS TO TEST COUNT
4443 021476 010021 MOV R0,(R1)+ ;LOAD SLAVE # INTO SLAVE TABLE
4444 021500 005200 4$: INC R0 ;STEP TO NEXT SLAVE
4445 021502 022700 000010 CMP #10,R0 ;BRANCH IF ALL SLAVE NOT DONE
4446 021506 001362 BNE 3$
4447 021510 005737 004730 5$: TST REOTC ;SEE IF FOUND ANY SLAVES
4448 021514 001727 BEQ 1$ ;IF NOT: BR
4449 021516 012711 177777 MOV #-1,(R1) ;TERMINATE SLAVE TABLE
4450 021522 000207 RTS PC ;RETURN TO SEQ

```

```
4451  
4452  
4453 ;SUBROUTINE TO SELECT NRZ AUTO TEST MODE*****  
4454 021524 005037 000654 AMOD1: CLR BLCNTR ;ASSURE BLOCK COUNTER IS 0  
4455 021530 012701 000742 MOV #UN1,R1 ;GET START OF SLAVE TABLE  
4456 021534 052721 001700 1$: BIS #1700,(R1)+ ;SET ALL SLAVE TO NRZ,NORM,ODD  
4457 021540 022711 177777 CMP #-1,(R1) ;LOOP UNTIL REACED END OF TABLE  
4458 021544 001373 BNE 1$  
4459 021546 004737 004744 JSR PC,RWDA ;GO REWIND ALL AVAIL SLAVES  
4460 021552 012737 000006 000736 MOV #6,ABLCNT ;SET NUMBER OF BLOCKS FOR MODE 1  
4461 021560 012737 174000 000556 MOV #-4000,FMCNT ;SET FC = 4000  
4462 021566 012737 000100 000554 MOV #100,RCNT ;SET REC CNTR = 100  
4463 021574 012737 000001 000560 MOV #1,PATRN ;SELECT PATTERN 1  
4464 021602 005037 000564 CLR TMEX ;ASSURE NO TMK  
4465 021606 005037 000570 CLR INTRF ;ASSURE NORMAL READ  
4466 021612 004737 003352 JSR PC,STAUTO ;GO DO AUTO MODE 1  
4467 021616 012737 000010 000560 MOV #10,PATRN ;SELECT PATTERN 10  
4468 021624 004737 003352 JSR PC,STAUTO ;GO DO PATTERN 10  
4469 021630 012737 000014 000560 MOV #14,PATRN ;SELECT PATTERN 14  
4470 021636 004737 003352 JSR PC,STAUTO  
4471 021642 012737 177777 000560 3$: MOV #-1,PATRN ;SELECT AUTO RANDOM DATA  
4472 021650 004737 003352 JSR PC,STAUTO  
4473 021654 000207 RTS ;RETURN TO SEQ
```

```
4474  
4475  
4476  
4477 021656 005037 000654          AMOD2: CLR      BLCNTR      ;CLEAR BLOCK CNTR  
4478 021662 012701 000742          MOV      #UN1,R1    ;SET START OF SLAVE TABLE  
4479 021666 042711 001700          1$:  BIC      #1700,(R1) ;CLEAR NRZ  
4480 021672 052721 002300          BIS      #2300,(R1)+ ;SET TO PE NORM, ODD  
4481 021676 022711 177777          CMP      #-1,(R1)   ;LOOP UNTIL END OF TABLE  
4482 021702 001371  
4483 021704 004737 004744          JSR      PC,RWDA    ;REWIND ALL SLAVES  
4484 021710 012737 000006 000736  MOV      #6,ABL CNT ;SET AUTO BLOCK COUNT  
4485 021716 012737 174000 000556  MOV      #-4000,FMCNT ;SET FC = 4000  
4486 021724 012737 000100 000554  MOV      #100,RCNT  ;SET REC CNTR TO 100  
4487 021732 012737 000010 000560  MOV      #10,PATRN  ;SELECT PATTERN 10  
4488 021740 004737 003352          JSR      PC,STAUTO  ;GO DO AUTO SEQ  
4489 021744 012737 000014 000560  MOV      #14,PATRN  ;SELECT PATTERN 14  
4490 021752 004737 003352          JSR      PC,STAUTO  
4491 021756 012737 000015 000560  MOV      #15,PATRN  ;SELECT PATTERN 15  
4492 021764 004737 003352          JSR      PC,STAUTO  
4493 021770 012737 177777 000736  MOV      #-1,ABL CNT ;FORCE TO END OF TAPE  
4494 021776 012737 177777 000560  MOV      #-1,PATRN  ;SELECT AUTO RANDOM DATA  
4495 022004 004737 003352          JSR      PC,STAUTO  
4496 022010 000207          3$:  RTS      PC      ;RETURN TO SEQ  
4497  
4498
```

```
4499
4500
4501
4502
4503
4504
4505
4506
4507
4508
4509
4510
4511
4512
4513
4514
4515 022012 000004 024754
4516 022016 013703 000550
4517 022022 104400
4518 022024 000004 024400
4519 022030 013703 000552
4520 022034 042703 177770
4521 022040 104400
4522 022042 000004 023522
4523 022046 013703 000552
4524 022052 000303
4525 022054 042703 177770
4526 022060 104400
4527 022062 000004 025071
4528 022066 005003
4529 022070 032737 000010 000552
4530 022076 001401
4531 022100 005203
4532 022102 104400
4533 022104 000004 025075
4534 022110 013703 000552
4535 022114 006003
4536 022116 006003
4537 022120 006003
4538 022122 006003
4539 022124 042703 177760
4540 022130 104400
4541 022132 000004 023565
4542 022136 005737 000560
4543 022142 100003
4544 022144 000004 023655
4545 022150 000403
4546 022152 013703 000560
4547 022156 104400
4548 022160 000004 023607
4549 022164 013703 000654
4550 022170 104400
4551 022172 000004 023615
4552 022176 010003
4553 022200 032737 000010 000672
4554 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
MOV UDES,R3
BIC #177770,R3 ;PRINT SLAVE NUMBER
TYPOCT ;TYPE DENSITY TAG '*DE'
TYPE,MSG1
MOV UDES,R3
SWAB R3
BIC #177770,R3 ;PRINT DENSITY
TYPOCT ;TYPE PARITY TAG '*P'
TYPE,MSG61
CLR R3
BIT #10,UDES
BEQ PAPRT0 ;SET PARITY INDICATOR = EVEN
INC R3 ;PRINT PARITY BIT STATE
PAPRT0: TYPOCT ;TYPE FORMAT TAG '*F'
TYPE,MSG62
MOV UDES,R3
ROR R3
ROR R3 ;POSTION FORMAT BITS
ROR R3
ROR R3
BIC #177760,R3 ;PRINT FORMAT
TYPOCT ;TYPE PATTERN # TAG '*PATRN'
TYPE,MSG8 TST PATRN ;BRANCH IF NOT RANDOM PATTERN
BPL PAPRTC ;TYPE 'R' FOR RANDOM
PAPRTA: TYPE,MSG17
BR PAPRTD
PAPRTC: MOV PATRN,R3 ;PRINT PATRN NUMBER
TYPOCT ;TYPE BLOCK # TAG '*BN'
PAPRTD: TYPE,MSG13
MOV BLCNTR,R3 ;PRINT NUMBER
TYPOCT ;TYPE RECORD # TAG '*RN'
TYPE,MSG14
MOV R0,R3 ;GET # OF RECORDS LEFT TO PROCESS
BIT #10,MTC1 ;SEE IF WRITE OPERATION
BEQ PAPRT1 ;IF SO: BR
```

```

4555 022210 022737 000030 000672      CMP      #30,MTC1      ;BRANCH IF SPACE FORWARD
4556 022216 001412                      BEQ      PAPRT1
4557 022220 005737 000562                      TST      RDCMD      ;BRANCH IF READ FORWARD
4558 022224 001407                      BEQ      PAPRT1
4559 022226 022737 000032 000672      CMP      #32,MTC1      ;BRANCH IF NOT SPACE REVERSE
4560 022234 001007                      BNE      PAPRT3
4561 022236 005737 000712                      TST      RTYFL      ;BRANCH IF NOT RETRYING
4562 022242 001404                      BEQ      PAPRT3
4563 022244 013703 000554      PAPRT1: MOV      RCNT,R3      ;GET # OF RECORDS TO PROCESS
4564 022250 160003      PAPRTY: SUB      R0,R3      ;FORM RECORD NUMBER
4565 022252 005203                      INC      R3      ;++MP ADJUST RECORD NUMBER IN FORWARD DIRECTION
4566 022254 104400      PAPRT3: TYPOCT      ;PRINT RECORD NUMBER
4567 022256 000004 023520      TYPE,DASH      ;TYPE A DASH '-'
4568 022262 013703 000554      MOV      RCNT,R3
4569 022266 104400      TYPOCT
4570 022270 000004 023560      TYPE,MSG7
4571 022274 013703 000556      MOV      FMCNT,R3
4572 022300 005403      NEG      R3
4573 022302 104400      TYPOCT
4574 022304 012737 000001 000664      MOV      #1,HDRFL
4575 022312 000207      RTS      PC      ;RETURN
4576
  
```

```
4577
4578
4579
4580
4581
4582
4583
4584
4585
4586 022314 063737 000630 000626 RANG: ADD RANSAV,RANBAS
4587 022322 063737 000626 000630 ADD RANBAS,RANSAV ;GET NEW NUMBER
4588 022330 023701 000630 CMP RANSAV,R1 ;SEE IF NUMBER TOO BIG
4589 022334 101367 BHI RANG ;IF SO: BR
4590 022336 020237 000630 CMP R2,RANSAV ;SEE IF NUMBER TOO SMALL
4591 022342 101364 BHI RANG ;IF SO: BR
4592 022344 000207 RTS PC ;EXIT
4593
4594 ;SUBROUTINE TO GET NEW SOFTWARE SWR
4595
4596 022346 022737 000176 000610 GTSWR: CMP #SWREG,SWR ;BRANCH IF SOFTWARE SWR
4597 022354 001025 BNE 1$ ;NOT INVOKED
4598 022356 004737 022432 JSR PC,..SAVE ;SAVE REGISTERS ON THE STACK
4599 022362 000004 023500 TYPE,$MSWR
4600 022366 017703 156216 MOV @SWR,R3 ;GET CURRENT SWR
4601 022372 104400 TYPOCT
4602 022374 000004 023510 TYPE,$MNEW ;REQUEST NEW SWR SETTING
4603 022400 013705 000610 MOV SWR,R5 ;TTR ROUTINE RETURNS VALUE TO (R5)
4604 022404 012701 000007 MOV #7,R1 ;LIMIT RESPONSE TO 7 CHARS
4605 022410 012702 177777 MOV #177777,R2 ;BETWEEN 0 AND 177777
4606 022414 012703 000000 MOV #0,R3
4607 022420 004737 022476 JSR PC,TTR ;GET RESPONSE
4608 022424 004737 022454 JSR PC,..RESTORE ;RESTORE REGISTERS
4609 022430 000207 1$: RTS PC ;RETURN
4610
4611 ;;ROUTINE TO SAVE REGISTERS ON THE STACK
4612 022432 010546 .SAVE: MOV %5,-(SP) ;;R5 IS SAVED AT 12(SP)
4613 022434 010446 MOV %4,-(SP) ;;R4 IS SAVED AT 10(SP)
4614 022436 010346 MOV %3,-(SP) ;;R3 IS SAVED AT 6(SP)
4615 022440 010246 MOV %2,-(SP) ;;R2 IS SAVED AT 4(SP)
4616 022442 010146 MOV %1,-(SP) ;;R1 IS SAVED AT 2(SP)
4617 022444 010046 MOV %0,-(SP) ;;R0 IS SAVED AT (SP)
4618 022446 016646 000014 MOV 14(SP),-(SP) ;;PUSH RETURN PC ON THE STACK
4619 022452 000207 RTS PC ;;RETURN TO CALLER
4620
4621 ;;ROUTINE TO RESTORE REGISTERS SAVED ON THE STACK
4622 022454 012666 000014 .RESTORE:MOV (SP)+,14(SP) ;;STORE RETURN PC ON STACK
4623 022460 012600 MOV (SP)+,%0
4624 022462 012601 MOV (SP)+,%1
4625 022464 012602 MOV (SP)+,%2
4626 022466 012603 MOV (SP)+,%3
4627 022470 012604 MOV (SP)+,%4
4628 022472 012605 MOV (SP)+,%5
4629 022474 000207 RTS PC ;;RETURN
4630
```

```

4631
4632
4633
4634
4635
4636
4637
4638
4639
4640
4641
4642
4643
4644
4645
4646
4647
4648 022476 010146
4649 022500 011601
4650 022502 005037 000644
4651 022506 005000
4652 022510 004737 022722
4653 022514 122737 000003 000642
4654 022522 001003
4655 022524 000005
4656 022526 000137 000200
4657 022532 122737 000015 000642 11$:
4658 022540 001004
4659 022542 005737 000644
4660 022546 001455
4661 022550 000447
4662 022552 122737 000025 000642 2$:
4663 022560 001003
4664 022562 000004 024155
4665 022566 000744
4666 022570 122737 000177 000642 21$:
4667 022576 001010
4668 022600 000241
4669 022602 006000
4670 022604 006200
4671 022606 006200
4672 022610 000004 026112
4673 022614 005201
4674 022616 000734
4675 022620 122737 000060 000642 3$:
4676 022626 101027
4677 022630 122737 000070 000642 4$:
4678 022636 101423
4679 022640 005237 000644 5$:
4680 022644 006300
4681 022646 006300
4682 022650 006300
4683 022652 042737 177770 000642
4684 022660 053700 000642
4685 022664 005301
4686 022666 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$: CMPB #60, TIB ;SEE IF CHAR IS LESS THAN 0
BHI TIBR
4$: CMPB #70, TIB ;SEE IF CHAR IS GREATER THAN 7
BLOS TIBR
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

```


4687	022670	020002		6\$:	CMP	R0,R2		:SEE IF EXCEEDED MAXIMUM LIMIT
4688	022672	101005			BHI	TINER		
4689	022674	020300		7\$:	CMP	R3,R0		:SEE IF BELOW MINIMUM LIMIT
4690	022676	101003			BHI	TINER		
4691	022700	010015		8\$:	MOV	R0,(R5)		:LOAD VALUE
4692	022702	005726		9\$:	TST	(SP)+		:POP CHAR COUNT OFF STACK
4693	022704	000207			RTS	PC		:EXIT
4694								
4695	022706	000004	024646	TINER:	TYPE,#MSG43			:TYPE '?'
4696	022712	005726			TST	(SP)+		:POP CHAR COUNT OFF STACK
4697	022714	162716	000020		SUB	#20,(SP)		:RESET SP TO START OF VALUE ROUTINE
4698	022720	000207			RTS	PC		:REDO VALUE ENTRY

4699
4700
4701
4702
4703
4704
4705
4706
4707
4708
4709
4710
4711
4712
4713
4714
4715
4716
4717
4718
4719
4720
4721
4722
4723
4724
4725
4726
4727
4728
4729
4730
4731
4732
4733
4734
4735
4736
4737
4738
4739
4740
4741
4742
4743
4744
4745

022722 005277 155664
022726 105777 155660
022732 100375
022734 017737 155654 000642
022742 042737 177600 000642
022750 022737 000015 000642
022756 001003
022760 000004 024155
022764 000402
022766 000004 000642
022772 000207

022774 010446
022776 010346
023000 017604 000004
023004 062766 000002 000004
023012 111437 000640
023016 001431
023020 122724 000045
023024 001403
023026 004737 023110
023032 000767

023034 112737 000015 000640
023042 004737 023110
023046 012703 000006
023052 005037 000640
023056 004737 023110
023062 005303
023064 001372
023066 112737 000012 000640
023074 004737 023110
023100 000744
023102 012603
023104 012604
023106 000002

023110 105777 155502
023114 100375
023116 113777 000640 155474
023124 000207

```

;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
    
```

```

4746                                     ;OCTAL OUTPUT SUBROUTINE*****
4747
4748 023126 005037 023302      OCTP:  CLR      OFL      ;CLEAR FLAG FOR LEADING ZERO
4749 023132 010304              MOV      R3,R4      ;SEE IF NUMBER IS ZERO
4750 023134 001003              BNE      1$         ;IF NOT ZERO: BR
4751 023136 000004 026337      TYPE,DIGIT0
4752 023142 000434              BR       4$         ;SPACE AND EXIT
4753 023144 100004              1$:  BPL      3$         ;BRANCH IF MSD IS A '0'
4754 023146 012704 000001      MOV      #1,R4
4755 023152 004737 023242      JSR      PC,OCTPG   ;PRINT 1
4756 023156 006004              3$:  ROR      R4
4757 023160 006004              ROR      R4
4758 023162 006004              ROR      R4         ;POSITION DIGIT
4759 023164 006004              ROR      R4
4760 023166 000304              SWAB     R4
4761 023170 004737 023242      JSR      PC,OCTPG   ;PRINT DIGIT 2
4762 023174 006004              ROR      R4
4763 023176 000304              SWAB     R4
4764 023200 004737 023242      JSR      PC,OCTPG   ;PRINT DIGIT 3
4765 023204 006104              ROL      R4
4766 023206 006104              ROL      R4
4767 023210 000304              SWAB     R4
4768 023212 004737 023242      JSR      PC,OCTPG   ;PRINT DIGIT 4
4769 023216 006004              ROR      R4
4770 023220 006004              ROR      R4
4771 023222 006004              ROR      R4
4772 023224 004737 023242      JSR      PC,OCTPG   ;PRINT DIGIT 5
4773 023230 004737 023242      JSR      PC,OCTPG   ;PRINT DIGIT 6
4774 023234 000004 026335      4$:  TYPE,SPACE ;TYPE A SPACE
4775 023240 000002              RTI         ;EXIT
4776
4777 023242 042704 177770      OCTPG:  BIC      #177770,R4
4778 023246 001003              BNE      1$
4779 023250 005737 023302      TST      OFL
4780 023254 001410              BEQ      2$
4781 023256 005237 023302      1$:  INC      OFL
4782 023262 052704 000260      BIS      #260,R4
4783 023266 010437 000640      MOV      R4,TOB
4784 023272 004737 023110      JSR      PC,TOG
4785 023276 010304              2$:  MOV      R3,R4
4786 023300 000207              RTS      PC
4787 023302 000000      OFL:  0          ;FIRST CHAR FLAG
4788
4789
4790                                     ;DATA CHARACTER OUTPUT SUBROUTINE*****
4791
4792 023304 012704 000010      DOUT:  MOV      #10,R4      ;SET NUMBER TO PRINT
4793 023310 110346              MOVB     R3,-(SP)      ;GET CHAR TO OUTPUT
4794 023312 106316              1$:  ASLB     (SP)        ;BRANCH IF BIT IS A ZERO
4795 023314 103003              BCC      2$
4796 023316 000004 026341      TYPE,DIGIT1
4797 023322 000402              BR       3$
4798 023324 000004 026337      2$:  TYPE,DIGIT0
4799 023330 005304              3$:  DEC      R4
4800 023332 001367              BNE      1$
4801 023334 005726              TST      (SP)+        ;POP STACK

```

```
4802 023336 000207          RTS      PC
4803
4804 023340 113703 000651    DOUTD:  MOVB   TEMP3+1,R3
4805 023344 004737 023304      JSR     PC,DOUT
4806 023350 013703 000650      MOV     TEMP3,R3
4807 023354 004737 023304      JSR     PC,DOUT
4808 023360 000207          RTS      PC
4809
4810                          ;TU16 SERIAL NUMBER PRINT SUBROUTINE*****
4811
4812 023362 017703 155152    SNPT:   MOV     @SN,R3          ;GET CONTENTS OF SERIAL # REG
4813 023366 000004 023575      TYPE,MSG9          ;TYPE SN TAG
4814 023372 010304          MOV     R3,R4
4815 023374 000304          SWAB    R4
4816 023376 006004          ROR     R4
4817 023400 006004          ROR     R4
4818 023402 006004          ROR     R4
4819 023404 006004          ROR     R4
4820 023406 004737 023454    JSR     PC,SNPG      ;PRINT FIRST DIGIT
4821 023412 010304          MOV     R3,R4
4822 023414 000304          SWAB    R4
4823 023416 004737 023454    JSR     PC,SNPG      ;PRINT SECOND DIGIT
4824 023422 010304          MOV     R3,R4
4825 023424 006004          ROR     R4
4826 023426 006004          ROR     R4
4827 023430 006004          ROR     R4
4828 023432 006004          ROR     R4
4829 023434 004737 023454    JSR     PC,SNPG      ;PRINT THIRD DIGIT
4830 023440 010304          MOV     R3,R4
4831 023442 004737 023454    JSR     PC,SNPG      ;PRINT FOURTH DIGIT
4832 023446 000004 024155    TYPE,MSG28         ;TYPE <CR><LF>
4833 023452 000207          RTS      PC          ;EXIT
4834 023454 012737 000260 000640 SNPG:  MOV     #260,TOB      ;SET NUMBER BASE
4835 023462 042704 177760      BIC     #177760,R4   ;MASK NUMBER
4836 023466 050437 000640      BIS     R4,TOB       ;BUILD DIGIT
4837 023472 004737 023110      JSR     PC,TOG       ;GO TYPE
4838 023476 000207          RTS      PC          ;RETURN
4839
```

4840
4841 :ERROR MESSAGES*****
4842
4843 023500 051445 051127 036440 \$MSWR: .ASCIZ /%SWR = /
4844 023506 000040
4845 023510 047040 053505 036440 \$MNEW: .ASCIZ / NEW = /
4846 023516 000040
4847 023520 000055 DASH: .ASCIZ /-/
4848 023522 042052 020105 000 MSG1: .ASCIZ /*DE /
4849 023527 045 035507 000040 MSG2: .ASCIZ /%G; /
4850 023534 041045 020073 000 MSG3: .ASCIZ /%B; /
4851 023541 045 047103 000040 MSG4: .ASCIZ /%CN /
4852 023546 053452 020105 000 MSG5: .ASCIZ /*WE /
4853 023553 052 042522 000040 MSG6: .ASCIZ /*RE /
4854 023560 051052 020123 000 MSG7: .ASCIZ /*RS /
4855 023565 052 040520 051124 MSG8: .ASCIZ /*PATRN /
4856 023572 020116 000
4857 023575 123 035116 000040 MSG9: .ASCIZ /SN: /
4858 023602 051452 020105 000 MSG10: .ASCIZ /*SE /
4859 023607 045 041052 020116 MSG13: .ASCIZ /%*BN /
4860 023614 000
4861 023615 052 047122 000040 MSG14: .ASCIZ /*RN /
4862 023622 020045 020040 020040 MSG15: .ASCIZ /% BAD RECORD%/
4863 023630 020040 020040 041040
4864 023636 042101 051040 041505
4865 023644 051117 022504 000045
4866 023652 043040 000 MSG16: .ASCIZ / F /
4867 023655 040 000122 MSG17: .ASCIZ / R /
4868 023660 042440 052117 021440 MSG20: .ASCIZ / EOT # /
4869 023666 000040
4870 023670 047111 042524 041522 MSG21: .ASCIZ /INTERCHANGE READ? /
4871 023676 040510 043516 020105
4872 023704 042522 042101 020077
4873 023712 000
4874 023713 045 046111 042514 MSG22: .ASCIZ /%ILLEGAL BOT: HALT%/
4875 023720 040507 020114 047502
4876 023726 035124 044040 046101
4877 023734 022524 000
4878 023737 045 051503 020061 MSG23: .ASCIZ /%CS1 /
4879 023744 000
4880 023745 045 041527 000040 MSG23A: .ASCIZ /%WC /
4881 023752 041045 020101 000 MSG23B: .ASCIZ /%BA /
4882 023757 045 041506 000040 MSG23C: .ASCIZ /%FC /
4883 023764 041445 031123 000040 MSG23D: .ASCIZ /%CS2 /
4884 023772 042045 020123 000 MSG23E: .ASCIZ /%DS /
4885 023777 045 051105 000040 MSG23F: .ASCIZ /%ER /
4886 024004 040445 020123 000 MSG23G: .ASCIZ /%AS /
4887 024011 045 045503 000040 MSG23H: .ASCIZ /%CK /
4888 024016 042045 020102 000 MSG23I: .ASCIZ /%DB /
4889 024023 045 051115 000040 MSG23J: .ASCIZ /%MR /
4890 024030 042045 020124 000 MSG23K: .ASCIZ /%DT /
4891 024035 045 041524 000040 MSG23L: .ASCIZ /%TC /
4892 024042 047045 020117 047111 MSG24: .ASCIZ /%NO INTERRUPT%/
4893 024050 042524 051122 050125
4894 024056 022524 000
4895 024061 045 046123 053101 MSG25: .ASCIZ /%SLAVE UNSAFE-TEST DISCONTINUED ON SLAVE%/

4896	024066	020105	047125	040523	
4897	024074	042506	052055	051505	
4898	024102	020124	044504	041523	
4899	024110	047117	044524	052516	
4900	024116	042105	047440	020116	
4901	024124	046123	053101	022505	
4902	024132	000			
4903	024133	045	051104	050117	MSG26: .ASCIZ /%DROPS: /
4904	024140	035123	000040		
4905	024144	050045	041511	051513	MSG27: .ASCIZ /%PICKS: /
4906	024152	020072	000		
4907	024155	045	000		MSG28: .ASCIZ /%/
4908	024157	045	052045	047515	MSG30: .ASCIZ '%TM03-TE16/TU77 AUTO SEQUENCE (CZTEDCO)%';++B
4909	024164	026463	042524	033061	
4910	024172	052057	033525	020067	
4911	024200	052501	047524	051440	
4912	024206	050505	042525	041516	
4913	024214	020105	041450	052132	
4914	024222	042105	030103	022451	
4915	024230	000			
4916	024231	045	052045	030115	MSG31: .ASCIZ '%TM03-TE16/TU77 DATA RELIABILITY TEST (CZTEDCO)%';++B
4917	024236	026463	042524	033061	
4918	024244	052057	033525	020067	
4919	024252	040504	040524	051040	
4920	024260	046105	040511	044502	
4921	024266	044514	054524	052040	
4922	024274	051505	020124	041450	
4923	024302	052132	042105	030103	
4924	024310	022451	000		
4925	024313	124	050131	020105	MSG31A: .ASCIZ /TYPE <CR> TO TERMINATE ALL REQUESTS & ^C TO RESTART%/
4926	024320	041474	037122	052040	
4927	024326	020117	042524	046522	
4928	024334	047111	052101	020105	
4929	024342	046101	020114	042522	
4930	024350	052521	051505	051524	
4931	024356	023040	057040	020103	
4932	024364	047524	051040	051505	
4933	024372	040524	052122	000045	
4934	024400	046123	053101	020105	MSG32: .ASCIZ /SLAVE # = /
4935	024406	020043	020075	000	
4936	024413	104	047105	044523	MSG33: .ASCIZ /DENSITY = /
4937	024420	054524	036440	000040	
4938	024426	040520	044522	054524	MSG34: .ASCIZ /PARITY = /
4939	024434	036440	000040		
4940	024440	042522	047503	042122	MSG35: .ASCIZ /RECORD COUNT = /
4941	024446	041440	052517	052116	
4942	024454	036440	000040		
4943	024460	044103	051101	041440	MSG36: .ASCIZ /CHAR COUNT = /
4944	024466	052517	052116	036440	
4945	024474	000040			
4946	024476	040520	052124	051105	MSG37: .ASCIZ /PATTERN # = /
4947	024504	020116	020043	020075	
4948	024512	000			
4949	024513	123	047111	046107	MSG38: .ASCIZ /SINGLE PASS? /
4950	024520	020105	040520	051523	
4951	024526	020077	000		

4952	024531	103	041522	041440	MSG39:	.ASCIZ	/CRC CORRECTION (YES=1,NO=0)? /
4953	024536	051117	042522	052103			
4954	024544	047511	020116	054450			
4955	024552	051505	030475	047054			
4956	024560	036517	024460	020077			
4957	024566	000					
4958	024567	045	042445	052116	MSG40:	.ASCIZ	/%ENTER STALLS%READ = /
4959	024574	051105	051440	040524			
4960	024602	046114	022523	042522			
4961	024610	042101	036440	000040			
4962	024616	051127	052111	020105	MSG41:	.ASCIZ	/WRITE = /
4963	024624	020075	000				
4964	024627	124	051125	020116	MSG42:	.ASCIZ	/TURN AROUND = /
4965	024634	051101	052517	042116			
4966	024642	036440	000040				
4967	024646	037445	000045		MSG43:	.ASCIZ	/%?%/
4968	024652	042445	052116	051105	MSG44:	.ASCIZ	/%ENTER YOZZLE STALL = /
4969	024660	054440	055117	046132			
4970	024666	020105	052123	046101			
4971	024674	020114	020075	000			
4972	024701	045	051105	020122	MSG45:	.ASCIZ	/%ERR AMT /
4973	024706	046501	020124	000			
4974	024713	045	047516	020124	MSG49:	.ASCIZ	/%NOT AVAIL /
4975	024720	053101	044501	020114			
4976	024726	000					
4977	024727	045	046111	042514	MSG50:	.ASCIZ	/%ILLEGAL DRIVE TYPE /
4978	024734	040507	020114	051104			
4979	024742	053111	020105	054524			
4980	024750	042520	000040				
4981	024754	022445			MSG52:	.ASCII	/%%/
4982	024756	051104	053111	020105	MSG52A:	.ASCIZ	/DRIVE (TM03) # = /
4983	024764	052050	030115	024463			
4984	024772	021440	036440	000040			
4985	025000	047506	046522	052101	MSG53:	.ASCIZ	/FORMAT = /
4986	025006	036440	000040				
4987	025012	053452	020105	046524	MSG54:	.ASCIZ	/*WE TM/
4988	025020	000					
4989	025021	052	042523	052040	MSG55:	.ASCIZ	/*SE TM/
4990	025026	000115					
4991	025030	052040	000115		MSG56:	.ASCIZ	/ TM/
4992	025034	047045	047117	042455	MSG57:	.ASCIZ	/%NON-EXIST SLAVE/
4993	025042	044530	052123	051440			
4994	025050	040514	042526	000			
4995	025055	045	051103	020103	MSG58:	.ASCIZ	/%CRC /
4996	025062	000					
4997	025063	045	051114	020103	MSG59:	.ASCIZ	/%LRC /
4998	025070	000					
4999	025071	052	020120	000	MSG61:	.ASCIZ	/*P /
5000	025075	052	020106	000	MSG62:	.ASCIZ	/*F /
5001	025101	045	047452	044522	MSG64:	.ASCIZ	/%*ORIGINAL ERROR*/
5002	025106	044507	040516	020114			
5003	025114	051105	047522	025122			
5004	025122	000					
5005	025123	045	042522	051124	MSG65:	.ASCIZ	/%RETRY: /
5006	025130	035131	000040				
5007	025134	051452	020105	052122	MSG66:	.ASCIZ	/*SE RTRY /

5008	025142	054522	000040						
5009	025146	042452	040522	042523	MSG67:	.ASCIZ	/*ERASE/		
5010	025154	000							
5011	025155	045	042522	042522	MSG68:	.ASCIZ	/%REREV: /		
5012	025162	035126	000040						
5013	025166	040524	042520	046440	MSG69:	.ASCIZ	/TAPE MARK? /		
5014	025174	051101	037513	000040					
5015	025202	047045	047117	042455	MSG71:	.ASCIZ	/%NON-EXIST DRIVE/		
5016	025210	044530	052123	042040					
5017	025216	044522	042526	000					
5018	025223	045	042522	053506	MSG72:	.ASCIZ	/%REFWD: /		
5019	025230	035104	000040						
5020	025234	053445	042524	051122	MSG73:	.ASCIZ	/%WTERR: /		
5021	025242	020072	000						
5022	025245	045	042522	044507	MSG74:	.ASCIZ	/%REGISTER START = /		
5023	025252	052123	051105	051440					
5024	025260	040524	052122	036440					
5025	025266	000040							
5026	025270	042526	052103	051117	MSG75:	.ASCIZ	/VECTOR ADRS = /		
5027	025276	040440	051104	020123					
5028	025304	020075	000						
5029	025307	045	042504	042522	MSG76:	.ASCIZ	/%DEREV: /		
5030	025314	035126	000040						
5031	025320	042045	043105	042127	MSG77:	.ASCIZ	/%DEFWD: /		
5032	025326	020072	000						
5033	025331	045	047516	026516	MSG78:	.ASCIZ	/%NON-RETRYABLE WRITE ERROR: ER /		
5034	025336	042522	051124	040531					
5035	025344	046102	020105	051127					
5036	025352	052111	020105	051105					
5037	025360	047522	035122	042440					
5038	025366	020122	000						
5039	025371	045	047516	026516	MSG79:	.ASCIZ	/%NON-RETRYABLE READ ERROR: ER /		
5040	025376	042522	051124	040531					
5041	025404	046102	020105	042522					
5042	025412	042101	042440	051122					
5043	025420	051117	020072	051105					
5044	025426	000040							
5045	025430	042445	042116	047440	MSG100:	.ASCIZ	/%END OF PASS %/		
5046	025436	020106	040520	051523					
5047	025444	022440	000						
5048	025447	045	025045	025052	MSG101:	.ASCIZ	/%*****%/		
5049	025454	025052	025052	025052					
5050	025462	025052	025052	025052					
5051	025470	025052	025052	022452					
5052	025476	000							
5053	025477	101	052125	020117	MSG104:	.ASCIZ	/AUTO CONT.? /		
5054	025504	047503	052116	037456					
5055	025512	000040							
5056	025514	051045	041505	053117	MSG105:	.ASCIZ	/%RECOVERED/		
5057	025522	051105	042105	000					
5058	025527	052	040502	020104	MSG106:	.ASCIZ	/*BAD TAPE OVERFLOW/		
5059	025534	040524	042520	047440					
5060	025542	042526	043122	047514					
5061	025550	000127							
5062	025552	051045	053505	047111	MSG16A:	.ASCIZ	/%REWIND TAPE; RESTART AT BLOCK 1/		
5063	025560	020104	040524	042520					

5064	025566	020073	042522	052123	
5065	025574	051101	020124	052101	
5066	025602	041040	047514	045503	
5067	025610	030440	000		
5068	025613	045	047125	042522	MSG107: .ASCII /%UNRECOVERABLE BAD SPOT/
5069	025620	047503	042526	040522	
5070	025626	046102	020105	040502	
5071	025634	020104	050123	052117	
5072	025642	041045	042101	051040	.ASCIZ /%BAD RECORD LEFT ON TAPE%/
5073	025650	041505	051117	020104	
5074	025656	042514	052106	047440	
5075	025664	020116	040524	042520	
5076	025672	000045			
5077	025674	050052	051517	052111	MSG109: .ASCIZ /*POSITION LOST IN RETRY/
5078	025702	047511	020116	047514	
5079	025710	052123	044440	020116	
5080	025716	042522	051124	000131	
5081	025724	051445	051525	042520	MSG110: .ASCIZ /%SUSPECT BAD TAPE/
5082	025732	052103	041040	042101	
5083	025740	052040	050101	000105	
5084	025746	051045	050105	040505	MSG111: .ASCIZ /%REPEAT: /
5085	025754	035124	000040		
5086	025760	041040	042101	052040	MSG112: .ASCIZ / BAD TAPE SPOTS%/
5087	025766	050101	020105	050123	
5088	025774	052117	022523	000	
5089					
5090	026001	045	051440	043117	MSG113: .ASCIZ /% SOFT: /
5091	026006	035124	000040		
5092					
5093	026012	020045	040510	042122	MSG114: .ASCIZ /% HARD: /
5094	026020	020072	000		
5095					
5096	026023	045	040510	042122	MSG115: .ASCIZ /%HARD READ ERROR/
5097	026030	051040	040505	020104	
5098	026036	051105	047522	000122	
5099	026044	051445	040514	042526	MSG116: .ASCIZ /%SLAVE REWINDING: WILL RESTART AT BOT/
5100	026052	051040	053505	047111	
5101	026060	044504	043516	020072	
5102	026066	044527	046114	051040	
5103	026074	051505	040524	052122	
5104	026102	040440	020124	047502	
5105	026110	000124			
5106	026112	000134			MSG118: .ASCIZ /\
5107	026114	051045	046505	053117	MSG120: .ASCIZ /%REMOVE TMDP FROM SLAVE TO BE TESTED%/
5108	026122	020105	046524	050104	
5109	026130	043040	047522	020115	
5110	026136	046123	053101	020105	
5111	026144	047524	041040	020105	
5112	026152	042524	052123	042105	
5113	026160	000045			
5114	026162	044045	051101	053504	MSG121: .ASCIZ /%HARDWARE SWR IN USE%/
5115	026170	051101	020105	053523	
5116	026176	020122	047111	052440	
5117	026204	042523	000045		
5118	026210	047516	051440	040514	MSG122: .ASCIZ /NO SLAVES LEFT TO TEST: HALT%/
5119	026216	042526	020123	042514	

5120 026224 052106 052040 020117
5121 026232 042524 052123 020072
5122 026240 040510 052114 000045
5123 026246 040445 052125 026517
5124 026254 042523 035121 052040
5125 026262 051505 020124 044527
5126 026270 046114 051040 051505
5127 026276 040524 052122 000045
5128 026304 041445 051117 042522
5129 026312 052103 042105 050040
5130 026320 020105 040504 040524
5131 026326 042440 051122 051117
5132 026334 000
5133 026335 040 000
5134 026337 060 000
5135 026341 061 000
5136
5137 026344
5138 026344 000000
5139
5140 032352
5141 032352 000000
5142
5143 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	1802	3469#						
DAT15	014272	1803	3481#						
DAT2	014036	1792	3388#						
DAT3	014042	1793	3393#						
DAT3A	014050	3395#	3406						
DAT4	014066	1794	3404#						
DAT5	014076	1795	3411#						
DAT6	014104	1796	3416#						
DAT7	014112	1797	3421#						
DB	000532	1487#							
DCHK	014750	2679	2872	3613#					
DCHKO	014776	3617	3619#						
DEREV1	001164	1665#	1984	3699*					
DEREX	015754	3766	3784	3786	3794	3801	3804	3806#	
DEREX1	016006	3807	3810	3812	3814#				
DERFL	000704	1547#	3614*	3690	3815*				
DERR	015354	3683	3730#						
DERRO	015364	3732#	3813						
DERROA	015412	3734	3738#						
DERROB	015440	3743	3746#						
DERROC	015460	3749	3752#						
DERROD	015462	3751	3753#						
DERR1	015504	3756	3759#						
DERR2	015506	3758	3760#						
DERR3	015520	3763#							
DERR4	015522	3731	3762	3764#					
DERR4A	015646	3778	3787#						
DERR4B	015710	3773	3795#						
DERR5	015736	3798	3802#						
DERR6	015750	3775	3796	3805#					
DFX	015352	3691	3693	3698	3700#				
DFO	015250	3639	3678#	3687					
DFOA	015144	3649	3651#	3688					
DFOAO	015166	3655	3657#						
DFOA1	015202	3660	3662#						
DFOA2	015216	3665	3667#						
DFOA3	015232	3670	3672#						
DFOA4	015236	3652	3674#						
DFOB	015104	3640#							
DFOB0	015126	3643	3646#						
DFOC	015066	3632	3636#						
DFOCO	015076	3622	3624	3626	3638#				
DFOD	015052	3628	3633#						
DFOE	015044	3630#	3635						
DFOF	015036	3627#	3631						
DF1	015262	3675	3679	3682#					
DF2	015272	3677	3681	3684#					
DF3	015306	3685	3689#						
DF4	015346	3696	3699#						
DIGIT0	026337	4751	4798	5134#					
DIGIT1	026341	4796	5135#						
DOUT	023304	3753	3760	4792#	4805	4807			
DOUTD	023340	4804#							
DPC	016126	3855#	3899						
DPCG	016134	3856	3858#						
DPCO	016142	3859#	3891						

MSG14	023615	2452	4551	4861#				
MSG15	023622	3769	4862#					
MSG16	023652	4217	4866#					
MSG16A	025552	2037	5062#					
MSG17	023655	4214	4544	4867#				
MSG2	023527	3747	4849#					
MSG20	023660	2032	4868#					
MSG21	023670	3215	4870#					
MSG22	023713	2625	4874#					
MSG23	023737	4105	4878#					
MSG23A	023745	4132	4880#					
MSG23B	023752	4121	4881#					
MSG23C	023757	4129	4882#					
MSG23D	023764	4108	4883#					
MSG23E	023772	4113	4884#					
MSG23F	023777	4116	4885#					
MSG23G	024004	4886#						
MSG23H	024011	4159	4887#					
MSG23I	024016	4888#						
MSG23J	024023	4889#						
MSG23K	024030	4890#						
MSG23L	024035	4891#						
MSG24	024042	4323	4892#					
MSG25	024061	4227	4895#					
MSG26	024133	3901	4903#					
MSG27	024144	3915	4905#					
MSG28	024155	2460	2474	4664	4709	4832	4907#	
MSG3	023534	3754	4850#					
MSG30	024157	1841	3047	4908#				
MSG31	024231	3044	4916#					
MSG31A	024313	3051	3052*	4925#				
MSG32	024400	3102	4400	4518	4934#			
MSG33	024413	3133	4936#					
MSG34	024426	3145	4938#					
MSG35	024440	3174	4940#					
MSG36	024460	3184	4943#					
MSG37	024476	3196	4946#					
MSG38	024513	3224	4949#					
MSG39	024531	3233	4952#					
MSG4	023541	3738	4851#					
MSG40	024567	3246	4958#					
MSG41	024616	3255	4962#					
MSG42	024627	3264	4964#					
MSG43	024646	4695	4967#					
MSG44	024652	4357	4968#					
MSG45	024701	4972#						
MSG49	024713	4974#						
MSG5	023546	2227	4852#					
MSG50	024727	3127	4977#					
MSG52	024754	4515	4981#					
MSG52A	024756	3088	4396	4982#				
MSG53	025000	3156	4985#					
MSG54	025012	2277	4100	4987#				
MSG55	025021	2907	4989#					
MSG56	025030	4102	4991#					
MSG57	025034	3121	4992#					

RWND1A	005064	2127	2133	2137#										
RWND2	005102	2125	2140#											
RWND3	005106	2141#	2155											
RWND5	005172	2146	2151	2153#										
SCVFL	013444	1472*	3053	3242	3272*	3274#	4353*							
SERFL	000706	1548#	2249	2253*	2296	2337	2401*	2413	2423*	2432	2665	2680*	2730*	2743*
		2746	3692	4071*	4079*									
SN	000540	1490#	4812											
SNPG	023454	4820	4823	4829	4831	4834#								
SNPT	023362	3131	4812#											
SPACE	026335	4399	4774	5133#										
SPFLG	000572	1506#	2090	3225	3227									
STAL	000666	1540#	1930*	1937*	1940*	2046*	2134*	2244*	2402*	2523*	2545*	2685*	2808*	2817*
		2821*	2823*	2894*	2929*	2952*	4305*	4313*						
STALL	011654	1938	1941	2245	2403	2412	2524	2546	2686	2809	2824	2895	2930	2952#
		2953												
START	003022	1449	1473	1823#										
STARTA	003144	1456	1854#	2627										
STARTB	003150	1847	1855#											
STARTC	003136	1452	1850#											
STARTD	003250	1851	1874#											
STARTE	003242	1873#	2105											
START1	003406	1897#	1951											
START2	003522	1915	1917#											
START3	003536	1918	1920#											
START4	003552	1902	1921	1923#										
START7	003734	1924	1943	1950#	2055									
START8	003742	1951#												
STAR1A	003420	1899#												
STAR1B	003440	1900	1903#											
STAR1C	003474	1905	1907	1911#										
STAUT	003122	1468	1839	1845#										
STAUTO	003352	1890#	4466	4468	4470	4472	4488	4490	4492	4495				
STFLG	000640	1528#	1855	1856										
STP	003744	1946	1954#	2039										
STTBL	001044	1609#	1862	1863										
SWR	000610	1513#	1830*	1876	1880	1884*	1914	1917	1920	1942	2116	2223	2241	2254
		2265	2283	2300	2313	2321	2342	2358	2510	2513	2515	2519	2526	2534
		2536	2541	2628	2675	2682	2721	2739	2748	2770	2840	2868	2874	2905
		3730	3767	3806	3811	3865	3872	4042	4081	4164	4169	4316	4324	4340
		4342*	4348*	4355	4596	4600	4603							
SWREG	000176	1445#	1830	1876	1884	4340	4348	4596						
TAPG	020372	2232	2282	2410	2430	2613	2818	2839	2904	2927	4268#			
TAPG3	020420	4271	4273#											
TAPG3A	020466	4280	4283#											
TAPG3B	020502	4282	4285	4287#										
TAPG3C	020522	4290	4293#											
TAPG3D	020566	4297	4299	4301#										
TAPG3E	020612	4304	4306#											
TAPG3F	020450	4274	4279#											
TAPG4	020640	4311#	4312	4314										
TAPG5	020654	4315#												
TAPG6	020720	4317	4324#											
TAPG7	020730	4325	4327#											
TC	000542	1491#	1927*	2008*	2060*	2129*	2147*	2160*	2174*	3118*	4184*	4439*	3839*	3847*
TEMP1	000644	1531#	3108	3349*	3355	3359*	3787*	3788*	3789*	3790	3792	3836*		

CZTEDCO TMO3-TE16/TU77 DRT		MACY11 30A(1052) 10-MAY-79 10:46		H 10 PAGE 125										
CZTEDC.P11 10-MAY-79 10:32		CROSS REFERENCE TABLE -- USER SYMBOLS												
		SEQ 0124												
TEMP2	000646	3853	3896	4268*	4310*	4311*	4650*	4659	4679*	3360*	3367	3837*	3840*	3848*
		1532#	3101	3103*	3134*	3146*	3157*	3278*	3282					
		3854	3897											
TEMP3	000650	1533#	2350*	2356*	2367	2369*	2762*	2768*	2775	2777*	3838*	3861	3898*	4804
		4806												
TEND	004662	2092#	4414											
TIB	000642	1530#	4653	4657	4662	4666	4675	4677	4683*	4684	4705*	4706*	4707	4711
TINER	022706	4676	4678	4688	4690	4695#								
TINF	000636	1527#	1845*	1850*	1854*	2082*	3039	3244						
TINP	011750	1888	3039#											
TINPX	013436	3243	3245	3272#										
TINPO	012302	3097	3101#	3111	3122	3168								
TINPOB	012364	3109	3114#											
TINPOD	012434	3120	3123#											
TINPOE	012470	3126	3131#											
TINP1	012474	3133#												
TINP2	012534	3142#												
TINP2A	012612	3156#												
TINP2B	012652	3164#												
TINP2C	012700	3113	3166	3171#										
TINP3	012712	3174#												
TINP3A	013226	3233#	4354											
TINP4	013270	3041	3242#											
TKB	000614	1515#	4331	4705										
TKS	000612	1514#	1897*	4702*	4703									
TMEX	000564	1503#	2274	2594	2701	2899	3207	3209	4464*					
TMFLG	000676	1544#	2276*	2318*	2332	2591*	2596*	2618	2630	2677	2689*	2703	2705*	2708*
		2819	2842	2870	3988	4010	4016	4028	4046	4098	4187			
TOB	000640	1529#	4720*	4727*	4730*	4734*	4743	4783*	4834*	4836*				
TOG	023110	4724	4728	4731	4735	4741#	4742	4784	4837					
TPB	000620	1517#	4743*											
TPOS	013446	3140	3154	3163	3278#	3280								
TPS	000616	1516#	4741											
TSTAL	000600	1509#	1937	1940	2402	2523	2545	2894	2929	3265	3267			
TTIN	022722	4652	4702#											
TTINT	020734	1439	4331#											
TTOUT	022774	1418	4716#											
TTR	022476	3062	3071	3093	3107	3138	3150	3161	3181	3192	3204	3213	3222	3231
		3240	3253	3262	3271	4364	4392	4607	4648#					
TYPE =	000004	1421#	1836	1841	1955	1959	1962	1965	1968	1971	1974	1977	1980	1983
		2029	2032	2037	2092	2260	2267	2306	2315	2344	2345	2354	2360	2361
		2364	2449	2452	2460	2474	2481	2490	2625	2731	2736	2741	2750	2751
		2766	2772	2782	3048	3051	3055	3064	3088	3098	3102	3121	3127	3133
		3145	3156	3174	3184	3196	3206	3215	3224	3233	3246	3255	3264	3736
		3738	3747	3754	3769	3901	3915	4095	4102	4105	4108	4113	4116	4121
		4124	4129	4132	4137	4141	4146	4149	4159	4218	4227	4230	4235	4276
		4320	4323	4344	4357	4387	4395	4396	4399	4400	4515	4518	4522	4527
		4533	4541	4544	4548	4551	4567	4570	4599	4602	4664	4672	4695	4709
		4711	4751	4774	4796	4798	4813	4832						
TYPOCT=	104400	1425#	1958	1961	1964	1967	1970	1973	1976	1979	1982	1985	2036	2347
		2363	2366	2448	2451	2455	2480	2753	2774	2792	3057	3066	3130	3176
		3187	3198	3208	3217	3226	3235	3248	3257	3266	3746	3909	3917	4107
		4110	4115	4118	4123	4126	4131	4134	4140	4143	4148	4151	4162	4359
		4398	4404	4517	4521	4526	4532	4540	4547	4550	4566	4569	4573	4601
UDES	000552	1498#	1901*	1913*	1927	2008	2059*	2060	2128*	2129	2144*	2147	2159*	2160
		2171	2604	2647	2653	2659	2830	2849	2854	2860	3314	3334	3564	3578

