

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

REM %

IDENTIFICATION

PRODUCT CODE. AC-9471C-MC  
PRODUCT NAME CZTUGCO TM02/TE16 DR FCNT TMP  
DATE CREATED AUG 77  
MAINTAINER. DIAGNOSTIC GROUP  
AUTHOR. R BARNES  
REVISED. 11-NOV-77 BY CLEM WALSH

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

THE SOFTWARE DESCRIBED IN THIS DOCUMENT IS FURNISHED UNDER A LICENSE AND MAY ONLY BE USED OR COPIED IN ACCORDANCE WITH THE TERMS OF SUCH LICENSE.

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

COPYRIGHT (C) 1977 BY DIGITAL EQUIPMENT CORPORATION

46  
47  
48  
49  
50  
51  
52  
53  
54  
55  
56  
57  
58  
59  
60  
61  
62  
63  
64  
65  
66  
67  
68  
69  
70  
71  
72  
73  
74  
75  
76  
77  
78  
79  
80  
81  
82  
83  
84  
85  
86  
87  
88  
89  
90  
91

TMO2 DRIVE FUNCTION TIMER  
TABLE OF CONTENTS

PAGE 2

TABLE OF CONTENTS

ABSTRACT

CHAPTER 1 REQUIREMENTS

- 1.1 EQUIPMENT
- 1.2 MEMORY STORAGE
- 1.3 PRELIMINARY PROGRAMS

CHAPTER 2 LOADING AND STARTING PROCEDURE

- 2.1 ACT11 OPERATION

CHAPTER 3 SWITCH SETTINGS

CHAPTER 4 ERRORS

- 4.1 ERROR TYPEOUT FORMAT (HARDWARE)
- 4.2 ERROR TYPEOUT FORMAT (FUNCTION OUT OF RANGE)

CHAPTER 5 SUBROUTINE ABSTRACTS

CHAPTER 6 MISCELLANEOUS

- 6.1 STACK POINTER
- 6.2 EXECUTION TIME

CHAPTER 7 PROGRAM DESCRIPTION

- 7.1 FUNCTION TIME DOCUMENT
- 7.2 TEST SEQUENCE / RELATED ADJUSTMENTS / ASSOCIATED HARDWARE
- 7.3 TEST DESCRIPTIONS

TMO2 DRIVE FUNCTION TIMER  
ABSTRACT

PAGE 3

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

ABSTRACT

PROGRAM DZTUG MEASURES THE TIME REQUIRED AND GAP SIZES PRODUCED BY THE TMO2/TE16 MAGTAPE DRIVE/SLAVE.

THE TEST WILL CHECK BOTH THE LOGIC GENERATED TIME DELAYS, AND THE DISTANCES TRAVLED BY THE TAPE IN RESPONSE.

ACTUAL TAPE SPEED MAY ALSO BE CHECKED BY USING THE SPEED TESTS WITH AN 800 BPI SKEW TAPE.

DEVICE ERRORS ARE CHECKED AND PRINTED AS THEY OCCUR. IF THE ERROR IS DATA RELATED(PARITY; ETC) THEY ARE PRINTED AS SOFT ERRORS

IF THE TIME CHECK IS OUT OF RANGE, IT IS PRINTED AS AN OUT OF RANGE ERROR

137  
138  
139  
140  
141  
142  
143  
.  
146  
147  
148  
149  
150  
151  
152  
153  
154  
155  
156  
157  
158  
159  
160  
161  
162  
163  
164  
165  
166  
167  
168  
169  
170  
171  
172  
173  
174  
175  
176  
177  
178  
179  
180  
181  
182  
183

TMO2 DRIVE FUNCTION TIMER  
REQUIREMENTS

PAGE 4

CHAPTER 1  
REQUIREMENTS

PDP-11 FAMILY CENTRAL PROCESSOR WITH 4K MEMORY WITH UP TO 64 TMO2/TE16  
CONTROLLER/MAGTAPE STATIONS.

\*\*\*PROGRAM CAN BE RUN ON A PROCESSOR THAT DOES NOT HAVE A HARDWARE SWITCH REGISTER  
A SOFTWARE SWITCH REGISTER (SWREG) LOC. 176 IS AUTOMATICALLY SELECTED(REFER TO  
CHAPTER 3 FOR DESCRIPTION OF HOW TO DYNAMICALLY LOAD LOC 176)\*\*\*

1 1 OPTIONAL EQUIPMENT USED

1 NONE

1 2 STORAGE

PROGRAM LOADS AND RUNS IN THE FIRST 4K OF MEMORY

1 3 PRELIMINARY PROGRAMS (TO ASSURE HARDWARE OPERATION)

MAINDEC-11-DZTUC CONTROL LOGIC TEST  
MAINDEC-11-DZTUB BASIC FUNCTION TEST

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  
220  
221  
222  
223  
224  
225  
226  
227  
228  
229  
230  
231  
232  
233  
234  
235

TMO2 DRIVE FUNCTION TIMER  
LOADING AND STARTING PROCEDURE

PAGE 5

CHAPTER 2

LOADING AND STARTING PROCEDURE

THE PROCEDURE IS AS FOLLOWS:

LOAD PROGRAM USING THE ABSOLUTE LOADER  
LOAD ADDRESS = 200  
SET OPERATING SWITCHES  
PRESS START

\*\*\*IF THE SOFTWARE SWITCH REGISTER IS USED THEN THE PROGRAM WILL TYPE SWR=XXXXXX NEW=  
THIS WILL ALLOW LOC. 176 TO BE CHANGED BEFORE THE START OF THE TESTING (REFER TO CHAP

PROGRAM WILL REQUEST DRIVE (TMO2) AND SLAVE (TE16) NUMBERS TO BE  
TESTED TYPE DRIVE/SLAVE NUMBERS WITH A COMMA (,) BETWEEN EACH  
DRIVE/SLAVE TO BE TESTED

REQUESTS FOR TAPE SPEED TESTS AND NRZ ONLY MODE WILL BE MADE  
RESPONSE TO TAPE SPEED ONLY REQUEST WITH A ONE (1) WILL CAUSE  
THE PROGRAM TO EXECUTE TEST 31 AND 32 ONLY. THIS IS THE ONLY WAY  
TO TEST TAPE SPEED.  
NRZ ONLY MODE WILL CAUSE THE PROGRAM TO SKIP THE 1600 BPI DATA TIME TEST  
TYPE CONTROL U (U) TO DELETE LINE TYPED OR RUBOUT TO DELETE LAST  
CHARACTER(S)

PROGRAM WILL PUBLISH TIMES REQUIRED AND REPORT ERRORS

2 1 ACT11 OPERATION

IF THE PROGRAM IS RUN IN QUICK VERIFY MODE, FUNCTION TESTS ARE NOT  
ITERATED

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  
271  
272  
273  
274  
275  
276  
277  
278  
279  
280  
281  
282  
283  
284  
285  
286  
287  
288

TMO2 DRIVE FUNCTION TIMER  
SWITCH SETTINGS

PAGE 6

CHAPTER 3  
SWITCH SETTINGS

IF THE DIAGNOSTIC IS RUN ON A CPU WITHOUT A SWITCH REGISTER THEN A SOFTWARE SWITCH REGISTER IS USED WHICH ALLOWS THE USER THE SAME SWITCH OPTIONS AS THE HARDWARE SWITCH REGISTER IF THE HARDWARE SWITCH REGISTER DOES NOT EXIST OR IF ONE DOES AND IT CONTAINS ALL ONES (177777) THEN THE SOFTWARE SWITCH REGISTER (LOC 176) IS USED.

CONTROL.

THIS PROGRAM ALSO SUPPORTS THE DYNAMIC LOADING OF THE SOFTWARE SWITCH REGISTER (LOC 176) FROM THE TTY THIS CAN BE ACCOMPLISHED BY DOING THE FOLLOWING:

- 1) TYPE CONTROL G < G>; THIS WILL ALLOW THE TTY TO ENTER DATA INTO LOC. 176 AT SELECTED POINTS WITHIN THE PROGRAM
- 2) THE MACHINE WILL THEN TYPE: SWR=XXXXXXNEW= (XXXXXX IS THE OCTAL CONTENTS OF THE SOFTWARE SWITCH REGISTER )
- 3) 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 LOC 176 FOLLOWED BY A <CR> (ONLY NUMBERS BETWEEN 0-7 WILL BE ACCEPTED AND ONLY 6 NUMBERS WILL BE ALLOWED)  
IF A <CR> IS THE FIRST KEY DEPRESSED THE SOFTWARE SWITCH REGISTER CONTENTS WILL NOT BE CHANGED
  - B) IF A CONTROL U < U> IS DEPRESSED THEN THE PROGRAM WILL SEND YOU BACK TO STEP 2

289	SW15	HALT ON ERROR	THIS SWITCH WHEN SET WILL HALT THE
290	(100000)		PROCESSOR WHEN AN ERROR IS DETECTED
291			THE PC+2 AND PSW AT THE TIME OF THE
292			ERROR IS STORED ON THE STACK. PRESSING
293			CONTINUE WILL CAUSE THE ERROR TO BE
294			TYPED (IF SELECTED) AND FURTHER TESTING
295			RESUMED
296			
297	SW14	LOOP SUBTEST	THIS SWITCH WHEN SET LOOPS THE CURRENT
298	(040000)		SUBTEST REGARDLESS OF ERROR CONDITION
299			
300	SW13	INHIBIT ERROR	THIS SWITCH WHEN SET INHIBITS ERROR
301	(020000)	TYPEOUT	TYPEOUT.
302			
303	SW11	INHIBIT SUB-	THIS SWITCH WHEN SET CAUSES EACH SUBTEST
304	(004000)	TEST ITERATION	TO BE EXECUTED ONLY ONCE (INITIAL
305			STARTUP ONLY).
306			
307	SW10	INHIBIT	THIS SWITCH WHEN SET WILL INHIBIT THE
308	(002000)	FUNCTION TIME	PRINTING OF THE FUNCTION TIMES (SEE
309		PUBLICATION	CHAPTER 8.)
310			
311	SW09	RING BELL	THIS SWITCH WHEN SET WILL RING THE BELL
312	(001000)	ON ERROR	ON THE TTY WHEN AN ERROR IS DETECTED
313			
314	SW07	HALT AFTER	THIS SWITCH WHEN SET WILL CAUSE THE
315	(000200)	SELECTED TEST	PROGRAM TO HALT AFTER THE TEST SELECTED
316			IN SW05-SW00 IS EXECUTED
317			
318	SW06	CONTINUOUS	THIS SWITCH WHEN SET WILL CAUSE THE
319	(000100)	CYCLE	PROGRAM TO RUN CONTINUOUSLY UNTIL
320			STOPPED BY THE OPERATOR
321			
322	SW5-0	TEST SELECT	THE PROGRAM WILL HALT AFTER EXECUTION
323			OF THE TEST SELECTED WHEN SW07 IS SET

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

TMO2 DRIVE FUNCTION TIMER  
ERRORS

PAGE 7

CHAPTER 4  
ERRORS

TWO TYPES OF ERRORS ARE DETECTED BY THIS PROGRAM, HARDWARE ERRORS AND INCORRECT FUNCTION TIMES.

4 1 ERROR TYPEOUT FORMAT (HARDWARE): DATA RELATED ERRORS (IE. PARITY ERROR) ARE PRINTED AND HAVE NO EFFECT ON TIME

TEST # XXXXXX DEVICE ERROR

CS1	WE	BA	FC	CS2	DS	ER1
AAAAAA	BBBBBB	CCCCCC	DDDDDD	EEEEEE	FFFFFF	GGGGGG

WHERE

XXXXXX = TEST NUMBER  
AAAAAA-111111 = CONTENTS OF TAPE REGISTER 172440-172454

4 2 ERROR TYPEOUT FORMAT (FUNCTION TIME OUT OF RANGE)

TEST # XXXXXX OUT OF RANGE ERROR

RANGE = <AAAAAA-BBBBBB> ACTUAL = CCCCCC



372  
373  
374  
375  
376  
377  
378  
379  
380  
381  
382  
383  
384  
385  
386  
387  
388  
389  
390  
391  
392  
393  
394  
395  
396  
397  
398  
399  
400  
401  
402  
403  
404  
405  
406  
407  
408  
409  
410  
411  
412  
413  
414  
415  
416  
417  
418  
419  
420  
421  
422  
423  
424  
425  
426  
427

TMO2 DRIVE FUNCTION TIMER  
SUBROUTINE ABSTRACTS

PAGE 8

CHAPTER 5  
SUBROUTINE ABSTRACTS

5 1 . SCOPE

THE SCOPE ROUTINE IS CALLED BY THE SCOPE (EMT) INSTRUCTION AT THE START OF EACH SUBTEST THE SCOPE ROUTINE PERFORMS THE FOLLOWING FUNCTIONS

- 1 LOADS R5 WITH BASE ADDRESS
- 2 TYPES TIME LINE <SW08>
- 3 PROVIDES CONTINUOUS LOOP <SW14>
- 4 MOVES FUNCTION TIME INTO TABLE
- 5 OUTPUTS LINE ITEM IF SELECTED
- 6 PROVIDES HALT ON TEST <SW07>
7. DELAYS 350MS BEFORE STARTING TEST
8. INIT'S DRIVE/SLAVE
- 9 CLEARS THE ERROR FLAG (ERFLG)

THE ROUTINE MONITORS SW14, SW11, SW10, SW08, AND SW07

\*\*\*THIS ROUTINE WILL CHECK FOR CNTL G< G> BY DOING A JSR PC.CKSWR  
(REFER TO CHAPTER 3 FOR DESCRIPTION).

5 2 PUBLISH

THE PUBLISH ROUTINE IS CALLED FROM THE SCOPE ROUTINE IF SW10 IS EQUAL TO 0 (PUBLISH TIME DOCUMENT) THE ROUTINE WILL PRINT A "SINGLE LINE ITEM" EACH TIME IT IS CALLED

428  
429  
430  
431  
432  
433  
434  
435  
436  
437  
438  
439  
440  
441

TMO2 DRIVE FUNCTION TIMER  
SUBROUTINE ABSTRACTS

PAGE 9

5 3 .HLT

THE HLT ROUTINE IS CALLED BY THE HLT (TRAP) INSTRUCTION WHEN AN ERROR IS DETECTED. A HLT (TRAP) INSTRUCTION FORMATS THE ERROR INFORMATION AS SHOWN IN SEC 4.1 A HLT+1 (TRAP+1) FORMATS THE ERROR AS SHOWN IN SEC 4.2

\*\*\*THIS ROUTINE WILL CHECK FOR A CNTL G < G> BY DOING A JSR PC,CKSWR (REFER TO CHAPTER 3 FOR DESCRIPTION)

442 TMO2 DRIVE FUNCTION TIMER PAGE 10  
443 MISCELLANEOUS  
444  
445  
446  
447  
448  
449  
450  
451  
452  
453  
454  
455  
456  
457

CHAPTER 6  
MISCELLANEOUS

6 1 STACK POINTER

THE STACK POINTER IS INITIALLY SET TO 500 AND IS RESET TO 500 BY THE SCOPEA ROUTINE

6 2 EXECUTION TIME

WHEN SW11=1 (INHIBIT ITERATIONS) THE TIME REQUIRED IS 2 MIN.

WHEN SW11=0 (ITERATE SUBTESTS) THE TIME REQUIRED IS 9 MIN

468  
469  
470  
471  
472  
473  
474  
475

476  
477  
478  
479  
480  
481  
482  
483  
484  
485  
486  
487  
488  
489  
490  
491  
492  
493  
494  
495  
496  
497  
498  
499  
500  
501  
502  
503  
504  
505  
506  
507  
508  
509  
510  
511  
512  
513  
514  
515  
516  
517  
518  
519  
520  
521  
522  
523  
524  
525  
526  
527  
528  
529  
530  
531

TMO2 DRIVE FUNCTION TIMER  
PROGRAM DESCRIPTION

PAGE 11

CHAPTER 7

PROGRAM DESCRIPTION

7 1 SAMPLE TIME DOCUMENT

TYPE FIRST ADDRESS OF CONTROLLER 172440  
TYPE TMO2 DRIVE #'S TO BE TESTED 0  
FOR TMO2 DRIVE 0- TYPE SLAVE #'S TO BE TESTED 7  
TAPE SPEED TESTS ONLY? (YES/NO = 1/0) 0  
NRZ ONLY? (YES/NO = 1/0) 0

\*\*\*\*\*  
\* TMO2 DRIVE FUNCTION TIMES- DRIVE # 0 SLAVE # 7 9 CHAN SER # 5009  
\*

* FUNCTION	TIME (SPECIFICATION)	TIME (ACTUAL)
* WRITE FROM BOT	RANGE=<154000-150000>	ACTUAL=152740
* WRITE START	RANGE=<009500-008700>	ACTUAL=009120
* WRITE SHUTDOWN	RANGE=<008900-008500>	ACTUAL=008840
* WRITE SETTLEDOWN	RANGE=<013500-008100>	ACTUAL=010970
* READ FROM BOT	RANGE=<037000-033000>	ACTUAL=035580
* READ START	RANGE=<003200-002600>	ACTUAL=002740
* READ SHUTDOWN	RANGE=<004650-004250>	ACTUAL=004360
* READ SETTLEDOWN	RANGE=<013500-008100>	ACTUAL=010970
* READ REV START	RANGE=<003200-002600>	ACTUAL=002740
* READ REV SHUTDOWN	RANGE=<003700-003300>	ACTUAL=003520
* READ REV SETTLEDOWN	RANGE=<013500-008100>	ACTUAL=010970
* TURN AROUND DELAY F-R	RANGE=<016700-010700>	ACTUAL=013600
* TURN AROUND DELAY R-F	RANGE=<016700-010700>	ACTUAL=013660
* GAP SIZE-STOP HALF	RANGE=<012900-009500>	ACTUAL=012200
* GAP SIZE-START HALF	RANGE=<011800-008500>	ACTUAL=010520
* GAP SIZE-INTERRECORD	RANGE=<014800-013700>	ACTUAL=014500
* GAP CONSISANCY	RANGE=<014000-012400>	ACTUAL=013040
* DATA TIME-200 BPI	RANGE=<024100-023100>	ACTUAL=023460
* DATA TIME-556 BPI	RANGE=<024000-023000>	ACTUAL=023350
* DATA TIME-800BPI	RANGE=<024000-023000>	ACTUAL=023400
* DATA TIME-1600BPI	RANGE=<025100-024100>	ACTUAL=024470
* ERASE GAP TIME	RANGE=<101000-099000>	ACTUAL=099510
* WRITE FILE MARK	RANGE=<105000-103000>	ACTUAL=103990

532

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

TMO2 DRIVE FUNCTION TIMER

PAGE 12

7.1.1 SAMPLE TIME DOCUMENT FOR TAPE SPEED TESTS

TYPE FIRST ADDRESS OF CONTROLLER 172440  
TYPE TMO2 DRIVE #'S TO BE TESTED 0  
FOR TMO2 DRIVE 0- TYPE SLAVE #'S TO BE TESTED 7  
SPEED TESTS ONLY? (YES/NO = 1/0) 1  
\*\*\*\*\*  
\*TMO2 DRIVE FUNCTION TIMES- DRIVE # 0 SLAVE # 7 9 CHAN SER. # 5009  
\*

*FUNCTION	TIME(SPECIFICATION)	TIME(ACTUAL)
*TAPE SPEED FWD	RANGE=<022700-021700>	ACTUAL=022500
*TAPE SPEED REV	RANGE=<022700-021700>	ACTUAL=022500

TMO2 DRIVE FUNCTION TIMER

PAGE 13

7 2 TEST SEQUENCE WITH RELATED ADJUSTMENTS AND ASSOCIATED HARDWARE

TEST NO. /NAME RELATED ADJUSTMENTS

549  
550  
551  
552  
553  
554  
555  
556  
557  
558  
559  
560  
561  
562  
563  
564  
565  
566  
567  
568  
569  
570  
571  
572  
573  
574  
575  
576  
577  
578  
579  
580  
581  
582  
583  
584  
585  
586  
587  
588  
589  
590  
591  
592  
593  
594  
595  
596  
597  
598  
599  
600  
601  
602  
603

1	WRITE FROM BOT	*NONE
2	WRITE START	* "
3	WRITE SHUTDOWN	* "
4	WRITE SETTLEDOWN	* "
5	READ FROM BOT	* "
6	READ START	* "
7	READ SHUTDOWN	* "
10	READ SETTLEDOWN	* "
11.	READ REVERSE START	* "
12.	READ REVERSE SHUTDOWN	* "
13.	READ REVERSE SETTLEDOWN	* "
14.	TURN AROUND F-P	* "
15.	TURN AROUND R-F	* "
16	GAP SIZE-STOP HALF	*FWD/REV SPEED-START/STOP-RAMPS
17.	GAP SIZE-START HALF	*SAME AS IN TEST 16
20	GAP SIZE INTERRECORD	*FWD/REV SPEED

	TMO2 DRIVE FUNCTION TIMER	PAGE 14
604		
605		
606	21 GAP CONSISTENCY	*SAME AS IN TEST 16
607		
608	*TEST NUMBER 22 IS RESERVED FOR FUTURE USE	
609		
610	23 DATA TIME 200 BPI	*NONE
611		
612		
613	24 DATA TIME 556 BPI	* "
614		
615		
616	25 DATA TIME 800 BPI	* "
617		
618		
619	26 DATA TIME 1600 BPI	* "
620		
621		
622	27. ERASE GAP TIME	* "
623		
624		
625	30 WRITE FILE MARK	* "
626		
627		
628	31. TAPE SPEED-FORWARD	*FWD SPEED
629		
630		
631	32. TAPE SPEED-REVERSE	*PEVERSE SPEED
632		
633		
634	*****NOTE. IF TIME PROBLEMS APPEAR IN T1 THRU T3C RUN TAPE SPEED TESTS FIRST*****	



635  
636  
637  
638  
639  
640  
641  
642  
643  
644  
645  
646  
647  
648  
649  
650  
651  
652  
653  
654  
655  
656  
657  
658  
659  
660  
661  
662  
663  
664  
665  
666  
667  
668  
669  
670  
671  
672  
673  
674  
675  
676  
677  
678  
679  
680

TMO2 DRIVE FUNCTION TIMER

PAGE 15

7 3 TEST DESCRIPTIONS.

THE FIRST THIRTEEN (13) TESTS (T1 - T15) ARE CHECKS OF THE ROM CIRCUITS IN THE TE16 (M9811), THE ACCL COUNTER IN THE TMO2 (M8903), AND THE SETTLEDOWN ONE SHOT (M8910)

T1. WRITE FROM BOT.

THIS TEST WILL MEASURE ACCELERATION DELAY REQUIRED TO MOVE THE TAPE APPROXIMATELY SEVEN (7) INCHES FORWARD FROM DEAD STOP AT BOT BEFORE STARTING TO TRANSFER DATA

- 1 ASSURE TAPE IS STOPPED AT BOT.
- 2 ISSUE A WRITE COMMAND
- 3 MONITOR BIT 15 OF TC (ACCL)
4. TIME FROM GO TO ACCL RESET IS BOT DELAY
5. STOP

T2. WRITE START:

THIS TEST WILL MEASURE ACCELERATION DELAY JUST AS IN T1. HOWEVER THE TIME WILL BE LESS WHEN NOT STARTING FROM BOT.

- 1 LEAVE TAPE AT ITS PRESENT POSITION ASSURE THAT IT IS STOPPED
2. ISSUE A WRITE COMMAND
- 3 MONITOR BIT 15 OF TC (ACCL)
- 4 TIME FROM GO TO RESET OF ACCL IS START DELAY
5. STOP

T3. WRITE SHUTDOWN.

THIS TEST WILL MEASURE THE TIME FROM EOR (LAST CHARACTER WRITTEN ON TAPE) TO THE START OF SETTLEDOWN TIME THIS ASSURES, IN PART, A PROPER INTERROCORD GAP

- 1 LEAVE TAPE AT ITS PRESENT POSITION ASSURE THAT IT IS STOPPED
2. ISSUE A WRITE COMMAND
- 3 MONITOR FRAME COUNTER AND BIT 4 OF DS (SDWN)
4. TIME FROM FC=0 TO ASSERTION OF SDWN IS THE SHUTDOWN TIME
- 5 STOP

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

T4 WRITE SETTLEDOWN

THIS TEST WILL MEASURE THE SLOWDOWN TIME THE TIME FROM THE START OF SLOWDOWN UNTIL THE TAPE SHOULD BE STOPPED. THIS IS A PART OF THE GAP TIMING IN LOGIC THE MECHANICAL POSITIONING OF THE TAPE IN THE GAP DISTANCE WILL BE MEASURED IN A LATER TEST.

1. LEAVE TAPE AT ITS PRESENT POSITION ASSURE THAT IT IS STOPPED
2. ISSUE A WRITE COMMAND
3. MONITOR BIT 4 OF DS (SDWN)
4. TIME FROM SET OF SDWN TO RESET OF SDWN IS THE SETTLEDOWN DELAY
5. STOP

T5 READ FROM BOT

THIS MEASUREMENT IS MADE EXACTLY AS THE WRITE MEASUREMENT IN T1 USE THE SAME RECORD THAT WAS WRITTEN IN T1

1. REWIND TO BOT
2. ASSURE TAPE HAS HAD TIME TO COME TO A COMPLETE STOP
3. READ FORWARD 1 RECORD
4. MONITOR BIT 15 OF TC (ACCL)
5. TIME FROM GO TO ACCL IS BOT DELAY
6. STOP

712 T6 READ START  
713 THIS TEST MEASURES THE SAME DELAY AS IN T2  
714  
715 1 WRITE 1 RECORD, THEN BACKSPACE OVER IT, ASSURE TAPE IS STOPPED  
716 2 ISSUE A READ FORWARD OF THE RECORD WRITTEN IN STEP 1  
717 3 MONITOR BIT 15 OF TC (ACCL)  
718 4 TIME FROM GO TO RESET OF ACCL IS START DELAY  
719 5 STOP  
720  
721 T7 READ SHUTDOWN  
722 THIS TEST MEASURES THE SAME DELAY AS IN T3  
723  
724 1 WRITE 1 RECORD, THEN BACKSPACE OVER IT, ASSURE TAPE IS STOPPED  
725 2 READ FORWARD THE RECORD WRITTEN IN STEP 1  
726 3 MONITOR FRAME COUNT AND BIT 4 OF DS (SDWN).  
727 4 TIME FROM FC=RECORD SIZE (LAST FRAME READ) TO SDWN=1  
728 IS THE SHUTDOWN TIME  
729 5 STOP  
730  
731 T10 READ SETTLEDOWN.  
732 THIS TEST MEASURES THE SAME DELAY AS IN T4  
733  
734 1 WRITE 1 RECORD, THEN BACKSPACE OVER IT, ASSURE TAPE IS STOPPED  
735 2 READ FORWARD THE RECORD WRITTEN IN STEP 1  
736 3 MONITOR BIT 4 OF DS (SDWN)  
737 4 TIME FROM SET OF SDWN TO RESET OF SDWN IS THE  
738 SETTLEDOWN DELAY  
739 5 STOP  
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  
779  
780  
781  
782  
783  
784  
785  
786  
787  
788  
789  
790  
791  
792  
793  
794  
795  
796  
797  
798

TMO2 DRIVE FUNCTION TIMER

PAGE 17

T11 READ REVERSE START

THIS TEST WILL MEASURE THE START DELAY IN THE REVERSE DIRECTION

- 1 WRITE 1 RECORD, ASSURE TAPE IS STOPPED
- 2 READ REVERSE THE RECORD WRITTEN IN STEP 1
- 3 MONITOR BIT 15 OF TC (ACCL)
- 4 THE TIME FROM GO TO RESET OF ACCL IS THE START TIME
- 5 STOP

T12 READ REVERSE SHUTDOWN

THIS TEST WILL MEASURE THE READ SHUTDOWN IN THE REVERSE DIRECTION

- 1 WRITE 1 RECORD, ASSURE TAPE IS STOPPED
- 2 READ REVERSE THE RECORD WRITTEN IN STEP 1
- 3 MONITOR FRAME COUNTER AND BIT 4 OF DS (SDWN)
- 4 TIME FROM FC=RECORD SIZE (LAST FRAME READ) TO SDWN=1 IS THE READ REVERSE SHUTDOWN TIME
- 5 STOP

T13 READ REVERSE SETTLEDOWN.

THIS TEST WILL MEASURE THE READ SETTLEDOWN IN THE PEVERSE DIRECTION

- 1 WRITE 1 RECORD, ASSURE TAPE IS STOPPED
- 2 READ REVERSE THE RECORD WRITTEN IN STEP 1
- 3 MONITOR BIT 4 OF DS (SDWN)
- 4 TIME FROM SET OF SDWN TO RESET OF SDWN IS THE SETTLEDOWN DELAY
- 5 STOP

T14. TURN AROUND DELAY-FORWARD TO REVERSE

THIS TEST WILL MEASURE THE TIME REQUIRED FOR THE TAPE TO CHANGE DIRECTION.

- 1 LEAVE TAPE AT ITS PRESENT POSITION ASSURE THAT IT IS STOPPED
- 2 ISSUE A WRITE FORWARD OF AT LEAST 20 FRAMES
- 3 MONITJR BIT 7 OF DS (DRY)
- 4 WHEN DRY IS ASSERTED (EOR), IMMEDIATELY ISSUE A READ REVERSE OF THAT RECORD
- 5 MONITOR BIT 15 OF TC (ACCL)
- 6 TIME FROM GO OF READ REVERSE TO RESET OF ACCL IS THE TURNAROUND TIME
- 7 STOP

799  
800  
801  
802  
803  
804  
805  
806  
807  
808  
809  
810  
811  
812  
813  
814

TMO2 DRIVE FUNCTION TIMER

PAGE 18

T15 TURN AROUND DELAY-REVERSE TO FORWARD

THIS TEST WILL MEASURE THE TIME FS IN T14, BUT IN THE  
OPPOSITE DIRECTION

- 1 WRITE 1 RECORD
- 2 ASSURE TAPE IS STOPPED
- 3 READ REVERSE
- 4 MONITOR DRY (BIT 7 OF DS)
- 5 WHEN DRY = 1, ISSUE A READ FORWARD
- 6 MONITOR ACCL (BIT 15 OF TC)
- 7 TIME FROM GO FORWARD TO ACCL = 1 IS THE TURN AROUND TIME
- 8 STOP

TMO2 DRIVE FUNCTION TIMER

PAGE 19

815  
816  
817  
818  
819  
820  
821  
822  
823  
824  
825  
826  
827  
828  
829  
830  
831  
832  
833  
834  
835  
836  
837  
838  
839  
840  
841  
842  
843  
844  
845  
846  
847  
848  
849  
850  
851  
852  
853  
854  
855  
856  
857  
858  
859  
860  
861  
862  
863  
864  
865  
866  
867  
868  
869  
870

GAP MEASUREMENTS.

THE PREVIOUS THIRTEEN (13) TESTS WERE MEASUREMENTS OF LOGIC DELAYS PERFORMED BY THE TMO2 OR TE16 IN ORDER TO ALLOW FOR PROPER ACCELERATION AND DECELERATION OF TAPE ACCORDING TO THE DESIRED INTERCORD GAP (.6 INCHES) THIS TEST, HOWEVER, WILL MEASURE THE PHYSICAL SIZE OF THE INTERCORD GAP THAT EXISTS ON TAPE AS A RESULT OF THE START/STOP TIMES OF THE CAPSTAN ITSELF. BECAUSE THE INTERCORD GAP IS CREATED BY TWO ACTIONS, THE START OF MOTION AND THE STOP OF MOTION IT IS NECESSARY TO MAKE TWO SEPERATE MEASUREMENTS. A THIRD MEASUREMENT, MADE ON THE FLY, OF THE ENTIRE LENGTH OF THE GAP WILL ALSO BE MADE

T16. GAP SIZE (STOP HALF)

THIS TEST WILL MEASURE THE DISTANCE TRAVLED BY THE TAPE IN A STOP CYCLE. IN OTHER WORDS, THE DISTANCE INTO THE IRG.

1. WRITE 1 RECORD.
2. ASSURE TAPE IS STOPPED.
3. ISSUE A READ REVERSE OVER THE RECORD
4. MONITOR THE FRAME COUNT FOR THE FIRST FRAME READ (FC = 1)
5. THE TIME FROM GO=1 TO FC=1 IS THE LENGTH OF THE GAP
6. STOP

T17. GAP SIZE (START HALF)

THIS TEST WILL MEASURE THE DISTANCE OF TAPE TRAVEL DURING START UP.

1. WRITE 1 RECORD, THEN REVERSE OVER IT, ASSURE TAPE IS STOPPED
2. ISSUE A READ FORWARD
3. MONITOR FC FOR FC=1
4. TIME FROM GO=1 TO FC=1 IS START DISTANCE
5. STOP

T20. GAP SIZE (INTERRECORD)

THIS TEST WILL MEASURE THE ENTIRE LENGTH OF THE IRG ON THE FLG. THE TIME VALUE OF THIS TEST SHOULD NOT BE EQUAL TO A SUMMATION OF T16 AND T17 DUE TO THE FACT THAT THE ACCELERATION AND DECELERATION CURVES ARE NOT IN EFFECT. THE VALUE HERE SHOULD ACTUALLY BE LESS THAN THE SUM OF T16 AND T17

1. WRITE 2 RECORDS
2. READ REVERSE OVER THE SECOND RECORD
3. MONITOR DRY (BIT 7 OF DS)
4. WHEN DRY = 1, ISSUE A SECOND READ REVERSE
5. MONITOR FRAME COUNT
6. TIME FROM GO=1 OF SECOND READ REVERSE TO FC=1 IS THE LENGTH OF THE GAP

871

7 STOP

872  
873  
874  
875  
876  
877  
878  
879  
880  
881  
882  
883  
884  
885  
886  
887  
888  
889  
890  
891  
892  
893  
894  
895  
896  
897  
898  
899  
900  
901  
902  
903  
904  
905  
906  
907  
908  
909  
910  
911  
912  
913  
914  
915  
916  
917  
918  
919  
920  
921  
922  
923  
924  
925  
926

TMO2 DRIVE FUNCTION TIMER

PAGE 20

T21. GAP CONSISTENCY.

NOW THAT WE HAVE ESTABLISHED THAT THE INTERCORD GAP IS THE PROPER SIZE, LET US DETERMINE THE CONSISTENCY OF THE GAP UNDER VARIOUS COMMAND EXECUTION TIMES. BY WRITING A SERIES OF RECORDS, EACH WITH A DIFFERENT DELAY BETWEEN EXECUTION, WE CAN ESTABLISH THE CONSISTENCY OF THE GAPS BY READING THESE RECORDS AND MONITORING THEIR INTERRECORD GAPS, ON THE FLY.

- 1 REWIND TAPE TO BOT.
  - 2 WRITE ONE (1) RECORD TO GET TAPE OFF BOT
  - 3 WRITE SIXTEEN (16) RECORDS WITH A PROGRESSIVE DELAY OF FROM 0 TO 16 MILLISECONDS (APPROX) BETWEEN COMMANDS
  - 4 BACKSPACE 16 RECORDS AND ALLOW THE TAPE TO STOP
  5. READ FORWARD (NON-STOP) OVER THESE 16 RECORDS, EACH TIME MONITORING THE TIME FROM THE END OF RECORD (DRY) UNTIL THE FRAME COUNT NEXT GOES FROM 0 TO 1 (FC=1)
  6. THE TIMES FROM DRY TO FC=1 IS THE GAP TIME AND IT SHOULD REMAIN CONSISTANT FOR ALL RECORDS.
  7. STOP
- \*\*(SEE GTIMTBL IN DZTUG LISTING FOR GAP TIMES)\*\*

T22. RESERVED FOR FUTURE USE\*\*\*\*

T23. DATA TIME AT 200 BPI:

THIS TEST WILL MEASURE THE TIME REQUIRED TO WRITE ONE (1) INCH OF TAPE AT 200 BPI. BY WRITING A RECORD OF ENOUGH FRAMES TO MOVE THE TAPE 1 INCH (200 FRAMES), DATA RATE CAN BE VARIFIED

- 1 REWIND TO BOT AND ALLOW TAPE TO STOP
- 2 WRITE A RECORD AT 200 BPI.
- 3 MONITOR DRY (BIT 7 OF DS) FOR EACH RECORD
4. THE TIME FROM FC=FC+1 TO DRY WILL BE THE TIME REQUIRED FOR 1 INCH AT THE SELECTED DENSITY
5. STOP

T24. DATA TIME AT 556 BPI.  
REPEAT STEPS 1 THRU 5 OF T23 AT 556 BPI

T25. DATA TIME AT 800 BPI:  
REPEAT STEPS 1 THRU 5 AT 800 BPI

T26 DATA TIME AT 1600 BPI (PE):  
REPEAT STEPS 1 THRU 5 AT 1600 BPI  
\*\*THIS TEST IS NOT EXECUTED IF NR2 ONLY\*\*



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

TMO2 DRIVE FUNCTION TIMER

PAGE 21

T27. ERASE:

THE ERASE COMMAND WILL CAUSE AN AREA OF THE THREE (3) INCHES TO BE DC ERASED IN THE FORWARD DIRECTION. THIS TEST WILL ASSURE THAT THE PROPER DISTANCE IS ERASED.

1. LEAVE TAPE AT ITS PRESENT POSITION.
2. ISSUE AN ERASE COMMAND.
3. MONITOR DRY (BIT 7 OF DS)
4. THE TIME FROM GO TO DRY WILL BE THE TIME REQUIRED TO ERASE 3 INCHES OF TAPE AND WILL REFLECT THE DISTANCE. DENSITY IS NOT A FACTOR.
5. STOP

T30. TAPE MARK:

THIS TEST IS ALSO A CHECK ON THE THREE (3) INCH GAP WHEN A TAPE MARK IS WRITTEN, A 3 INCH GAP IS CREATED BEFORE DATA IS PUT ON TAPE

- 1 LEAVE TAPE AT ITS PRESENT POSITION
- 2 ISSUE A WRITE TAPE MARK COMMAND
3. MONITOR DRY (BIT 7 OF DS)
- 4 THE TIME FROM GO TO DRY WILL BE THE TIME REQUIRED TO WRITE THE TM RECORD PLUS THE 3 INCH GAP
- 5 STOP

TMO2 DRIVE FUNCTION TIMER

PAGE 22

959  
960  
961  
962  
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

T31. TAPE SPEED FORWARD:

THIS TEST REQUIRES THE USE OF AN 800 BPI SKEW TAPE!  
THE OPERATOR WILL BE REQUIRED TO MOUNT THE SKEW TAPE  
BEFORE EXECUTING THE TEST. THE SKEW TAPE IS THE ONLY  
WAY TO ASSURE THAT TAPE IS MOVING AT THE PROPER SPEED  
BECAUSE THE FREQUENCY OF FRAMES ON A SKEW TAPE IS  
GUARANTEED TO BE ACCURATE.

- 1 ASSURE TAPE IS STOPPED AT BOT.
- 2 ISSUE A READ FORWARD (800 BPI, NORMAL.)
- 3 MONITOR FC FOR FC = 800(10)
- 4 MONITOR FC FOR FC = 8800(10)
- 5 TIME FROM FC = 800 TO FC = 8800 IS THE TIME REQUIRED  
FOR TAPE TO TRAVEL 10 INCHES
- 6 DIVIDE THE TIME FOR 10 INCHES BY 10.
- 7 THE RESULT IS AN AVERAGE SPEED FOR 1 INCH
- 8 STOP.

T32. TAPE SPEED REVERSE:

THIS TEST IS THE SAME AS TEST 31, BUT SPEED IS  
MEASURED IN THE REVERSE DIRECTION

1. ADVANCE TAPE OFF OF BOT.
- 2 ISSUE A READ REVERSE.
3. REPEAT STEPS 3 THRU 6 IN THE REVERSE DIRECTION
- 4 STOP.

```
991
992          NLIST MC
993          LIST ME
994          ABS
995          MCALL $CPVEC,$CPREG,$CATCH,$TYPE
996          .TITLE DZTUG-B TMO2/TE16 DRIVE FUNCTION TIMER
997          .SBTTL STARTING INSTRUCTIONS
998          .LOADING AND STARTING PROCEEDURE
999          .LOAD PROGRAM USING ABS LOADER
1000         .LOAD ADDRESS 200
1001         .SET SWITCH OPTIONS
1002         .PRESS START
1003
1004         .GENERAL REGISTER USAGE.
1005         .R0=ADDRESS OF 'FC' REGISTER (SET BY SCOPE)
1006         .R1=ADDRESS OF 'DS' REGISTER (SET BY SCOPE)
1007         .R2=RETURN PC FROM TIMER (SET BY EACH TEST)
1008         .R3=INDEX INDICATING PREVIOUS OSCILLATOR POLARITY (SET BY TIMER)
1009         .R4=CONTAINS 'TICK' COUNT WHEN TIMER IS RUNNING (SET BY TIMER)
1010         .R5=ADDRESS OF CS1 (SET BY SCOPE)
1011
1012         .SWITCH REGISTER SWITCH ASSIGNMENTS
1013         100000 SW15= 100000 ;HALT ON ERROR
1014         040000 SW14= 040000 ;LOOP SUBTEST
1015         020000 SW13= 020000 ;INHIBIT ERROR TYPE OUT
1016         004000 SW11= 004000 ;INHIBIT SUBTEST ITERATION
1017         002000 SW10= 002000 ;INHIBIT PUBLICATION OF FUNCTION TIMES
1018         001000 SW09= 001000 ;RING BELL ON ERROR
1019         000400 SW08= 000400 ;TYPE LINE ITEM AFTER EACH ITERATION
1020         000200 SW07= 000200 ;HALT ON TEST SELECTED IN SW05-SW00
1021         000100 SW06= 000100 ;CONTINUOUS CYCLE
1022
1023
1024
1025
1026
```

1027  
1028  
1029  
1030  
1031  
1032 000764  
1033 000040  
1034 000040 000  
1035  
1036 000041  
1037 000041 000  
1038  
1039  
1040 000042  
1041 000042 000000  
1042  
1043  
1044 000046  
1045 000046 013264  
1046  
1047 000052  
1048 000052 000000  
1049  
1050  
1051 000764  
1052  
1053  
1054  
1055

\*\*\*\*\*  
ACT11 HOOKS  
\*\*\*\*\*

SSVPC=  
=40  
DRIVE BYTE 0 ,DRIVE # FOR XXDP LOAD MEDIUM  
=41  
MEDIUM BYTE 0 ,XXDP LOAD MEDIUM  
;ASSEMBLE AS A 0  
=42  
WORD 0 ;LOCATION INDICATOR - AUTOM/MAN MODE  
;ASSEMBLE AS A 0  
=46  
WORD SENDAD ,SET TO SENDAD IN SEOP  
=52  
WORD 0 ;CHARACTERISTICS OF PROGRAM  
;SET TO A 0  
=SSVPC ,RESTORE PC

\*\*\*\*\*

```
1056  
1057 , *****  
1058 , ACT11 MODE INDICATORS  
1059 , *****  
1060  
1061 000764 000000 AUTOM: . WORD 0 ;AUTOMATIC MODE INDICATOR  
1062 000766 000 ACT11M. BYTE 0 ;ACT11 AUTO MODE INDICATOR  
1063 000767 000 XXDPM . BYTE 0 ;XXDP AUTOM MODE INDICATOR  
1064 000770 000 ADUMPM. BYTE 0 ;ACT11 DUMP MODE INDICATOR  
1065 000771 000 XDUMPM BYTE 0 ;XXDP DUMP MODE INDICATOR  
1066  
1067 , *****  
1068
```

```
1069
1070          MACRO  SCHNMODE
1071
1072          ,      *****
1073
1074          ,      MACRO FOR SIZING DRIVES AND SLAVES
1075          ,      IN AUTOMATIC MODE.
1076
1077          ,      +
1078
1079          ,      +
1080
1081          ,      MOV      #-1, DRVNUM          , INIT DRIVE NUMBER
1082          ,      NXTDRV  MOV      #-1, SLVNUM         , INIT SLAVE NUMBER
1083          1$          MOV      #40, @CS2           , INIT CONTROLLER
1084          ,      INC      DRVNUM                , STEP DRIVE #
1085          ,      CMP      #10, DRVNUM           , ALL DRIVE TESTED?
1086          ,      BNE     2$                    , BRANCH - IF NO
1087          ,      JMP     FINISH                 , EXIT
1088          ,      2$      MOV      DRVNUM, @CS2       , LOAD DRIVE #
1089          ,      TST     @CS1                  , ACCESS DRIVE
1090          ,      BIT     #10000, @CS2          , NON-EXISTANT DRIVE?
1091          ,      BNE     1$                    , BRANCH - IF YES
1092          ,      1$      NXTSLV  INC      SLVNUM         , STEP SLAVE #
1093          ,      BNE     1$                    , BRANCH IF NOT SLAVE 0
1094          ,      TST     DRVNUM                , DRIVE 0?
1095          ,      BNE     1$                    , BRANCH - IF NO
1096          ,      TSTB   XXDPM                 , XXDP?
1097          ,      BEQ    1$                    , BRANCH - IF NO
1098          ,      INC     SLVNUM                , STEP TO SLAVE #1
1099          ,      1$      CMP      #10, SLVNUM       , ALL DRIVES TESTED?
1100          ,      BEQ    NXTDRV                 , BRANCH - IF YES
1101          ,      MOV     SLVNUM, @TC           , LOAD SLAVE UNIT?
1102          ,      BIT     #2000, @DT            , SLAVE PRESENT?
1103          ,      BEQ    NXTSLV                 , BRANCH - IF NO (SPR=0)
1104          ,      BIT     #140000, @DT          , IS DRIVE A TAPE UNIT?
1105          ,      BEQ    NXTSLV                 , BRANCH IF NO
1106          ,      ,      *****
1107
1108          ,      ENDM  SCHNMODE
1109
```

```

1110
1111          SBTTL  MACRO DEFINITIONS
1112          MACRO  SAVE
1113          JSR    PC, SAVE          ,SAVE REGISTERS ON THE STACK
1114          ENDM  SAVE
1115          MACRO  RESTORE
1116          JSR    PC, RESTORE      ,RESTORE REGISTERS FROM THE STACK
1117          ENDM  RESTORE
1118          MACRO  INPUT
1119          JSR    PC, INPUT        ,GET USER INPUT
1120          ENDM  INPUT
1121          MACRO  REWIND
1122          JSR    PC, REWIND       ,REWIND SLAVE
1123          BVS    99$             ,BRANCH IF ERROR ON REWIND
1124          ENDM  REWIND
1125          MACRO  TIMEON
1126          JSR    PC, TIMEON      ,TURN TIMER ON
1127          ENDM  TIMEON
1128          MACRO  TIMCHK
1129          JMP    TIMER(R3)      ,GO TO TIMER & RETURN VIA R2
1130          ENDM  TIMCHK
1131          MACRO  SETGO
1132          INC    (R5)           ,SET 'GO' BIT
1133          ENDM  SETGO
1134

```

SBTTL REGISTER ASSIGNMENTS  
 DEFINITIONS AND REGISTER ASSIGNMENTS  
 GENERAL REGISTER ASSIGNMENTS

```

1138          000000  R0=%0
1139          000001  R1=%1
1140          000002  R2=%2
1141          000003  R3=%3
1142          000004  R4=%4
1143          000005  R5=%5
1144          000006  SP=%6
1145          000007  PC=%7
1146          000000  R10=%0
1147          000001  R11=%1
1148          000002  R12=%2
1149          000003  R13=%3
1150          000004  R14=%4
1151          000005  R15=%5

```

REGISTER ADDRESSES

```

1154          177776  PSW= 177776          ,,PROCESSER STATUS WORD
1155          177774  SLR= 177774          ,,STACK LIMIT REGISTER (11/40,11/45)
1156          177772  PIRQ= 177772        ,,PROGRAM INTERRUPT REQ (11/45)
1157          177770  UBREAK= 177770      ,,MICRO-BREAK REGISTER (11/45)
1158          177560  TKS= 177560         ,,KEYBOARD CSR
1159          177562  TKB= 177562         ,,KEYBOARD DATA BUFFER REGISTER
1160          177564  TPS= 177564         ,,TELEPRINTER CSR
1161          177566  TPB= 177566         ,,TELEPRINTER DATA BUFFER REGISTER

```

VECTOR ADDRESSES

```

1164          000004  ERRVEC=4           ,,ADDRESS OF ERROR VECTOR
1165          000010  RESVEC=10          ,,ADDRESS OF RESERVED INST TRAP VECTOR

```

1166	000014	TBITVEC=14	.. ADDRESS OF 'T' BIT TRAP VECTOR
1167	000014	TRTVEC=14	.. ADDRESS OF 'TRACE' TRAP VECTOR
1168	000014	BPTVEC=14	.. ADDRESS OF 'BREAKPOINT' TRAP VECTOR
1169	000020	IOTVEC=20	.. ADDRESS OF IOT TRAP VECTOR
1170	000024	PFVEC=24	.. ADDRESS OF POWER FAIL TRAP VECTOR
1171	000030	EMTVEC=30	.. ADDRESS OF EMT VECTOR
1172	000034	TRAPVEC=34	.. ADDRESS OF TRAP VECTOR
1173	0C0060	TKVEC= 60	.. ADDRESS OF TTY KEYBOARD INT. VECTOR
1174	000064	TPVEC=64	.. ADDRESS OF TTY PRINTER INTERRUPT VECTOR
1175	000114	PARVEC= 114	.. ADDRESS OF MA/MF PARITY ERROR VECTOR
1176	000240	PIRVEC=240	.. ADDRESS OF PIRQ VECTOR
1177	000244	FPEVEC=244	.. ADDRESS OF FLOATING POINT INT VECTOR
1178	00C250	MMVEC=250	.. ADDRESS OF MEM MGMT ERROR TRAP VECTOR
1179			
1180		.CLOCK ADDRESS AND VECTORS	
1181	172540	PLKCSR= 172540	.KW11-P
1182	000104	PLKVEC= 104	
1183	177546	LKS= 177546	.KW11-L
1184	000100	LKVEC= 100	
1185	177514	LPS= 177514	.LP11
1186	177516	LPB= 177516	
1187			
1188		.RH11 TMO2/TE16 REGISTERS	
1189	172440	TMCS1= 172440	
1190			
1191		TMO2 TE16 INDEX VALUES	
1192	000000	CS1= 00	.CONTROL STATUS #1
1193	000002	WC= 02	
1194	000004	BA= 04	.BUS ADDRESS REGISTER
1195	000006	FC= 06	.FRAME COUNT
1196	000010	CS2= 10	.CONTROL STATUS #2
1197	000012	DS= 12	.DRIVE STATUS
1198	000014	ER= 14	.ERROR REG #1
1199	000016	AS= 16	.ATTENTION SUMMARY
1200	000022	DB= 22	.DATA BUFFER REG
1201	000024	MR= 24	.MAINTENANCE REG
1202	000026	DT= 26	.DRIVE TYPE REG
1203	000030	SN= 30	.SERIAL NUMBER REGISTER
1204	000032	TC= 32	.TAPE CONTROL REG
1205			
1206		SBTTL TMO2/TE16 REGISTER BITS	
1207		PHCS1-CS1(R5)	
1208	000001	GO= 1	
1209	000000	NOP= 0	
1210	000002	PWDOFF= 2	
1211	000006	RWD= 6	
1212	000010	DRYCLR= 10	
1213	000026	WFMK= 26	
1214	000024	ERASE= 24	
1215	000030	SPCFWD= 30	
1216	000032	SPCREV= 32	
1217	000050	WCHKF= 50	
1218	000056	WCHKR= 56	
1219	000060	WFWD= 60	
1220	000070	RDFWD= 70	
1221	000076	RDREV= 76	



1222	000100	IE=	100
1223	000200	RDY=	200
1224	000400	A16=	400
1225	001000	A17=	1000
1226	002000	PSEL=	2000
1227	004000	DVA=	4000
1228	020000	MCPE=	20000
1229	040000	TRE=	40000
1230	100000	SC=	100000
1231		.RHCS2-CS2(R5)	
1232	000000	DV0=	0
1233	000001	DV1=	1
1234	000002	DV2=	2
1235	000003	DV3=	3
1236	000004	DV4=	4
1237	000005	DV5=	5
1238	000006	DV6=	6
1239	000007	DV7=	7
1240	000010	BA1=	10
1241	000020	PAT=	20
1242	000040	CLR=	40
1243	000100	IR=	100
1244	000200	OR=	200
1245	000400	MDPE=	400
1246	001000	MXF=	1000
1247	002000	PGE=	2000
1248	004000	NEM=	4000
1249	010000	NED=	10000
1250	020000	UPE=	20000
1251	040000	WCE=	40000
1252	100000	DLT=	100000
1253		.RHDS-DS(R5)	
1254	000001	SLA=	1
1255	000002	BOT=	2
1256	000004	TMK=	4
1257	000010	IDB=	10
1258	000020	SDWN=	20
1259	000040	PES=	40
1260	000100	SSC=	100
1261	000200	DRY=	200
1262	000400	DPR=	400
1263	002000	EOT=	2000
1264	004000	WRL=	4000
1265	010000	MOL=	10000
1266	020000	PIP=	20000
1267	040000	ERR=	40000
1268	100000	ATA=	100000
1269		.RHER-ER(R5)	
1270	000001	ILF=	1
1271	000002	ILR=	2
1272	000004	RMR=	4
1273			
1274	000020	FMT=	20
1275	000100	INCVAE=	100
1276	000200	PEFLRC=	200
1277	000400	NSG=	400

1278 001000  
1279 002000  
1280 004000  
1281 010000  
1282 020000  
1283 040000  
1284  
1285  
1286 000100  
1287  
1288  
1289 002000  
1290 010000  
1291 040000  
1292  
1293  
1294 000300  
1295 000320  
1296 000000  
1297 000400  
1298 001000  
1299 002000  
1300 100000  
1301  
1302  
1303 104400  
1304 104000  
1305 000004  
1306  
1307  
1308 005730  
1309 177400  
1310 177600  
1311  
1312 000003  
1313 000011  
1314 000012  
1315 000015  
1316 000017  
1317 000025

FCE= 1000  
CS1TM= 2000  
NEF= 4000  
DTE= 10000  
OP1= 20000  
UNS= 40000  
  
RHRM-MR(R5)  
OSC= 100  
  
RHDT-DT(R5)  
SPR= 2000  
CH7= 10000  
TAP= 40000  
  
RHTC-TC(R5)  
NORM11= 300  
CDM11= 320  
BP1200= 0  
BP1556= 000400  
BP1800= 001000  
PE1600= 002000  
ACCL= 100000  
  
INSTRUCTION EQUATES  
HLT= TRAP  
SCOPE= EMT  
TYPE= 10T  
  
MISCELLANEOUS EQUATES  
OUTBUF= INIT  
FRMCNT= -256  
WRDCNT= -128  
  
ASCII EQUATES  
CNTRLC= 3  
HT= 11  
LF= 12  
CR= 15  
CNTRL0= 17  
CNTRPLU= 25

OUTPUT BUFFER START AT BEG OF PROGRAM  
FRAME COUNT  
WORD COUNT  
  
ASCII CODE FOR CONTROL C ( C )  
ASCII CODE FOR HORIZONTAL TAB  
ASCII CODE FOR LINE FEED  
ASCII CODE FOR CARRIAGE RETURN  
ASCII CODE FOR CONTROL O ( O )  
ASCII CODE FOR CONTROL U ( U )

```

1318          , SETUP TRAP VECTORS
1319          =TBIVVEC
1320          WORD +2          , SET 'T' TRAP TO TIMER ROUTINE
1321          WORD HALT        , PRIORITY LEVEL 7
1322          WORD .TYPE      , SET IOT TRAP TO TYPE ROUTINE
1323          WORD 0           , PRIORITY LEVEL 0
1324          WORD PFVEC+2    , POWER FAIL TRAP TO HALT
1325          WORD HALT        , AT PFVEC+2
1326          WORD SCOPE      , SET EMT TRAP TO SCOPE ROUTINE
1327          WORD 340        , PRIORITY LEVEL 7
1328          WORD .HLT        , SET TRAP TRAP TO HLT ROUTINE
1329          WORD 340        , PRIORITY LEVEL 7
1330          =TKVEC
1331          WORD TKISR
1332          WORD 340
1333
1334          , SOFTWARE SWITCH REGISTER LOC 176
1335          =176
1336          SWREG: 0          , SOFTWARE SWITCH REGISTER
1337
1338          =200
1339
1340          CLR PCNTR          , INIT PROGRAM PASS INDICATOR
1341          JMP @#INIT        , GO TO START OF PROGRAM
1342
1343          =500
1344          STKPTR= 600       , STACK
1345
1346          =1000
1347          , PROGRAM TAGS
1348          SWR: 177570      , SWITCH REGISTER
1349
1350          PAFLG WORD 0      , PASS INDICATOR
1351          PCNTR WORD 0      , PASS COUNT INDICATOR
1352
1353          SCPADR WORD 0
1354          DRVNUM BYTE 0     , TMO2 DRIVE UNDER TEST
1355          SLVNUM BYTE 0     , TE16 SLAVE UNDER TEST
1356          SLVPTR WORD 0    , POINTER TO SLAVE TABLE (SLVTBL) BELOW
1357          TMBASE WORD TMCS1 , BASE ADDRESS OF TMO2/TE16 REGISTERS
1358          ATIME WORD 0      , CONTAINS 'TICK' COUNT
1359          ATIMTBL BLKW 16   , EACH ENTRY CONTAINS TIME FOR FUNCTION
1360                                     , ENTRIES ARE MADE BY 'SCOPE' ROUTINE
1361          GAP TBL 16         , TIMES RECORDED BY 'GAP CONSISTANCY' TEST
1362          DELTIM WORD 0     , VARIABLE DELAY
1363          OCTALO WORD 0
1364          GAP: BYTE 0       , CONTAINS GAP # (USED FOR TST 021)
1365          ITCNT: BYTE 0     , ITERATION COUNT
1366          TSTNUM BYTE 0    , TEST #
1367          ERFLG: BYTE 0     , ERROR FLAG
1368          PRGFLG: BYTE 0    , PROGRAM FLAG
1369          UNTFND: BYTE 0    , UNIT FOUND INDICATOR
1370          TYPFLG: BYTE 0
1371          NRZFLG: BYTE 0    , INDICATES IF DRIVE IS NRZ ONLY
1372          ASFLG: BYTE 0    , 1'0 = YES/NO
1373          EVEN
  
```

1374	001136	030460		DIGTAB	"01		
1375	001140	031462			"23		
1376	001142	032464			"45		
1377	001144	033466			"67		
1378	001146	034470			"89		
1379	001150	000006		ODIGITS	. BLKB	6	, RESERVE SPACE FOR CONVERTED DIGITS
1380	001156	000			. BYTE	0	; TERMINATOR
1381		001160			. EVEN		
1382	001160	000010		DRVTBL.	. BLKB	8	; A 0/-1 = DRIVE NOT TO BE/TO BE TESTED
1383	001170	000100		SLVTBL	. BLKB	64	; A 0/-1 = SLAVE NOT TO BE/TO BE TESTED
1384	001270	000110		INBUF	. BLKB	72	; TELETYPE INPUT BUFFER
1385	001400	005015	000	CRLF:	. ASCIZ	<CR><LF>	; MISCELLANEOUS ASCII CHARACTERS
1386	001403	134	000	BKSLSH	. ASCIZ	' '	
1387	001405	060	000	ECHO	. ASCIZ	'0'	
1388	001407	007	000	BELL.	. ASCIZ	<7>	
1389	001411	055	000	DASH	. ASCIZ	' - '	
1390	001413	040		SPACE2:	. ASCII	' '	
1391	001414	000040		SPACE	. ASCIZ	' '	
1392	001416	004476	000	ANGTAB.	. ASCIZ	' '><HT>	
1393		001422			. EVEN		

1394  
 1395  
 1396  
 1397  
 1398  
 1399  
 1400  
 1401  
 1402  
 1403  
 1404  
 1405  
 1406  
 1407  
 1408  
 1409  
 1410  
 1411  
 1412  
 1413  
 1414  
 1415  
 1416  
 1417  
 1418  
 1419  
 1420  
 1421  
 1422  
 1423  
 1424  
 1425  
 1426  
 1427  
 1428  
 1429

SBTTL TIME SPECIFICATION TABLE  
 THE BELOW TABLE CONTAINS THE SPECIFIED FUNCTION TIMES IN TENS OF  
 MICROSECONDS. NOTE THAT WHEN TIMES ARE TYPED THAT THEY ARE TYPED IN  
 MICROSECONDS (BY APPENDING A 0)  
 FORMAT IS

WORD	MAX.	MIN	TIME IN MS	FUNCTION	TEST #
STIMTBL.	WORD	0.0	SPARE		
	WORD	15400. 15000	154.0-150.0	WRITE FROM BOT	TST001
	WORD	00950. 00870.	9.5-8.7	WRITE START	TST002
	WORD	00890. 00850.	8.9-8.5	WRITE SHUTDOWN	TST003
	WORD	01350. 00810	13.5-8.1	WRITE STLDOWN	TST004
	WORD	03700. 03300.	37.0-33.0	READ FROM BOT	TST005
	WORD	00320. 00260.	3.2-2.6	READ START	TST006
	WORD	00465. 00425.	4.65-4.25	READ SHUTDOWN	TST007
	WORD	01350. 00810.	13.5-8.1	READ SETTLEDOWN	TST010
	WORD	00320. 00260.	3.2-2.6	RD REV START	TST011
	WORD	00370. 00330	3.7-3.3	RD REV SHTDWN	TST012
	WORD	01350. 00810.	13.5-8.1	RD REV STLDWN	TST013
	WORD	01670. 01070.	16.7-10.7	TRN RND DLY F-R	TST014
	WORD	01670. 01070.	16.7-10.7	TRN RND DLY R-F	TST015
	WORD	01290. 00950.	12.9-9.5	GAP SIZE STOP	TST016
	WORD	01180. 00850	11.8-8.5	GAP SIZE STRT	TST017
	WORD	01480. 01370.	14.8-13.7	GAP SIZE INTER	TST020
	WORD	01380. 01240.	13.8-12.4	GAP CONSISANCY	TST021
	WORD	0.0	0.0-0.0	DUMMY	TST022
	WORD	02410. 02310	24.1-23.1	DAT TIME 200BPI	TST023
	WORD	02400. 02300.	24.0-23.0	DAT TIME 556BPI	TST024
	WORD	02400. 02300	24.0-23.0	DAT TIME 800BPI	TST025
	WORD	02510. 02410.	25.1-24.1	DAT TIME 1600PE	TST026
	WORD	10100. 09900.	101.0-99.0	ERASE	TST027
	WORD	10500. 10300	105.0-103.0	WRT FILE MARK	TST030
	WORD	02270. 02170	22.7-21.7	READ 1" TAPE	TST031
	WORD	02270. 02170	22.7-21.7	RD REV 1" TAPE	TST032

NOTE TEST 31 AND 32 REQUIRE PRERECORDED 800BPI SKEW TAPE

SBTTL GAP TIME SPECIFICATION TABLE  
 THIS TABLE CONTAINS THE GAP SIZES (IN TENS OF MICROSECONDS) FOR EACH  
 OF THE 16 GAPS RECORDED BY THE GAP CONSISTANCY TEST (TST021)  
 NOTE: GAP #'S ARE IN OCTAL.

	WORD	MAX.	MIN(10)	TIME IN MS(10)	GAP #	DELAY IN MS(10)
1430						
1431						
1432						
1433						
1434						
1435						
1436						
1437	001576	002602	002412	14.1-12.9	GAP-0	0 MS
1438	001602	002652	002506	14.5-13.5	GAP-1	1 0 MS
1439	001606	002734	002532	15.0-13.7	GAP-2	2.0 MS
1440	001612	002734	002532	15.0-13.7	GAP-3	3.0 MS
1441	001616	002734	002424	15.0-13.0	GAP-4	4.0 MS
1442	001622	002652	002260	14.5-12.0	GAP-5	5.0 MS
1443	001626	002652	002260	14.5-12.0	GAP-6	6.0 MS
1444	001632	002652	002260	14.5-12.0	GAP-7	7.0 MS
1445	001636	002532	002260	13.7-12.0	GAP-10	8.0 MS
1446	001642	002532	002260	13.7-12.0	GAP-11	9 0 MS
1447	001646	002532	002260	13.7-12.0	GAP-12	10 0 MS
1448	001652	002532	002260	13.7-12.0	GAP-13	11.0 MS
1449	001656	002532	002260	13.7-12.0	GAP-14	12.0 MS
1450	001662	002532	002260	13.7-12.0	GAP-15	13 1 MS
1451	001666	002532	002260	13.7-12.0	GAP-16	14 1 MS
1452	001672	002532	002260	13.7-12.0	GAP-17	15 1 MS

1453  
1454  
1455 001676 000000  
1456 001700 015405  
1457 001702 015427  
1458 001704 015447  
1459 001706 015471  
1460 001710 015515  
1461 001712 015537  
1462 001714 015556  
1463 001716 015600  
1464 001720 015623  
1465 001722 015645  
1466 001724 015672  
1467 001726 015721  
1468 001730 015752  
1469 001732 016003  
1470 001734 016031  
1471 001736 016060  
1472 001740 016110  
1473 001742 000000  
1474 001744 016133  
1475 001746 016157  
1476 001750 016203  
1477 001752 016227  
1478 001754 016254  
1479 001756 016276  
1480 001760 016321  
1481 001762 016343

.SBTTL TEST HEADER POINTERS  
.THE BELOW TABLE CONTAINS POINTERS TO EACH TEST'S DESCRIPTOR

NAMPTR: .WORD 0  
.WORD A T001  
.WORD A T002  
.WORD A T003  
.WORD A T004  
.WORD A T005  
.WORD A T006  
.WORD A T007  
.WORD A T010  
.WORD A T011  
.WORD A T012  
.WORD A T013  
.WORD A T014  
.WORD A T015  
.WORD A T016  
.WORD A T017  
.WORD A T020  
.WORD A T021  
.WORD 0  
.WORD A T023  
.WORD A T024  
.WORD A T025  
.WORD A T026  
.WORD A T027  
.WORD A T030  
.WORD A T031  
.WORD A T032

; DUMMY TEST

```

1482
1483      ,CHECK SWITCH REGISTER ROUTINE CHECKS FOR G TO ALLOW CHANGING
1484      ,OF LOC. 176.
1485      ;CALL IS BY WAY OF JSR PC,CKSWR
1486      ,LOCATIONS USED.
1487      TIB:      .WORD 0
1488      TEMPST.  .WORD 0
1489      COUNT.   .WORD 0
1490      RDSW.    .WORD 0
1491
1492      001764 000000      000176 176776 CKSWR  CMP      #SWREG,SWR      ,SOFTWARE SWITCH REG PRESENT
1493      002002 001120      BNE      OUT        ,NO, GET OUT
1494      002004 016767 175552 177752      MOV      TKB,TIB    ;AND STRIP OFF
1495      002012 042767 177600 177744      BIC      #177600,TIB ,THE GARBAGE
1496      002020 022767 000007 177736      CMP      #7,TIB     ,IS IT A < G>
1497      002026 001106      BNE      OUT
1498      002030 000004 016365      TYPE,L,CNTG
1499      002034 000004 016372      CNTLU.  TYPE,L,SWR
1500      002040 017702 176734      MOV      @SWR,R2
1501      002044 004767 000564      JSR      PC,TYPOCT
1502      002050 000004 016401      TYPE,L,NEH
1503
1504      002054 005067 177706      CLR      TEMPST
1505      002060 012767 000007 177702      MOV      #7,COUNT
1506      002066 004767 000154      JSR      PC,TTIN    ;GO READ A CHARACTER
1507      002072 042767 177600 177664      BIC      #177600,TIB ,STRIP OFF GARBAGE
1508      002100 122767 000025 177656      CMPB     #25,TIB    ;IS IT A U?
1509      002106 001001      BNE      2$        ;BRANCH IF NOT
1510      002110 000751      BR       3$        ;START OVER
1511      002112 122767 000015 177644      CMPB     #15,TIB    ;IS IT A <CR>?
1512      002120 001012      BNE      4$        ;BRANCH IF NOT
1513      002122 012767 000200 177642      MOV      #200,RDSW
1514      002130 004767 000230      JSR      PC,TCRLF   ;ECHO IT WITH <LF>
1515      002134 022767 000007 177626      CMP      #7,COUNT  ;WAS IT FIRST CHARACTER
1516      002142 001034      BNE      7$        ;CHANGE SWR IF NOT FIRST ONE
1517      002144 000437      BR       OUT       ;GET OUT
1518      002146 122767 000060 177610      CMPB     #60,TIB
1519      002154 003004      BGT      5$
1520      002156 122767 000067 177600      CMPB     #67,TIB
1521      002164 002003      BGE      6$
1522      002166 000004 016411      TYPE,L,QUEST
1523      002172 000746      BR       3$        ;START OVER IF NOT LEGAL CHARACTER
1524      002174 006367 177566      ASL      TEMPST
1525      002200 006367 177562      ASL      TEMPST
1526      002204 006367 177556      ASL      TEMPST
1527      002210 142767 000060 177546      BICB     #60,TIB    ;GET NITTY-GRITTY
1528      002216 156767 177542 177542      BISB     TIB,TEMPST
1529      002224 005367 177540      DEC      COUNT     ;ONLY WANT 6 DIGITS
1530      002230 001756      BEQ      5$
1531      002232 000715      BR       1$
1532      002234 016777 177526 176536      MOV      TEMPST,@SWR ;CHANGE SWITCH REGISTER CONTENTS
1533      002242 000740      BR       8$
1534      002244 000207      OUT     RTS      PC
1535

```



```
1536
1537
1538
1539 002246 005067 175306
1540 002252 005067 175304
1541 002256 005067 177502
1542 002262 005267 175272
1543 002266 105767 175266
1544 002272 100375
1545 002274 016767 175262 177462
1546 002302 105767 175256
1547 002306 100375
1548 002310 116767 177450 175250
1549 002316 000207
1550
1551
1552
1553
1554
1555
1556
1557
1558
1559
1560
1561
1562
1563
1564
1565
1566
1567
1568
1569
1570
1571
1572
1573
1574
1575
1576
1577
1578
1579
1580
1581
1582
1583
1584
1585
1586
1587
1588
1589
1590
1591
```

```

      .TTY READ SUBROUTINE*****
TTIN:  CLR     TKS
      CLR     TKB
      CLR     TIB
      INC     TKS
TTIN1: TSTB   TKS
      BPL    TTIN1
      MOV    TKB,TIB
TTIN2: TSTB   TPS
      BPL    TTIN2
      MOVB   TIB,TPB
      RTS    PC

      .SBTTL PROGRAM SUBROUTINES
      SBTTL TYPE SUBROUTINE
      .ROUTINE TO TYPE ASCII MESSAGE MESSAGE MUST TERMINATE WITH A 0 BYTE
      .THE ROUTINE WILL INSERT A NUMBER OF NULL CHARACTERS AFTER A LINE FEED
      .CALL TYPE
      .MESADR
      .A TRAP TYPE INSTRUCTION
      .MESADR IS FIRST ADDRESS OF ASCII STRING

      .TAGS USED BY THE TYPE ROUTINE BELOW
      .SHT=11
      .HORIZONTAL TAB
      .CONTAINS NULL CHARACTER
      .CONTAINS # OF FILLER CHARACTERS
      .CONTAINS TELEPRINTER AVAILABLE FLAG
      .0/377 = AVAIL/NOT AVAIL
      .CONTAINS KEYBOARD AVAILABLE FLAG
      .ADDRESS OF TELEPRINTER STATUS REGISTER
      .ADDRESS OF TELEPRINTER DATA BUFFER
      .CONTAINS # OF CHARS TYPED
      .CONTAINS CONTROL 0 CHAR (IF TYPED)

      .TYPE. MOV RO, -(SP)
      .SAVE RO
      .MOV @2(SP),RO
      .GET MESSAGE ADDRESS
      .ADD #2,2(SP)
      .ADJUST RETURN PC
      .CLRB %CNTRLO

TYPE1: TSTB %CNTRLO
      .BRANCH IF CONTROL 0(0) WASN'T TYPED
      BEQ TYPE2

TCRLF: TYPE, %CNTRLO
      .TYPE <CR><LF>
      TSTB RDSW
      BPL TYPE3
      CLR RDSW
      RTS PC

TYPE2: MOVB (RO)+, -(SP)
      .PUSH CHARACTER TO BE TYPED ONTO STACK
      BNE TYPE4
      .BRANCH IF NOT THE TERMINATOR
      TST (SP)+
      .POP TERMINATOR CHAR OFF THE STACK
TYPE3: MOV (SP)+, RO
      .RESTORE RO
      RTI
      .RETURN TO CALLER

TYPE4: CMPB #SHT, (SP)
      .BRANCH IF HORIZONTAL TAB (HT)
      BEQ 95
```

```

1592 002424 004767 000026      JSR    PC,5$      ;;TYPE CHARACTER
1593 002430 122726 000012      CMPB   #12,(SP)+  ;;CHECK IF CHARACTER WAS A LINE FEED
1594 002434 001350              BNE    TYPE1      ;;BRANCH IF NOT LINE FEED
1595 002436 016746 177656      MOV    $NULL,-(SP) ;;GET # OF FILLERS REQUIRED AND FILLER
1596                                ;;CHARACTER
1597
1598 002442 105366 000001      4$    DECB   1(SP)  ;;DECREMENT FILLERS REQ COUNT
1599 002446 002770              BLT    3$         ;;BRANCH IF NO MORE FILLERS ARE REQUIRED
1600 002450 004767 000002      JSR    PC,5$      ;;TYPE FILLER CHARACTER
1601 002454 000772              BR     4$
1602
1603 002456 105777 177642      5$    TSTB   @5TPS    ;;WAIT FOR OUTPUT DEVICE
1604 002462 100375              BPL   -4
1605 002464 122737 000017 002331  CMPB   #17,@#SCNTRLO ;;CHECK IF CONTROL O WAS TYPED
1606 002472 001403              BEQ   6$         ;;STOP TYPING MESSAGE IF O WAS TYPED
1607 002474 116677 000002 177624  MOVB   2(SP),@5TPB  ;;OUTPUT CHARACTER
1608 002502 122766 000015 000002  6$    CMPB   #15,2(SP)  ;;BRANCH IF NOT <CR>
1609 002510 001003              BNE   7$
1610 002512 105067 177612      CLRB   $CHARCNT   ;;CLEAR CHARACTERS TYPED COUNT
1611 002516 000406              BR    8$
1612 002520 122766 000012 000002  7$    CMPB   #12,2(SP)  ;;BRANCH IF <LF> OR 'NULL'
1613 002526 002002              BGE   8$
1614 002530 105267 177574      INCB   $CHARCNT   ;;INCREMENT CHARACTER TYPED COUNT
1615 002534 000207      8$    RTS    PC
1616
1617                                ;; HORIZONTAL TAB <HT> PROCESSER
1618 002536 112716 000040      9$    MOVB   #40,(SP)  ;;LOAD 'SPACE'
1619 002542 004767 177710      10$   JSR    PC,5$      ;;TYPE 'SPACE'
1620 002546 132767 000007 177554  BITB   #7,$CHARCNT ;;TYPE SPACES UNTIL A MULTIPLE
1621 002554 001372              BNE   10$        ;;OF 8 CHARACTERS HAVE BEEN TYPED
1622 002556 105726              TSTB   (SP)+      ;;POP SPACE
1623 002560 000676              BR     TYPE1      ;;GET NEXT CHARACTER
1624
1625                                ; SUBROUTINE TO SAVE GENERAL REGISTERS ON THE STACK
1626                                ; CALL SAVE
1627 002562 010546      SAVE   MOV    R5,-(SP)      ; SAVE REGISTERS ON THE STACK
1628 002564 010446      MOV    R4,-(SP)
1629 002566 010346      MOV    R3,-(SP)
1630 002570 010246      MOV    R2,-(SP)
1631 002572 010146      MOV    R1,-(SP)
1632 002574 010046      MOV    R0,-(SP)
1633 002576 016646 000014      MOV    14(SP),-(SP)  ; GET RETURN PC
1634 002602 000207      RTS    PC           ; RETURN
1635
1636                                ; SUBROUTINE TO RESTORE GENERAL REGISTERS FROM THE STACK
1637                                ; CALL RESTORE
1638 002604 012666 000014      RESTORE MOV   (SP)+,14(SP)  ; MOVE RETURN PC
1639 002610 012600      MOV   (SP)+,R0        ; RESTORE REGISTERS
1640 002612 012601      MOV   (SP)+,R1
1641 002614 012602      MOV   (SP)+,R2
1642 002616 012603      MOV   (SP)+,R3
1643 002620 012604      MOV   (SP)+,R4
1644 002622 012605      MOV   (SP)+,R5
1645 002624 000207      RTS    PC           ; RETURN
1646
1647                                ; SUBROUTINE TO CONVERT OCTAL DATA TO ASC I
    
```

```

1648      ,CALL  MOV    NUMBER,R2      ;MOVE NUMBER TO R2
1649      ,      JSR    PC,CNVCT
1650
1651 002626 110667 176300      CNVCT  MOVB  SP,TYPFLG      ;SET DO NOT TYPE FLAG
1652 002632 000402              BR      CNVTO
1653
1654      .SBTTL      OCTAL TO ASCII & TYPE ROUTINE
1655 ,SUBROUTINE TO CONVERT OCTAL NUMBER TO ASCII AND TYPE IT OUT
1656 ,CALL  MOV    NUMBER,R2      ;PUT # IN R2
1657 ,      JSR    PC,TYPOCT      ;CALL ROUTINE
1658
1659 002634 105037 001132      TYPOCT CLRB  @#TYPFLG      ;SET TYPE FLAG
1660 002640              CNVTO
1661 002640 004767 177716      JSR    PC,.SAVE          ;SAVE REGISTERS ON THE STACK
1662 002644 012704 001150      MOV    #ODIGITS,R4      ;SET PTR TO OUTPUT
1663 002650 005003              CLR    R3                ;R3 WILL CONTAIN OCTAL DIGIT
1664 002652 010201              MOV    R2,R1            ;GET # TO BE TYPED
1665 002654 006302      1$    ASL    R2                ;SHIFT #
1666 002656 006103              ROL    R3
1667 002660 012700 000C06      MOV    #6,R0            ;SET DIGIT COUNTER
1668 002664 000404              BR      3$
1669
1670 002666 006302      2$    ASL    R2                ;SHIFT # 3 PLACES LEFT
1671 002670 006103              ROL    R3
1672 002672 005301              DEC    R1
1673 002674 001374              BNE    2$
1674 002676 012700 000003      3$    MOV    #3,R1            ;SET SHIFT COUNTER
1675 002702 116324 001136      MOVB  DIGTAB(R3),(R4)+  ;MOVE ASCII EQUIV TO OUTPUT
1676 002706 005003              CLR    R3
1677 002710 005300              DEC    R0                ;DECREMENT DIGIT COUNT
1678 002712 001365              BNE    2$                ;GET NEXT DIGIT
1679 002714 105737 001132      TSTB  @#TYPFLG          ;BRANCH IF ASCII IS
1680 002720 001002              BNE    4$                ;NOT TO BE TYPED
1681 002722 000004 001150      TYPE,ODIGITS
1682 002726              4$
1683 002726 004767 177652      JSR    PC,RESTORE      ;RESTORE REGISTERS FROM THE STACK
1684 002732 000207              RTS    PC
1685
1686
1687 ,SUBROUTINE TO CONVERT OCTAL DATA TO DECIMAL ASCII
1688 ,CALL  MOV    NUMBER,R2      ;MOVE NUMBER TO R2
1689 ,      JSR    PC,CNVDEC
1690
1691 002734 110637 001132      CNVDEC MOVB  SP,@#TYPFLG  ;SET DO NOT TYPE FLAG
1692 002740 000402              BR      CNVTD
1693
1694      .SBTTL      OCTAL TO DECIMAL & TYPE ROUTINE
1695 ,THIS ROUTINE CONVERTS AN OCTAL # TO DECIMAL ASCII AND TYPES IT OUT
1696 ,CALL  MOV    NUMBER,R2      ;PUT # IN R2
1697 ,      JSR    PC,TYPDEC      ;CALL ROUTINE
1698 002742 105037 001132      TYPDEC CLRB  @#TYPFLG      ;SET TYPE FLAG
1699 002746              CNVTD
1700 002746 004767 177610      JSR    PC,.SAVE          ;SAVE REGISTERS ON THE STACK
1701 002752 005000              CLR    R0                ;R0 IS INDEX TO DECIMAL CONSTANT
1702 002754 012704 001150      MOV    #ODIGITS,R4      ;SET OUTPUT PTR
1703 002760 005003      1$    CLR    R3                ;R3 CONTAINS DECIMAL DIGIT
    
```

```

1704 002762 166002 003042      25  SUB      DCONST(R0),R2      ;SUBTRACT DECIMAL CONSTANT UNTIL
1705 002766 103402              BLO      3$              ;INPUT # GOES NEGATIVE
1706 002770 005203              INC      R3              ;KEEPING TRACK OF SUBTRACTIONS
1707 002772 000773              BR       2$              ;
1708 002774 066002 003042      35  ADD      DCONST(R0),R2      ;ADD BACK CONSTANT WHEN NEGATIVE
1709 003000 116324 001136      MOVVB   DIGTAB(R3),(R4)+  ;MOVE ASCII EQUIVALENT
1710 003004 062700 000002      ADD      #2,R0           ;NEXT CONSTANT
1711 003010 005760 003042      TST     DCONST(R0)      ;UNTIL ALL CONSTANTS DONE
1712 003014 001361              BNE     1$              ;
1713 003016 112724 000060      MOVVB   #'0,(R4)+      ;LAST DIGIT IS 0
1714 003022 105737 001132      TSTB   @#TYPFLG       ;BRANCH IF ASCII IS
1715 003026 001002              BNE     4$              ;NOT TO BE TYPED
1716 003030 000004 001150      TYPE,ODIGITS
1717                                4$
1718 003034 004767 177544      JSR     PC,RESTORE     ;RESTORE REGISTERS FROM THE STACK
1719 003040 000207              RTS     PC
1720
1721 003042 023420      DCONST  WORD 10000
1722 003044 001750      WORD 1000
1723 003046 000144      WORD 100
1724 003050 000012      WORD 10
1725 003052 000001      WORD 1
1726 003054 000000      WORD 0      ;TERMINATOR
1727
1728                                SBTTL      TYPE SPECIFIED TIMES ROUTINE
1729                                ;THIS SUBROUTINE OUTPUTS THE TIME SPECIFICATIONS FOR THE TEST
1730                                ;AND ALSO THE ACTUAL TIME RECORDED (ATIME)
1731                                ;FORMAT OF LINE TYPED
1732                                ;RANGE=<AAAAAA-BBBBBB>      ACTUAL=CCCCC
1733                                ;WHERE.      AAAAAA IS MAXIMUM TIME FOR TEST (STIMTBL(TSTNUMX4))
1734                                ;      BBBBBB IS MINIMUM TIME FOR TEST (STIMTBL(TSTNUMX4+2))
1735                                ;      CCCCC IS ACTUAL TIME RECORDED BY TEST (ATIME)
1736                                ;CALL      MOVVB   TEST NUMBER,R2      ;LOAD TEST NUMBER
1737                                ;      MOV      #TIME,@#ATIME      ;MOVE TIME TO ATIME
1738                                ;      JSR     PC,OUTSPC
1739                                OUTSPC  MOV      R2,-(SP)      ;SAVE R2 & R3 ON THE STACK
1740                                MOV      R3,-(SP)
1741                                ASL      R2      ;MULTIPLY TEST # TIMES 4
1742                                ASL      R2      ;TO FORM INDEX INTO STIMTBL
1743                                MOV      R2,R3      ;R3 CONTAINS INDEX INTO TABLE
1744                                TYPE,L RNG
1745                                MOV      STIMTBL(R3),P2      ;GET MAXIMUM SPEC TIME
1746                                JSR     PC,TYPDEC      ;CONVERT TO DECIMAL & TYPE
1747                                TYPE,DASH
1748                                MOV      STIMTBL+2(R3),R2      ;GET MINIMUM TIME
1749                                JSR     PC,TYPDEC      ;CONVERT TO DECIMAL & TYPE
1750                                TYPE,ANGTAB
1751                                TYPE,L ACT
1752                                MOV      @#ATIME,R2      ;GET ACTUAL TIME
1753                                JSR     PC,TYPDEC      ;CONVERT TO DECIMAL & TYPE
1754                                TYPE,CRLF
1755                                MOV      (SP)+,R3
1756                                MOV      (SP)+,R2
1757                                RTS     PC      ;RETURN
1758
1759                                SBTTL      TYPE GAP TIMES SUBROUTINE
    
```

```

1760 , THIS SUBROUTINE IS USED TO TYPE THE SPECIFIED GAP SIZES (RECORDED IN
1761 , TSTQ21) IT IS CALLED BY THE GAPOK ROUTINE IF THE GAP SIZE IS OUT OF
1762 , RANGE VIA THE HLT ROUTINE (HLT+2)
1763 , CALL MOV#B #GAP, GAP , LOAD GAP # INTO GAP
1764 , MOV #TIME, ATIME , LOAD ACTUAL TIME INTO ATIME
1765 , JSR PC, OUTGAP
1766
1767 OUTGAP MOV R2, -(SP) , SAVE R2 AND R3
1768 MOV R3, -(SP)
1769 MOV#B GAP, R3 , GET GAP #
1770 ASL R3
1771 ASL R3
1772 TYPE, L. RNG
1773 MOV GTIMTBL(R3), R2 ; GET MAX TIME
1774 JSR PC, TYPDEC ; CONVERT TO DECIMAL & TYPE
1775 TYPE, DASH
1776 MOV GTIMTBL+2(R3), R2 ; GET MIN TIME
1777 JSR PC, TYPDEC ; CONVERT TO DECIMAL & TYPE
1778 TYPE, ANGTAB ; TYPE <
1779 TYPE, L. ACT
1780 MOV @ATIME, R2 ; GET ACTUAL TIME
1781 JSR PC, TYPDEC ; CONVERT TO DECIMAL & TYPE
1782 TYPE, E. GAP
1783 MOV#B #GAP, R2 ; GET GAP #
1784 JSR PC, TYPOCT ; TYPE GAP #
1785 TYPE, CRLF
1786 MOV (SP)+, R3 , RESTORE R3 AND R2
1787 MOV (SP)+, R2
1788 RTS PC
1789
1790 , SBTTL ASCII TO OCTAL CONVERT SUBROUTINE
1791 , SUBROUTINE TO CONVERT ASCII DATA TO OCTAL CONVERTED OCTAL DATA
1792 , IS LEFT IN OCTALO <15-00>
1793 CNVTAO
1794 JSR PC, .SAVE , SAVE REGISTERS ON THE STACK
1795 MOV #INBUF, R0 ; SET PTR TO ASCII DATA
1796 MOV #OCTALO, R1 ; GET ADDRESS OF OCTAL DATA
1797 CLR (R1) ; CLEAR OUT OLD OCTAL DATA
1798 CLR 2(R1)
1799 CMPB #CR, (R0) ; <CR> TERMINATES INPUT
1800 BEQ 3$
1801 MOV#B (R0)+, R2 ; GET 'OCTAL' DATA
1802 BIC #17770, R2 ; STRIP UNUSED BITS
1803 MOV #3, R3 ; SET SHIFT COUNT
1804 ASL (R1) ; SHIFT LAST
1805 ROL 2(R1) ; OCTAL DIGIT
1806 DEC R3
1807 BNE 2$
1808 BIS R2, (R1) ; AND INSERT THIS DIGIT
1809 BR 1$ ; GO GET NEXT DIGIT
1810 3$
1811 JSR PC, .RESTORE , RESTORE REGISTERS FROM THE STACK
1812 RTS PC ; RETURN
1813
1814 , SBTTL PUBLISH SUBROUTINE
1815 THE PUBLISH SUBROUTINE AVERAGES THE RECORDED TIMES FOR EACH TEST IT-
    
```

```

1816          ,ERATION (IF 16. ITERATIONS) AND PLACES THE AVERAGE RESULT IN 'ATIME'
1817          ,IT TYPES THE NAME OF THE FUNCTION THAT WAS TIMED, THE TIME SPEC-
1818          ,IFICATION AND THE ACTUAL TIME
1819
1820 003352          PUBLISH
1821 003352 004767 177204          JSR      PC, .SAVE          ,SAVE REGISTERS ON THE STACK
1822 003356 012700 001020          MOV      #ATIMTBL, R0      ,GET TABLE ADDRESS CONTAINING TIMES
1823 003362 113701 001125          MOVB    @#ITCNT, R1      ,GET # OF ENTRIES (GIVEN BY ITERATION COUNT)
1824 003366 122701 000001          CMPB    #1, R1          ,BRANCH IF SINGLE ITERATION
1825 003372 001423          BEQ     4$
1826 003374 005002          CLR     R2          ,CLEAR 'SUM' REGISTERS
1827 003376 005003          CLR     R3
1828 003400 122701 000020          CMPB    #16, R1        ,BRANCH IF 16 ITERATIONS
1829 003404 001402          BEQ     1$
1830 003406 000000          HALT
1831 003410 000777          BR
1832
1833
1834 003412 062002          1$      ADD     (R0)+, R2      ,SUM INDIVIDUAL TIMES
1835 003414 005503          ADC     R3
1836 003416 005301          DEC     R1
1837 003420 001374          BNE     1$
1838
1839 003422 012700 000004          2$      MOV     #4, R0
1840 003426 006203          3$      ASR     R3          ,SHIFT TIME IN R3 & R2 4 PLACES
1841 003430 006002          ROR     R2          ,RIGHT = DIVIDE BY 16
1842 003432 005300          DEC     R0
1843 003434 001374          BNE     3$
1844 003436 010237 001016          MOV     R2, @#ATIME      ,MOVE AVERAGED TIMES
1845
1846 003442 113700 001126          4$      MOVB    @#TSTNUM, R0      ,GET TEST #
1847 003446 006300          ASL     R0
1848 003450 016067 001676 000002          MOV     NAMPTR(R0), 5$      ,GET TEST NAME STRING ADDRESS
1849 003456 000004          TYPE
1850 003460 000000          5$      WORD    0
1851 003462 113702 001126          MOVB    @#TSTNUM, R2      ,GET TEST #
1852 003466 004767 177364          JSR     PC, OUTSPC        ,OUTPUT TIMES
1853 003472 004767 177106          JSR     PC, RESTORE      ,RESTORE REGISTERS FROM THE STACK
1854 003476 000207          RTS     PC
1855
1856          SBTTL          INPUT SUBROUTINE
1857          ,SUBROUTINE TO GET TTY INPUT
1858          ,CALL JSR PC, INPUT
1859          INPUT DATA IS RETURNED IN BUFFER BEGINNING AT INBUF
1860
1861 003500 010046          INPUT  MOV     R0, -(SP)      ,SAVE R0 ON THE STACK
1862 003502 012700 001270          1$      MOV     #INBUF, R0
1863 003506 105737 177560          2$      TSTB   @#TKS
1864 003512 100375          BPL     2$
1865
1866 003514 113746 177562          MOVB    @#TKB, -(SP)      ,GET CHARACTER
1867 003520 042716 000200          BIC     #200, (SP)
1868 003524 122716 000177          CMPB    #177, (SP)      ,CHECK RUBOUT
1869 003530 001004          BNE     3$
1870 003532 124026          CMPB    -(R0), (SP)+      ,REMOVE CHARACTER FROM INPUT
1871 003534 000004 001402          TYPE   BKSLSH
    
```

```

1872 003540 000762          BR      2$          ,WAIT FOR NEXT CHARACTER
1873 003542 122716 000025 3$      CMPB   #CNTRLU,(SP) ,CHECK CONTROL U ( U)
1874 003546 001004          BNE    4$
1875 003550 005726          TST   (SP)+
1876 003552 000004 001400          TYPE,CRLF
1877 003556 000751          BR      1$
1878 003560 111637 001405 4$      MOVB   (SP),@#ECHO
1879 003564 111620          MOVB   (SP),(RO)+
1880 003566 122726 000015          CMPB   #CR,(SP)+
1881 003572 001403          BEQ   5$
1882 003574 000004 001405          TYPE,ECHO
1883 003600 000742          BR      2$
1884 003602 000004 001400 5$      TYPE,CRLF
1885 003606 012600          MOV   (SP)+,RO
1886 003610 000207          RTS    PC
1887
1888          ,KEYBOARD INTERRUPT SERVICE ROUTINE
1889 003612 113746 177562 TKISR MOVB   @#TKB,-(SP) ,GET TYPED CHARACTER
1890 003616 042716 000200          BIC   #200,(SP) ,STRIP PARITY BIT
1891 003622 122716 000017          CMPB   #CNTRL0,(SP) ,BRANCH IF NOT CONTROL 0 ( 0)
1892 003626 001003          BNE   1$
1893 003630 112667 176475          MOVB   (SP)+,#CNTRL0 ,SET CONTROL 0 INDICATOR IN TYPE ROUTINE
1894 003634 000002          RTI   ,EXIT
1895
1896 003636 122726 000003 1$      CMPB   #3,(SP)+ ,BRANCH IF NOT CONTROL C ( C)
1897 003642 001003          BNE   2$
1898 003644 000005          RESET
1899 003646 000137 005730          JMP   @#INIT ,RESTART PROGRAM
1900 003652 000002 2$      RTI   ,EXIT
    
```

```

1901          SBTTL          ERROR SERVICE ROUTINES
1902          ,ROUTINE TO PROCESS ERROR TRAPS (TRAPS TO 4)
1903 003654 000000          ERRTRP: HALT
1904
1905          ,ERROR SERVICE ROUTINE
1906          ,THIS ROUTINE PROCESSES TWO TYPES OF ERRORS (OUT OF RANGE AND HARDWARE)
1907          ;THE CALLS FOR AN OUT OF RANGE ERROR ARE <HLT+1>, <HLT+2> AND, FOR A
1908          ,HARDWARE ERROR THE CALL IS <HLT>
1909
1910 003656 004767 176112          HLT      JSR      PC,CKSWR          ,CHECK FOR CNTL G
1911 003662 004767 176674          JSR      PC,SAVE          ,SAVE REGISTERS ON THE STACK
1912 003666 110637 001127          1$     MOVB   SP,@#ERFLG          ,SET ERROR FLAG
1913 003672 032777 020000 175100          BIT      #SW13,@SWR          ,BRANCH IF NO TYP0UT
1914 003700 001075          BNE      4$
1915 003702 000004 014655          TYPE,E HDR
1916 003706 113702 001126          MOVB   @#TSTNUM,R2          ,GET TEST #
1917 003712 004767 176716          JSR      PC,TYPOCT          ,AND TYPE IT
1918 003716 016600 000016          MOV     16(SP),R0          ,GET RETURN PC
1919 003722 162700 000002          SUB     #2,R0              ;NOW PC OF HLT CALL
1920 003726 111000          MOVB   (R0),R0            ;NOW HLT CALL ITSELF
1921 003730 001417          BEQ     2$                ;BRANCH IF HLT
1922 003732 000004 014740          TYPE,E HDR2
1923 003736 122700 000002          CMPB   #2,R0              ,BRANCH IF NOT HLT+2
1924 003742 001005          BNE     10$
1925 003744 004767 177202          JSR      PC,OUTGAP          ,TYPE GAP SPECIFIED TIMES
1926 003750 000004 001400          TYPE,CRLF
1927 003754 000447          BR      4$
1928 003756 004767 177074          10$     JSR      PC,OUTSPC          ,TYPE SPECIFIED TIMES
1929 003762 000004 001400          TYPE,CRLF
1930 003766 000442          BR      4$
1931 003770 016500 000014          2$     MOV     ER(R5),R0
1932 003774 032765 002000 000032          BIT      #PE1600,TC(R5)
1933 004002 001403          BEQ     20$
1934 004004 042700 102100          BIC     #102100,R0
1935 004010 000402          BR      21$
1936 004012 042700 102300          20$     BIC     #102300,R0
1937 004016 005700          21$     TST     R0
1938 004020 001003          BNE     22$
1939 004022 000004 014631          TYPE,E SFT          ,TYPE SOFT ERROP MESSAGE
1940 004026 000434          BR      6$
1941
1942 004030 000004 014665          22$     TYPE,E HDR1
1943 004034 010500          MOV     R5,R0              ,GET FIRST ADDRESS OF REGS
1944 004036 012701 000007          MOV     #7,R1              TYPE FIRST 7 REGS
1945 004042 012002          3$     MOV     (R0)+,R2          ,GET REG CONTENTS
1946 004044 004767 176564          JSR      PC,TYPOCT          ,AND TYPE IT
1947 004050 000004 001413          TYPE,SPACE2
1948 004054 005301          DEC     R1
1949 004056 001371          BNE     3$
1950 004060 016502 000032          MOV     TC(R5),R2          ,GET CONTENTS OF TC REGISTER
1951 004064 004767 176544          JSR      PC,TYPOCT
1952 004070 000004 001400          TYPE,CRLF
1953
1954 004074 032777 001000 174676 4$          BIT      #SW09,@SWR          ,BRANCH IF NO RING THE BELL
1955 004102 001402          BEQ     5$
1956 004104 000004 001407          TYPE,BELL

```



ERROR SERVICE ROUTINES

1957	004110	005777	174664	55	TST	@SWR	.HALT ON ERROR?
1958	004114	100001			BPL	65	
1959	004116	000000			HALT		
1960	004120	004767	175650	65	JSR	PC, CKSWR	.CHECK FOR CNTL G
1961	004124	004767	176454		JSR	PC, RESTORE	;RESTORE REGISTERS FROM THE STACK
1962	004130	000002			RTI		.RETURN
1963							
1964							

```

1965          .SBTTL          SCOPE SUBROUTINE
1966          .SCOPE ROUTINE
1967          .THIS ROUTINE IS ENTERED UPON COMPLETION OF EACH SUBTEST
1968          .THE SCOPE ROUTINE:
1969          .   OUTPUTS TIME SPEC ON EACH ITERATION IF SW08 IS SET
1970          .   REPEATS TEST IF SW14 IS SET
1971          .   STORES ACTUAL TIME FOR FUNCTION IN TIME TABLE (ATIMTBL)
1972          .   PUBLISHES TIME IF SW10=0
1973          .   UPDATES ITERATION COUNT AND IF ITERATIONS COMPLETE CONTINUES
1974          .   TO NEXT TEST, OTHERWISE REPEATS TEST.
1975          .   DELAYS BEFORE CONTINUING OR REPEATING TEST
1976          .   INITIALIZES DRIVE
1977          .RETURNS      R5=BASE ADDRESS OF TMO2 REGISTERS (ADDRESS OF CS1)
1978          .              R1='DS' REG ADDRESS
1979          .              RO='FC' REG ADDRESS
1980
1981 004132 004767 175636          SCOPE JSR      PC,CKSWR          ;CHECK FOR CNTL G
1982 004136 013705 001014          MOV      @#TMBASE,R5          ;SET R5 TO FIRST TM REG
1983 004142 032777 000400 174630  BIT      #SW08,@SWR          ;BRANCH IF SPECIFICATION LINE
1984 004150 001404          BEQ      10$                ;NOT DESIRED ON EACH ITERATION
1985 004152 113702 001126          MOVB    @#TSTNUM,R2          ;GET TEST NUMBER
1986 004156 004767 176674          JSR      PC,OUTSPC          ;OUTPUT TIME RECORDED
1987 004162 032777 040000 174610 10$  BIT      #SW14,@SWR          ;BRANCH IF CONTINUOUS LOOP
1988 004170 001417          BEQ      2$                ;NOT DESIRED
1989 004172 004767 000546          JSR      PC,DELAY          ;DELAY 350 MS
1990 004176 004767 000776          JSR      PC,RHINIT          ;INIT
1991 004202 105037 001127          CLRB    @#ERFLG            ;CLEAR ERROR FLAG
1992 004206 016716 174574          MOV      SCPADR,(SP)
1993 004212 010501          MOV      R5,R1
1994 004214 062701 000012          ADD      #DS,R1            ;ADDRESS OF 'DS' REG IS IN R1
1995 004220 010500          MOV      R5,RO
1996 004222 062700 000006          ADD      #FC,RO            ;ADDRESS OF 'FC' REG IS IN RO
1997 004226 000002          RTI
1998
1999 004230 105737 001127          2$    TSTB    @#ERFLG          ;BRANCH IF ERROR FLAG IS SET
2000 004234 001006          BNE      3$
2001 004236 113700 001125          MOVB    @#ITCNT,RO          ;GET ITERATION COUNT
2002 004242 006300          ASL      RO                ;STORE TIME IN TABLE
2003 004244 013760 001016 001020  MOV      @#ATIME,ATIMTBL(RO)
2004 004252 105737 001125          3$    INCB    @#ITCNT          ;INCREMENT ITERATION COUNT
2005 004256 032777 004000 174514  BIT      #SW11,@SWR          ;BRANCH IF SINGLE ITERATION DESIRED
2006 004264 001004          BNE      4$
2007 004266 122737 000020 001125  CMPB    #16,@#ITCNT          ;BRANCH IF ITERATIONS INCOMPLETE
2008 004274 001336          BNE      1$
2009 004276 011637 001006          4$    MOV      (SP),@#SCPADR          ;SET SCOPE ADDRESS TO NEXT TEST
2010 004302 032777 002000 174470  BIT      #SW10,@SWR          ;BRANCH IF NO PUBLICATION DESIRED
2011 004310 001002          BNE      5$
2012 004312 004767 177034          JSR      PC,PUBLISH          ;GO PUBLISH TEST DATA
2013 004316 105037 001125          5$    CLRB    @#ITCNT          ;RESET ITERATION COUNT
2014 004322 105777 174452          TSTB    @SWR                ;BRANCH IF USER DOES NOT WANT TO
2015 004326 001721          BEQ      1$                ;HALT ON A SELECTED TEST
2016 004330 017746 174444          MOV      @SWR,-(SP)          ;GET SWITCHES
2017 004334 042716 177740          BIC      #177740,(SP)          ;CLEAR ALL BUT TEST #
2018 004340 005316          DEC      (SP)                ;FORM TEST # -1
2019 004342 123726 001126          CMPB    @#TSTNUM,(SP)+          ;BRANCH IF NOT AT TEST
2020 004346 001311          BNE      1$

```

```

2021 004350 000000          HALT
2022 004352 004767 175416   JSR      PC,CKSWR          ,CHECK FOR CNTL G
2023 004356 000705          BR       1$
2024
2025          ,SBTTL  TIMER SUBROUTINES
2026
2027          ,SUBROUTINE TO SYNCHRONIZE THE TIMER AND TURN IT ON.
2028          ,REGISTER 4 IS CLEARED, AND THE OSCILLATOR POLARITY IS MONITORED
2029          ,THE ROUTINE IS EXITED WHEN THE OSCILLATOR POLARITY CHANGES WITH R3
2030          ,SET TO INDICATE THE POLARITY OF THE OSCILLATOR.
2031          ,CALL JSR      PC,TIMON
2032          ,RETURNS   R3 SET TO INDICATE LAST POLARITY (+24/-24=0/1)
2033          ;          R4 = 0
2034
2035 004360 005004          TIMON CLR      R4          ;CLEAR TIME COUNT
2036 004362 012703 000024   MOV     #24,R3        ;SET POLARITY TO '0' STATE
2037 004366 032765 000100 000024 BIT     #OSC,MR(R5)    ;BRANCH IF POLARITY IS '0'
2038 004374 001405          BEQ     2$
2039 004376 032765 000100 000024 1$ BIT     #OSC,MR(R5)    ;WAIT FOR OSCILLATOR TO RETURN
2040 004404 001374          BNE     1$
2041 004406 000405          BR       4$
2042
2043 004410 005403          2$ NEG     R3          ;NEGATE PREV POLARITY INDICATOR
2044 004412 032765 000100 000024 3$ BIT     #OSC,MR(R5)    ;WAIT FOR OSCILLATOR TO RETURN
2045 004420 001774          BEQ     3$          ,TO '1' STATE
2046 004422 000207          4$ RTS      PC
2047
2048          ,SUBROUTINE TO COUNT TIME
2049          ,EACH TIME THE OSCILLATOR TOGGLES (BIT <06> IN MR REG) REGISTER
2050          ,R4 IS INCREMENTED, AND THE REGISTER R3 IS NEGATED TO INDICATE
2051          ,THE LAST STATE OF THE OSCILLATOR.
2052          ,CALL JMP     TIMER(R3)          ,R3 IS SET BY TIMON ROUTINE
2053          ,R2=RETURN ADDRESS TO CALLER
2054          ,NOTE THE TIME TO EXECUTE THIS ROUTINE IS VERY CRITICAL IT MUST BE
2055          ,LESS THAN 40 US.
2056
2057          ,ENTER HERE VIA JMP TIMER(R3) WHEN R3=-24 (PREV STATE=1)
2058 004424 032765 000100 000024 TIMER1: BIT     #OSC,MR(R5)    ;BRANCH IF CURRENT STATE IS '0'
2059 004432 001406          BEQ     TIMER          ,GO INCREMENT TIME
2060 004434 000112          JMP     (R2)          ;RETURN TO TEST
2061
2062          =TIMER1+24
2063 004450 005403          TIMER NEG     R3          ;NEGATE PREV STATE INDICATOR
2064 004452 005204          INC     R4          ,INCREMENT 'TICK' COUNT
2065 004454 100401          BMI     TIMERR        ,BRANCH ON OVERFLOW
2066 004456 000112          JMP     (R2)          ;RETURN TO TEST
2067 004460 000004 014766   TIMERR TYPE,E TIMOV    ;TYPE 'TIMER OVERFLOWED'
2068 004464 104400          HLT
2069 004466 000177 174314   JMP     @SCPADR        ;RETURN TO BEGINNING OF TEST
2070
2071          =TIMER+24
2072          ,ENTER HERE VIA JMP TIMER(R3) WHEN R3=+24 (PREV STATE=0)
2073 004474 032765 000100 000024 TIMERO BIT     #OSC,MR(R5)    ;BRANCH IF CURRENT STATE = '1'
2074 004502 001362          BNE     TIMER
2075 004504 000112          JMP     (R2)
2076

```

```

2077 ;SUBROUTINE TO CHECK TIME RECORDED BY SUBTEST
2078 ;THIS SUBROUTINE COMPUTES THE ACTUAL TIME (IN MICROSECONDS) AND CHECKS
2079 ;THAT THE TIME RECORDED BY THE SUBTEST IS CORRECT BY COMPARING THE TIME
2080 ;WITH THE HIGH LIMIT (STIMTBL(RO)) AND THE LOW LIMIT (STIMTBL+2(RO))
2081 ;IF THE TIME IS OUT OF RANGE AN OUT OF RANGE ERROR TYPEOUT RESULTS
2082 ;THE SUBROUTINE IS ENTERED WITH:
2083 ;
2084 ; R4=TICK COUNT
2085 TIMOK.
2086 004506 004767 176050 JSR PC, SAVE ;SAVE REGISTERS ON THE STACK
2087 004512 012700 000070 MOV #56,RO ;GET TIME PER TICK
2088 004516 010401 MOV R4,R1 ;GET TICKS COUNT
2089 004520 005002 CLR R2 ;CLEAR SUMMING REGISTERS
2090 004522 005003 CLR R3
2091 15. ADD RO,R2 ;MULTIPLY TIME PER TICK
2092 004526 005503 ADC R3 ;BY TICK COUNT
2093 004530 005301 DEC R1
2094 004532 001374 BNE 15
2095 004534 010246 MOV R2,-(SP) ;DIVIDE COUNT BY 10
2096
2097 004536 010346 MOV R3,-(SP)
2098 004540 012746 000012 MOV #10,-(SP)
2099 004544 004767 000262 JSR PC,DIVIDE
2100 004550 005726 TST (SP)+ ;DISCARD REMAINDER
2101 004552 012637 001016 MOV (SP)+,@#ATIME ;STORE QUOTIENT
2102 004556 113700 001126 MOVB @#TSTNUM,RO ;GET TEST #
2103 004562 006300 ASL 0
2104 004564 006300 ASL 0
2105 004566 023760 001016 001422 CMP @#ATIME,STIMTBL(RO) ;CHECK THAT TIME IS WITHIN
2106 004574 101004 BHI 25 ;LIMITS SPECIFIED
2107 004576 023760 001016 001424 CMP @#ATIME,STIMTBL+2(RO)
2108 004604 101001 BHI 35
2109 004606 104401 25 HLT+1 ;CALL ERROR ROUTINE
2110 004610 35
2111 004610 004767 175770 JSR PC,RESTORE ;RESTORE REGISTERS FROM THE STACK
2112 004614 000207 RTS PC ;RETURN
2113
2114 ;SUBROUTINE TO CHECK INDIVIDUAL GAP TIMES (PRODUCED BY TST021)
2115 ;SUBROUTINE COMPUTES THE ACTUAL TIME (IN MICROSECONDS) AND CHECKS
2116 ;THAT THE GAP TIME RECORDED BY THE SUBTEST (TST021) BY COMPARING THE
2117 ;TIME WITH THE MAX LIMIT (GTIMTBL-GAPTBL(R1)) AND THE MIN LIMIT
2118 ;(GTIMTBL+2-GAPTBL(R1)).
2119 ;CALL MOV #TICK COUNT,R4 ;R4 CONTAINS TICK COUNT
2120 ; MOVB #GAP,@#GAP ;LOCATION GAP CONTAINS GAP #
2121 ; JSR PC,GAPOK
2122
2123 GAPOK
2124 004616 004767 175740 JSR PC,SAVE ;SAVE REGISTERS ON THE STACK
2125 004622 012700 000070 MOV #56,RO ;GET TIME PER TICK
2126 004626 010401 MOV R4,R1 ;GET TICK COUNT
2127 004630 005002 CLR R2 ;CLEAR SUMMING REGISTERS
2128 004632 005003 CLR R3
2129 15. ADD RO,R2 ;MULTIPLY TICK COUNT
2130 004636 005503 ADC R3 ;BY TIME PER TICK
2131 004640 005301 DEC R1
2132 004642 001374 BNE 15

```

```

2133
2134 004644 010246          MOV     R2, -(SP)          ; DIVIDE TIME BY 10
2135 004646 010346          MOV     R3, -(SP)
2136 004650 012746 000012    MOV     #10, -(SP)
2137 004654 004767 000152    JSR     PC, DIVIDE
2138 004660 005726          TST     (SP)+             ; DISCARD REMAINDER
2139 004662 012637 001016    MOV     @#ATIME           ; STORE QUOTIENT
2140 004666 113703 001124    MOVB   @#GAP, R3         ; GET GAP #
2141 004672 006303          ASL     R3                ; MULTIPLY BY 4
2142 004674 006303          ASL     R3                ; TO GET AT TABLE ENTRY
2143 004676 023763 001016 001576  CMP     @#ATIME, GTIMTBL(R3) ; CHECK TIME (MAX)
2144 004704 101004          BHI     2$
2145 004706 023763 001016 001600  CMP     @#ATIME, GTIMTBL+2(R3) ; CHECK TIME (MIN)
2146 004714 101002          BHI     3$
2147 004716 104402          HLT+2  2$                ; REPORT OUT OF RANGE ERROR
2148 004720 000406          BR      100$
2149 004722 032777 000400 174050 3$    BIT     #SW08, @SWR      ; BRANCH IF TIMES NOT WANTED
2150 004730 001402          BEQ     100$
2151 004732 004767 176214          JSR     PC, OUTGAP       ; TYPE GAP TIMES
2152
2153 004736          100$
2154 004736 004767 175642          JSR     PC, RESTORE     ; RESTORE REGISTERS FROM THE STACK
2155 004742 000207          RTS     PC              ; RETURN TO TEST
2156
2157          SB^TL          DELAY SUBROUTINES
2158          ; THIS SUBROUTINE CAUSES A DELAY OF 350 MS
2159 004744 004767 177410  DELAY  JSR     PC, TIMON
2160 004750 010246          MOV     R2, -(SP)       ; SAVE R2 ON THE STACK
2161 004752 012702 004762          MOV     #2$, R2        ; SET RETURN ADDRESS FOR TIMER
2162 004756          1$.
2163 004756 000163 004450          JMP     TIMER(R3)      ; GO TO TIMER & RETURN VIA R2
2164 004762 032704 004000  2$.    BIT     #4000, R4
2165 004766 001773          BEQ     1$
2166 004770 012602          MOV     (SP)+, R2      ; RESTORE R2
2167 004772 000207          RTS     PC
2168
2169          ; THIS SUBROUTINE ALLOWS A CALLER SPECIFIED DELAY (UP TO 65MS )
2170          ; CALL. MOV     DELAY TIME, DELTIM      ; LOAD DELAY TIME (IN US)
2171          ; JSR     PC, DELAYV
2172 004774 005767 174120  DELAYV TST     DELTIM      ; BRANCH IF 0 DELAY
2173 005000 001413          BEQ     3$
2174 005002 004767 177352          JSR     PC, TIMON      ; TURN TIMER ON
2175 005006 010246          MOV     R2, -(SP)       ; SAVE R2 ON THE STACK
2176 005010 012702 005020          MOV     #2$, R2        ; SET RETURN ADDRESS FROM TIMER
2177 005014          1$.
2178 005014 000163 004450          JMP     TIMER(R3)      ; GO TO TIMER & RETURN VIA R2
2179 005020 023704 001120  2$.    CMP     @#DELTIM, R4
2180 005024 101373          BHI     1$
2181 005026 012602          MOV     (SP)+, R2      ; RESTORE R2
2182 005030 000207          3$.    RTS     PC
2183
2184          SB^TL          DIVIDE SUBROUTINE
2185          ; THIS SUBROUTINE DIVIDES A DOUBLE PRECISION # AND RETURNS THE RESULT
2186          ; TO THE CALLER ON THE STACK BOTH DIVIDEND & DIVISOR MUST BE POSITIVE
2187          ; CALL MOV     LEAST SIGNIFICANT HALF DIVIDEND, -(SP)
2188          ; MOV     #MOST SIGNIFICANT HALF DIVIDEND, -(SP)

```

```
2189      ,      MOV      #DIVISOR, -(SP)
2190      ,      JSR      PC, DIVIDE
2191      ;RETURN
2192      ,      (SP)=REMAINDER ON STACK
2193      ,      2(SP)=QUOTIENT
2194
2195      ;NOTE   THIS SUBROUTINE DESTROYS PREVIOUS CONTENTS OF R0, R1, R2 & R3
2196
2197 005032 005046      DIVIDE. CLR      -(SP)      ;SAVE LOC FOR SIGNS
2198 005034 012746 000021  MOV      #17, -(SP)    ;SET ITERATION COUNT
2199 005040 016601 000012  MOV      12(SP), R1    ;GET LSH DIVIDEND
2200 005044 016600 000010  MOV      10(SP), R0    ;GET MSH DIVIDEND
2201 005050 016602 000006  MOV      6(SP), R2     ;GET DIVISOR
2202 005054 005402      NEG      R2            ;NEGATE DIVISOR
2203 005056 000241      CLC                      ;CLEAR 'C' BIT IN PSW
2204 005060 000405      BR      2$
2205 005062 006100 1$     ROL      R0            ;ROTATE MSH DIVIDEND
2206 005064 010003      MOV      R0, R3       ;SAVE IN R3
2207 005066 060203      ADD      R2, R3       ;SUBTRACT DIVISOR FROM MSH DIVIDEND
2208 005070 103001      BCC     2$           ;BRANCH IF DIVIDEND > DIVISOR
2209 005072 010300      MOV      R3, R0       ;SAVE REMAINDER IN R0
2210 005074 006101 2$     ROL      R1            ;ROTATE LSH DIVIDEND
2211 005076 005316      DEC      (SP)         ;DECREMENT ITERATION COUNT
2212 005100 001370      BNE     1$
2213 005102 005726      TST     (SP)+         ;POP ITERATION COUNTER
2214 005104 005726      TST     (SP)+         ;POP SIGN CORRECTION
2215 005106 010166 000006  MOV      R1, 6(SP)     ;PUSH REMAINDER ON STACK
2216 005112 010066 000004  MOV      R0, 4(SP)     ;PUSH QUOTIENT ONTO STACK
2217 005116 012616      MOV      (SP)+, (SP)
```

```

2218 005120 000207          RTS      PC
2219
2220          SBTTL    DRIVE SUBROUTINES
2221          ,SUBROUTINE TO CHECK IF DRIVE IS AVAILABLE
2222          ,CALL    MOVB    DRIVE#,DRVNUM
2223          ,        JSR     PC,DRVAVA
2224          ,RETURN   'C' BIT SET IF NOT AVAILABLE
2225 005122 113765 001010 000010 DRVAVA MOVB    @DRVNUM,(S2(R5)) ;LOAD DRIVE #
2226 005130 032765 040000 000026          BIT     #TAP,DT(R5) ;CHECK IF TAPE UNIT
2227 005136 001003          BNE     1$
2228 005140 004767 000034          JSR     PC,RHINIT
2229 005144 000262          SEV     ;SET 'V' TO IND NOT AVAIL
2230 005146 000207          1$     RTS      PC ;RETURN
2231
2232          ,SUBROUTINE TO CHECK IF TE16 SLAVE IS AVAILABLE FOR TEST
2233          ,CALL    MOVB    DRIVE #,@DRVNUM ;PASS DRIVE # VIA DRVNUM
2234          ,        MOVB    SLAVE #,@SLVNUM ;PASS SLAVE # VIA SLVNUM
2235          ,        JSR     PC,SLVAVA ;CALL SUBROUTINE
2236 005150 113765 001010 000010 SLVAVA: MOVB    @DRVNUM,(S2(R5)) ;LOAD DRIVE #
2237 005156 113765 001011 000032          MOVB    @SLVNUM,TC(R5) ;AND SLAVE #
2238 005164 032765 002000 000026          BIT     #SPR,DT(R5) ;BRANCH IF SLAVE PRESENT
2239 005172 001001          BNE     1$
2240 005174 000262          SEV     ;SET 'V' TO INDICATE NO SLAVE
2241 005176 000207          1$     RTS      PC
2242
2243          ,SUBROUTINE TO INITIALIZE RH CONTROLLER
2244          ,CALL    JSR     PC,RHINIT
2245
2246 005200 012765 000040 000010 RHINIT MOV     #40,(S2(R5))
2247 005206 113765 001010 000010          MOVB    @DRVNUM,(S2(R5))
2248 005214 005046          CLR     -(SP)
2249 005216 113716 001011          MOVB    @SLVNUM,(SP)
2250 005222 012665 000032          MOV     (SP)+,TC(R5) ;LOAD SLAVE # INTO TC REG
2251 005226 052765 000300 000032          BIS     #NORM11,TC(R5)
2252 005234 000207          RTS      PC
2253
2254          ,SUBROUTINE TO WAIT FOR DRIVE READY (DRY)
2255 005236 005027          WAITRDY CLR    (PC)+ ;CLEAR WAIT TIMER
2256 005240 000000          WAITTIM WORD 0
2257 005242 105765 000012          1$     TSTB    DS(R5) ;WAIT FOR READY TO SET
2258 005246 100406          BMI     2$
2259 005250 005267 177764          INC     WAITTIM ;INCREMENT WAIT TIMER
2260 005254 001372          BNE     1$ ;BRANCH IF TIME HAS NOT EXPIRED
2261 005256 000004 015013          TYPE,E. TIMEEXP ;TYPE 'TIME EXPIRED WAITING FOR RDY'
2262 005262 000425          BR     99$ ;TAKE ERROR EXIT
2263 005264 032765 002000 000012 2$     BIT     #EOT,DS(R5) ;CHECK FOR END OF TAPE
2264 005272 001415          BEQ     3$ ;BRANCH IF NO EOT
2265 005274 000004 014060          TYPE,M. NAM
2266 005300 000004 014376          TYPE,M. EOT ;TYPE 'END OF TAPE'
2267 005304 004767 000032          JSR     PC,REWIND ;REWIND SLAVE
2268 005310 102412          BVS     99$ ;BRANCH IF ERROR ON REWIND
2269 005312 004767 000106          JSR     PC,WRITE ;WRITE A RECORD
2270 005316 005215          INC     (R5) ;SET 'GO' BIT
2271 005320 004767 177712          JSR     PC,WAITRDY ;WAIT FOR READY
2272 005324 000404          BR     99$ ;TAKE ERROR EXIT
2273 005326 032765 040000 000012 3$     BIT     #ERR,DS(R5) ;CHECK ERROR EXIT
  
```

2274	005334	001401		BEQ	100%
2275	005336	000262	99%	SEV	
2276	005340	000207	100%	RTS	PC
2277					



2278  
2279

,SUBROUTINE TO REWIND A UNIT (DRIVE/SLAVE COMBINATION)  
;CALL MOV8 DRIVE #, @#DRVNUM

```
2280      ,      MOVB      SLAVE #, @#SLVNUM
2281      ,      JSR      PC, REWIND
2282      , SUBROUTINE RETURNS TO CALLER WITH SELECTED SLAVE AT 'BOT', & 'V' SET IF
2283      , AN ERROR OCCURS
2284
2285 005342 004767 177632      REWIND JSR      PC, RHINIT      , INITIALIZE CONTROLLER
```

```

2286 005346 004367 000206 JSR R3, TMCMD , GO TO TM COMMAND SUBROUTINE
2287 005352 000000 WORD 0 , BUS ADDRESS (NOT USED)
2288 005354 000000 WORD 0 , WORD COUNT (NOT USED)
2289 005356 000000 WORD 0 , FRAME COUNT (NOT USED)
2290 005360 000006 WORD RWD , REWIND COMMAND
2291 005362 005215 INC (R5) , SET 'GO' BIT
2292 005364 032765 000002 000012 1$ BIT #BOT, DS(R5) , BRANCH IF 'BOT' SET
2293 005372 001005 BNE 2$
2294 005374 032765 040000 000012 BIT #ERR, DS(R5) , CHECK ERROR BIT
2295 005402 001006 BNE 99$ , BRANCH IF ERROR BIT SET
2296 005404 000767 BR 1$
2297
2298 005406 032765 020000 000012 2$ BIT #PIP, DS(R5) , WAIT FOR TAPE MOTION TO STOP
2299 005414 001374 BNE 2$
2300 005416 000401 BR 100$
2301 005420 000262 99$ SEV
2302 005422 000207 100$ RTS PC
2303
2304 , SUBROUTINE TO WRITE 256 WORD RECORD
2305 , CALL JSR PC, WRITE
2306
2307 005424 004367 000130 WRITE JSR R3, TMCMD , GO TO TM COMMAND SUBROUTINE
2308 005430 016420 WORD WTBUF , BUS ADDRESS
2309 005432 177600 WORD WRDCNT , WORD COUNT
2310 005434 177400 WORD FRMCNT , FRAME COUNT
2311 005436 000060 WORD WFWD , WRITE FORWARD COMMAND
2312 005440 000207 RTS PC
2313
2314 , SUBROUTINE TO READ A 256 WORD RECORD
2315 , CALL JSR PC, READ
2316
2317 005442 004337 005560 READ JSR R3, @#TMCMD
2318 005446 016420 WORD RDBUF , ADDRESS OF READ BUFFER
2319 005450 177600 WORD WRDCNT , 2'S COMPLEMENT OF WORD COUNT
2320 005452 177400 WORD FRMCNT , 2'S COMPLEMENT OF FRAME COUNT
2321 005454 000070 WORD RDFWD , READ FORWARD COMMAND
2322 005456 000207 RTS PC
2323
2324 , SUBROUTINE TO INITIATE READ REVERSE COMIAND
2325 , CALL JSR PC, REVRD
2326
2327 005460 004367 000074 REVRD JSR R3, TMCMD
2328 005464 017020 WORD RDBUF+256 , ADDRESS OF READ REVERSE BUFFER
2329 005466 177600 WORD WRDCNT , 2'S COMPLEMENT OF WORD COUNT
2330 005470 177400 WORD FRMCNT , 2'S COMPLEMENT OF FRAME COUNT
2331 005472 000076 WORD RDREV , READ REVERSE COMMAND
2332 005474 000207 RTS PC
2333
2334 , SUBROUTINE TO SPACE FORWARD 1 RECORD
2335 005476 012765 177777 000006 FWDSPC MOV #-1, FC(R5) , LOAD RECORD COUNT
2336 005504 012715 000031 MOV #SPCFWD+1, (R5) , LOAD COMMAND
2337 005510 004767 177522 JSR PC, WAITRDY , WAIT FOR READY
2338 005514 000207 RTS PC , RETURN
2339
2340 SUBROUTINE TO WRITE A RECORD AND BACK SPACE OVER THE RECORD
2341 005516 004767 177702 WRT BK JSR PC, WRITE , WRITE THE RECORD
    
```

2342	005522	005215		INC	(R5)		.SET 'GO' BIT
2343	005524	004767	177506	JSR	PC, WAITRDY		
2344	005530	102412		BVS	ZS		
2345	005532	012765	177777	MOV	#-1, FC(R5)		.LOAD RECORD COUNT
2346	005540	012715	000033	MOV	#SPCREV+1, (R5)		.LOAD COMMAND
2347	005544	004767	177466	JSR	PC, WAITRDY		
2348	005550	102402		BVS	ZS		
2349	005552	004767	177166	JSR	PC, DELAY		.WAIT FOR TAPE MOTION TO STOP
2350	005556	000207		RTS	PC		
2351							
2352							.SUBROUTINE TO LOAD A COMMAND
2353				.CALL	JSR	R3, TMCMD	
2354					WORD	BUS ADDRESS	
2355					WORD	WORD COUNT (2'S COMPLEMENT)	
2356					WORD	FRAME COUNT (2 S COMPLEMENT)	
2357					WORD	COMMAND	
2358							
2359	005560	012365	000004	TMCMD	MOV	(R3)+, BA(R5)	.LOAD BUS ADDRESS
2360	005564	012365	000002		MOV	(R3)+, WC(R5)	.LOAD WORD COUNT
2361	005570	012365	000006		MOV	(R3)+, FC(R5)	.LOAD FRAME COUNT
2362	005574	012315			MOV	(R3)+, (R5)	.LOAD COMMAND
2363	005576	000203			RTS	R3	.RETURN
2364							
2365							.SUBROUTINE TO PRINT TE16 SERIAL NUMBER
2366				.JSR	PC, SNPT		
2367							
2368	005600	016503	000030	SNPT	MOV	SN(R5), R3	
2369	005604	012701	001150		MOV	#ODIGITS, R1	
2370	005610	000303			SWAB	R3	
2371	005612	006003			ROR	R3	
2372	005614	006003			ROR	R3	
2373	005616	006003			ROR	R3	
2374	005620	006003			ROR	R3	GET FIRST DIGIT
2375	005622	042703	177760		BIC	#177760, R3	
2376	005626	052703	000260		BIS	#260, R3	
2377	005632	110321			MOVB	R3, (R1)+	.FILL FIRST DIGIT
2378	005634	016503	000030		MOV	SN(R5), R3	
2379	005640	000303			SWAB	R3	
2380	005642	042703	177760		BIC	#177760, R3	
2381	005646	052703	000260		BIS	#260, R3	
2382	005652	110321			MOVB	R3, (R1)+	.GET SECOND DIGIT
2383	005654	016503	000030		MOV	SN(R5), R3	
2384	005660	006003			ROR	R3	
2385	005662	006003			ROR	R3	
2386	005664	006003			ROR	R3	
2387	005666	006003			ROR	R3	
2388	005670	042703	177760		BIC	#177760, R3	
2389	005674	052703	000260		BIS	#260, R3	
2390	005700	110321			MOVB	R3, (R1)+	.GET THIRD DIGIT
2391	005702	016503	000030		MOV	SN(R5), R3	
2392	005706	042703	177760		BIC	#177760, R3	
2393	005712	052703	000260		BIS	#260, R3	
2394	005716	110321			MOVB	R3, (R1)+	.GET FOURTH DIGIT
2395	005720	105011			CLRB	(R1)	
2396	005722	000004	001150		TYPE, ODIGITS		.TYPE SERIAL NUMBER
2397	005726	000207			RTS	PC	.RETURN

2398

```
2399
2400          SBTTL PROGRAM INITIAL'ZATION
2401
2402 005730 012706 000600      INIT:  MOV      #STKPTR,SP          ,SET STACK PTR
2403
2404          ; ***** MODIFIED 11-8-77 *****
2405
2406          ;          DIAGNOSTIC SETUP FOR EXECUTION
2407          ;          UNDER ACT11
2408
2409          ;          +
2410          ;          +
2411
2412 005734 004767 005732      JSR      PC,CKMODE          ;CHECK FOR MODE OF OPERATION
2413 005740 005767 173020      TST      AUTOM          ;AUTOMATIC MODE?
2414 005744 001001              BNE      1$              ;BRANCH - IF YES
2415 005746 000410              BR       SUSWR          ;CHECK SWITCH REGISTER IN DUMP MODE
2416 005750 032737 020000 000052 1$  BIT      #20000, @#52      ;MANUAL INTERVENTION?
2417 005756 001404              BEQ      SUSWR          ;BRANCH - IF NO
2418 005760 000004 014036      TYPE,MSGB          ;TYPE ABORT MESSAGE
2419 005764 000167 005774      JMP      ABORT          ;AND ABORT THE PROGRAM
2420
2421          ; *****
2422
2423 005770 013746 000006      SUSWR  MOV      @#6, -(SP)          ,SAVE VECTORS
2424 005774 013746 000004      MOV      @#4, -(SP)
2425 006000 012737 006020 000004      MOV      #61$, @#4          ,SET UP FOR TIMEOUT
2426 006006 022777 177777 172764      CMP      #-1, @SWR          ;REFERENCE HARDWARE SWITCH REGISTER
2427 006014 001402              BEQ      60$
2428 006016 000404              BR       62$
2429 006020 022626      61$:  CMP      (SP)+, (SP)+          ;ADJUST STACK
2430 006022 012767 000176 172750 60$  MOV      #SWREG, SWR          ;POINT TO SOFTWARE SWITCH REG
2431 006030 012637 000004      62$:  MOV      (SP)+, @#4          ;RESTORE VECTORS
2432 006034 012637 000006      MOV      (SP)+, @#6
2433 006040 022737 000176 001000      CMP      #SWREG, @#SWR
2434 006046 001002              BNE      64$
2435 006050 004767 173760      JSR      PC,CNTLU
2436 006054 105037 001130      64$:  CLRB    @#PRGFLG          ;CLEAR PROGRAM FLAG
2437 006060 105037 001125      CLRB    @#ITCNT          ;CLEAR ITERATION COUNT
2438 006064 105037 001126      CLRB    @#TSTNUM          ;SET TEST # 0
2439 006070 105037 001127      CLRB    @#ERFLG          ;CLEAR ERROR FLAG
2440 006074 105067 173034      CLRB    ASFLG          ;CLEAR ASK FLAG
2441 006100 012737 000006 000004      MOV      #ERRVEC+2, @#ERRVEC
2442 006106 012737 000002 000006      MOV      #RT1, @#ERRVEC+2          ;CHECK IF 'LP' IS AVAILABLE
2443 006114 005037 001270      2$:  CLR     @#INBUF
2444
2445          ; ***** MODIFIED 11-8-77 *****
2446
2447          ;          IF IN ACT11 MODE INHIBIT TYPING PROGRAM
2448          ;          IDENTIFICATION AND MANUAL INTERVENTION
2449
2450 006120 005767 172642      TST      ACT11M          ;AUTOMATIC MODE?
2451 006124 001402              BEQ      7$              ;BRANCH - IF NO
2452 006176 000167 000662      JMP      SIZE          ;START THERE
2453
2454          ; *****
```

```

2455
2456 006132          75
2457
2458 006132 000004 001400      TYPE,CRLF
2459 006136 000004 014060      TYPE,M.NAM      ;TYPE TITLE
2460 006142 000004 014116      TYPE,I REG      ;ASK USER TO TYPE CONT BASE ADRS
2461 006146 004767 175326      JSR   PC, INPUT ;GET USER INPUT
2462 006152 004767 175106      JSR   PC,CNVTAO ;CONVERT ASCII TO OCTAL
2463 006156 013737 001122 001014  MOV   @#OCTALO,@#TMBASE ;SET NEW ADDRESS
2464 006164 013705 001014  55    MOV   @#TMBASE,R5
2465
2466      ;ROUTINE TO CHECK IF CONTROLLER (RH11) IS AVAILAABLE
2467 006170 000261      SEC
2468 006172 005715      TST   (R5)      ;SET 'C' IN PSW
2469 006174 103003      BCC   65        ;BRANCH IF CONTROLLER AVAIL
2470 006176 000004 014435      TYPE,E NCON
2471 006202 000652      BR    INIT
2472 006204 012737 003654 000004 65    MOV   #ERRTRP,@#ERRVEC      ;SET ERROR TRAP VECTOR
    
```

```

2473 ,ROUTINE TO GET TMO2 DRIVES USER DESIRES TO TEST
2474 006212 105037 001127 DRIVES CLR B @#ERFLG ;CLEAR ERROR FLAG
2475 006216 012701 001160 MOV #DRVTBL,R1 ;MARK ALL DRIVES AS NOT TO
2476 006222 012700 000004 MOV #4,RO ;BE TESTED A '0' INDICATES
2477 006226 005021 1$ CLR (R1)+ ;THAT A DRIVE IS NOT TO BE
2478 006230 005300 DEC RO ;TESTED
2479 006232 001375 BNE 1$
2480 006234 000004 014163 TYPE,1,DRVS
2481 006240 004767 175234 JSR PC, INPUT ;GET USER INPUT
2482 006244 012700 001270 MOV #INBUF,RO
2483 006250 122710 000101 CMPB #'A,(RO) ;AN 'A' SPECIFIES ALL
2484 006254 001013 BNE 3$ ;DRIVES TO BE TESTED
2485 006256 110667 172646 MOVB SP,PRGFLG ;SET FLAG TO IND ALL DRIVES
2486 006262 012701 001160 MOV #DRVTBL,R1 ;MARK ALL DRIVES TO BE TESTED
2487 006266 012700 000004 MOV #4,RO ;A '-1' INDICATES THAT A DRIVE
2488 006272 012721 177777 2$ MOV #-1,(R1)+ ;IS TO BE TESTED
2489 006276 005300 DEC RO
2490 006300 001374 BNE 2$
2491 006302 000417 BR CHKDRV ;GO CHECK DRIVE AVAILABILITY
2492
2493 ,GET USER SELECTED DRIVES AND MARK EACH DRIVE SELECTED TO BE TESTED
2494 006304 122710 000015 3$ CMPB #CR,(RO)
2495 006310 001414 BEQ CHKDRV
2496 006312 121027 000054 CMPB (RO),#', ;CHECK IF 'COMMA'
2497 006316 001001 BNE 4$
2498 006320 105720 TSTB (RO)+ ;STEP PTR PAST 'COMMA'
2499 006322 112001 4$ MOVB (RO)+,R1
2500 006324 042701 177770 BIC #177770,R1
2501 006330 112761 177777 001160 MOVB #-1,DRVTBL(R1)
2502 006336 000240 NOP
2503 006340 000761 BR 3$
2504
2505 ,ASCERTAIN THAT DRIVES (TMO2'S) SPECIFIED ARE AVAILABLE
2506 006342 005000 CHKDRV CLR RO ;A 0/-1 INDICATES THAT THE
2507 006344 105760 001160 1$ TSTB DRVTBL(RO) ;DRIVE IS NOT/IS TO BE TESTED
2508 006350 001005 BNE 3$
2509 006352 005200 2$ INC RO
2510 006354 122700 000010 CMPB #8,RO
2511 006360 001371 BNE 1$
2512 006362 000421 BR 4$
2513 006364 110037 001010 3$ MOVB RO,@#DRVNUM
2514 006370 104737 005122 JSR PC,@#DRVAVA ;CHECK IF AVAILABLE
2515 006374 102366 BVC 2$ ;'V' BIT SET INDICATES NOT AVAIL
2516 006376 000004 014502 TYPE,E,NDRV
2517 006402 116037 001136 014534 MOVB DIGTAB(RO),@#E.NAVA ;SET DRIVE # IN MESSAGE
2518 006410 000004 014534 TYPE,E,NAVA
2519 006414 110637 001127 MOVB SP,@#ERFLG ;SET 'ERROR' FLAG
2520 006420 105060 001160 CLR B DRVTBL(RO) ;MARK DRIVE UNAVAILABLE
2521 006424 000752 BR 2$ ;CHECK NEXT DRIVE
2522 006426 105737 001127 4$ TSTB @#ERFLG ;GO GET SLAVES IF NO ERROR
2523 006432 001403 BEQ SLAVES
2524 006434 105737 001130 TSTB @#PRGFLG ;ASK USER TO RETYPE DRIVES IF
2525 006440 001664 BEQ DRIVES ;'ALL' NOT SPECIFIED
2526
2527 ,ROUTINE TO GET SLAVES (TE16'S) USER DESIRES TO TEST
2528 006442 105037 001127 SLAVES CLR B @#ERFLG ;CLEAR 'ERROR' FLAG
    
```



```

2529 006446 012701 001170      MOV      #SLVTBL,R1      ;MARK ALL SLAVES (64.) AS NOT
2530 006452 012700 000040      MOV      #32.,RO        ;TO BE TESTED. A 0 INDICATES THAT
2531 006456 005021      1$      CLR      (R1)+          ;A DRIVE'S SLAVE IS NOT TO BE
2532 006460 005300      DEC      RO              ;TESTED
2533 006462 001375      BNE     1$
2534 006464 005000      CLR      RO              ;RO = DRIVE # FOR SLAVES
2535 006466 012701 001170      MOV      #SLVTBL,R1      ;R1 POINTS TO DRIVE'S SLAVE
2536 006472 105760 001160      2$      TSTB     DRVTBL(RO)     ;IF DRIVE IS TO BE TESTED
2537 006476 001007      BNE     4$              ;GO TO 4$ OTHERWISE
2538 006500 062701 000010      3$      ADD      #8.,R1         ;STEP SLAVE PTR TO NEXT DRIVE'S
2539 006504 005200      INC      RO              ;SLAVES AND INCREMENT DRIVE #
2540 006506 122700 000010      CMPB    #8.,RO          ;CHECK ALL DRIVES
2541 006512 001367      BNE     2$              ;AND WHEN ALL DRIVES CHECKED
2542 006514 000454      BR      CHKSLV          ;GO CHECK SLAVE AVAILABILITY
2543
2544 006516 105737 001130      4$:     TSTB     @#PRGFLG      ;BRANCH IF USER SELECTED ALL
2545 006522 001020      BNE     5$              ;DRIVES
2546 006524 110067 172260      MOVB    RO,DRVNUM       ;GET DRIVE #
2547 006530 116037 001136      014244 MOVB    DIGTAB(RO),@#1 DRV ;PREPARE USER ACTION MESSAGE
2548 006536 000004 014225      TYPE,1,SLVS
2549 006542 004767 174732      JSR     PC.,INPUT       ;GET USER INPUT
2550 006546 012703 001270      MOV      #INBUF,R3      ;SET PTR TO USER INPUT
2551 006552 122710 000101      CMPB    #'A,(RO)        ;BRANCH IF USER DOES NOT WANT
2552 006556 001015      BNE     7$              ;'ALL' SLAVES
2553 006560 110637 001130      MOVB    SP,@#PRGFLG     ;SET 'ALL' INDICATOR
2554 006564 012701 001170      5$      MOV      #SLVTBL,R1     ;MARK ALL SLAVES FOR ALL
2555 006570 012700 000040      MOV      #32.,RO        ;DRIVES AS TO BE TESTED
2556 006574 012721 177777      6$      MOV      #-1,(R1)+
2557 006600 005300      DEC      RO
2558 006602 001374      BNE     6$
2559 006604 105737 001130      TSTB    @#PRGFLG       ;BRANCH IF ALL WAS SELECTED
2560 006610 001016      BNE     CHKSLV
2561
2562 006612 122713 000015      7$      CMPB    #CR.(R3)        ;GET USER SELECTED SLAVES FOR
2563 006616 001730      BEQ     3$              ;DRIVE
2564 006620 121327 000054      CMPB    (R3),#',        ;STEP PTR PAST 'COMMA'
2565 006624 001001      BNE     8$
2566 006626 105723      TSTB    (R3)+
2567 006630 112304      8$      MOVB    (R3)+,R4        ;AND MARK SELECED SLAVE
2568 006632 042704 177770      BIC     #177770,R4     ;AS TO BE TESTED
2569 006636 060104      ADD     R1,R4
2570 006640 112714 177777      MOVB    #-1,(R4)
2571 006644 000762      BR      7$
2572
2573      ;ASCERTAIN THAT SLAVES (TE16'S) SELECTED ARE AVAILABLE
2574 006546 005000      CHKSLV CLR      RO        ;RO WILL CONTAIN THE DRIVE #
2575 006650 005001      CLR     R1              ;AND R1 THE SLAVE #
2576 006652 012702 001170      MOV     #SLVTBL,R2      ;SET PTR TO SLAVE TABLE
2577 006656 105760 001160      1$     TSTB    DRVTBL(RO)     ;BRANCH IF DRIVE SELECTED
2578 006662 001007      BNE     3$              ;& AVAILABLE FOR TEST
2579 006664 005200      2$     INC     RO              ;INCREMENT DRIVE #
2580 006666 062702 000010      ADD     #8.,R2         ;STEP SLAVE PTR TO NEXT DRIVE'S
2581 006672 022700 000010      CMP     #8.,RO          ;SLAVES BRANCH TO 1$ IF NOT ALL
2582 006676 001367      BNE     1$              ;DRIVES CHECKED OTHERWISE EXIT
2583 006700 000434      BR      7$
2584

```

```

2585 006702 005001 35 CLR R1 ;SET SLAVE # 0
2586 006704 105712 45 TSTB (R2) ;BRANCH IF DRIVE'S SLAVE IS SEL-
2587 006706 001006 BNE 65 ;ECTED FOR TEST
2588 006710 005201 55 INC R1 ; INCREMENT SLAVE #
2589 006712 005202 INC R2 ;STEP PTR TO NEXT SLAVE
2590 006714 022701 000010 CMP #8,R1 ;GO TO 45 IF ALL SLAVES NOT
2591 006720 001371 BNE 45 ;CHECKED
2592 006722 000760 BR 25 ;OTHERWISE GO TO 25 ABOVE
2593
2594 006724 110037 001010 65 MOVB RO,@#DRVNUM ;PASS DRIVE & SLAVE #
2595 006730 110137 001011 MOVB R1,@#SLVNUM
2596 006734 004737 005150 JSR PC,@#SLVAVA ;AND CHECK IF AVAILABLE
2597 006740 102363 BVC 55 ;'V' BIT SET ON RETURN IND-
2598 006742 116037 001136 014524 MOVB DIGTAB(RO),@#E.DRV ;ICATES ERROR PREPARE ERROR
2599 006750 116137 001136 014534 MOVB DIGTAB(R1),@#E.NAVA ;MESSAGE
2600 006756 000004 014516 TYPE,E.NSLV
2601 006762 110637 001127 MOVB SP,@#ERFLG ;SET ERROR INDICATOR
2602 006766 105012 CLR B (R2) ;CLEAR SLAVE TABLE ENTRY
2603 006770 000747 BR 55 ;GET NEXT SLAVE
2604
2605 006772 105737 001127 75 TSTB @#ERFLG ;BRANCH IF NO ERROR
2606 006776 001403 BEQ 100$
2607 007000 105737 001130 TSTB @#PRGFLG ;BRANCH IF NOT 'ALL'
2608 007004 001616 BEQ SLAVES ;ASK USER TO RETYPE SLAVES
2609 007006 012737 003654 000004 100$ MOV #ERPTR,@#ERRVEC
2610
2611 007014 SIZE
2612
2613 ;SCAN DRIVE AND SLAVE TABLE FOR DRIVE/SLAVE COMBINATION TO TEST
2614
2615 007014 105037 001010 CLR B @#DRVNUM ;SET DRIVE AND SLAVE # 0
2616 007020 105037 001011 CLR B @#SLVNUM
2617 007024 012737 001170 001012 MOV #SLVTBL,@#SLVPTR ;SET PTR TO SLAVE TABLE
2618 007032 105037 001131 CLR B @#UNTFND ;CLEAR 'UNIT FOUND' IND
2619
2620
2621 / *****
2622 /
2623 / MACRO FOR SIZING DRIVES AND SLAVES
2624 / IN AUTOMATIC MODE
2625 /
2626 / +
2627 /
2628 / +
2629 /
2630 007036 012767 177777 171744 MOV #-1,DRVNUM ;INIT DRIVE NUMBER
2631 007044 012767 177777 171737 NXTDRV MOV #-1,SLVNUM ;INIT SLAVE NUMBER
2632 007052 012777 000040 170730 1$ MOV #40,@CS2 ;INIT CONTROLLER
2633 007060 005267 171724 INC DRVNUM ;STEP DRIVE #
2634 007064 022767 000010 171716 CMP #10,DRVNUM ;ALL DRIVE TESTED?
2635 007072 001002 BNE 25 ;BRANCH - IF NO
2636 007074 000167 003746 JNZ FINISH ;EXIT
2637 007100 016777 171704 170702 25 MOV DRVNUM,@CS2 ;LOAD DRIVE #
2638 007106 005777 170666 TST @CS1 ;ACCESS DRIVE
2639 007112 032777 010000 170670 BIT #10000,@CS2 ;NON-EXISTANT DRIVE?
2640 007120 001354 BNE 1$ ;BRANCH - IF YES
    
```

```
2641 007122 005267 171663      NXTSLV: INC      SLVNUM      ,STEP SLAVE #
2642 007126 001010                BNE      1$          ,BRANCH IF NOT SLAVE 0
2643 007130 005767 171654                TST      DRVNUM     ,DRIVE 0?
2644 007134 001005                BNE      1$          ,BRANCH - IF NO
2645 007136 105767 171625                TSTB     XXDPM      ,XXDP?
2646 007142 001402                BEQ      1$          ,BRANCH - IF NO
2647 007144 005267 171641                INC      SLVNUM     ,STEP TO SLAVE #1
2648 007150 022767 000010 171633 1$      CMP      #10,SLVNUM ,ALL DRIVES TESTED?
2649 007156 001732                BEQ      NXTDRV     ,BRANCH - IF YES
2650 007160 016777 171625 170644                MOV      SLVNUM,@TC ,LOAD SLAVE UNIT?
2651 007166 032777 002000 170632                BIT      #2000,@DT  ,SLAVE PRESENT?
2652 007174 001752                BEQ      NXTSLV     ,BRANCH - IF NO (SPR=0)
2653 007176 032777 140000 170622                BIT      #140000,@DT ,IS DRIVE A TAPE UNIT?
2654 007204 001746                BEQ      NXTSLV     ,BRANCH IF NO
2655
2656 , *****
2657
2658 007206 113700 001010      BEGIN  MOVB     @#DRVNUM,R0 ,GET DRIVE #
2659 007212 113701 001011      MOVB     @#SLVNUM,R1 ,AND SLAVE #
```

2660	007216	013702	001012		MOV	@#SLVPTR,R2	,GET SLAVE PTR
2661	007222	105760	001160	1\$	TSTB	DRVTBL(RO)	,BRANCH IF DRIVE AVAIL TO TEST
2662	007226	001011			BNE	3\$	
2663	007230	005001			CLR	R1	,CLEAR SLAVE #
2664	007232	062702	000010		ADD	#8,R2	,AND STEP PTR TO NEXT DRIVE'S
2665	007236	005200		2\$	INC	RO	,SLAVES AND INCREMENT DRIVE #
2666	007240	022700	000010		CMP	#8,RO	,EXIT TEST IF ALL DRIVES
2667	007244	001366			BNE	1\$	,CHECKED OTHERWISE CONTINUE
2668	007246	000137	013162		JMP	@#END	,SCAN FOR NEXT 'UNIT'
2669							
2670	007252	105712		3\$	TSTB	(R2)	,BRANCH IF SLAVE ON DRIVE IS
2671	007254	001007			BNE	4\$	,AVAILABLE THERWISE STEP
2672	007256	005202			INC	R2	,PTR TO NEXT SLAVE
2673	007260	005201			INC	R1	,INCREMENT SLAVE #
2674	007262	122701	000010		CMPB	#8,R1	,UNTIL ALL SLAVES CHECKED
2675	007266	001371			BNE	3\$	,WHEN ALL SLAVES CHECKED
2676	007270	005001			CLR	R1	,SET SLAVE # 0
2677	007272	000761			BR	2\$	,AND CONTINUE SCAN
2678							
2679	007274	110637	001131	4\$	MOVB	SP,@#UNTFND	,INDICATE THAT A 'UNIT' IS FOUND
2680	007300	110037	001010		MOVB	RO,@#DRVNUM	,SET DRIVE 3

```

2681 007304 110137 001011      MOVB   R1, @#SLVNUM      ; SET SLAVE #
2682 007310 010237 001012      MOV    R2, @#SLVPTR      ; SAVE SLAVE PTR
2683
2684 007314 105737 001134      5$    TSTB  @#ASFLG
2685 007320 001047              BNE    7$
2686 007322 112767 000001 171604  MOVB   #1, ASFLG
2687 007330 105037 001130      CLRB   @#PRGFLG          ; CLEAR PROGRAM INDICATOR
2688
2689      , ***** MODIFIED 8-11-77 *****
2690
2691      ,
2692      , CHECK FOR AUTOMATIC MODE
2693 007334 005767 171424      TST    AUTOM              ; AUTOMATIC MODE?
2694 007340 001037              BNE    TYPHDR             ; BRANCH - IF YES
2695
2696      , *****
2697
2698 007342 000004 014304      TYPE, I SKEW              ; ASK USER IF HE WANTS TO RUN SKEW TESTS
2699 007346 004767 174126      JSR    PC, INPUT          ; GET USER INPUT
2700 007352 012703 001270      MOV    #INBUF, R3         ; GET REPLY
2701 007356 122713 000060      CMPB   #'0', (R3)         ; BRANCH IF 'NO' (0)
2702 007362 001406              BEQ    6$
2703 007364 122713 000061      CMPB   #'1', (R3)         ; CHECK IF 'YES' (1)
2704 007370 001351              BNE    5$                 ; NEITHER SO ASK AGAIN
2705 007372 111337 001130      MOVB   (R3), @#PRGFLG     ; SET INDICATOR
2706 007376 000420              BR     7$
2707
2708 007400 105037 001133      6$    CLRB   @#NRZFLG        ; CLEAR NRZ INDICATOR
2709 007404 000004 014345      TYPE, I NRZ              ; ASK USER IF DRIVE 'NRZ' ONLY
2710 007410 004767 174064      JSR    PC, INPUT          ; GET USER INPUT
2711 007414 012703 001270      MOV    #INBUF, R3         ; GET REPLY
2712 007420 122713 000060      CMPB   #'0', (R3)         ; BRANCH IF 'NO' (0)
2713 007424 001405              BEQ    7$
2714 007426 122713 000061      CMPB   #'1', (R3)         ; CHECK IF 'YES' (1)
2715 007432 001362              BNE    6$                 ; ASK AGAIN IF NEITHER
2716 007434 111337 001133      MOVB   (R3), @#NRZFLG    ; SET INDICATOR
2717
2718 007440      7$
2719
2720 007440      TYPHDR
2721
2722 007440 052737 000100 177560  BIS    #100, @#TKS        ; SET KEYBOARD IE BIT
2723 007446 000004 015064      TYPE, L HDR1
2724 007452 116037 001136 015244  MOVB   DIGTAB(R0), @#L DRV ; SET DRIVE #
2725 007460 116137 001136 015256  MOVB   DIGTAB(R1), @#L SLV ; AND SLAVE #
2726 007466 112737 000071 015261  MOVB   #'9, @#L.CHAN      ; GET SLAVES CHANNEL TYPE
2727 007474 032765 010000 000026  BIT    #CH7, DT(R5)
2728 007502 001403              BEQ    1$
2729 007504 112737 000067 015261  MOVB   #'7, @#L.CHAN      ; SET 7 CHANNEL
2730 007512 000004 015177      1$    TYPE, L HDR2
2731 007516 004767 176056      JSR    PC, SNPT           ; GO PRINT SERIAL NUMBER
2732 007522 000004 015300      TYPE, L HDR3
2733 007526 012737 007566 001006  MOV    #TST001, @#SCPADR  ; SET 'SCOPE' ADDRESS FOR FIRST TEST
2734 007534 010500              MOV    R5, R0
2735 007536 062700 000006      ADD    #FC, R0            ; R0 CONTAINS ADDRESS OF FC REG
2736 007542 010501              MOV    R5, R1
  
```

2737 007544 062701 000012  
2738 007550 012703 004450  
2739 007554 105737 001130  
2740 007560 001402  
2741 007562 000137 013326

ADD #DS,R1  
MOV #TIMER,R3  
TSTB @#PRGFLG  
BEQ TST001  
JMP @#SKEWTST

,R1 CONTAINS ADDRESS OF DS REG  
,SET JUMP ADDRESS TO TIMER  
,BRANCH IF NOT SKEW TESTS

```

2742          SBTTL START OF TESTS
2743          , TEST 001 - WRITE FROM BOT
2744          , THIS TEST WILL MEASURE ACCELERATION DELAY REQUIRED TO
2745          , MOVE THE TAPE APPROXIMATELY SEVEN (7) INCHES FORWARD
2746          , FROM DEAD STOP BEFORE STARTING TO TRANSFER DATA
2747
2748          , THIS TEST MEASURES TIME FROM 'GO'=1 TO 'ACCL'=0.
2749 007566 112737 0000C 001126 TST001 MOVB #1,@TSTNUM , SET TEST #
2750 007574 012702 007620      MOV #15,R2 , SET RETURN PC FROM TIMER
2751 007600 004767 175536      JSR PC,REWIND , REWIND SLAVE
2752 007604 102420      BVS 99$ , BRANCH IF ERROR ON REWIND
2753 007606 004767 175612      JSR PC,WRITE , GO SETUP WRITE COMMAND
2754 007612 004767 174542      JSR PC,TIMON , TURN TIMER ON
2755 007616 005215      INC (R5) , SET 'GO' BIT
2756
2757 007620 005765 000032 15 TST TC(R5) , BRANCH WHEN 'ACCL'=0
2758 007624 100002      BPL 2$
2759 007626 000163 004450      JMP TIMER(R3) , GO TO TIMER & RETURN VIA R2
2760
2761 007632 004767 175400 25 JSR PC,WAITRDY , WAIT FOR COMMAND TO FINISH
2762 007636 102403      BVS 99$ , BRANCH IF ERROR
2763 007640 004767 174642      JSR PC,TIMOK , GO CHECK TIME
2764 007644 000401      BR 100$
2765 007646 104400      99$ HLT
2766 007650 104000      100$ SCOPE
2767
2768          , TEST 002 - WRITE START
2769          , THIS TST MEASURES TIME FROM 'GO'=1 TO 'ACCL'=0
2770 007652 112737 000002 001126 TST002 MOVB #2,@TSTNUM , SET TEST # 2
2771 007660 004767 175540      JSR PC,WRITE , INITIATE WRITE COMMAND
2772 007664 012702 007676      MOV #15,R2 , SET RETURN PC FROM TIMER
2773 007670 004767 174464      JSR PC,TIMON
2774 007674 005215      INC (R5) , SET 'GO' BIT
2775
2776 007676 005765 000032 15 TST TC(R5) , BRANCH WHEN 'ACCL'=0
2777 007702 100002      BPL 2$
2778 007704 000163 004450      JMP TIMER(R3) , GO TO TIMER & RETURN VIA R2
2779
2780 007710 004767 175322 25 JSR PC,WAITRDY , WAIT FOR READY
2781 007714 102403      BVS 99$ , BRANCH IF ERROR
2782 007716 004767 174564      JSR PC,TIMOK , GO CHECK TIME RECORDED
2783 007722 000401      BR 100$ , EXIT VIA SCOPE
2784
2785 007724 104400      99$ HLT , REPORT ERROR
2786 007726 104000      100$ SCOPE
2787
2788          , TEST 003- WRITE SHUTDOWN
2789          , THIS TEST MEASURES TIME FROM 'FC REG'=0 TO 'SWDN'=1
2790 007730 112737 000003 001126 TST003 MOVB #3,@TSTNUM , SET TEST#3
2791 007736 004767 175462      JSR PC,WRITE , INITIATE WRITE COMMAND
2792 007742 005215      INC (R5) , SET 'GO' BIT
2793
2794 007744 005710 0400CF 15 TST (R0) , BRANCH WHEN WRITING FIN SHED
2795 007746 001404      BEQ 2$
2796 007750 032711      BIT #ERR,(R1) , MONITOR ERROR BIT
2797 007754 001017      BNE 99$
  
```

2798	007756	000772			BR	1%	
2799							
2800	007760			2%			
2801	007760	004767	174374		JSR	PC, TIMON	, TURN TIMER ON
2802	007764	010702			MOV	PC, R2	, LOAD RETURN PC FROM TIMER
2803	007766	032711	000020	3%	BIT	#SDWN, (R1)	, BRANCH WHEN DS <SDWN> SETS
2804	007772	001002			BNE	4%	
2805	007774	000163	004450		JMP	TIMER(R3)	, GO TO TIMER & RETURN VIA R2
2806							
2807	010000	004767	175232	4%	JSR	PC, WAITRDY	, WAIT FOR READY
2808	010004	102403			BVS	99%	
2809	010006	004767	174474		JSR	PC, TIMOK	, GO CHECK TIME RECORDED
2810	010012	000401			BR	100%	
2811	010014	104400		99%	HLT		, REPORT ERROR
2812	010016	104000		100%	SCOPE		
2813							
2814							
2815							, TEST 004 - WRITE SETTLEDOWN
2816	010020	112737	000004	001126	TST004	MOVB #4, @TSTNUM	, THIS TEST MEASURES TIME FROM 'SDWN'=1 TO 'SDWN'=0
2817	010026	004767	175372		JSR	PC, WRITE	
2818	010032	005215			INC	(R5)	, SET 'GO' BIT
2819							
2820	010034	005710		1%	TST	(R0)	, BRANCH WHEN WRITING FINISHED
2821	010036	001404			BEQ	2%	
2822	010040	032711	040000		BIT	#ERR, (R1)	, CHECK ERROR BIT
2823	010044	001026			BNE	99%	
2824	010046	000772			BR	1%	
2825							
2826	010050	032711	000020	2%	BIT	#SDWN, (R1)	, WAIT FOR ASSERTION OF 'SDWN'
2827	010054	001004			BNE	3%	
2828	010056	032711	040000		BIT	#ERR, (R1)	, MONITOR ERROR BIT
2829	010062	001017			BNE	99%	
2830	010064	000771			BR	2%	
2831							
2832	010066			3%			
2833	010066	004767	174266		JSR	PC, TIMON	, TURN TIMER ON
2834	010072	010702			MOV	PC, R2	, SET RETURN PC FROM TIMER
2835	010074	032711	000020		BIT	#SDWN, (R1)	, BRANCH WHEN SDWN CLEARS
2836	010100	001402			BEQ	5%	
2837	010102	000163	004450		JMP	TIMER(R3)	, GO TO TIMER & RETURN VIA R2
2838							
2839	010106	004767	175124	5%	JSR	PC, WAITRDY	, WAIT FOR READY
2840	010112	102403			BVS	99%	
2841	010114	004767	174366		JSR	PC, TIMOK	
2842	010120	000401			BR	100%	
2843							
2844	010122	104400		99%	HLT		
2845	010124	104000		100%	SCOPE		
2846							
2847							
2848							, TEST 005 - READ FROM BOT
2849	010126	112737	000005	001126	TST005	MOVB #5, @TSTNUM	, THIS TEST MEASURES TIME FROM 'GO'=1 TO 'ACCL'=0
2850	010134	004767	175202		JSR	PC, REWIND	, SET TEST #5
2851	010140	102422			BVS	99%	, REWIND SLAVE
2852	010142	004767	175274		JSR	PC, READ	, BRANCH IF ERROR ON REWIND
2853	010146	012702	010160		MOV	#15, R2	, SET RETURN PC FROM TIMER



2854	010152	004767	174202		JSR	PC, TIMON	, TURN TIMER ON
2855	010156	005215			INC	(R5)	, SET 'GO' BIT
2856							
2857	010160	005765	000032	15	TST	TC(R5)	, BRANCH WHEN 'ACCL' RESETS
2858	010164	100002			BPL	25	
2859	010166	000163	004450		JMP	TIMER(R3)	, GO TO TIMER & RETURN VIA R2
2860							
2861	010172	004767	175040	25	JSR	PC, WAITRDY	, WAIT FOR READY
2862	010176	102403			BVS	995	, BRANCH IF ERROR
2863	010200	004767	174302		JSR	PC, TIMOK	, CHECK RECORDED TIME
2864	010204	000401			BR	1005	
2865							
2866	010206	104400		995	HLT		
2867	010210	104000		1005	SCOPE		
2868							
2869							, TEST 006 - READ START
2870							, THIS TEST MEASURES TIME FROM 'GO'=1 TO 'ACCL'=0
2871	010212	112737	000006	001126	TST006	MOVB #6, @TSTNUM	, SET TEST #6
2872	010220	004767	175272		JSR	PC, WRT. BK	, WRITE A RECORD & BACK SPACE
2873	010224	102422			BVS	995	
2874	010226	004767	175210		JSR	PC, READ	
2875	010232	012702	010244		MOV	#15, R2	, SET RETURN PC FROM TIMER
2876	010236	004767	174116		JSR	PC, TIMON	, TURN TIMER ON
2877	010242	005215			INC	(R5)	, SET 'GO' BIT
2878							
2879	010244	005765	000032	15	TST	TC(R5)	, BRANCH WHEN 'ACCL' RESETS
2880	010250	100002			BPL	25	
2881	010252	000163	004450		JMP	TIMER(R3)	, GO TO TIMER & RETURN VIA R2
2882							
2883	010256	004767	174754	25	JSR	PC, WAITRDY	
2884	010262	102403			BVS	995	
2885	010264	004767	174216		JSP	PC, TIMOK	
2886	010270	000401			BR	1005	
2887							
2888	010272	104400		995	HLT		
2889	010274	104000		1005	SCOPE		
2890							
2891							, TEST 007 - READ SHUTDOWN
2892							, THIS TEST MEASURES TIME FROM 'FC REG'=FRAME COUNT TO 'SDWN'=1
2893	010276	112737	000007	001126	TST007	MOVB #7, @TSTNUM	, SET TEST #7
2894	010304	004767	175206		JSR	PC, WRT BK	, WRITE A RECORD & BACK SPACE
2895	010310	102430			BVS	995	, BRANCH IF ERROR
2896	010312	004767	175124		JSR	PC, READ	
2897	010316	005215			INC	(R5)	, SET 'GO' BIT
2898							
2899	010320	022710	000400	15	CMP	#-FRMCNT (R0)	, WAIT FOR FRAME COUNT TO
2900	010324	001404			BEQ	25	, = # OF FRAMES WRITTEN
2901	010326	032711	040000		BIT	#ERR, (R1)	, MONITOR ERROR BIT
2902	010332	001017			BNE	995	
2903	010334	000771			BR	15	
2904							
2905	010336			25			
2906	010336	004767	174016		JSR	PC, TIMON	, TURN TIMER ON
2907	010342	010702			MOV	PC, R2	, SET RETURN PC FROM TIMER
2908	010344	032711	000020		BIT	#SDWN, (R1)	, BRANCH WHEN SDWN SETS
2909	010350	001002			BNE	35	

```

2910 010352 000163 004450          JMP      TIMER(R3)          ,GO TO TIMER & RETURN VIA R2
2911
2912 010356 004767 174654          3$      JSR      PC, WAITRDY
2913 010362 102403          BVS     99$
2914 010364 004767 174116          JSR     PC, TIMOK
2915 010370 000401          BR      100$
2916
2917 010372 104400          99$     HLT
2918 010374 104000          100$    SCOPE          ,REPORT ERROR
2919
2920
2921          , TEST 010 - READ SETTLEDOWN
2922          , THIS TEST MEASURES TIME FROM 'SWDN'=1 TO 'SWDN'=0
2922 010376 112737 000010 001126 TST010  MOVB   #10, @#TSTNUM      ;SET TEST #10
2923 010404 012702 010462          MOV     #4$, R2          ;SET RETURN PC FROM TIMER
2924 010410 004767 175102          JSR     PC, WRT BK       ;WRITE A RECORD & BACK SPACE
2925 010414 102436          BVS     99$
2926 010416 004767 175020          JSR     PC, READ
2927 010422 005215          INC     (R5)            ,SET 'GO' BIT
2928
2929 010424 105711          1$      TSTB   (R1)          ;WAIT FOR READY
2930 010426 100404          BMI     2$              ;BRANCH WHEN SET
2931 010430 032711 040000          BIT     #ERR, (R1)      ;CHECK ERROR BIT
2932 010434 001026          BNE     99$
2933 010436 000772          BR      1$
2934
2935 010440 032711 000020          2$      BIT     #SDWN, (R1)     ;WAIT FOR ASSERTION OF 'SDWN'
2936 010444 001004          BNE     3$
2937 010446 032711 040000          BIT     #ERR, (R1)     ;MONITOR ERROR BIT
2938 010452 001017          BNE     99$
2939 010454 000771          BR      2$
2940
2941 010456          3$
2942 010456 004767 173676          JSR     PC, TIMON       ;TURN TIMER ON
2943 010462 032765 000020 000012 4$      BIT     #SDWN, DS(R5)   ;WAIT FOR NEGATION OF SDWN
2944 010470 001402          BEQ     5$
2945 010472 000163 004450          JMP     TIMER(R3)      ,GO TO TIMER & RETURN VIA R2
2946
2947 010476 004767 174534          5$      JSR     PC, WAITRDY
2948 010502 102403          BVS     99$
2949 010504 004767 173776          JSR     PC, TIMOK
2950 010510 000401          BR      100$
2951
2952 010512 104400          99$     HLT
2953 010514 104000          100$    SCOPE
2954
2955
2956          , TEST 011-READ REVERSE START
2957          , THIS TEST MEASURES TIME FROM 'GO'=1 TO 'ACCL'=0
2958 010516 112737 000011 001126 TST011  MOVB   #11, @#TSTNUM      ;SET RETURN PC FROM TIMER
2959 010524 012702 010562          MOV     #1$, R2          ;WRITE A RECORD
2960 010530 004767 174670          JSR     PC, WRITE
2961 010534 005215          INC     (R5)            ;SET 'GO' BIT
2962 010536 004767 174474          JSR     PC, WAITRDY
2963 010542 102422          BVS     99$
2964 010544 004767 174174          JSR     PC, DELAY
2965 010550 004767 174704          JSR     PC, REVRO      ,WAIT FOR TAPE MOTION TO STOP
    
```

```

2966 010554 004767 173600          JSR   PC,TIMON          ,TURN TIMER ON
2967 010560 005215                   INC   (R5)              ;SET 'GO' BIT
2968
2969 010562 005765 000032          1$   TST   TC(R5)       ,BRANCH WHEN 'ACCL' = 0
2970 010566 100002                   BPL   2$                ;
2971 010570 000163 004450          JMP   TIMER(R3)        ;GO TO TIMER & RETURN VIA R2
2972
2973 010574 004767 174436          2$   JSR   PC,WAITRDY    ,BRANCH IF ERROR
2974 010600 102403                   BVS   99$
2975 010602 004767 173700          JSR   PC,TIMOK
2976 010606 000401                   BR    100$
2977
2978 010610 104400          99$   HLT
2979 010612 104000          100$  SCOPE
2980
2981          ;TEST 012-READ REVERSE SHUTDOWN
2982          ;THIS TEST MEASURES TIME FROM 'FC REG' = FRAME COUNT TO 'SDWN'=1
2983 010614 112737 000012 001126  TST012: MOVB  #12, @#TSTNUM
2984 010622 012702 010672          MOV   #3$, R2          ,SET RETURN PC FROM TIMER
2985 010626 004767 174572          JSR   PC,WRITE         ,WRITE A RECORD
2986 010632 005215                   INC   (R5)              ;SET 'GO' BIT
2987 010634 004767 174376          JSR   PC,WAITRDY
2988 010640 102427                   BVS   99$
2989 010642 004767 174612          JSR   PC,REVRD
2990 010646 005215                   INC   (R5)              ;SET 'GO' BIT
2991
2992 010650 022710 000400          1$   CMP   #-FRMNT, (R0)   ;BRANCH WHEN FRAME COUNT
2993 010654 001404                   BEQ   2$                ;= # OF RECORD WRITTEN
2994 010656 032711 040000          BIT   #ERR, (R1)       ;MONITOR ERROR BIT IN 'DS' REG
2995 010662 001016                   BNE   99$
2996 010664 000771                   BR    1$
2997
2998 010666          2$
2999 010666 004767 173466          JSR   PC,TIMON          ,TURN TIMER ON
3000 010672 032711 000020          3$   BIT   #SDWN, (R1)   ,BRANCH WHEN SDWN SETS
3001 010676 001002                   BNE   4$
3002 010700 000163 004450          JMP   TIMER(R3)        ;GO TO TIMER & RETURN VIA R2
3003
3004 010704 004767 174326          4$   JSR   PC,WAITRDY    ,WAIT FOR READY
3005 010710 102403                   BVS   99$
3006 010712 004767 173570          JSR   PC,TIMOK
3007 010716 000401                   BR    100$
3008
3009 010720 104400          99$   HLT
3010 010722 104000          100$  SCOPE
3011
3012          ;TEST 013-READ REVERSE SETTLEDOWN
3013          ;THIS TEST MEASURES TIME FROM 'SDWN'=1 TO 'SDWN'=0
3014 010724 112737 000013 001126  TST013: MOVB  #13, @#TSTNUM
3015 010732 012702 011016          MOV   #4$, R2          ,SET RETURN PC FROM TIMER
3016 010736 004767 174462          JSR   PC,WRITE         ,WRITE A RECORD
3017 010742 005215                   INC   (R5)              ;SET 'GO' BIT
3018 010744 004767 174266          JSR   PC,WAITRDY
3019 010750 102435                   BVS   99$
3020 010752 004767 174502          JSR   PC,REVRD
3021 010756 005215                   INC   (R5)              ;SET 'GO' BIT

```

```

3022
3023 010760 105711          1$   TSTB  (R1)          ; BRANCH WHEN
3024 010762 100404          BMI   2$           ; READY SETS
3025 010764 032711 040000   BIT   #ERR, (R1)
3026 010770 001025          BNE   99$
3027 010772 000772          BR    1$
3028
3029 010774 032711 000020   2$   BIT   #SDWN, (R1)
3030 011000 001004          BNE   3$
3031 011002 032711 040000   BIT   #ERR, (R1)
3032 011006 001016          BNE   99$
3033 011010 000771          BR    2$
3034
3035 011012          3$
3036 011012 004767 173342   JSR   PC, TIMON      ; TURN TIMER ON
3037 011016 032711 000020   4$   BIT   #SDWN, (R1)   ; BRANCH WHEN SWDN = 0
3038 011022 001402          BEQ   5$
3039 011024 000163 004450   JMP   TIMER(R3)      ; GO TO TIMER & RETURN VIA R2
3040
3041 011030 004767 174202   5$   JSR   PC, WAITRDY    ; WAIT FOR READY
3042 011034 102403          BVS   99$
3043 011036 004767 173444   JSR   PC, TIMOK
3044 011042 000401          BR    100$
3045
3046 011044 104400          99$   HLT
3047 011046 104000          100$  SCOPE
3048
3049          ; REWIND DRIVE
3050 011050          A
3051 011050 004767 174266   JSR   PC, .REWIND    ; REWIND SLAVE
3052 011054 102401          BVS   99$            ; BRANCH IF ERROR ON REWIND
3053 011056 102002          BVC   100$
3054 011060 104400          99$   HLT
3055 011062 000772          BR    A
3056 011064          100$
3057
3058          ; TEST 014-TURN AROUND DELAY (FORWARD-REVERSE)
3059          ; THIS TEST MEASURES TIME FROM 'GO'=1 (READ REVERSE) TO 'ACCL'=0
3060 011064 112737 000014 001126 TST014 MOV8  #14, #TSTNUM
3061 011072 012702 011124   MOV   #2$, R2        ; SET RETURN PC FROM TIMER
3062 011076 004767 174322   JSR   PC, WRITE      ; WRITE A RECORD
3063 011102 005215          INC   (R5)           ; SET 'GO' BIT
3064 011104 004767 174126   JSR   PC, WAITRDY
3065 011110 102420          BVS   99$
3066
3067 011112 004767 174342   1$   JSR   PC, REVURD     ; READ THE RECORD (REVERSE)
3068 011116 004767 173236   JSR   PC, TIMON      ; TURN TIMER ON
3069 011122 005215          INC   (R5)           ; SET 'GO' BIT
3070
3071 011124 005765 000032   2$   TST   TC(R5)        ; WAIT FOR 'ACCL' = 0
3072 011130 100002          BPL   3$
3073 011132 000163 004450   JMP   TIMER(R3)      ; GO TO TIMER & RETURN VIA R2
3074
3075 011136 004767 174074   3$   JSR   PC, WAITRDY
3076 011142 102403          BVS   99$
3077 011144 004767 173336   JSR   PC, TIMOK
  
```

```

3078 011150 000401          BR      100$
3079
3080 011152 104400          99$:  HLT
3081 011154 104000          100$: SCOPE
3082
3083          , TEST 015- TURN AROUND DELAY (REVERSE-FORWARD)
3084          , THIS TEST MEASURES TIME FROM 'GO'=1 (READ) TO 'ACCL'=0
3085 011156 112737 000015 001126 TST015: MOVB  #15, @#TSTNUM
3086 011164 012702 011232      MOV  #15, R2          ; SET RETURN PC FROM TIMER
3087 011170 004767 174230      JSR  PC, WRITE        ; WRITE A RECORD
3088 011174 005215          INC  (R5)             ; SET 'GO' BIT
3089 011176 004767 174034      JSR  PC, WAITRDY      ; WAIT FOR READY
3090 011202 102426          BVS  99$
3091 011204 004767 174250      JSR  PC, REVRD        ; READ A RECORD IN THE
3092 011210 005215          INC  (R5)             ; SET 'GO' BIT
3093
3094 011212 004767 174020      JSR  PC, WAITRDY
3095 011216 102420          BVS  99$
3096
3097 011220 004767 174216      1$   JSR  PC, READ        ; READ RECORD FORWARD
3098 011224 004767 173130      JSR  PC, TIMON        ; TURN TIMER ON
3099 011230 005215          INC  (R5)             ; SET 'GO' BIT
3100
3101 011232 005765 000032      2$   TST  TC(R5)        ; WAIT FOR 'ACCL' = 0
3102 011236 100002          BPL  3$
3103 011240 000163 004450      JMP  TIMER(R3)        ; GO TO TIMER & RETURN VIA R2
3104
3105 011244 004767 173766      3$   JSR  PC, WP TRDY
3106 011250 102403          BVS  99$
3107 011252 004767 173230      JSR  PC, TIMOK
3108 011256 000401          BR      100$
3109
3110 011260 104400          99$:  HLT
3111 011262 104000          100$: SCOPE
3112
3113          , TEST 016-GAP SIZE (STOP HALF)
3114 011264 112737 000016 001126 TST016: MOVB  #16, @#TSTNUM
3115 011272 012702 011330      MOV  #15, R2          ; SET RETURN PC FROM TIMER
3116 011276 004767 174122      JSR  PC, WRITE        ; WRITE A RECORD
3117 011302 005215          INC  (R5)             ; SET 'GO' BIT
3118 011304 004767 173726      JSR  PC, WAITRDY
3119 011310 102421          BVS  99$
3120 011312 004767 173426      JSR  PC, DELAY        ; DELAY 350 MS
3121 011316 004767 174136      JSR  PC, REVRD        ; READ REVERSE RECORD
3122 011322 004767 173032      JSR  PC, TIMON        ; TURN TIMER ON
3123 011326 005215          INC  (R5)             ; SET 'GO' BIT
3124
3125 011330 005710          1$   TST  (R0)            ; WAIT FOR FRAME COUNT > 0
3126 011332 001002          BNE  2$
3127 011334 000163 004450      JMP  TIMER(R3)        ; GO TO TIMER & RETURN VIA R2
3128
3129 011340 004767 173672      2$   JSR  PC, WAITRDY      ; WAIT FOR READY BIT TO SET
3130 011344 102403          BVS  99$
3131 011346 004767 173134      JSR  PC, TIMOK        ; CHECK TIME
3132 011352 000401          BR      100$
3133
    
```

3134	011354	104400		99%	HLT	
3135	011356	104000		100%	SCOPE	
3136						
3137						
3138	011360	112737	000017	001126	TST017	MOV8 #17, @#TSTNUM
3139	011366	012702	011440		MOV	#15, R2 ; SET RETURN PC FROM TIMER
3140	011372	004767	174026		JSR	PC, WRITE ; WRITE A RECORD
3141	011376	005215			INC	(R5) ; SET 'GO' BIT
3142	011400	004767	173632		JSR	PC, WAITRDY ; WAIT FOR READY
3143	011404	102427			BVS	99%
3144	011406	004767	174046		JSR	PC, REVRD ; READ REVERSE THE RECORD
3145	011412	005215			INC	(R5) ; SET 'GO' BIT
3146	011414	004767	173616		JSR	PC, WAITRDY ; WAIT FOR READY
3147	011420	102421			BVS	99% ; BRANCH ON ERROR
3148	011422	004767	173316		JSR	PC, DELAY ; WAIT FOR TAPE MOTION TO STOP
3149	011426	004767	174010		JSR	PC, READ ; READ RECORD
3150	011432	004767	172722		JSR	PC, TIMON ; TURN TIMER ON
3151	011436	005215			INC	(R5) ; SET 'GO' BIT
3152						
3153	011440	005710		15	TST	(R0) ; WAIT FOR FRAME COUNT > 0
3154	011442	001002			BNE	2%
3155	011444	000163	004450		JMP	TIMER(R3) ; GO TO TIMER & RETURN VIA R2
3156						
3157	011450	004767	173562	25	JSR	PC, WAITRDY ; WAIT FOR READY
3158	011454	102403			BVS	99%
3159	011456	004767	173024		JSR	PC, TIMOK ; CHECK TIME
3160	011462	000401			BR	100%
3161						
3162	011464	104400		99%	HLT	
3163	011466	104000		100%	SCOPE	
3164						
3165						
3166						
3167	011470	112737	000020	001126	TST020	MOV8 #20, @#TSTNUM
3168	011476	012702	011560		MOV	#15, R2 ; SET RETURN PC FROM TIMER
3169	011502	004767	173716		JSR	PC, WRITE ; WRITE A RECORD
3170	011506	005215			INC	(R5) ; SET 'GO' BIT
3171	011510	004767	173522		JSR	PC, WAITRDY ; WAIT FOR READY
3172	011514	102433			BVS	99%
3173	011516	004767	173702		JSR	PC, WRITE ; WRITE SECOND RECORD
3174	011522	005215			INC	(R5) ; SET 'GO' BIT
3175	011524	004767	173506		JSR	PC, WAITRDY ; WAIT FOR READY
3176	011530	102425			BVS	99%
3177	011532	004767	173722		JSR	PC, REVRD ; READ REVERSE SECOND RECORD
3178	011536	005215			INC	(R5) ; SET 'GO' BIT
3179	011540	004767	173472		JSR	PC, WAITRDY ; WAIT FOR READY
3180	011544	102417			BVS	99%
3181	011546	004767	173706		JSR	PC, REVRD ; READ REVERSE FIRST RECORD
3182	011552	004767	172602		JSR	PC, TIMON ; TURN TIMER ON
3183	011556	005215			INC	(R5) ; SET 'GO' BIT
3184						
3185	011560	005710		15	TST	(R0) ; WAIT FOR FRAME COUNT > 0
3186	011562	001002			BNE	2%
3187	011564	000163	004450		JMP	TIMER(R3) ; GO TO TIMER & RETURN VIA R2
3188						
3189	011570	004767	173442	25	JSR	PC, WAITRDY ; WAIT FOR READY

3190 011574 102403  
 3191 011576 004767 172704  
 3192 011602 000401  
 3193  
 3194 011604 104400  
 3195 011606 104000

BVS 99%  
 JSR PC,TIMOK  
 BR 100%  
 99% HLT  
 100% SCOPE

3196  
 3197  
 3198  
 3199  
 3200  
 3201  
 3202  
 3203  
 3204  
 3205  
 3206  
 3207  
 3208

, TEST 021- GAP CONSISTANCY  
 , THIS TEST MEASURES TIME FROM 'GO'=1 TO 'FC REG' > 0  
 , THE TEST REWINDS THE TAPE, WRITES 17 RECORDS WITH A DELAY FROM 1-16 MS  
 , BETWEEN EACH WRITE COMMAND. AFTER THE 17 RECORDS ARE WRITTEN THE  
 , PROGRAM READ REVERSES 16 RECORDS. AT THIS POINT THE TAPE IS STOPPED BE-  
 , TWEEN THE FIRST AND SECOND RECORD. A READ COMMAND IS EXECUTED TO READ  
 , THE 16 RECORDS WITH THE TIME BETWEEN GO=1 TO FC > 0 STORED IN 'GAPTBL'  
 , FOR EACH RECORD READ. AFTER 16 RECORDS HAVE BEEN READ THE TIME IS VER-  
 , IFIED FOR EACH READ. AFTER ALL RECORD TIMES ARE VERIFIED THEY ARE AVER-  
 , AGED AND PLACED IN THE 'ATIMTBL' (BY SCOPE). THE ABOVE PROCESS IS RE-  
 , PEATED FOR EACH ITERATION.

3209 011610 112737 000021 001126  
 3210 011616 012702 011754  
 3211 011622 004767 173514  
 3212 011626 102530  
 3213 011630 005067 167264  
 3214 011634 012700 000021  
 3215 011640 004767 173560  
 3216 011644 005215  
 3217 011646 004767 173364  
 3218 011652 102516  
 3219 011654 004767 173114  
 3220 011660 062767 000022 167232  
 3221 011666 005300  
 3222 011670 001363

TST021: MOV# #21, @#TSTNUM  
 MOV #45, R2 ; SET RETURN PC FROM TIMER  
 JSR PC, .REWIND ; REWIND SLAVE  
 BVS 99% ; BRANCH IF ERROR ON REWIND  
 CLR DELTIM ; CLEAR VARIABLE DELAY TIME  
 MOV #17, RO ; SET # OF RECORDS TO WRITE  
 JSR PC, WRITE ; WRITE 17 RECORDS  
 INC (R5) ; SET 'GO' BIT  
 JSR PC, WAITRDY ; WAIT FOR READY  
 BVS 99%  
 JSR PC, DELAYV ; DELAY BEFORE WRITING NEXT REC  
 ADD #18, DELTIM ; SET NEXT DELAY TIME  
 DEC RO ; DECREMENT RECORDS WRITTEN COUNT  
 BNE 15

3224 011672 012700 000021  
 3225 011676 004767 173556  
 3226 011702 005215  
 3227 011704 004767 173326  
 3228 011710 102477  
 3229 011712 005300  
 3230 011714 001370

MOV #17, RO ; SET # OF RECS TO REVERSE READ  
 JSR PC, REVRD ; REVERSE READ 17 RECORDS  
 INC (R5) ; SET 'GO' BIT  
 JSR PC, WAITRDY ; WAIT FOR READY  
 BVS 99%  
 DEC RO DECREMENT RECORD COUNT  
 BNE 25

3232 011716 012700 000020  
 3233 011722 012701 001060  
 3234 011726 004767 173510  
 3235 011732 005215

MOV #16, RO ; SET # OF RECORDS TO READ  
 MOV #GAPTBL, R1 ; SET PTR TO GAP TABLE FOR TEST  
 JSR PC, READ ; READ A RECORD  
 INC (R5) ; SET 'GO' BIT

3237 011734 004767 173276  
 3238 011740 102463  
 3239 011742 004767 173474  
 3240 011746 004767 172406  
 3241 011752 005215

JSR PC, WAITRDY ; WAIT FOR READY  
 BVS 99%  
 JSR PC, READ ; READ NEXT RECORD  
 JSR PC, TIMON ; TURN TIMER ON  
 INC (R5) ; SET 'GO' BIT

3243 011754 005765 000006  
 3244 011760 001002  
 3245 011762 000163 004450

TST FC(R5) ; WAIT FOR FRAME COUNT > 0  
 BNE 55  
 JMP TIMER(R3) ; GO TO TIMER & RETURN VIA R2

```

3246
3247 011766 004767 173244      55      JSR      PC, WAITRDY      ; WAIT FOR READY
3248 011772 102446              BVS      99$
3249 011774 010421              MOV      R4, (R1)+      ; STORE TIME IN GAPTBL
3250 011776 005300              DEC      R0              ; DECREMENT # OF RECORDS READ
3251 012000 001355              BNE      3$
3252
3253 012002 105037 001124      CLR      @#GAP          ; SET GAP # 0
3254 012006 012700 000020      MOV      #16, R0
3255 012012 012701 001060      MOV      #GAPTBL, R1
3256
3257 012016 012104              MOV      (R1)+, R4      ; GET GAP TICK COUNT
3258 012020 004767 172572      JSR      PC, GAPOK      ; CHECK TIME
3259 012024 105237 001124      INCB     @#GAP          ; INCREMENT GAP #
3260 012030 122737 000020 001124  CMPB     #16, @#GAP      ; BRANCH IF ALL GAPS NOT CHECKED
3261 012036 001367              BNE      6$
3262
3263 012040 012700 000020      MOV      #16, R0          ; SETUP TO AVERAGE GAP SIZES
3264 012044 012701 001060      MOV      #GAPTBL, R1     ; SET PTR TO TABLE
3265 012050 005002              CLR      R2              ; CLEAR 'SUM' REGISTERS
3266 012052 005003              CLR      R3
3267 012054 062102              ADD      (R1)+, R2      ; ADD ALL GAP SIZES TOGETHER
3268 012056 005503              ADC      R3
3269 012060 005300              DEC      R0
3270 012062 001374              BNE      7$
3271 012064 012700 000004      MOV      #4, R0          ; NOW DIVIDE BY 16.
3272 012070 006203              ASR      R3              ; BY SHIFTING 4 PLACES RIGHT
3273 012072 006002              ROR      R2
3274 012074 005300              DEC      R0
3275 012076 001374              BNE      8$
3276 012100 010204              MOV      R2, R4          ; MOVE AVERAGED TIMES TO R4
3277 012102 004767 172400      JSR      PC, TIMOK      ; CHECK AVERAGED TIMES
3278 012106 000401              BR       100$
3279
3280 012110 104400              99$.    HLT
3281 012112 104000              100$.   SCOPE
3282
3283      ; TEST 022-DUMMY TEST
3284      ; THIS TEST MEASURES NOTHING
3285 012114 112737 000022 001126  TST022  MOV      #22, @#TSTNUM
3286
3287      ; TEST 023-DATA TIME (200BP1)
3288      ; THIS TEST MEASURES TIME FROM 'FC REG' CHANGES TO 'RDY'=1
3289 012122 112737 000023 001126  TST023  MOV      #23, @#TSTNUM
3290 012130 012702 012202      MOV      #3$, R2          ; SET RETURN PC FROM TIMER
3291 012134 004767 173202      JSR      PC, REWIND      ; REWIND SLAVE
3292 012140 102437              BVS      99$            ; BRANCH IF ERROR ON REWIND
3293 012142 004367 173412      JSR      R3, TMCMD      ; WRITE 800 WORD RECORD
3294 012146 016420              .WORD   WTBUF          ; SET WRITE BUFFER ADDRESS
3295 012150 176340              .WORD   -800           ; WORD COUNT
3296 012152 174700              .WORD   -1600          ; FRAME COUNT
3297 012154 000060              .WORD   WFWD           ; WRITE COMMAND
3298 012156 005215              INC      (R5)           ; SET 'GO' BIT
3299
3300 012160 022710 174700      1$      CMP      #-1600, (R0)    ; WAIT FOR FRAME COUNT TO CHANGE
3301 012164 001004              BNE      2$

```



3302	012166	032711	040000		BIT	#ERR, (R1)		; MONITOR ERROR BIT
3303	012172	001022			BNE	99\$		
3304	012174	000771			BR	1\$		
3305								
3306	012176			2\$				
3307	012176	004767	172156		JSR	PC, TIMON		; TURN TIMER ON
3308	012202	105711		3\$	TSTB	(R1)		; WAIT FOR READY TO SET
3309	012204	100402			BMI	4\$		
3310	012206	000163	004450		JMP	TIMER(R3)		; GO TO TIMER & RETURN VIA R2
3311	012212	012700	000003	4\$	MOV	#3, R0		; SET TO DIVIDE BY 8
3312	012216	006204		5\$	ASR	R4		; BY SHIFTING RIGHT 3 PLACES
3313	012220	005300			DEC	R0		
3314	012222	001375			BNE	5\$		
3315	012224	004767	173006		JSR	PC, WAITRDY		
3316	012230	102403			BVS	99\$		
3317	012232	004767	172250		JSR	PC, TIMOK		; CHECK TIME
3318	012236	000401			BR	100\$		
3319								
3320	012240	104400		99\$	HLT			
3321	012242	104000		100\$	SCOPE			
3322								
3323								
3324	012244	112737	000024	001126	TEST 024-DATA TIME (556BP1)			
3325	012252	012702	012332		TST024 MOV	#24, #TSTNUM		
3326	012256	004767	173060		MOV	#3\$ R2		; SET RETURN PC FROM TIMER
3327	012262	102442			JSR	PC, REWIND		; REWIND SLAVE
3328	012264	052765	000700	000032	BVS	99\$		; BRANCH IF ERROR ON REWIND
3329	012272	004367	173262		BIS	#BP1556+NORM11, TC(R5)		; LOAD TAPE CONTROL REGISTER
3330	012276	016420			JSR	R3, TMCMD		; WRITE 2224. WORD RECORD
3331	012300	173520			WORD	WTBUF		
3332	012302	167240			WORD	-2224.		
3333	012304	000060			WORD	-4448		
3334	012306	005215			WORD	WFWD		
3335					INC	(R5)		; SET 'GO' BIT
3336	012310	022710	167240	1\$	CMP	#-4448, (R0)		; BRANCH WHEN WRITING BEGINS
3337	012314	001004			BNE	2\$		
3338	012316	032711	040000		BIT	#ERR, (R1)		; MONITOR ERROR BIT
3339	012322	001022			BNE	99\$		
3340	012324	000771			BR	1\$		
3341								
3342	012326			2\$				
3343	012326	004767	172026		JSR	PC, TIMON		; TURN TIMER ON
3344	012332	105711		3\$	TSTB	(R1)		; BRANCH WHEN READY SETS
3345	012334	100402			BMI	4\$		
3346	012336	000163	004450		JMP	TIMER(R3)		; GO TO TIMER & RETURN VIA R2
3347								
3348	012342	012700	000003	4\$	MOV	#3, R0		; SET SHIFT COUNT
3349	012346	006204		5\$	ASR	R4		
3350	012350	005300			DEC	R0		
3351	012352	001375			BNE	5\$		
3352	012354	004767	172656		JSR	PC, WAITRDY		
3353	012360	102403			BVS	99\$		
3354	012362	004767	172120		JSR	PC, TIMOK		; CHECK TIME
3355	012366	000401			BR	100\$		
3356								
3357	012370	104400		99\$	HLT			

```

3358 012372 104000          100$ SCOPE
3359
3360          , TEST 025-DATA TIME (800BPI)
3361 012374 112737 000025 001126 TST025 MOVB #025, @#TSTNUM
3362 012402 012702 012462          MOV #3$, R2          ; SET RETURN PC FROM TIMER
3363 012406 004767 172730          JSR PC, .REWIND      ; REWIND SLAVE
3364 012412 102442          BVS 99$             ; BRANCH IF ERROR ON REWIND
3365 012414 052765 001300 000032 BIS #BP1800+NORM11, TC(R5) ; SET 800 BPI
3366 012422 004367 173132          JSR R3, TMCMD       ; WRITE 3200 WORD RECORD
3367 012426 016420          WORD WTBUF
3368 012430 171600          WORD -3200.
3369 012432 163400          WORD -6400
3370 012434 000060          WORD WFWD
3371 012436 005215          INC (R5)           ; SET 'GO' BIT
3372
3373 012440 022710 163400          1$ CMP #-6400, (R0)    ; WAIT FOR WRITING TO START
3374 012444 001004          BNE 2$
3375 012446 032711 040000          BIT #ERR, (P1)     ; MONITOR ERROR BIT
3376 012452 001022          BNE 99$
3377 012454 000771          BR 1$
3378
3379          2$
3380 012456 004767 171676          JSR PC, TIMON      ; TURN TIMER ON
3381 012462 105711          3$ TSTB (R1)         ; BRANCH WHEN READY SETS
3382 012464 100402          BMI 4$
3383 012466 000163 004450          JMP TIMER(R3)     ; GO TO TIMER & RETURN VIA R2
3384
3385 012472 012700 000003          4$ MOV #3, R0        ; SET SHIFT COUNT
3386 012476 006204          5$ ASR R4
3387 012500 005300          DEC R0
3388 012502 001375          BNE 5$
3389 012504 004767 172526          JSR PC, WAITRDY
3390 012510 102403          BVS 99$
3391 012512 004767 171770          JSR PC, TIMOK     ; CHECK TIME
3392 012516 000401          BR 100$
3393
3394 012520 104400          99$ HLT
3395 012522 104000          100$ SCOPE
3396
3397          , TEST 026-DATA TIME (1600BPI)
3398 012524 112737 000026 001126 TST026 MOVB #026, @#TSTNUM
3399 012532 105737 001133          TSTB @#NRZFLG     ; BRANCH IF DRIVE 'NRZ ONLY'
3400 012536 001046          BNE TST027
3401 012540 012702 012620          MOV #3$, R2          ; SET RETURN PC FROM TIMER
3402 012544 004767 172572          JSR PC, .REWIND      ; REWIND SLAVE
3403 012550 102437          BVS 99$             ; BRANCH IF ERROR ON REWIND
3404 012552 052765 002300 000032 BIS #PE1600+NORM11, TC(R5) ; SET 1600 BPI
3405 012560 004367 172774          JSR R3, TMCMD       ; WRITE 3200 WORD RECORD
3406 012564 016420          WORD WTBUF
3407 012566 171600          WORD -3200.
3408 012570 163400          WORD -6400
3409 012572 000060          WORD WFWD
3410 012574 005215          INC (R5)           ; SET 'GO' BIT
3411
3412 012576 022710 163400          1$ CMP #-6400, (R0)    ; BRANCH WHEN WRITING STARTS
3413 012602 001004          BNE 2$
    
```

3414	012604	032711	040000		BIT	#ERR, (R1)		. MONITOR ERROR BIT
3415	012610	001017			BNE	99%		
3416	012612	000771			BR	15		
3417								
3418	012614			25				
3419	012614	004767	171540		JSR	PC, TIMON		. TURN TIMER ON
3420	012620	105711		35	TSTB	(R1)		. BRANCH WHEN READY SETS
3421	012622	100402			BMI	45		
3422	012624	000163	004450		JMP	TIMER(R3)		. GO TO TIMER & RETURN VIA R2
3423								
3424	012630	006204		45	ASR	R4		. DIVIDE TIME BY 4
3425	012632	006204			ASR	R4		
3426	012634	004767	172376		JSR	PC, WAITRDY		
3427	012640	102403			BVS	99%		
3428	012642	004767	171640		JSR	PC, TIMOK		. CHECK TIME
3429	012646	000401			BR	100%		
3430								
3431	012650	104400		99%	HLT			
3432	012652	104000		100%	SCOPE			
3433								
3434								. TEST 027-ERASE
3435								. THIS TST MEASURES TIME FROM 'GO'=1 TO 'RDY'=1
3436	012654	112737	000027	001126	TST027	MOVB #27, @TSTNUM		
3437	012662	012702	012710		MOV	#15, R2		. SET RETURN PC FROM TIMER
3438	012666	004337	005560		JSR	R3, @TMCMD		
3439	012672	000000			WORD	0		
3440	012674	000000			WORD	0		
3441	012676	000000			WORD	0		
3442	012700	000024			WORD	ERASE		
3443	012702	004767	171452		JSR	PC, TIMON		. TURN TIMER ON
3444	012706	005215			INC	(R5)		. SET 'GO' BIT
3445								
3446	012710	105711		15	TSTB	(R1)		. BRANCH WHEN READY SETS
3447	012712	100402			BMI	25		
3448	012714	000163	004450		JMP	TIMER(R3)		. GO TO TIMER & RETURN VIA R2
3449								
3450	012720	004767	172312	25	JSR	PC, WAITRDY		
3451	012724	102403			BVS	99%		
3452	012726	004767	171554		JSR	PC, TIMOK		
3453	012732	000401			BR	100%		
3454								
3455	012734	104400		99%	HLT			
3456	012736	104000		100%	SCOPE			
3457								
3458								. TEST-030 TAPE MARK
3459								. THIS TEST MEASURES TIME FROM 'GO'=1 TO 'RDY'=1
3460	012740	112737	000030	001126	TST030	MOVB #30, @TSTNUM		
3461	012746	012702	013010		MOV	#15, R2		. SET RETURN PC FROM TIMER
3462	012752	004767	172446		JSR	PC, WRITE		. WRITE A RECORD
3463	012756	005215			INC	(R5)		. SET 'GO' BIT
3464	012760	004767	172252		JSR	PC, WAITRDY		
3465	012764	102423			BVS	99%		
3466	012766	004337	005560		JSR	R3, @TMCMD		
3467	012772	000000			WORD	0		
3468	012774	000000			WORD	0		
3469	012776	000000			WORD	0		

3470	013000	000026			. WORD	WFMK	
3471	013002	004767	171352		JSR	PC, TIMON	. TURN TIMER ON
3472	013006	005215			INC	(R5)	. SET 'GO' BIT
3473							
3474	013010	105711		15	TSTB	(R1)	. BRANCH WHEN READY SETS
3475	013012	100402			BMI	25	
3476	013014	000163	004450		JMP	TIMER(R3)	. GO TO TIMER & RETURN VIA R2
3477							
3478	013020	004767	172212	25	JSR	PC, WAITRDY	
3479	013024	102403			BVS	995	
3480	013026	004767	171454		JSR	PC, TIMOK	
3481	013032	000401			BR	1005	
3482							
3483	013034	104400		995	HLT		
3484	013036			1005			
3485	013036	004767	172300		JSR	PC, REWIND	. REWIND SLAVE
3486	013042	102774			BVS	995	. BRANCH IF ERROR ON REWIND
3487	013044	104000			SCOPE		
3488							

```

3489 013046 012700 000012 FINISH MOV #10 ,R0 , SET LINE FEED COUNT
3490 013052 000004 001400 1$ TYPE, CRLF
3491 013056 005300 DEC R0
3492 013060 001374 BNE 1$
3493 013062 032777 000100 165710 BIT #SW06, @SWR
3494 013070 001410 BEQ 2$
3495 013072 113700 001010 MOVB @#DRVNUM, R0
3496 013076 113701 001011 MOVB @#SLVNUM, R1
3497 013102 113702 001012 MOVB @#SLVPTR, R2
3498 013106 000137 007440 JMP @#TYPHDR
3499 013112 105237 001011 2$ INCB @#SLVNUM , SET NEXT SLAVE #
3500 013116 005237 001012 INC @#SLVPTR ; AND ITS POINTER
3501 013122 122737 000010 001011 CMPB #8, @#SLVNUM ; BRANCH IF LAST SLAVE (7)
3502 013130 001402 BEQ 3$
3503 013132 000137 007206 JMP @#BEGIN ; BEGIN TEST ON NEXT SLAVE
3504 013136 105037 001011 3$ CLRB @#SLVNUM , SET SLAVE #0
3505 013142 105237 001010 INCB @#DRVNUM ; AND INCREMENT DRIVE #
3506 013146 122737 000010 001010 CMPB #8, @#DRVNUM ; AND CHECK IF LAST DRIVE
3507 013154 001402 BEQ END
3508 013156 000137 007206 JMP @#BEGIN
3509
3510 013162 END
3511
3512 ***** MODIFIED 8-11-77 *****
3513 ACT11 END OF PASS
3514
3515 +
3516 +
3517
3518
3519 013162 005767 165576 TST AUTOM ; AUTOMATIC MODE?
3520 013166 001015 BNE 2$ ; BRANCH - IF YES
3521 013170 105737 001131 TSTB @#UNTFND ; BRANCH IF A UNIT WAS FOUND
3522 013174 001004 BNE 1$
3523 013176 000004 014567 TYPE, E UNIT
3524 013202 000137 005730 JMP @#INIT
3525 013206 000000 1$ HALT
3526 013210 004767 166560 JSR PC, CKSWR ; CHECK FOR CNTL G
3527 013214 000005 RESET
3528 013216 000137 005730 JMP @#INIT ; RESTART
3529
3530 013222 000004 014014 2$ TYPE, MSGA ; PRINT END OF PASS
3531 013226 005267 165552 INC PCNTR ; BUMP PASS COUNTER
3532 013232 013702 001004 MOV @#PCNTR, R2 ; GET END OF PASS COUNT
3533 013236 004767 167372 JSR PC, TYPOCT ; AND TYPE IT
3534 013242 005767 165534 TST PAFLG ; PASS INDICATOR SET?
3535 013246 001002 BNE 3$ ; BRANCH - IF YES
3536 013250 005267 165526 INC PAFLG ; SET PASS INDICATOR
3537 013254 013704 000042 3$ MOV @#42, R4 ; CONTENTS OF 42 TO R4
3538 013260 001405 BEQ HERE ; BRANCH IF NOT AUTO MODE
3539 013262 000005 RESET ; CLEAR THE WORLD
3540 013264 004714 SENDAD JSR PC, (R4) ; RETURN TO MONITOR
3541 013266 000240 NOP
3542 013270 000240 NOP
3543 013272 000240 NOP
3544 013274 HERE
  
```

3545	013274	005767	165464	TST	AUTOM	;CHECK FOR AUTOM MODE
3546	013300	001010		BNE	1\$	;BRANCH - IF YES
3547	013302	032777	094000 165470	BIT	#4000, @SWR	;SEE IF HALT ON PASS
3548	013310	001004		BNE	1\$	;BRANCH - IF NOT
3549	013312	000000		HALT		
3550	013314	004767	166454	JSR	PC, CKSWR	;CHECK FOR CONTROL G
3551	013320	000005		RESET		;CLEAR THE WORLD
3552	013322	000137	005730	JMP:	@#INIT	;RESTART

3553  
3554  
3555  
3556

\*\*\*\*\*

```

3557 ,SKEW TAPE TIMING TESTS
3558 ,THE FOLLOWING TESTS REQUIRE A SPECIALLY WRITTEN 800 BPI SKEW TAPE
3559 013326 012737 013334 001006 SKEWTST MOV #TST031,@#SCPADR ;SET SCOPE POINTER
3560
3561 ,TEST 031- SKEW TAPE SPEED TEST-FORWARD
3562 ,THIS TEST READS 32" OF TAPE (26400.-800. = 25600. FRAMES), THEN
3563 ,DIVIDES TIME BY 32. TO GET TIME TO READ 1" (800. FRAMES) OF TAPE
3564 013334 112737 000031 001126 TST031 MOVB #31,@#TSTNUM
3565 013342 012702 013420 MOV #25,R2 ;SET RETURN PC FROM TIMER
3566 013346 004767 171770 JSR PC,REWIND ;REWIND SLAVE
3567 013352 102441 BVS 99% ;BRANCH IF ERROR ON REWIND
3568 013354 052765 001300 000032 BIS #BP1800+NORM11,TC(R5) ;SET 800 BPI
3569 013362 052765 000010 000010 BIS #BA1,(S2(R5)) ;INHIBIT BUS ADDRESS INCREMENT
3570 013370 004337 005560 JSR R3,@#TMCMD ;READ 32" OF TAPE-FORWARD
3571 013374 016420 WORD RDBUF
3572 013376 177777 WORD -1.
3573 013400 063440 10% WORD 26400 ;FRAME COUNT
3574 013402 000070 WORD RDFWD
3575 013404 005215 INC (R5) ;SET 'GO' BIT
3576
3577 013406 022710 001440 1% CMP #800,(R0) ;WAIT FOR FIRST 800 FRAMES
3578 013412 101375 BHI 1% ;TO BE READ
3579
3580 013414 004767 170740 JSR PC,TIMON ;TURN TIMER ON
3581 013420 023710 013400 2% CMP @#10%,(R0) ;WAIT FOR READING TO FINISH
3582 013424 103402 BLO 3%
3583 013426 000163 004450 JMP TIMER(R3) ;GO TO TIMER & RETURN VIA R2
3584
3585 013432 012700 000005 3% MOV #5,R0 ;DIVIDE TIME BY 32
3586 013436 006204 4% ASR P4
3587 013440 005300 DEC R0
3588 013442 001375 BNE 4%
3589 013444 004767 171530 JSR PC,RHINIT ;INIT DRIVE
3590 013450 004767 171032 JSR PC,TIMOK ;CHECK TIME
3591 013454 000401 BR 100%
3592
3593 013456 104400 99% HLT
3594 013460 104000 100% SCOPE
3595
3596 ,TEST 032-SKEW TAPE SPEED TEST-REVERSE
3597 ,THIS TEST READS FORWARD 40" (32000. FRAMES) OF TAPE, THEN READS REVERSE
3598 ,32" (26400.-800. = 25600. FRAMES) OF TAPE THE TIME IS THEN DIVIDED BY
3599 ,32 TO GET TIME TO READ 1" (800 FRAMES) OF TAPE
3600 013462 112737 000032 001126 TST032 MOVB #32,@#TSTNUM
3601 013470 012702 013616 MOV #35,R2 ;SET RETURN PC FROM TIMER
3602 013474 004767 171642 JSR PC,REWIND ;REWIND SLAVE
3603 013500 102465 BVS 99% ;BRANCH IF ERROR ON REWIND
3604 013502 052765 001300 000032 BIS #BP1800+NORM11,TC(R5)
3605 013510 052765 000010 000010 BIS #BA1,(S2(R5))
3606 013516 004337 005560 JSR R3,@#TMCMD ;READ FORWARD 32000 FRAMES
3607 013522 016420 WORD RDBUF
3608 013524 177777 WORD -1.
3609 013526 076400 10% WORD 32000 ;FRAME COUNT
3610 013530 000070 WORD RDFWD ;READ FORWARD
3611 013532 005215 INC (R5) ;SET 'GO' BIT
3612
  
```





3651  
3652  
3653  
3654  
3655  
3656  
3657  
3658  
3659  
3660  
3661  
3662  
3663  
3664  
3665  
3666  
3667  
3668  
3669  
3670  
3671  
3672  
3673  
3674  
3675  
3676  
3677

013672 005067 165066  
013676 005737 000042  
013702 001417  
013704 005267 165054  
013710 023737 000042 000046  
013716 001403  
013720 105267 165043  
013724 000416  
013726 105267 165034 1\$  
013732 052777 104100 165040  
013740 000410  
013742 105737 000041 2\$  
013746 001003  
013750 105267 165014  
013754 000402  
013756 105267 165007 3\$  
013762 000207 5\$

```
CKMODE CLR AUTOM ;INIT AUTO MODE
TST @#42 ;AUTOMATIC MODE?
BEQ 2$ ;BRANCH - IF NOT
INC AUTOM ;SET AUTO MODE INDICATOR
CMP @#42,@#46 ;ACT11 MODE?
BEQ 1$ ;BRANCH - IF YES
INCB XXDPM ;INDICATE XXDP AUTO MODE
BR 5$ ;AND EXIT
INCB ACT11M ;INDICATE ACT11 AUTO MODE
BIS #104100,@SWR ;CON: CYCLE,HALT ON ERROR,INHIBIT SUBTEST
BR 5$ ;AND EXIT
TSTB @#41 ;MAN: MODE VIA ACT11/PAPER TAPE?
BNE 3$ ;BRANCH - IF NOT
INCB ADUMPM ;INDICATE MAN MODE VIA ACT11/PAPER TAPE
BR 5$ ;AND EXIT
INCB XDUMPM ;INDICATE MANUAL MODE VIA XXDP
RTS PC ;RETURN
```

\*\*\*\*\*INSERTED 11-8-77 \*\*\*\*\*

CHECK FOR DUMP MODE OR AUTOMATIC MODE

+  
+

\*\*\*\*\*











```
3750 014376 005015 047105 020104 M EOT .ASCIZ <CR><LF>'END OF TAPE'<CR><LF>
3751 014404 043117 052040 050101
3752 014412 006505 000012
3753
3754 , ERROR MESSAGES
3755 014416 005015 051124 050101 E TRP4 .ASCIZ <CR><LF>'TRAPPED TO 4'
3756 014424 042520 020104 047524
3757 014432 032040 000
3758 014435 116 020117 047503 E.NCON .ASCIZ 'NO CONTROLLER AT ADDRESS SPECIFIED'<CR><LF>
3759 014442 052116 047522 046114
3760 014450 051105 040440 020124
3761 014456 042101 051104 051505
3762 014464 020123 050123 041505
3763 014472 043111 042511 006504
3764 014500 000012
3765 014502 046524 031060 042040 E NDRV. .ASCIZ 'TMO2 DRIVE '
3766 014510 044522 042526 000040
3767 014516 051104 053111 020105 E NSLV .ASCIZ 'DRIVE '
```

```
3768 014524 020060 046123 053101 E.DRV: .ASCII 'O SLAVE '  
3769 014532 020105  
3770 014534 020060 047516 020124 E.NAVA ASCIIZ 'O NOT AVAILABLE FOR TEST'<CR><LF>  
3771 014542 053101 044501 040514  
3772 014550 046102 020105 047506  
3773 014556 020122 042524 052123  
3774 014564 005015 000  
3775 014567 116 020117 046524 E.UNIT ASCIIZ 'NO TMO2/TE16 UNIT FOUND TO TEST'<CR><LF>  
3776 014574 031060 052057 030505  
3777 014602 020066 047125 052111  
3778 014610 043040 052517 042116  
3779 014616 052040 020117 042524  
3780 014624 052123 005015 000  
3781 014631 123 043117 020124 E.SFT ASCIIZ 'SOFT ERROR (DATA)'<CR><LF>  
3782 014636 051105 047522 020122  
3783 014644 042050 052101 024501  
3784 014652 005015 000  
3785 014655 124 051505 020124 E.HDR .ASCIIZ 'TEST # '  
3786 014662 020043 000  
3787 014665 040 042504 044526 E.HDR1 ASCII 'DEVICE ERROR'<CR><LF>  
3788 014672 042503 042440 051122  
3789 014700 051117 005015  
3790 014704 051503 004461 041527 ASCIIZ 'CS1'<HT>'WC'<HT>'BA'<HT>'FC'<HT>'CS2'<HT>'DS'<HT>'ER'<HT>'TC'<CR><LF>  
3791 014712 041011 004501 041506  
3792 014720 041411 031123 042011  
3793 014726 004523 051105 052011  
3794 014734 006503 000012  
3795 014740 047440 052125 047440 E.HDR2 ASCIIZ 'OUT OF RANGE ERPOP'<CR><LF>  
3796 014746 020106 040522 043516  
3797 014754 020105 051105 047522  
3798 014762 006522 000012  
3799 014766 005015 044524 042515 E.TIMOV ASCIIZ <CR><LF>'TIMER OVERFLOWED'<CR><LF>  
3800 014774 020122 053117 051105  
3801 015002 046106 053517 042105  
3802 015010 005015 000  
3803 015013 015 052012 046511 E.TIMEX ASCIIZ <CR><LF>'TIME EXPIED WAITING FOR RDY'<CR><LF>  
3804 015020 020105 054105 044520  
3805 015026 042522 020104 040527  
3806 015034 052111 047111 020107  
3807 015042 047506 020122 042122  
3808 015050 006531 000012  
3809 015054 043440 050101 021440 E.GAP ASCIIZ 'GAP # '  
3810 015062 000040  
3811  
3812  
3813 015064 025052 025052 025052 .TIME DOCUMENT LINES  
3814 015072 025052 025052 025052 L.HDR1 ASCIIZ '*****  
3815 015100 025052 025052 025052  
3816 015106 025052 025052 025052  
3817 015114 025052 025052 025052  
3818 015122 025052 025052 025052  
3819 015130 025052 025052 025052  
3820 015136 025052 025052 025052  
3821 015144 025052 025052 025052  
3822 015152 025052 025052 025052  
3823 015160 025052 025052 025052
```



3824	015166	025052	025052	025052			
3825	015174	005015	000				
3826	015177	052	052040	030115	L HDR2	ASCII	'* TMO2 DRIVE FUNCTION TIMES- DRIVE #
3827	015204	020062	051104	053111			
3828	015212	020105	052506	041516			
3829	015220	044524	047117	052040			
3830	015226	046511	051505	020055			
3831	015234	051104	053111	020105			
3832	015242	020043					
3833	015244	020060	046123	053101	L DRV	ASCII	'0 SLAVE #
3834	015252	020105	020043				
3835	015256	020060	040		L SLV	ASCII	'0 '
3836	015261	071	041440	040510	L CHAN	ASCIZ	'9 CHAN SER # '
3837	015266	027116	051440	051105			
3838	015274	021440	000040				
3839	015300	006440	025012	005015	L HDR3	ASCII	' '<CR><LF>'*<CR><LF>
3840	015306	020052	052506	041516		ASCIZ	'* FUNCTION'<HT><HT>'TIME(SPECIFICATION) <HT>'TIME(ACTUAL)'<CR><LF>
3841	015314	044524	047117	004411			
3842	015322	044524	042515	051450			
3843	015330	042520	044503	044506			
3844	015336	040503	044524	047117			
3845	015344	004451	044524	042515			
3846	015352	040450	052103	040525			
3847	015360	024514	005015	000			
3848							
3849	015365	122	047101	042507	L RNG	ASCIZ	'RANGE=<'
3850	015372	036075	000				
3851	015375	101	052103	040525	L ACT	ASCIZ	'ACTUAL='
3852	015402	036514	000				
3853							
3854							
3855	015405	052	053440	044522	A T001	ASCIZ	'* WRITE FROM BOT'<HT>
3856	015412	042524	043040	047522			
3857	015420	020115	047502	004524			
3858	015426	000					
3859	015427	052	053440	044522	A T002	ASCIZ	'* WRITE START'<HT><HT>
3860	015434	042524	051440	040524			
3861	015442	052122	004411	000			
3862	015447	052	053440	044522	A T003	ASCIZ	'* WRITE SHUTDOWN'<HT>
3863	015454	042524	051440	052510			
3864	015462	042124	053517	004516			
3865	015470	000					
3866	015471	052	053440	044522	A T004	ASCIZ	'* WRITE SETTLEDOWN'<HT>
3867	015476	042524	051440	052105			
3868	015504	046124	042105	053517			
3869	015512	004516	000				
3870	015515	052	051040	040505	A T005	ASCIZ	'* READ FROM BOT'<HT><HT>
3871	015522	020104	051106	046517			
3872	015530	041040	052117	004411			
3873	015536	000					
3874	015537	052	051040	040505	A T006	ASCIZ	'* READ START'<HT><HT>
3875	015544	020104	052123	051101			
3876	015552	004524	000011				
3877	015556	020052	042522	042101	A T007	ASCIZ	'* READ SHUTDOWN'<HT><HT>
3878	015564	051440	052510	042124			
3879	015572	053517	004516	000011			

3880	015600	020052	042522	042101	A. T010	ASCIZ	'* READ SETTLEDOWN'<HT>
3881	015606	051440	052105	046124			
3882	015614	042105	053517	004516			
3883	015622	000					
3884	015623	052	051040	040505	A T011	ASCIZ	'* READ REV START'<HT>
3885	015630	020104	042522	020126			
3886	015636	052123	051101	004524			
3887	015644	000					
3888	015645	052	051040	040505	A. T012.	ASCIZ	'* READ REV SHUTDOWN'<HT>
3889	015652	020104	042522	020126			
3890	015660	044123	052125	047504			
3891	015666	047127	000011				
3892	015672	020052	042522	042101	A T013	ASCIZ	'* READ REV SETTLEDOWN'<HT>
3893	015700	051040	053105	051440			
3894	015706	052105	046124	042105			
3895	015714	053517	004516	000			
3896	015721	052	052040	051125	A T014	ASCIZ	'* TURN AROUND DELAY F-R'<HT>
3897	015726	020116	051101	052517			
3898	015734	042116	042040	046105			
3899	015742	054501	043040	051055			
3900	015750	000011					
3901	015752	020052	052524	047122	A T015	ASCIZ	'* TURN AROUND DELAY R-F'<HT>
3902	015760	040440	047522	047125			
3903	015766	020104	042504	040514			
3904	015774	020131	026522	004506			
3905	016002	000					
3906	016003	052	043440	050101	A T016	ASCIZ	'* GAP SIZE-STOP HALF'<HT>
3907	016010	051440	055111	026505			
3908	016016	052123	050117	044040			
3909	016024	046101	004506	000			
3910	016031	052	043440	050101	A T017	ASCIZ	'* GAP SIZE-START HALF'<HT>
3911	016036	051440	055111	026505			
3912	016044	052123	051101	020124			
3913	016052	040510	043114	000011			
3914	016060	020052	040507	020120	A T020	ASCIZ	'* GAP SIZE-INTERRECORD'<HT>
3915	016066	044523	042532	044455			
3916	016074	052116	051105	042522			
3917	016102	047503	042122	000011			
3918	016110	020052	040507	020120	A T021	ASCIZ	'* GAP CONSISTANCY'<HT>
3919	016116	047503	051516	051511			
3920	016124	040524	041516	004531			
3921	016132	000					
3922	016133	052	042040	052101	A T023	ASCIZ	'* DATA TIME-200BPI'<HT>
3923	016140	020101	044524	042515			
3924	016146	031055	030060	050102			
3925	016154	004511	000				
3926	016157	052	042040	052101	A T024	ASCIZ	'* DATA TIME-556BPI'<HT>
3927	016164	020101	044524	042515			
3928	016172	032455	033065	050102			
3929	016200	004511	000				
3930	016203	052	042040	052101	A T025	ASCIZ	'* DATA TIME-800BPI'<HT>
3931	016210	020101	044524	042515			
3932	016216	034055	030060	050102			
3933	016224	004511	000				
3934	016227	052	042040	052101	A T026	ASCIZ	'* DATA TIME-1600BPI'<HT>
3935	016234	020101	044524	042515			

DZTUG-B TMO2/TE16 DRIVE FUNCTION TIMER MACY11 30(1046) 02-DEC-77 09 48 PAGE 99  
CZTUGC P11 02-DEC-77 09 46 PROGRAM MESSAGES

SEQ 0099

3936 016242 030455 030066 041060  
3937 016250 044520 000011

3938	016254	020052	051105	051501	A T027	ASCIZ	'* ERASE GAP TIME'<HT>
3939	016262	020105	040507	020120			
3940	016270	044524	042515	000011			
3941	016276	020052	051127	052111	A. T030	ASCIZ	'* WRITE FILE MARK'<HT>
3942	016304	020105	044506	042514			
3943	016312	046440	051101	004513			
3944	016320	000					
3945	016321	052	052040	050101	A T031	ASCIZ	'* TAPE SPEED-FWD'<HT>
3946	016326	020105	050123	042505			
3947	016334	026504	053506	004504			
3948	016342	000					
3949	016343	052	052040	050101	A T032	ASCIZ	'* TAPE SPEED-REV'<HT>
3950	016350	020105	050123	042505			
3951	016356	026504	042522	004526			
3952	016364	000					
3953							
3954	016365	015	057012	000107	L CNTG	ASCIZ	<CR><LF>' G'
3955	016372	005015	053523	036522	L. SWR	ASCIZ	<CR><LF>'SWR='
3956	016400	000					
3957	016401	040	047040	053505	L NEW	ASCIZ	' NEW= '
3958	016406	020075	000				
3959	016411	015	037412	005015	L QUEST	ASCIZ	<CR><LF>'?'<CR><LF>
3960	016416	000					
3961		016420					
3962		016420					
3963		016420					
3964	016420	000200					
3965		000001					

EVEN  
 ROBUF=  
 WTBUF=  
 BLKW 128  
 END







LPS =	177514	1185#							
L ACT	015375	1751	1779	3851#					
L CHAN	015261	2726*	2729*	3836#					
L CNTG	016365	1498	3954#						
L DRV	015244	2724*	3833#						
L HDR1	015064	2723	3813#						
L HDR2	015177	2730	3826#						
L HDR3	015300	2732	3839#						
L NEW	016401	1502	3957#						
L QUES	016411	1522	3959#						
L RNG	015365	1744	1772	3849#					
L SLV	015256	2725*	3835#						
L SWR	016372	1499	3955#						
MCPE =	020000	1228#							
MDPE =	000400	1245#							
MEDIUM	000041	1037#							
MMVEC =	000250	1178#							
MOL =	010000	1265#							
MR =	000024	1201#	2037	2039	2044	2058	2073		
MSGA	014014	3530	3701#						
MSGB	014036	2418	3685	3704#					
MXF =	001000	1246#							
M EOT	014376	2266	3750#						
M NAM	014060	2265	2459	3712#					
NAMPTR	001676	1455#	1848						
NED =	010000	1249#							
NEF =	004000	1280#							
NEM =	004000	1248#							
NOP =	000000	1209#							
NORM11=	000300	1294#	2251	3328	3365	3404	3568	3604	3618
NRZFLG	001133	1371#	2708*	2716*	3399				
NSG =	000400	1277#							
NXTDRV	007044	2631#	2649						
NXTSLV	007122	2641#	2652	2654					
OCTALO	001122	1363#	1796	2463					
ODIGIT	001150	1379#	1662	1681	1702	1716	2369	2396	
OPI =	020000	1282#							
OR =	000200	1244#							
OSC =	000100	1286#	2037	2039	2044	2058	2073		
OUT	002244	1493	1497	1517	1534#				
OUTBUF=	005730	1308#							
OUTGAP	003152	1767#	1925	2151					
OUTSPC	003056	1739#	1852	1928	1986				
PAFLG	001002	1350#	3534	3536*					
PARVEC=	000114	1175#							
PAT =	000020	1241#							
PCNTR	001004	1340*	1351#	3531*	3532				
PEFLRC=	000200	1276#							
PES =	000040	1259#							
PE1600=	002000	1299#	1932	3404					
PFVEC =	000024	1170#	1324						
PGE =	002000	1247#							
PIP =	020000	1266#	2298						
PIRQ =	177772	1156#							
PIRVEC=	000240	1176#							
PLKCSR=	172540	1181#							











DZTUG-B TMO2/TE16 DRIVE FUNCTION TIMER MACY11 30(1046) 02-DEC-77 09 48 PAGE 111  
CZTUGC P11 02-DEC-77 09 46 CROSS REFERENCE TABLE -- MACRO NAMES

SEQ 0109

INPUT	1118#	2461	2481	2549	2699	2710										
RESTOR	1115#	1682	1717	1810	1853	1961	2110	2153								
REWIND	1121#	2267	2751	2850	3050	3211	3291	3326	3363	3402	3484	3566	3602	3644		
SAVE	1112#	1660	1699	1793	1820	2085	2123									
SETGO	1131#	2270	2291	2342	2877	2897	2927	2961	2967	2986	2990	3017	3021	3063	3069	
	3088	3092	3099	3117	3123	3141	3145	3151	3170	3174	3178	3183	3216	3226	3235	
	3241	3298	3334	3371	3410	3444	3463	3472	3575	3611	3625					
TIMCHK	1128#	2162	2177	2759	2778	2805	2837	2859	2881	2910	2945	2971	3002	3039	3073	
	3103	3127	3155	3187	3245	3310	3346	3383	3422	3448	3476	3583	3633			
TIMEON	1125#	2174	2754	2800	2832	2854	2876	2905	2941	2966	2998	3035	3068	3098	3122	
	3150	3182	3240	3306	3342	3379	3418	3443	3471	3580	3630					
SCATCH	995#	1024														
SCHNMO	1070#	2620														
SLPREG	995#	1136														
SCP/EC	995#	1163														
ATYPE	995#	1553														

ABS 017020 000

ERRORS DETECTED 0

CZTUGC, CZTUGC SEQ/SOL/NL TOC/DOC=CZTUGC SML/ML, CZTUGC P11

RUN-TIME 47.8 SECONDS

RUN-TIME RATIO 440/13=33.5

CORE USED 8K (15 PAGES)

DOCUMENT PAGES 109

mv taz P 45 S  
< &S (v m' va qaz N



