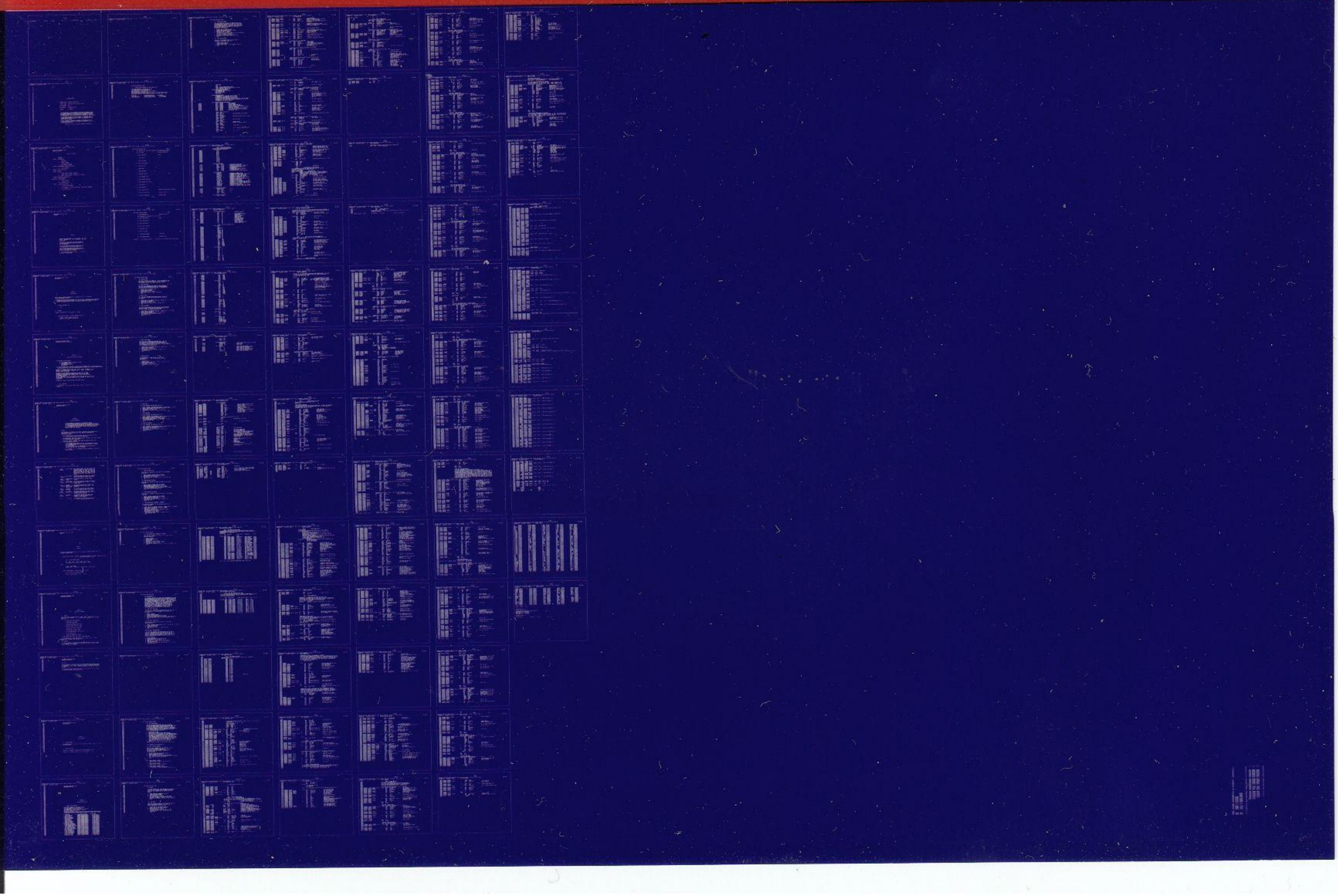
TM02/TU16

DRIVE FUNCTION TIMER
MD-11-DZTUG-B

EP-DZTUG-B-DL-A

FICHE 1 OF 1

OCT 1977
digital
MADE IN USA



B01

EOF1DZRMJASEQ	00010000	770804	PDP10 411	Z:HDR1DZTUGBSEQ	00010000	770804

```
                        .REM %
```

IDENTIFICATION

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

PRODUCT NAME: TM02/TE16 DRIVE FUNCTION TIMER

DATE CREATED: AUG 77

MAINTAINER: DIAGNOSTIC GROUP

AUTHOR: R. BARNES

TABLE OF CONTENTS

```
                                TM02 DRIVE FUNCTION TIMER                                PAGE 3
                                ABSTRACT


                                                ABSTRACT


                                PROGRAM DZTUG MEASURES THE TIME REQUIRED  AND  GAP
                                SIZES    PRODUCED   BY   THE   TM02/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.
```

CHAPTER 1

REQUIREMENTS

PDP-11 FAMILY CENTRAL PROCESSOR WITH 4K MEMORY WITH UP TO 64 TM02/TE16 CONTROLLER/MAGTAPE STATIONS.

PROGRAM CAN BE RUN ON A PROCESSOR THAT DOES NOT HAVE A HARDWARE SWITCCH 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

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 (TM02) 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.

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.

SW15 (100000)	HALT ON ERROR	THIS SWITCH WHEN SET WILL HALT THE PROCESSOR WHEN AN ERROR IS DETECTED. THE PC+2 AND PSW AT THE TIME OF THE ERROR IS STORED ON THE STACK. PRESSING CONTINUE WILL CAUSE THE ERROR TO BE TYPED (IF SELECTED) AND FURTHER TESTING RESUMED.
SW14 (040000)	LOOP SUBTEST	THIS SWITCH WHEN SET LOOPS THE CURRENT SUBTEST REGARDLESS OF ERROR CONDITION.
SW13 (020000)	INHIBIT ERROR TYPEOUT	THIS SWITCH WHEN SET INHIBITS ERROR TYPEOUT.
SW11 (004000)	INHIBIT SUB-TEST ITERATION	THIS SWITCH WHEN SET CAUSES EACH SUBTEST TO BE EXECUTED ONLY ONCE. (INITIAL STARTUP ONLY).
SW10 (002000)	INHIBIT FUNCTION TIME PUBLICATION	THIS SWITCH WHEN SET WILL INHIBIT THE PRINTING OF THE FUNCTION TIMES. (SEE CHAPTER 8.)
SW09 (001000)	RING BELL ON ERROR	THIS SWITCH WHEN SET WILL RING THE BELL ON THE TTY WHEN AN ERROR IS DETECTED.
SW07 (000200)	HALT AFTER SELECTED TEST	THIS SWITCH WHEN SET WILL CAUSE THE PROGRAM TO HALT AFTER THE TEST SELECTED IN SW05-SW00 IS EXECUTED.
SW06 (000100)	CONTINUOUS CYCLE	THIS SWITCH WHEN SET WILL CAUSE THE PROGRAM TO RUN CONTINUOUSLY UNTIL STOPPED BY THE OPERATOR.
SW5-0	TEST SELECT	THE PROGRAM WILL HALT AFTER EXECUTION OF THE TEST SELECTED WHEN SW07 IS SET.

CHAPTER 4

ERRORS

TWO TYPES OF ERRORS ARE DETECTED BY THIS PROGRAM, HARDWARE ERRORS AND INCORECT 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-IIIIII = 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
```

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.

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))

CHAPTER 6

MISCELLANEOUS

6.1 STACK POINTER

THE STACK POINTER IS INITAILLY 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.

CHAPTER 7

PROGRAM DESCRIPTION

7.1 SAMPLE TIME DOCUMENT

```
TYPE FIRST ADDRESS OF CONTROLER 172440
TYPE TM02 DRIVE #'S TO BE TESTED 0
FOR TM02 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
*****************************************************************
* TM02 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=(004450-004250)       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
```

```
   531
```

```
532
533                                 TM02 DRIVE FUNCTION TIMER                                   PAGE 12
534
535                                 7.1.1   SAMPLE TIME DOCUMENT FOR TAPE SPEED TESTS
536
537                                 TYPE FIRST ADDRESS OF CONTROLLER 172440
538                                 TYPE TM02 DRIVE #'S TO BE TESTED 0
539                                 FOR TM02 DRIVE 0- TYPE SLAVE #'S TO BE TESTED 7
540                                 SPEED TESTS ONLY? (YES/NO = 1/0) 1
541                                 ************************************************************
542                                 *TM02 DRIVE FUNCTION TIMES- DRIVE # 0 SLAVE # 7 9 CHAN. SER. # 5009
543                                 *
544                                 *FUNCTION               TIME(SPECIFICATION)     TIME(ACTUAL)
545                                 *TAPE SPEED FWD         RANGE=<022700-021700>       ACTUAL=022500
546                                 *TAPE SPEED REV         RANGE=<022700-021700>       ACTUAL=022500
547
```

7.2 TEST SEQUENCE WITH RELATED ADJUSTMENTS AND ASSOCIATED HARDWARE

TEST NO./NAME	RELATED ADJUSTMENTS
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-R	* "
15. TURN AROUND R-F	* "
16. GAP SIZE-STOP HALF	*FWRD/REV SPEED-START/STOP-RAMPS
17. GAP SIZE-START HALF	*SAME AS IN TEST 16
20. GAP SIZE INTERRECORD	*FWD/REV SPEED

```
                                TM02 DRIVE FUNCTION TIMER                                PAGE 14

                        21. GAP CONSISTENCY                 *SAME AS IN TEST 16

                        *TEST NUMBER 22 IS RESERVED FOR FUTURE USE

                        23. DATA TIME 200 BPI               *NONE

                        24. DATA TIME 556 BPI               *  "

                        25. DATA TIME 800 BPI               *  "

                        26. DATA TIME 1600 BPI              *  "

                        27. ERASE GAP TIME                  *  "

                        30. WRITE FILE MARK                 *  "

                        31. TAPE SPEED-FORWARD              *FWD SPEED

                        32. TAPE SPEED-REVERSE              *REVERSE SPEED

                *****NOTE: IF TIME PROBLEMS APPEAR IN T1 THRU T30, RUN TAPE SPEED TESTS FIRST*******
```

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 TM02 (M8903), AND THE SETTLEDOWN ONE SHOT (M8910).

T1. WRITE FROM BOT:

THIS TEST WILL MEASURE ACCELLERATION 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 ACCELLERATION 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

```
                                        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
```

```
                                T6. READ START

                                THIS TEST MEASURES THE SAME DELAY AS IN T2.

                                1.  WRITE 1 RECORD, THEN BACKSPACE OVER IT, ASSURE TAPE IS STOPPED.
                                2.  ISSUE A READ FORWARD OF THE RECORD WRITTEN IN STEP 1.
                                3.  MONITOR BIT 15 OF TC (ACCL)
                                4.  TIME FROM GO TO RESET OF ACCL IS START DELAY
                                5.  STOP

                                T7. READ SHUTDOWN:

                                THIS TEST MEASURES THE SAME DELAY AS IN T3.

                                1.  WRITE 1 RECORD, THEN BACKSPACE OVER IT, ASSURE TAPE IS STOPPED.
                                2.  READ FORWARD THE RECORD WRITTEN IN STEP 1.
                                3.  MONITOR FRAME COUNT AND BIT 4 OF DS (SDWN).
                                4.  TIME FROM FC=RECORD SIZE (LAST FRAME READ) TO SDWN=1
                                    IS THE SHUTDOWN TIME.
                                5.  STOP

                                T10. READ SETTLEDOWN:

                                THIS TEST MEASURES THE SAME DELAY AS IN T4.

                                1.  WRITE 1 RECORD, THEN BACKSPACE OVER IT, ASSURE TAPE IS STOPPED.
                                2.  READ FORWARD 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
```

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 REVERSE 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. MONITOR 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

```
798
799                                 TM02 DRIVE FUNCTION TIMER                                   PAGE 18
800
801                                         T15. TURN AROUND DELAY-REVERSE TO FORWARD
802
803                                         THIS TEST WILL MEASURE THE TIME AS IN T14, BUT IN THE
804                                         OPPOSITE DIRECTION.
805
806                                         1.  WRITE 1 RECORD.
807                                         2.  ASSURE TAPE IS STOPPED
808                                         3.  READ REVERSE
809                                         4.  MONITOR DRY (BIT 7 OF DS)
810                                         5.  WHEN DRY = 1, ISSUE A READ FORWARD
811                                         6.  MONITOR ACCL (BIT 15 OF TC)
812                                         7.  TIME FROM GO FORWARD TO ACCL = 1 IS THE TURN AROUND TIME.
813                                         8.  STOP.
```

```
                        GAP MEASUREMENTS:

                        THE PREVIOUS THIRTEEN (13) TESTS WERE MEASUREMENTS OF LOGIC
                        DELAYS PERFORMED BY THE TM02 OR TE16 IN ORDER TO ALLOW FOR
                        PROPER ACCELERATION AND DECELLERATION OF TAPE ACCORDING TO
                        THE DESIRED INTEROCORD 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 INTERRCORD 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 ACCELLERATION AND DECELLERATION 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.
```

```
870                                     7.  STOP
```

T21. GAP CONSISTENCY:

NOW THAT WE HAVE ESTABLISHED THAT THE INTERRCORD
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 NRZ ONLY

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

```
                        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.

                %
```

```
                                        .NLIST  MC
                                        .LIST   ME
                                        .ABS
                                        .MCALL  $CPVEC,$CPREG,$CATCH,$TYPE
                                        .TITLE  DZTUG-8 TM02/TE16 DRIVE FUNCTION TIMER
                                        .SBTTL  STARTING INSTRUCTIONS
                                ;LOADING AND STARTING PROCEEDURE
                                ;       LOAD PROGRAM USING ABS LOADER
                                ;       LOAD ADDRESS 200
                                ;       SET SWITCH OPTIONS
                                ;       PRESS START

                                ;GENERAL REGISTER USAGE:
                                ;       R0=ADDRESS OF 'FC' REGISTER (SET BY SCOPE)
                                ;       R1=ADDRESS OF 'DS' REGISTER (SET BY SCOPE)
                                ;       R2=RETURN PC FROM TIMER (SET BY EACH TEST)
                                ;       R3=INDEX INDICATING PREVIOUS OSCILLATOR POLARITY (SET BY TIMER)
                                ;       R4=CONTAINS 'TICK' COUNT WHEN TIMER IS RUNNING (SET BY TIMER)
                                ;       R5=ADDRESS OF CSI (SET BY SCOPE)

                                ;SWITCH REGISTER SWITCH ASSIGNMENTS
                                        SW15=   100000          ;HALT ON ERROR
                                        SW14=   040000          ;LOOP SUBTEST
                                        SW13=   020000          ;INHIBIT ERROR TYPE OUT
                                        SW11=   004000          ;INHIBIT SUBTEST ITERATION
                                        SW10=   002000          ;INHIBIT PUBLICATION OF FUNCTION TIMES
                                        SW09=   001000          ;RING BELL ON ERROR
                                        SW08=   000400          ;TYPE LINE ITEM AFTER EACH ITERATION
                                        SW07=   000200          ;HALT ON TEST SELECTED IN SW05-SW00
                                        SW06=   000100          ;CONTINUOUS CYCLE

                                .SBTTL  MACRO DEFINITIONS
                                .MACRO  SAVE
                                JSR     PC,.SAVE                ;SAVE REGISTERS ON THE STACK
                                .ENDM   SAVE
                                .MACRO  RESTORE
                                JSR     PC,.RESTORE             ;RESTORE REGISTERS FROM THE STACK
                                .ENDM   RESTORE
                                .MACRO  INPUT
                                JSR     PC,.INPUT               ;GET USER INPUT
                                .ENDM   INPUT
                                .MACRO  REWIND
                                JSR     PC,.REWIND              ;REWIND SLAVE
                                BVS     99$                     ;BRANCH IF ERROR ON REWIND
                                .ENDM   REWIND
                                .MACRO  TIMEON
                                JSR     PC,TIMON                ;TURN TIMER ON
                                .ENDM
                                .MACRO  TIMCHK
                                JMP     TIMER(R3)               ;GO TO TIMER & RETURN VIA R2
                                .ENDM
                                .MACRO  SETGO
                                INC     (R5)                    ;SET 'GO' BIT
                                .ENDM
```

```
                                        .SBTTL  REGISTER ASSIGNMENTS
                                ;;DEFINITIONS AND REGISTER ASSIGNMENTS
                                ;;GENERAL REGISTER ASSIGNMENTS
                                        R0=%0
                                        R1=%1
                                        R2=%2
                                        R3=%3
                                        R4=%4
                                        R5=%5
                                        SP=%6
                                        PC=%7
                                        R10=%0
                                        R11=%1
                                        R12=%2
                                        R13=%3
                                        R14=%4
                                        R15=%5

                                ;;REGISTER ADDRESSES
                                        PSW=    177776          ;;PROCESSER STATUS WORD
                                        SLR=    177774          ;;STACK LIMIT REGISTER (11/40,11/45)
                                        PIRQ=   177772          ;;PROGRAM INTERRUPT REQ. (11/45)
                                        UBREAK= 177770          ;;MICRO-BREAK REGISTER (11/45)
                                        TKS=    177560          ;;KEYBOARD CSR
                                        TKB=    177562          ;;KEYBOARD DATA BUFFER REGISTER
                                        TPS=    177564          ;;TELEPRINTER CSR
                                        TPB=    177566          ;;TELEPRINTER DATA BUFFER REGISTER

                                ;;VECTOR ADDRESSES
                                        ERRVEC=4                ;;ADDRESS OF ERROR VECTOR
                                        RESVEC=10               ;;ADDRESS OF RESERVED INST. TRAP VECTOR
                                        TBITVEC=14              ;;ADDRESS OF 'T' BIT TRAP VECTOR
                                        TRTVEC=14               ;;ADDRESS OF 'TRACE' TRAP VECTOR
                                        BPTVEC=14               ;;ADDRESS OF 'BREAKPOINT' TRAP VECTOR
                                        IOTVEC=20               ;;ADDRESS OF IOT TRAP VECTOR
                                        PFVEC=24                ;;ADDRESS OF POWER FAIL TRAP VECTOR
                                        EMTVEC=30               ;;ADDRESS OF EMT VECTOR
                                        TRAPVEC=34              ;;ADDRESS OF TRAP VECTOR
                                        TKVEC=  60              ;;ADDRESS OF TTY KEYBOARD INT. VECTOR
                                        TPVEC=64                ;;ADDRESS OF TTY PRINTER INTERRUPT VECTOR
                                        PARVEC= 114             ;;ADDRESS OF MA/MF PARITY ERROR VECTOR
                                        PIRVEC=240              ;;ADDRESS OF PIRQ VECTOR
                                        FPEVEC=244              ;;ADDRESS OF FLOATING POINT INT. VECTOR
                                        MMVEC=250               ;;ADDRESS OF MEM MGMT ERROR TRAP VECTOR

                                ;CLOCK ADDRESS AND VECTORS
                                        PLKCSR= 172540          ;KW11-P
                                        PLKVEC= 104
                                        LKS=    177546          ;KW11-L
                                        LKVEC=  100
                                        LPS=    177514          ;LP11
                                        LPB=    177516

                                ;RH11,TM02/TE16 REGISTERS
                                        TMCS1=  172440
```

```
                                ;TM02/TE16 INDEX VALUES
000000                  CS1=    00              ;CONTROL STATUS #1
000002                  WC=     02
000004                  BA=     04              ;BUS ADDRESS REGISTER
000006                  FC=     06              ;FRAME COUNT
000010                  CS2=    10              ;CONTROL STATUS #2
000012                  DS=     12              ;DRIVE STATUS
000014                  ER=     14              ;ERROR REG #1
000016                  AS=     16              ;ATTENTION SUMMARY
000022                  DB=     22              ;DATA BUFFER REG
000024                  MR=     24              ;MAINTENANCE REG
000026                  DT=     26              ;DRIVE TYPE REG
000030                  SN=     30              ;SERIAL NUMBER REGISTER
000032                  TC=     32              ;TAPE CONTROL REG

                        .SBTTL  TM02/TE16 REGISTER BITS
                ;RHCS1-CS1(RS)
000001                  GO=     1
000000                  NOP=    0
000002                  RWDOFF= 2
000006                  RWD=    6
000010                  DRYCLR= 10
000026                  WFMK=   26
000024                  ERASE=  24
000030                  SPCFWD= 30
000032                  SPCREV= 32
000050                  WCHKF=  50
000056                  WCHKR=  56
000060                  WFWD=   60
000070                  RDFWD=  70
000076                  RDREV=  76
000100                  IE=     100
000200                  RDY=    200
000400                  A16=    400
001000                  A17=    1000
002000                  PSEL=   2000
004000                  DVA=    4000
020000                  MCPE=   20000
040000                  TRE=    40000
100000                  SC=     100000
                ;RHCS2-CS2(RS)
000000                  DV0=    0
000001                  DV1=    1
000002                  DV2=    2
000003                  DV3=    3
000004                  DV4=    4
000005                  DV5=    5
000006                  DV6=    6
000007                  DV7=    7
000010                  BAI=    10
000020                  PAT=    20
000040                  CLR=    40
000100                  IR=     100
000200                  OR=     200
000400                  MDPE=   400
001000                  MXF=    1000
```

```
1158            002000                  PGE=    2000
1159            004000                  NEM=    4000
1160            010000                  NED=    10000
1161            020000                  UPE=    20000
1162            040000                  WCE=    40000
1163            100000                  DLT=    100000
1164                            ;RHDS-DS(RS)
1165            000001                  SLA=    1
1166            000002                  BOT=    2
1167            000004                  TMK=    4
1168            000010                  IDB=    10
1169            000020                  SDWN=   20
1170            000040                  PES=    40
1171            000100                  SSC=    100
1172            000200                  DRY=    200
1173            000400                  DPR=    400
1174            002000                  EOT=    2000
1175            004000                  WRL=    4000
1176            010000                  MOL=    10000
1177            020000                  PIP=    20000
1178            040000                  ERR=    40000
1179            100000                  ATA=    100000
1180                            ;RHER-ER(RS)
1181            000001                  ILF=    1
1182            000002                  ILR=    2
1183            000004                  RMR=    4
1184
1185            000020                  FMT=    20
1186            000100                  INCVAE= 100
1187            000200                  PEFLRC= 200
1188            000400                  NSG=    400
1189            001000                  FCE=    1000
1190            002000                  CSITM=  2000
1191            004000                  NEF=    4000
1192            010000                  DTE=    10000
1193            020000                  OPI=    20000
1194            040000                  UNS=    40000
1195
1196                            ;RHMR-MR(RS)
1197            000100                  OSC=    100
1198
1199                            ;RHDT-DT(RS)
1200            002000                  SPR=    2000
1201            010000                  CH7=    10000
1202            040000                  TAP=    40000
1203
1204                            ;RHTC-TC(RS)
1205            000300                  NORM11= 300
1206            000320                  CDM11=  320
1207            000000                  BPI200= 0
1208            000400                  BPI556= 000400
1209            001000                  BPI800= 001000
1210            002000                  PE1600= 002000
1211            100000                  ACCL=   100000
1212
1213                            ;INSTRUCTION EQUATES
```

```
                                HLT=    TRAP
                                SCOPE=  EMT
                                TYPE=   IOT

                        ;MISCELLANEOUS EQUATES
                                OUTBUF=INIT             ;OTPUT BUFFER START AT BEG OF PROGRAM
                                FRMCNT= -256.           ;FRAME COUNT
                                WRDCNT= -128.           ;WORD COUNT
                        ;ASCII EQUATES
                                CNTRLC= 3               ;ASCII CODE FOR CONTROL C (↑C)
                                HT=     11              ;ASCII CODE FOR HORIZONTAL TAB
                                LF=     12              ;ASCII CODE FOR LINE FEED
                                CR=     15              ;ASCII CODE FOR CARRIAGE RETURN
                                CNTRLO= 17              ;ASCII CODE FOR CONTROL O (↑O)
                                CNTRLU= 25              ;ASCII CODE FOR CONTROL U (↑U)
```

```
                                        ;SETUP TRAP VECTORS
                    000014                      .=TBITVEC
000014  000016                          .WORD   .+2             ;SET 'T' TRAP TO TIMER ROUTINE
000016  000000                          .WORD   HALT            ;PRIORITY LEVEL 7
000020  002332                          .WORD   .TYPE           ;SET IOT TRAP TO .TYPE ROUTINE
000022  000000                          .WORD   0               ;PRIOTITY LEVEL 0
000024  000026                          .WORD   PFVEC+2         ;POWER FAIL TRAP TO HALT
000026  000000                          .WORD   HALT            ;AT PFVEC+2
000030  004126                          .WORD   .SCOPE          ;SET EMT TRAP TO .SCOPE ROUTINE
000032  000340                          .WORD   340             ;PRIORITY LEVEL 7
000034  003652                          .WORD   .HLT            ;SET TRAP TRAP TO .HLT ROUTINE
000036  000340                          .WORD   340             ;PRIORITY LEVEL 7
                    000060                      .=TKVEC
000060  003604                          .WORD   TKISR
000062  000340                          .WORD   340

                                        ;SOFTWARE SWITCH REGISTER LOC. 176
                    000176              .=176
000176  000000                  SWREG:  0               ;SOFTWARE SWITCH REGISTER

                    000200              .=200
000200  000137  005724                  JMP     @#INIT                  ;GO TO START OF PROGRAM

                    000500              .=500
                    000600      STKPTR= 600                     ;STACK

                    001000              .=1000
                                ;PROGRAM TAGS
001000  177570              SWR:    177570                      ;SWITCH REGISTER
001002  000000              SCPADR: .WORD   0
001004  000                 DRVNUM: .BYTE   0                   ;TM02 DRIVE UNDER TEST
001005  000                 SLVNUM: .BYTE   0                   ;TE16 SLAVE UNDER TEST
001006  000000              SLVPTR: .WORD   0                   ;POINTER TO SLAVE TABLE (SLVTBL) BELOW
001010  172440              TMBASE: .WORD   TMCS1               ;BASE ADDRESS OF TM02/TE16 REGISTERS
001012  000000              ATIME:  .WORD   0                           ;CONTAINS 'TICK' COUNT
001014  000020              ATIMTBL:.BLKW   16.                 ;EACH ENTRY CONTAINS TIME FOR FUNCTION
                                                                ;ENTRIES ARE MADE BY 'SCOPE' ROUTINE
001054  000020              GAPTBL: .BLKW   16.                 ;TIMES RECORDED BY 'GAP CONSISTANCY' TEST
001114  000000              DELTIM: .WORD   0                   ;VARIBLE DELAY
001116  000000              OCTALD: .WORD   0
001120  000                 GAP:    .BYTE   0                   ;CONTAINS GAP # (USED FOR TST 021)
001121  000                 ITCNT:  .BYTE   0                   ;ITERATION COUNT
001122  000                 TSTNUM: .BYTE   0                   ;TEST #
001123  000                 ERFLG:  .BYTE   0                   ;ERROR FLAG
001124  000                 PRGFLG: .BYTE   0                   ;PROGRAM FLAG
001125  000                 UNTFND: .BYTE   0                   ;UNIT FOUND INDICATOR
001126  000                 TYPFLG: .BYTE   0
001127  000                 NRZFLG: .BYTE   0                   ;INDICATES IF DRIVE IS NRZ ONLY.
001130  000                 ASFLG:  .BYTE   0                   ;1/0 = YES/NO.
                    001132              .EVEN
001132  030460              DIGTAB: "01
                                    "23
                                    "45
                                    "67
                                    "89
001144  000006              ODIGITS:.BLKB   6                   ;RESERVE SPACE FOR CONVERTED DIGITS
```

```
1285  001152    000                         .BYTE   0               ;TERMINATOR
1286            001154                      .EVEN
1287  001154    000010              DRVTBL: .BLKB   8.              ;A 0/-1 = DRIVE NOT TO BE/TO BE TESTED
1288  001164    000100              SLVTBL: .BLKB   64.             ;A 0/-1 = SLAVE NOT TO BE/TO BE TESTED
1289  001264    000110              INBUF:  .BLKB   72.             ;TELETYPE INPUT BUFFER
1290  001374    005015    000       CRLF:   .ASCIZ  <CR><LF>        ;MISCELLANEOUS ASCII CHARACTERS
1291  001377      134     000       BKSLSH: .ASCIZ  '\'
1292  001401      060     000       ECHO:   .ASCIZ  '0'
1293  001403      007     000       BELL:   .ASCIZ  <7>
1294  001405      055     000       DASH:   .ASCIZ  '-'
1295  001407      040               SPACE2: .ASCII  ' '
1296  001410    000040              SPACE:  .ASCIZ  ' '
1297  001412    004476    000       ANGTAB: .ASCIZ  '>'<HT>
1298            001416                      .EVEN
```

```
1299                                            .SBTTL  TIME SPECIFICATION TABLE
1300                                    ;THE BELOW TABLE CONTAINS THE SPECIFIED FUNCTION TIMES IN TENS OF
1301                                    ;MICROSECONDS. NOTE THAT WHEN TIMES ARE TYPED THAT THEY ARE TYPED IN
1302                                    ;MICROSECONDS (BY APPENDING A 0).
1303                                    ;FORMAT IS
1304                                    ;       .WORD   MAX,MIN         ;TIME IN MS     FUNCTION        TEST #
1305
1306  001416  000000  000000   STIMTBL:.WORD   0,0              ;SPARE
1307  001422  036050  035230           .WORD   15400.,15000.   ;154.0-150.0    WRITE FROM BOT  TST001
1308  001426  001666  001546           .WORD   00950.,00870.   ;9.5-8.7        WRITE START     TST002
1309  001432  001572  001522           .WORD   00890.,00850.   ;8.9-8.5        WRITE SHUTDOWN  TST003
1310  001436  002506  001452           .WORD   01350.,00810.   ;13.5-8.1        WRITE STLDOWN   TST004
1311  001442  007164  006344           .WORD   03700.,03300.   ;37.0-33.0       READ FROM BOT   TST005
1312  001446  000500  000404           .WORD   00320.,00260.   ;3.2-2.6        READ START      TST006
1313  001452  000721  000651           .WORD   00465.,00425.   ;4.65-4.25      READ SHUTDOWN   TST007
1314  001456  002506  001452           .WORD   01350.,00810.   ;13.5-8.1        READ SETTLEDOWN TST010
1315  001462  000500  000404           .WORD   00320.,00260.   ;3.2-2.6        RD REV START    TST011
1316  001466  000562  000512           .WORD   00370.,00330.   ;3.7-3.3        RD REV SHTDWN   TST012
1317  001472  002506  001452           .WORD   01350.,00810.   ;13.5-8.1        RD REV STLDWN   TST013
1318  001476  003206  002056           .WORD   01670.,01070.   ;16.7-10.7      TRN RND DLY F-R TST014
1319  001502  003206  002056           .WORD   01670.,01070.   ;16.7-10.7      TRN RND DLY R-F TST015
1320  001506  002412  001666           .WORD   01290.,00950.   ;12.9-9.5       GAP SIZE STOP   TST016
1321  001512  002234  001522           .WORD   01180.,00850.   ;11.8-8.5       GAP SIZE STRT   TST017
1322  001516  002710  002532           .WORD   01480.,01370.   ;14.8-13.7      GAP SIZE INTER  TST020
1323  001522  002544  002330           .WORD   01380.,01240.   ;13.8-12.4      GAP CONSISANCY  TST021
1324  001526  000000  000000           .WORD   0,0             ;0.0-0.0        DUMMY           TST022
1325  001532  004552  004406           .WORD   02410.,02310.   ;24.1-23.1      DAT TIME 2008PI TST023
1326  001536  004540  004374           .WORD   02400.,02300.   ;24.0-23.0      DAT TIME 5568PI TST024
1327  001542  004540  004374           .WORD   02400.,02300.   ;24.0-23.0      DAT TIME 8008PI TST025
1328  001546  004716  004552           .WORD   02510.,02410.   ;25.1-24.1      DAT TIME 1600PE TST026
1329  001552  023564  023254           .WORD   10100.,09900.   ;101.0-99.0      ERASE           TST027
1330  001556  024404  024074           .WORD   10500.,10300.   ;105.0-103.0     WRT FILE MARK   TST030
1331  001562  004336  004172           .WORD   02270.,02170.   ;22.7-21.7      READ 1" TAPE    TST031
1332  001566  004336  004172           .WORD   02270.,02170.   ;22.7-21.7      RD REV 1" TAPE  TST032
1333
1334                                    ;NOTE: TEST 31 AND 32 REQUIRE PRERECORDED 800BPI SKEW TAPE.
```

```
1335                                            .SBTTL  GAP TIME SPECIFICATION TABLE
1336                                    ;THIS TABLE CONTAINS THE GAP SIZES (IN TENS OF MICROSECONDS) FOR EACH
1337                                    ;OF THE 16. GAPS RECORDED BY THE GAP CONSISTANCY TEST (TST021).
1338                                    ;NOTE: GAP #'S ARE IN OCTAL.
1339
1340                                    ;       .WORD   MAX,MIN(10)     ;TIME IN MS(10) GAP #   DELAY IN MS(10)
1341
1342  001572  002602  002412           GTIMTBL:.WORD   01410.,01290.   ;14.1-12.9      GAP-0   0 MS
1343  001576  002652  002506                   .WORD   01450.,01350.   ;14.5-13.5      GAP-1   1.0 MS
1344  001602  002734  002532                   .WORD   01500.,01370.   ;15.0-13.7      GAP-2   2.0 MS
1345  001606  002734  002532                   .WORD   01500.,01370.   ;15.0-13.7      GAP-3   3.0 MS
1346  001612  002734  002424                   .WORD   01500.,01300.   ;15.0-13.0      GAP-4   4.0 MS
1347  001616  002652  002260                   .WORD   01450.,01200.   ;14.5-12.0      GAP-5   5.0 MS
1348  001622  002652  002260                   .WORD   01450.,01200.   ;14.5-12.0      GAP-6   6.0 MS
1349  001626  002652  002260                   .WORD   01450.,01200.   ;14.5-12.0      GAP-7   7.0 MS
1350  001632  002532  002260                   .WORD   01370.,01200.   ;13.7-12.0      GAP-10  8.0 MS
1351  001636  002532  002260                   .WORD   01370.,01200.   ;13.7-12.0      GAP-11  9.0 MS
1352  001642  002532  002260                   .WORD   01370.,01200.   ;13.7-12.0      GAP-12  10.0 MS
1353  001646  002532  002260                   .WORD   01370.,01200.   ;13.7-12.0      GAP-13  11.0 MS
1354  001652  002532  002260                   .WORD   01370.,01200.   ;13.7-12.0      GAP-14  12.0 MS
1355  001656  002532  002260                   .WORD   01370.,01200.   ;13.7-12.0      GAP-15  13.1 MS
1356  001662  002532  002260                   .WORD   01370.,01200.   ;13.7-12.0      GAP-16  14.1 MS
1357  001666  002532  002260                   .WORD   01370.,01200.   ;13.7-12.0      GAP-17  15.1 MS
```

```
1358                                    .SBTTL  TEST HEADER POINTERS
1359                                    ;THE BELOW TABLE CONTAINS POINTERS TO EACH TEST'S DESCRIPTOR
1360  001672  000000                    NAMPTR: .WORD   0
1361  001674  014665                            .WORD   A.T001
1362  001676  014707                            .WORD   A.T002
1363  001700  014727                            .WORD   A.T003
1364  001702  014751                            .WORD   A.T004
1365  001704  014775                            .WORD   A.T005
1366  001706  015017                            .WORD   A.T006
1367  001710  015036                            .WORD   A.T007
1368  001712  015060                            .WORD   A.T010
1369  001714  015103                            .WORD   A.T011
1370  001716  015125                            .WORD   A.T012
1371  001720  015152                            .WORD   A.T013
1372  001722  015201                            .WORD   A.T014
1373  001724  015232                            .WORD   A.T015
1374  001726  015263                            .WORD   A.T016
1375  001730  015311                            .WORD   A.T017
1376  001732  015340                            .WORD   A.T020
1377  001734  015370                            .WORD   A.T021
1378  001736  000000                            .WORD   0                       ;DUMMY TEST
1379  001740  015413                            .WORD   A.T023
1380  001742  015437                            .WORD   A.T024
1381  001744  015463                            .WORD   A.T025
1382  001746  015507                            .WORD   A.T026
1383  001750  015534                            .WORD   A.T027
1384  001752  015556                            .WORD   A.T030
1385  001754  015601                            .WORD   A.T031
1386  001756  015623                            .WORD   A.T032
```

```
1387
1388                                        ;CHECK SWITCH REGISTER ROUTINE. CHECKS FOR ↑G TO ALLOW CHANGING
1389                                        ;OF LOC.176.
1390                                        ;CALL IS BY WAY OF JSR  PC,CKSWR
1391                                        ;LOCATIONS USED:
1392  001760  000000                       TIB:    .WORD   0
1393  001762  000000                       TEMPST: .WORD   0
1394  001764  000000                       COUNT:  .WORD   0
1395  001766  000000                       RDSW:   .WORD   0
1396
1397  001770  022767  000176  177002  CKSWR:  CMP     #SWREG,SWR              ;SOFTWARE SWITCH REG PRESENT
1398  001776  001120                        BNE     OUT                     ;NO, GET OUT
1399  002000  016767  175556  177752        MOV     TKB,TIB         ;AND STRIP OFF
1400  002006  042767  177600  177744        BIC     #177600,TIB             ;THE GARBAGE
1401  002014  022767  000007  177736        CMP     #7,TIB          ;IS IT A ‹↑G›
1402  002022  001106                        BNE     OUT
1403  002024  000004  015645                TYPE,L.CNTG
1404  002030  000004  015652        CNTLU:  TYPE,L.SWR
1405  002034  017702  176740                MOV     @SWR,R2
1406  002040  004767  000564                JSR     PC,TYPOCT
1407  002044  000004  015661                TYPE,L.NEW
1408
1409  002050  005067  177706                CLR     TEMPST
1410  002054  012767  000007  177702        MOV     #7,COUNT
1411  002062  004767  000154        1$:     JSR     PC,TTIN         ;GO  READ A CHARACTER
1412  002066  042767  177600  177664        BIC     #177600,TIB     ;STRIP OFF GARBAGE
1413  002074  122767  000025  177656        CMPB    #25,TIB         ;IS IT A ↑U?
1414  002102  001001                        BNE     2$              ;BRANCH IF NOT
1415  002104  000751                3$:     BR CNTLU                ;START OVER
1416  002106  122767  000015  177644  2$:   CMPB    #15,TIB         ;IS IT A ‹CR›?
1417  002114  001012                        BNE     4$              ;BRANCH IF NOT
1418  002116  012767  000200  177642        MOV     #200,RDSW
1419  002124  004767  000230                JSR     PC,TCRLF        ;ECHO IT WITH ‹LF›
1420  002130  022767  000007  177626        CMP     #7,COUNT        ;WAS IT FIRST CHARACTER
1421  002136  001034                        BNE     7$              ;CHANGE SWR IF NOT FIRST ONE
1422  002140  000437                8$:     BR      OUT             ;GET OUT
1423  002142  122767  000060  177610  4$:   CMPB    #60,TIB
1424  002150  003004                        BGT     5$
1425  002152  122767  000067  177600        CMPB    #67,TIB
1426  002160  002003                        BGE     6$
1427  002162  000004  015671          5$:   TYPE,L.QUEST
1428  002166  000746                        BR      3$              ;START OVER IF NOT LEGAL CHARACTER
1429  002170  006367  177566          6$:   ASL     TEMPST
1430  002174  006367  177562                ASL     TEMPST
1431  002200  006367  177556                ASL     TEMPST
1432  002204  142767  000060  177546        BICB    #60,TIB         ;GET NITTY-GRITTY
1433  002212  156767  177542  177542        BISB    TIB,TEMPST
1434  002220  005367  177540                DEC     COUNT           ;ONLY WANT 6   DIGITS
1435  002224  001756                        BEQ     5$
1436  002226  000715                        BR      1$
1437  002230  016777  177526  176542  7$:   MOV     TEMPST,@SWR     ;CHANGE SWITCH REGISTER CONTENTS
1438  002236  000740                        BR      8$
1439  002240  000207                OUT:    RTS     PC
1440
```

```
1441
1442                                        ;TTY READ SUBROUTINE**********
1443
1444  002242  005067  175312        TTIN:   CLR     TKS
1445  002246  005067  175310                CLR     TKB
1446  002252  005067  177502                CLR     TIB
1447  002256  005267  175276                INC     TKS
1448  002262  105767  175272        TTIN1:  TSTB    TKS
1449  002266  100375                        BPL     TTIN1
1450  002270  016767  175266  177462        MOV     TKB,TIB
1451  002276  105767  175262        TTIN2:  TSTB    TPS
1452  002302  100375                        BPL     TTIN2
1453  002304  116767  177450  175254        MOVB    TIB,TPB
1454  002312  000207                        RTS     PC
1455
1456                                        .SBTTL  PROGRAM SUBROUTINES
1457                                        .SBTTL          TYPE SUBROUTINE
1458                                ;;ROUTINE TO TYPE ASCII MESSAGE. MESSAGE MUST TERMINATE WITH A 0 BYTE.
1459                                ;;THE ROUTINE WILL INSERT A NUMBER OF NULL CHARACTERS AFTER A LINE FEED.
1460                                ;;CALL: TYPE                    ;;A TRAP TYPE INSTRUCTION
1461                                ;;      MESADR                  ;;MESADR IS FIRST ADDRESS OF ASCIZ STRING
1462
1463                                ;;TAGS USED BY THE TYPE ROUTINE BELOW
1464          000011                        $HT=11                  ;;HORIZONTAL TAB
1465  002314     000                SNULL:  .BYTE   0               ;;CONTAINS NULL CHARACTER
1466  002315     002                $FILL:  .BYTE   2               ;;CONTAINS # OF FILLER CHARACTERS
1467  002316     000                $TPFLG: .BYTE   0               ;;CONTAINS TELEPRINTER AVAILABLE FLAG
1468                                                               ;;0/377 = AVAIL/NOT AVAIL
1469  002317     000                $TKFLG: .BYTE   0               ;;CONTAINS KEYBOARD AVAILABLE FLAG
1470  002320  177564                $TPS:   .WORD   177564          ;;ADDRESS OF TLEPRINTER STATUS REGISTER
1471  002322  177566                $TPB:   .WORD   177566          ;;ADDRESS OF TELEPRINTER DATA BUFFER
1472  002324     000                $CHARCNT:.BYTE  0               ;;CONTAINS # OF CHARS TYPED
1473  002325     000                $CNTRLO:.BYTE   0               ;;CONTAINS CONTROL O CHAR (IF TYPED)
1474  002326  005015     000        $CRLF:  .ASCIZ  <15><12>
1475          002332                        .EVEN
1476
1477  002332  010046                .TYPE:  MOV     R0,-(SP)        ;;SAVE R0
1478  002334  017600  000002                MOV     @2(SP),R0       ;;GET MESSAGE ADDRESS
1479  002340  062766  000002  000002        ADD     #2,2(SP)        ;;ADJUST RETURN PC
1480  002346  105067  177753                CLRB    $CNTRLO
1481
1482  002352  105767  177747        TYPE1:  TSTB    $CNTRLO         ;;BRANCH IF CONTROL O(10) WASN'T TYPED
1483  002356  001410                        BEQ     TYPE2
1484  002360  000004  002326        TCRLF:  TYPE,$CRLF              ;;TYPE <CR><LF>
1485  002364  105767  177376                TSTB    RDSW
1486  002370  100006                        BPL     TYPE3
1487  002372  005067  177370                CLR     RDSW
1488  002376  000207                        RTS     PC
1489  002400  112046                TYPE2:  MOVB    (R0)+,-(SP)     ;;PUSH CHARACTER TO BE TYPED ONTO STACK
1490  002402  001003                        BNE     TYPE4           ;;BRANCH IF NOT THE TERMINATOR
1491  002404  005726                        TST     (SP)+           ;;POP TERMINATOR CHAR OFF THE STACK
1492  002406  012600                TYPE3:  MOV     (SP)+,R0        ;;RESTORE R0
1493  002410  000002                        RTI                     ;;RETURN TO CALLER
1494
1495  002412  122716  000011        TYPE4:  CMPB    #$HT,(SP)       ;;BRANCH IF HORIZONTAL TAB <HT>
1496  002416  001445                        BEQ     9$
```

```
1497  002420  004767  000026          JSR     PC,5$           ;;TYPE CHARACTER
1498  002424  122726  000012    3$:   CMPB    #12,(SP)+       ;;CHECK IF CHARACTER WAS A LINE FEED
1499  002430  001350                  BNE     TYPE1           ;;BRANCH IF NOT LINE FEED
1500  002432  016746  177656          MOV     $NULL,-(SP)     ;;GET # OF FILLERS REQUIRED AND FILLER
1501                                                          ;;CHARACTER.
1502
1503  002436  105366  000001    4$:   DECB    1(SP)           ;;DECREMENT FILLERS REQ. COUNT
1504  002442  002770                  BLT     3$              ;;BRANCH IF NO MORE FILLERS ARE REQUIRED
1505  002444  004767  000002          JSR     PC,5$           ;;TYPE FILLER CHARACTER
1506  002450  000772                  BR      4$
1507
1508  002452  105777  177642    5$:   TSTB    @$TPS           ;;WAIT FOR OUTPUT DEVICE
1509  002456  100375                  BPL     .-4
1510  002460  122737  000017  002325  CMPB    #17,@#$CNTRLO   ;;CHECK IF CONTROL O WAS TYPED
1511  002466  001403                  BEQ     6$              ;;STOP TYPING MESSAGE IF ^O WAS TYPED
1512  002470  116677  000002  177624  MOVB    2(SP),@$TPB     ;;OUTPUT CHARACTER
1513  002476  122766  000015  000002 6$:   CMPB    #15,2(SP)       ;;BRANCH IF NOT <CR>
1514  002504  001003                  BNE     7$
1515  002506  105067  177612          CLRB    $CHARCNT        ;;CLEAR CHARACTERS TYPED COUNT
1516  002512  000406                  BR      8$
1517  002514  122766  000012  000002 7$:   CMPB    #12,2(SP)       ;;BRANCH IF <LF> OR 'NULL'
1518  002522  002002                  BGE     8$
1519  002524  105267  177574          INCB    $CHARCNT        ;;INCREMENT CHARACTER TYPED COUNT
1520  002530  000207            8$:   RTS     PC
1521
1522                                  ;;HORIZONTAL TAB <HT> PROCESSER
1523  002532  112716  000040    9$:   MOVB    #40,(SP)        ;;LOAD 'SPACE'
1524  002536  004767  177710    10$:  JSR     PC,5$           ;;TYPE 'SPACE'
1525  002542  132767  000007  177554  BITB    #7,$CHARCNT     ;;TYPE SPACES UNTIL A MULTIPLE
1526  002550  001372                  BNE     10$             ;;OF 8 CHARACTERS HAVE BEEN TYPED
1527  002552  105726                  TSTB    (SP)+           ;;POP SPACE
1528  002554  000676                  BR      TYPE1           ;;GET NEXT CHARACTER
1529
1530                                  ;SUBROUTINE TO SAVE GENERAL REGISTERS ON THE STACK
1531                                  ;CALL:  SAVE
1532  002556  010546            .SAVE: MOV     R5,-(SP)                ;SAVE REGISTERS ON THE STACK
1533  002560  010446                   MOV     R4,-(SP)
1534  002562  010346                   MOV     R3,-(SP)
1535  002564  010246                   MOV     R2,-(SP)
1536  002566  010146                   MOV     R1,-(SP)
1537  002570  010046                   MOV     R0,-(SP)
1538  002572  016646  000014           MOV     14(SP),-(SP)            ;GET RETURN PC
1539  002576  000207                   RTS     PC                      ;RETURN
1540
1541                                  ;SUBROUTINE TO RESTORE GENERAL REGISTERS FROM THE STACK
1542                                  ;CALL:  RESTORE
1543  002600  012666  000014 .RESTORE:MOV     (SP)+,14(SP)            ;MOVE RETURN PC
1544  002604  012600                   MOV     (SP)+,R0                ;RESTORE REGISTERS
1545  002606  012601                   MOV     (SP)+,R1
1546  002610  012602                   MOV     (SP)+,R2
1547  002612  012603                   MOV     (SP)+,R3
1548  002614  012604                   MOV     (SP)+,R4
1549  002616  012605                   MOV     (SP)+,R5
1550  002620  000207                   RTS     PC                      ;RETURN
1551
1552                                  ;SUBROUTINE TO CONVERT OCTAL DATA TO ASCII
```

```
1553                                    ;CALL:  MOV     NUMBER,R2               ;MOVE NUMBER TO R2
1554                                    ;       JSR     PC,CNVOCT
1555
1556  002622  110667  176300            CNVOCT: MOVB    SP,TYPFLG               ;SET DO NOT TYPE FLAG
1557  002626  000402                            BR      CNVTO
1558
1559                                            .SBTTL          OCTAL TO ASCII & TYPE ROUTINE
1560                                    ;SUBROUTINE TO CONVERT OCTAL NUMBER TO ASCII AND TYPE IT OUT
1561                                    ;CALL:  MOV     NUMBER,R2               ;PUT # IN R2
1562                                    ;       JSR     PC,TYPOCT               ;CALL ROUTINE
1563
1564  002630  105037  001126            TYPOCT: CLRB    @#TYPFLG                ;SET TYPE FLAG
1565  002634                            CNVTO:
1566  002634  004767  177716                    JSR     PC,.SAVE                ;SAVE REGISTERS ON THE STACK
1567  002640  012704  001144                    MOV     #ODIGITS,R4             ;SET PTR TO OUTPUT
1568  002644  005003                            CLR     R3                      ;R3 WILL CONTAIN OCTAL DIGIT
1569  002646  010201                            MOV     R2,R1                   ;GET # TO BE TYPED
1570  002650  006302                    1$:     ASL     R2                      ;SHIFT #
1571  002652  006103                            ROL     R3
1572  002654  012700  000006                    MOV     #6,R0                   ;SET DIGIT COUNTER
1573  002660  000404                            BR      3$
1574
1575  002662  006302                    2$:     ASL     R2                      ;SHIFT # 3 PLACES LEFT
1576  002664  006103                            ROL     R3
1577  002666  005301                            DEC     R1
1578  002670  001374                            BNE     2$
1579  002672  012701  000003            3$:     MOV     #3,R1                   ;SET SHIFT COUNTER
1580  002676  116324  001132                    MOVB    DIGTAB(R3),(R4)+        ;MOVE ASCII EQUIV TO OUTPUT
1581  002702  005003                            CLR     R3
1582  002704  005300                            DEC     R0                      ;DECREMENT DIGIT COUNT
1583  002706  001365                            BNE     2$                      ;GET NEXT DIGIT
1584  002710  105737  001126                    TSTB    @#TYPFLG                ;BRANCH IF ASCII IS
1585  002714  001002                            BNE     4$                      ;NOT TO BE TYPED
1586  002716  000004  001144                    TYPE,ODIGITS
1587  002722                            4$:
1588  002722  004767  177652                    JSR     PC,.RESTORE             ;RESTORE REGISTERS FROM THE STACK
1589  002726  000207                            RTS     PC
1590
1591
1592                                    ;SUBROUTINE TO CONVERT OCTAL DATA TO DECIMAL ASCII
1593                                    ;CALL:  MOV     NUMBER,R2               ;MOVE NUMBER TO R2
1594                                    ;       JSR     PC,CNVDEC
1595
1596  002730  110637  001126            CNVDEC: MOVB    SP,@#TYPFLG             ;SET DO NOT TYPE FLAG
1597  002734  000402                            BR      CNVTD
1598                                            .SBTTL          OCTAL TO DECIMAL & TYPE ROUTINE
1599                                    ;THIS ROUTINE CONVERTS AN OCTAL # TO DECIMAL ASCII AND TYPES IT OUT
1600                                    ;CALL:  MOV     NUMBER,R2               ;PUT # IN R2
1601                                    ;       JSR     PC,TYPDEC               ;CALL ROUTINE
1602
1603  002736  105037  001126            TYPDEC: CLRB    @#TYPFLG                ;SET TYPE FLAG
1604  002742                            CNVTD:
1605  002742  004767  177610                    JSR     PC,.SAVE                ;SAVE REGISTERS ON THE STACK
1606  002746  005000                            CLR     R0                      ;R0 IS INDEX TO DECIMAL CONSTANT
1607  002750  012704  001144                    MOV     #ODIGITS,R4             ;SET OUTPUT PTR
1608  002754  005003                    1$:     CLR     R3                      ;R3 CONTAINS DECIMAL DIGIT
```

```
1609  002756  166002  003036          2$:     SUB     DCONST(R0),R2           ;SUBTRACT DECIMAL CONSTANT UNTIL
1610  002762  103402                          BLO     3$                      ;INPUT # GOES NEGATIVE
1611  002764  005203                          INC     R3                      ;KEEPING TRACK OF SUBTRACTIONS
1612  002766  000773                          BR      2$
1613  002770  066002  003036          3$:     ADD     DCONST(R0),R2           ;ADD BACK CONSTANT WHEN NEGATIVE
1614  002774  116324  001132                  MOVB    DIGTAB(R3),(R4)+        ;MOVE ASCII EQUIVILENT
1615  003000  062700  000002                  ADD     #2,R0                   ;NEXT CONSTANT
1616  003004  005760  003036                  TST     DCONST(R0)              ;UNTIL ALL CONSTANTS DONE
1617  003010  001361                          BNE     1$
1618  003012  112724  000060                  MOVB    #'0,(R4)+               ;LAST DIGIT IS 0
1619  003016  105737  001126                  TSTB    @#TYPFLG                ;BRANCH IF ASCII IS
1620  003022  001002                          BNE     4$                      ;NOT TO BE TYPED
1621  003024  000004  001144                  TYPE,ODIGITS
1622  003030                          4$:
1623  003030  004767  177544                  JSR     PC,.RESTORE             ;RESTORE REGISTERS FROM THE STACK
1624  003034  000207                          RTS     PC
1625
1626  003036  023420          DCONST: .WORD   10000.
1627  003040  001750                  .WORD   1000.
1628  003042  000144                  .WORD   100.
1629  003044  000012                  .WORD   10.
1630  003046  000001                  .WORD   1.
1631  003050  000000                  .WORD   0                               ;TERMINATOR
1632
1633                                  .SBTTL          TYPE SPECIFIED TIMES ROUTINE
1634                          ;THIS SUBROUTINE OUTPUTS THE TIME SPECIFICATIONS FOR THE TEST
1635                          ;AND ALSO THE ACTUAL TIME RECORDED (ATIME)
1636                          ;FORMAT OF LINE TYPED
1637                          ;RANGE=<AAAAAA-BBBBBB>          ACTUAL=CCCCCC
1638                          ;WHERE:         AAAAAA IS MAXIMUM TIME FOR TEST (STIMTBL(TSTNUMX4)).
1639                          ;               BBBBBB IS MINIMUM TIME FOR TEST (STIMTBL(TSTNUMX4+2)).
1640                          ;               CCCCCC IS ACTUAL TIME RECORDED BY TEST (ATIME).
1641                          ;CALL:  MOVB    TEST NUMBER,R2  ;LOAD TEST NUMBER
1642                          ;       MOV     #TIME,@#ATIME           ;MOVE TIME TO ATIME
1643                          ;       JSR     PC,OUTSPC
1644  003052  010246          OUTSPC: MOV     R2,-(SP)                ;SAVE R2 & R3 ON THE STACK
1645  003054  010346                  MOV     R3,-(SP)
1646  003056  006302                  ASL     R2                      ;MULTIPLY TEST # TIMES 4
1647  003060  006302                  ASL     R2                      ;TO FORM INDEX INTO STIMTBL
1648  003062  010203                  MOV     R2,R3                   ;R3 CONTAINS INDEX INTO TABLE
1649  003064  000004  014645          TYPE,L.RNG
1650  003070  016302  001416          MOV     STIMTBL(R3),R2          ;GET MAXIMUM SPEC TIME
1651  003074  004767  177636          JSR     PC,TYPDEC               ;CONVERT TO DECIMAL & TYPE
1652  003100  000004  001405          TYPE,DASH
1653  003104  016302  001420          MOV     STIMTBL+2(R3),R2        ;GET MINIMUM TIME
1654  003110  004767  177622          JSR     PC,TYPDEC               ;CONVERT TO DECIMAL & TYPE
1655  003114  000004  001412          TYPE,ANGTAB
1656  003120  000004  014655          TYPE,L.ACT
1657  003124  013702  001012          MOV     @#ATIME,R2              ;GET ACTUAL TIME
1658  003130  004767  177602          JSR     PC,TYPDEC               ;CONVERT TO DECIMAL & TYPE
1659  003134  000004  001374          TYPE,CRLF
1660  003140  012603                  MOV     (SP)+,R3
1661  003142  012602                  MOV     (SP)+,R2
1662  003144  000207                  RTS     PC                      ;RETURN
1663
1664                                  .SBTTL          TYPE GAP TIMES SUBROUTINE
```

```
                                ;THIS SUBROUTINE IS USED TO TYPE THE SPECIFIED GAP SIZES (RECORDED IN
                                ;TST021). IT IS CALLED BY THE GAPOK ROUTINE IF THE GAP SIZE IS OUT OF
                                ;RANGE VIA THE HLT ROUTINE (HLT+2).
                                ;CALL:  MOVB    #GAP,GAP            ;LOAD GAP # INTO GAP
                                ;       MOV     #TIME,ATIME         ;LOAD ACTUAL TIME INTO ATIME
                                ;       JSR     PC,OUTGAP
                                
003146  010246                  OUTGAP: MOV     R2,-(SP)            ;SAVE R2 AND R3
003150  010346                          MOV     R3,-(SP)
003152  116703  175742                  MOVB    GAP,R3              ;GET GAP #
003156  006303                          ASL     R3
003160  006303                          ASL     R3
003162  000004  014645                  TYPE,L.RNG
003166  016302  001572                  MOV     GTIMTBL(R3),R2      ;GET MAX TIME
003172  004767  177540                  JSR     PC,TYPDEC           ;CONVERT TO DECIMAL & TYPE
003176  000004  001405                  TYPE,DASH
003202  016302  001574                  MOV     GTIMTBL+2(R3),R2    ;GET MIN TIME
003206  004767  177524                  JSR     PC,TYPDEC           ;CONVERT TO DECIMAL & TYPE
003212  000004  001412                  TYPE,ANGTAB                 ;TYPE <
003216  000004  014655                  TYPE,L.ACT
003222  013702  001012                  MOV     @#ATIME,R2          ;GET ACTUAL TIME
003226  004767  177504                  JSR     PC,TYPDEC           ;CONVERT TO DECIMAL & TYPE
003232  000004  014334                  TYPE,E.GAP
003236  113702  001120                  MOVB    @#GAP,R2            ;GET GAP #
003242  004767  177362                  JSR     PC,TYPOCT           ;TYPE GAP #
003246  000004  001374                  TYPE,CRLF
003252  012603                          MOV     (SP)+,R3            ;RESTORE R3 AND R2
003254  012602                          MOV     (SP)+,R2
003256  000207                          RTS     PC
                                
                                        .SBTTL          ASCII TO OCTAL CONVERT SUBROUTINE
                                ;SUBROUTINE TO CONVERT ASCII DATA TO OCTAL. CONVERTED OCTAL DATA
                                ;IS LEFT IN OCTALO <15-00>.
003260                          CNVTAO:
003260  004767  177272                  JSR     PC,.SAVE            ;SAVE REGISTERS ON THE STACK
003264  012700  001264                  MOV     #INBUF,R0           ;SET PTR TO ASCII DATA
003270  012701  001116                  MOV     #OCTALO,R1          ;GET ADDRESS OF OCTAL DATA
003274  005011                          CLR     (R1)                ;CLEAR OUT OLD OCTAL DATA
003276  005061  000002                  CLR     2(R1)
003302  122710  000015          1$:     CMPB    #CR,(R0)            ;<CR> TERMINATES INPUT
003306  001414                          BEQ     3$
003310  112002                          MOVB    (R0)+,R2            ;GET 'OCTAL' DATA
003312  042702  177770                  BIC     #177770,R2          ;STRIP UNUSED BITS
003316  012703  000003                  MOV     #3,R3               ;SET SHIFT COUNT
003322  006311                  2$:     ASL     (R1)                ;SHIFT LAST
003324  006161  000002                  ROL     2(R1)               ;OCTAL DIGIT
003330  005303                          DEC     R3
003332  001373                          BNE     2$
003334  050211                          BIS     R2,(R1)             ;AND INSERT THIS DIGIT
003336  000761                          BR      1$                  ;GO GET NEXT DIGIT
003340                          3$:
003340  004767  177234                  JSR     PC,.RESTORE         ;RESTORE REGISTERS FROM THE STACK
003344  000207                          RTS     PC                  ;RETURN
                                
                                        .SBTTL          PUBLISH SUBROUTINE
                                ;THE PUBLISH SUBROUTINE AVERAGES THE RECORDED TIMES FOR EACH TEST IT-
```

```
1721                                    ;ERATION (IF 16. ITERATIONS) AND PLACES THE AVERAGE RESULT IN 'ATIME'.
1722                                    ;IT TYPES THE NAME OF THE FUNCTION THAT WAS TIMED,THE TIME SPEC-
1723                                    ;IFICATION AND THE ACTUAL TIME .
1724
1725  003346                            PUBLISH:
1726  003346  004767  177204                  JSR     PC,.SAVE                ;SAVE REGISTERS ON THE STACK
1727  003352  012700  001014                  MOV     #ATIMTBL,R0     ;GET TABLE ADDRESS CONTAINING TIMES
1728  003356  113701  001121                  MOVB    @#ITCNT,R1      ;GET # OF ENTRIES (GIVEN BY ITERATION COUNT)
1729  003362  122701  000001                  CMPB    #1,R1                   ;BRANCH IF SINGLE ITERATION
1730  003366  001423                          BEQ     4$
1731  003370  005002                          CLR     R2                      ;CLEAR 'SUM' REGISTERS
1732  003372  005003                          CLR     R3
1733  003374  122701  000020                  CMPB    #16.,R1                 ;BRANCH IF 16. ITERATIONS
1734  003400  001402                          BEQ     1$
1735  003402  000000                          HALT                            ;ITERATION COUNT MUST BE 1 OR 16.
1736  003404  000777                          BR      .                       ;DO NOT CHANGE POSIT OF SW11
1737                                                                          ;WHEN TEST IS RUNNING.
1738
1739  003406  062002                    1$:   ADD     (R0)+,R2                ;SUM INDIVIDUAL TIMES
1740  003410  005503                          ADC     R3
1741  003412  005301                          DEC     R1
1742  003414  001374                          BNE     1$
1743
1744  003416  012700  000004           2$:   MOV     #4,R0
1745  003422  006203                    3$:   ASR     R3                      ;SHIFT TIME IN R3 & R2 4 PLACES
1746  003424  006002                          ROR     R2                      ;RIGHT = DIVIDE BY 16.
1747  003426  005300                          DEC     R0
1748  003430  001374                          BNE     3$
1749  003432  010237  001012                  MOV     R2,@#ATIME              ;MOVE AVERAGED TIMES
1750
1751  003436  113700  001122           4$:   MOVB    @#TSTNUM,R0             ;GET TEST #
1752  003442  006300                          ASL     R0
1753  003444  016067  001672  000002          MOV     NAMPTR(R0),5$           ;GET TEST NAME STRING ADDRESS
1754  003452  000004                          TYPE
1755  003454  000000                    5$:   .WORD   0
1756  003456  113702  001122                  MOVB    @#TSTNUM,R2             ;GET TEST #
1757  003462  004767  177364                  JSR     PC,OUTSPC               ;OUTPUT TIMES
1758  003466  004767  177106                  JSR     PC,.RESTORE             ;RESTORE REGISTERS FROM THE STACK
1759  003472  000207                          RTS     PC
1760
1761                                          .SBTTL          INPUT SUBROUTINE
1762                                    ;SUBROUTINE TO GET TTY INPUT
1763                                    ;CALL:  JSR     PC,.INPUT
1764                                    ;INPUT DATA IS RETURNED IN BUFFER BEGINNING AT INBUF.
1765
1766  003474  010046                    .INPUT: MOV     R0,-(SP)        ;SAVE R0 ON THE STACK
1767  003476  012700  001264           1$:     MOV     #INBUF,R0
1768  003502  105737  177560           2$:     TSTB    @#TKS
1769  003506  100375                          BPL     2$
1770
1771  003510  113746  177562                  MOVB    @#TKB,-(SP)     ;GET CHARACTER
1772  003514  042716  000200                  BIC     #200,(SP)
1773  003520  122716  000177                  CMPB    #177,(SP)       ;CHECK RUBOUT
1774  003524  001004                          BNE     3$
1775  003526  124026                          CMPB    -(R0),(SP)+     ;REMOVE CHARACTER FROM INPUT
1776  003530  000004  001377                  TYPE,BKSLSH
```

```
 1777  003534  000762                        BR      2$              ;WAIT FOR NEXT CHARACTER
 1778  003536  122716  000025        3$:     CMPB    #CNTRLU,(SP)    ;CHECK CONTROL U (↑U)
 1779  003542  001004                        BNE     4$
 1780  003544  005726                        TST     (SP)+
 1781  003546  000004  001374                TYPE,CRLF
 1782  003552  000751                        BR      1$
 1783  003554  111637  001401        4$:     MOVB    (SP),@#ECHO
 1784  003560  111620                        MOVB    (SP),(R0)+
 1785  003562  122726  000015                CMPB    #CR,(SP)+
 1786  003566  001403                        BEQ     5$
 1787  003570  000004  001401                TYPE,ECHO
 1788  003574  000742                        BR      2$
 1789  003576  000004  001374        5$:     TYPE,CRLF
 1790  003602  012600                        MOV     (SP)+,R0
 1791  003604  000207                        RTS     PC
 1792
 1793                                  ;KEYBOARD INTERRUPT SERVICE ROUTINE
 1794  003606  113746  177562        TKISR:  MOVB    @#TKB,-(SP)     ;GET TYPED CHARACTER
 1795  003612  042716  000200                BIC     #200,(SP)       ;STRIP PARITY BIT
 1796  003616  122716  000017                CMPB    #CNTRLO,(SP)    ;BRANCH IF NOT CONTROL O (↑O)
 1797  003622  001003                        BNE     1$
 1798  003624  112667  176475                MOVB    (SP)+,$CNTRLO   ;SET CONTROL O INDICATOR IN TYPE ROUTINE
 1799  003630  000002                        RTI                     ;EXIT
 1800
 1801  003632  122726  000003        1$:     CMPB    #3,(SP)+        ;BRANCH IF NOT CONTROL C (↑C)
 1802  003636  001003                        BNE     2$
 1803  003640  000005                        RESET
 1804  003642  000137  005724                JMP     @#INIT          ;RESTART PROGRAM
 1805  003646  000002                2$:     RTI                     ;EXIT
```

```
                                        .SBTTL          ERROR SERVICE ROUTINES
                                ;ROUTINE TO PROCESS ERROR TRAPS (TRAPS TO 4)
003650  000000                  ERRTRP: HALT

                                ;ERROR SERVICE ROUTINE
                                ;THIS ROUTINE PROCESSES TWO TYPES OF ERRORS (OUT OF RANGE AND HARDWARE)
                                ;THE CALLS FOR AN OUT OF RANGE ERROR ARE <HLT+1>,<HLT+2> AND, FOR A
                                ;HARDWARE ERROR THE CALL IS <HLT>.

003652  004767  176112          .HLT:   JSR     PC,CKSWR                ;CHECK FOR CNTL G
003656  004767  176674                  JSR     PC,.SAVE                ;SAVE REGISTERS ON THE STACK
003662  110637  001123          1$:     MOVB    SP,@#ERFLG              ;SET ERROR FLAG
003666  032777  020000  175104          BIT     #SW13,@SWR              ;BRANCH IF NO TYPOUT
003674  001075                          BNE     4$
003676  000004  014135                  TYPE,E.HDR
003702  113702  001122                  MOVB    @#TSTNUM,R2             ;GET TEST #
003706  004767  176716                  JSR     PC,TYPOCT               ;AND TYPE IT
003712  016600  000016                  MOV     16(SP),R0               ;GET RETURN PC
003716  162700  000002                  SUB     #2,R0                   ;NOW PC OF HLT CALL
003722  111000                          MOVB    (R0),R0                 ;NOW HLT CALL ITSELF
003724  001417                          BEQ     2$                      ;BRANCH IF HLT
003726  000004  014220                  TYPE,E.HDR2
003732  122700  000002                  CMPB    #2,R0                   ;BRANCH IF NOT HLT+2
003736  001005                          BNE     10$
003740  004767  177202                  JSR     PC,OUTGAP               ;TYPE GAP SPECIFIED TIMES
003744  000004  001374                  TYPE,CRLF
003750  000447                          BR      4$
003752  004767  177074          10$:    JSR     PC,OUTSPC               ;TYPE SPECIFIED TIMES
003756  000004  001374                  TYPE,CRLF
003762  000442                          BR      4$
003764  016500  000014          2$:     MOV     ER(R5),R0
003770  032765  002000  000032          BIT     #PE1600,TC(R5)
003776  001403                          BEQ     20$
004000  042700  102100                  BIC     #102100,R0
004004  000402                          BR      21$
004006  042700  102300          20$:    BIC     #102300,R0
004012  005700                  21$:    TST     R0
004014  001003                          BNE     22$
004016  000004  014111                  TYPE,E.SFT              ;TYPE SOFT ERROR MESSAGE
004022  000434                          BR      6$

004024  000004  014145          22$:    TYPE,E.HDR1
004030  010500                          MOV     R5,R0                   ;GET FIRST ADDRESS OF REGS.
004032  012701  000007                  MOV     #7.,R1                  ;TYPE FIRST 7 REGS.
004036  012002                  3$:     MOV     (R0)+,R2                ;GET REG CONTENTS
004040  004767  176564                  JSR     PC,TYPOCT               ;AND TYPE IT
004044  000004  001407                  TYPE,SPACE2
004050  005301                          DEC     R1
004052  001371                          BNE     3$
004054  016502  000032                  MOV     TC(R5),R2               ;GET CONTENTS OF TC REGISTER
004060  004767  176544                  JSR     PC,TYPOCT
004064  000004  001374                  TYPE,CRLF

004070  032777  001000  174702  4$:     BIT     #SW09,@SWR              ;BRANCH IF NO RING THE BELL
004076  001402                          BEQ     5$
004100  000004  001403                  TYPE,BELL
```

ERROR SERVICE ROUTINES

```
1862  004104  005777  174670          5$:     TST     @SWR                    ;HALT ON ERROR?
1863  004110  100001                          BPL     6$
1864  004112  000000                          HALT
1865  004114  004767  175650          6$:     JSR     PC,CKSWR        ;CHECK FOR CNTL G
1866  004120  004767  176454                  JSR     PC,.RESTORE             ;RESTORE REGISTERS FROM THE STACK
1867  004124  000002                          RTI                             ;RETURN
1868
1869
```

```
                                        .SBTTL          SCOPE SUBROUTINE
                                ;SCOPE ROUTINE
                                ;THIS ROUTINE IS ENTERED UPON COMPLETION OF EACH SUBTEST
                                ;THE SCOPE ROUTINE:
                                ;       OUTPUTS TIME SPEC ON EACH ITERATION IF SW08 IS SET
                                ;       REPEATS TEST IF SW14 IS SET
                                ;       STORES ACTUAL TIME FOR FUNCTION IN TIME TABLE (ATIMTBL)
                                ;       PUBLISHES TIME IF SW10=0
                                ;       UPDATES ITERATION COUNT AND IF ITERATIONS COMPLETE CONTINUES
                                ;       TO NEXT TEST, OTHERWISE REPEATS TEST.
                                ;       DELAYS BEFORE CONTINUING OR REPEATING TEST.
                                ;       INITIALIZES DRIVE
                                ;RETURNS:       R5=BASE ADDRESS OF TM02 REGISTERS (ADDRESS OF CS1)
                                ;               R1='DS' REG ADDRESS
                                ;               R0='FC' REG ADDRESS
                                ;
004126  004767  175636          .SCOPE: JSR     PC,CKSWR                ;CHECK FOR CNTL G
004132  013705  001010                  MOV     @#TMBASE,R5             ;SET R5 TO FIRST TM REG
004136  032777  000400  174634          BIT     #SW08,@SWR              ;BRANCH IF SPECIFICATION LINE
004144  001404                          BEQ     10$                     ;NOT DESIRED ON EACH ITERATION
004146  113702  001122                  MOVB    @#TSTNUM,R2             ;GET TEST NUMBER
004152  004767  176674                  JSR     PC,OUTSPC               ;OUTPUT TIME  RECORDED
004156  032777  040000  174614  10$:    BIT     #SW14,@SWR              ;BRANCH IF CONTINUOS LOOP
004164  001417                          BEQ     2$                      ;NOT DESIRED
004166  004767  000546          1$:     JSR     PC,DELAY                ;DELAY 350 MS
004172  004767  000776                  JSR     PC,RHINIT               ;INIT
004176  105037  001123                  CLRB    @#ERFLG                 ;CLEAR ERROR FLAG
004202  016716  174574                  MOV     SCPADR,(SP)
004206  010501                          MOV     R5,R1
004210  062701  000012                  ADD     #DS,R1                  ;ADDRESS OF 'DS' REG IS IN R1
004214  010500                          MOV     R5,R0
004216  062700  000006                  ADD     #FC,R0                  ;ADDRESS OF 'FC' REG IS IN R0
004222  000002                          RTI
                                ;
004224  105737  001123          2$:     TSTB    @#ERFLG                 ;BRANCH IF ERROR FLAG IS SET
004230  001006                          BNE     3$
004232  113700  001121                  MOVB    @#ITCNT,R0              ;GET ITERATION COUNT
004236  006300                          ASL     R0                      ;STORE TIME IN TABLE
004240  013760  001012  001014          MOV     @#ATIME,ATIMTBL(R0)
004246  105237  001121          3$:     INCB    @#ITCNT                 ;INCREMENT ITERATION COUNT
004252  032777  004000  174520          BIT     #SW11,@SWR              ;BRANCH IF SINGLE ITERATION DESIRED
004260  001004                          BNE     4$
004262  122737  000020  001121          CMPB    #16.,@#ITCNT            ;BRANCH IF ITERATIONS INCOMPLETE
004270  001336                          BNE     1$
004272  011637  001002          4$:     MOV     (SP),@#SCPADR           ;SET SCOPE ADDRESS TO NEXT TEST
004276  032777  002000  174474          BIT     #SW10,@SWR              ;BRANCH IF NO PUBLICATION DESIRED
004304  001002                          BNE     5$
004306  004767  177034                  JSR     PC,PUBLISH              ;GO PUBLISH TEST DATA
004312  105037  001121          5$:     CLRB    @#ITCNT                 ;RESET ITERATION COUNT
004316  105777  174456                  TSTB    @SWR                    ;BRANCH IF USER DOES NOT WANT TO
004322  001721                          BEQ     1$                      ;HALT ON A SELECTED TEST
004324  017746  174450                  MOV     @SWR,-(SP)              ;GET SWITCHES
004330  042716  177740                  BIC     #177740,(SP)            ;CLEAR ALL BUT TEST #
004334  005316                          DEC     (SP)                    ;FORM TEST # -1
004336  123726  001122                  CMPB    @#TSTNUM,(SP)+          ;BRANCH IF NOT AT TEST
004342  001311                          BNE     1$
```

```
                                        SCOPE SUBROUTINE

1926   004344  000000                   HALT
1927   004346  004767  175416           JSR     PC,CKSWR                ;CHECK FOR CNTL G
1928   004352  000705                   BR      1$
1929
1930                                    .SBTTL  TIMER SUBROUTINES
1931
1932                                    ;SUBROUTINE TO SYNCHRONIZE THE TIMER AND TURN IT ON.
1933                                    ;REGISTER 4 IS CLEARED, AND THE OSCILLATOR POLARITY IS MONITORED
1934                                    ;THE ROUTINE IS EXITED WHEN THE OSCILLATOR POLARITY CHANGES WITH R3
1935                                    ;SET TO INDICATE THE POLARITY OF THE OSCILLATOR.
1936                                    ;CALL:  JSR     PC,TIMON
1937                                    ;RETURNS:       R3 SET TO INDICATE LAST POLARITY (+24/-24=0/1)
1938                                    ;               R4 = 0
1939
1940   004354  005004           TIMON:  CLR     R4                      ;CLEAR TIME COUNT
1941   004356  012703  000024           MOV     #24,R3                  ;SET POLARITY TO '0' STATE
1942   004362  032765  000100  000024   BIT     #OSC,MR(R5)             ;BRANCH IF POLARITY IS '0'
1943   004370  001405                   BEQ     2$
1944   004372  032765  000100  000024   1$:     BIT     #OSC,MR(R5)             ;WAIT FOR OSCILLATOR TO RETURN
1945   004400  001374                   BNE     1$
1946   004402  000405                   BR      4$
1947
1948   004404  005403           2$:     NEG     R3                      ;NEGATE PREV POLARITY INDICATOR
1949   004406  032765  000100  000024   3$:     BIT     #OSC,MR(R5)             ;WAIT FOR OSCILLATOR TO RETURN
1950   004414  001774                   BEQ     3$                      ;TO '1' STATE
1951   004416  000207           4$:     RTS     PC
1952
1953                                    ;SUBROUTINE TO COUNT TIME
1954                                    ;EACH TIME THE OSCILLATOR TOGGLES (BIT <06> IN MR REG) REGISTER
1955                                    ;R4 IS INCREMENTED, AND THE REGISTER R3 IS NEGATED TO INDICATE
1956                                    ;THE LAST STATE OF THE OSCILLATOR.
1957                                    ;CALL   JMP     TIMER(R3)               ;R3 IS SET BY TIMON ROUTINE
1958                                    ;       R2=RETURN ADDRESS TO CALLER
1959                                    ;NOTE: THE TIME TO EXECUTE THIS ROUTINE IS VERY CRITICAL. IT MUST BE
1960                                    ;LESS THAN 40 US.
1961
1962                                    ;ENTER HERE VIA JMP  TIMER(R3) WHEN R3=-24 (PREV STATE=1)
1963   004420  032765  000100  000024   TIMER1: BIT     #OSC,MR(R5)             ;BRANCH IF CURRENT STATE IS '0'
1964   004426  001406                   BEQ     TIMER                   ;GO INCREMENT TIME
1965   004430  000112                   JMP     (R2)                    ;RETURN TO TEST
1966
1967           004444                   .=TIMER1+24
1968   004444  005403           TIMER:  NEG     R3                      ;NEGATE PREV STATE INDICATOR
1969   004446  005204                   INC     R4                      ;INCREMENT 'TICK' COUNT
1970   004450  100401                   BMI     TIMERR                  ;BRANCH ON OVERFLOW
1971   004452  000112                   JMP     (R2)                    ;RETURN TO TEST
1972   004454  000004  014246   TIMERR: TYPE,E.TIMOV                     ;TYPE 'TIMER OVERFLOWED'
1973   004460  104400                   HLT                             ;REPORT HARDWARE ERROR
1974   004462  000177  174314           JMP     @SCPADR                 ;RETURN TO BEGINNING OF TEST
1975
1976           004470                   .=TIMER+24
1977                                    ;ENTER HERE VIA JMP  TIMER(R3) WHEN R3=+24 (PREV STATE=0)
1978   004470  032765  000100  000024   TIMER0: BIT     #OSC,MR(R5)             ;BRANCH IF CURRENT STATE = '1'
1979   004476  001362                   BNE     TIMER
1980   004500  000112                   JMP     (R2)
1981
```

```
                                        ;SUBROUTINE TO CHECK TIME RECORDED BY SUBTEST.
                                        ;THIS SUBROUTINE COMPUTES THE ACTUAL TIME (IN MICROSECONDS) AND CHECKS
                                        ;THAT THE TIME RECORDED BY THE SUBTEST IS CORRECT BY COMPARING THE TIME
                                        ;WITH THE HIGH LIMIT (STIMTBL(R0)) AND THE LOW LIMIT (STIMTBL+2(R0)).
                                        ;IF THE TIME IS OUT OF RANGE AN OUT OF RANGE ERROR TYPEOUT RESULTS.
                                        ;THE SUBROUTINE IS ENTERED WITH:
                                        ;       R4=TICK COUNT
                                        ;
004502                                  TIMOK:
004502  004767  176050                          JSR     PC,.SAVE                ;SAVE REGISTERS ON THE STACK
004506  012700  000070                          MOV     #56.,R0                 ;GET TIME PER TICK
004512  010401                                  MOV     R4,R1                   ;GET TICKS COUNT
004514  005002                                  CLR     R2                      ;CLEAR SUMMING REGISTERS
004516  005003                                  CLR     R3
004520  060002                          1$:     ADD     R0,R2                   ;MULTIPLY TIME PER TICK
004522  005503                                  ADC     R3                      ;BY TICK COUNT
004524  005301                                  DEC     R1
004526  001374                                  BNE     1$
004530  010246                                  MOV     R2,-(SP)                ;DIVIDE COUNT BY 10.
                                        ;
004532  010346                                  MOV     R3,-(SP)
004534  012746  000012                          MOV     #10.,-(SP)
004540  004767  000262                          JSR     PC,DIVIDE
004544  005726                                  TST     (SP)+                   ;DISCARD REMAINDER
004546  012637  001012                          MOV     (SP)+,@#ATIME           ;STORE QUOTIENT
004552  113700  001122                          MOVB    @#TSTNUM,R0             ;GET TEST #
004556  006300                                  ASL     R0
004560  006300                                  ASL     R0
004562  023760  001012  001416                  CMP     @#ATIME,STIMTBL(R0)     ;CHECK THAT TIME IS WITHIN
004570  101004                                  BHI     2$                      ;LIMITS SPECIFIED
004572  023760  001012  001420                  CMP     @#ATIME,STIMTBL+2(R0)
004600  101001                                  BHI     3$
004602  104401                          2$:     HLT+1                           ;CALL ERROR ROUTINE
004604                                  3$:
004604  004767  175770                          JSR     PC,.RESTORE             ;RESTORE REGISTERS FROM THE STACK
004610  000207                                  RTS     PC                      ;RETURN
                                        ;
                                        ;SUBROUTINE TO CHECK INDIVIDUAL GAP TIMES (PRODUCED BY TST021)
                                        ;SUBROUTINE COMPUTES THE ACTUAL TIME (IN MICROSECONDS) AND CHECKS
                                        ;THAT THE GAP TIME RECORDED BY THE SUBTEST (TST021) BY COMPARING THE
                                        ;TIME WITH THE MAX LIMIT (GTIMTBL-GAPTBL(R1)) AND THE MIN LIMIT
                                        ;(GTIMTBL+2-GAPTBL(R1)).
                                        ;CALL:  MOV     #TICK COUNT,R4          ;R4 CONTAINS TICK COUNT
                                        ;       MOVB    #GAP,@#GAP              ;LOCATION GAP CONTAINS GAP #
                                        ;       JSR     PC,GAPOK
                                        ;
004612                                  GAPOK:
004612  004767  175740                          JSR     PC,.SAVE                ;SAVE REGISTERS ON THE STACK
004616  012700  000070                          MOV     #56.,R0                 ;GET TIME PER TICK
004622  010401                                  MOV     R4,R1                   ;GET TICK COUNT
004624  005002                                  CLR     R2                      ;CLEAR SUMMING REGISTERS
004626  005003                                  CLR     R3
004630  060002                          1$:     ADD     R0,R2                   ;MULTIPLY TICK COUNT
004632  005503                                  ADC     R3                      ;BY TIME PER TICK
004634  005301                                  DEC     R1
004636  001374                                  BNE     1$
```

```
2038
2039  004640  010246                    MOV     R2,-(SP)                ;DIVIDE TIME BY 10.
2040  004642  010346                    MOV     R3,-(SP)
2041  004644  012746  000012            MOV     #10.,-(SP)
2042  004650  004767  000152            JSR     PC,DIVIDE
2043  004654  005726                    TST     (SP)+                   ;DISCARD REMAINDER
2044  004656  012637  001012            MOV     (SP)+,@#ATIME           ;STORE QUOTIENT
2045  004662  113703  001120            MOVB    @#GAP,R3                ;GET GAP #
2046  004666  006303                    ASL     R3                      ;MULTPLY BY 4
2047  004670  006303                    ASL     R3                      ;TO GET AT TABLE ENTRY
2048  004672  023763  001012  001572    CMP     @#ATIME,GTIMTBL(R3)     ;CHECK TIME (MAX)
2049  004700  101004                    BHI     2$
2050  004702  023763  001012  001574    CMP     @#ATIME,GTIMTBL+2(R3)   ;CHECK TIME (MIN)
2051  004710  101002                    BHI     3$
2052  004712  104402            2$:     HLT+2                           ;REPORT OUT OF RANGE ERROR
2053  004714  000406                    BR      100$
2054  004716  032777  000400  174054  3$:     BIT     #SW08,@SWR              ;BRANCH IF TIMES NOT WANTED
2055  004724  001402                    BEQ     100$
2056  004726  004767  176214            JSR     PC,OUTGAP               ;TYPE GAP TIMES
2057
2058  004732                    100$:
2059  004732  004767  175642            JSR     PC,.RESTORE             ;RESTORE REGISTERS FROM THE STACK
2060  004736  000207                    RTS     PC                              ;RETURN TO TEST
2061
2062                                    .SBTTL          DELAY SUBROUTINES
2063                            ;THIS SUBROUTINE CAUSES A DELAY OF 350 MS.
2064  004740  004767  177410  DELAY:   JSR     PC,TIMON
2065  004744  010246                    MOV     R2,-(SP)                ;SAVE R2 ON THE STACK
2066  004746  012702  004756            MOV     #2$,R2                  ;SET RETURN ADDRESS FOR TIMER
2067  004752                    1$:
2068  004752  000163  004444            JMP     TIMER(R3)               ;GO TO TIMER & RETURN VIA R2
2069  004756  032704  004000    2$:     BIT     #4000,R4
2070  004762  001773                    BEQ     1$
2071  004764  012602                    MOV     (SP)+,R2                ;RESTORE R2
2072  004766  000207                    RTS     PC
2073
2074                            ;THIS SUBROUTINE ALLOWS A CALLER SPECIFIED DELAY (UP TO 65MS.)
2075                            ;CALL:  MOV     DELAY TIME,DELTIM       ;LOAD DELAY TIME (IN US)
2076                            ;       JSR     PC,DELAYV
2077  004770  005767  174120  DELAYV: TST     DELTIM                  ;BRANCH IF 0 DELAY
2078  004774  001413                    BEQ     3$
2079  004776  004767  177352            JSR     PC,TIMON                ;TURN TIMER ON
2080  005002  010246                    MOV     R2,-(SP)                ;SAVE R2 ON THE STACK
2081  005004  012702  005014            MOV     #2$,R2                  ;SET RETURN ADDRESS FROM TIMER
2082  005010                    1$:
2083  005010  000163  004444            JMP     TIMER(R3)               ;GO TO TIMER & RETURN VIA R2
2084  005014  023704  001114    2$:     CMP     @#DELTIM,R4
2085  005020  101373                    BHI     1$
2086  005022  012602                    MOV     (SP)+,R2                ;RESTORE R2
2087  005024  000207            3$:     RTS     PC
2088
2089                                    .SBTTL  DIVIDE SUBROUTINE
2090                            ;THIS SUBROUTINE DIVIDES A DOUBLE PRECISION # AND RETURNS THE RESULT
2091                            ;TO THE CALLER ON THE STACK. BOTH DIVIDEND & DIVISOR MUST BE POSITIVE.
2092                            ;CALL:  MOV     LEAST SIGNIFICANT HALF DIVIDEND,-(SP)
2093                            ;       MOV     #MOST SIGNIFICANT HALF DIVIDEND,-(SP)
```

```
2094                                    ;       MOV     #DIVISOR,-(SP)
2095                                    ;       JSR     PC,DIVIDE
2096                                    ;RETURN
2097                                    ;       (SP)=REMAINDER ON STACK
2098                                    ;       2(SP)=QUOTIENT
2099
2100                                    ;NOTE:  THIS SUBROUTINE DESTROYS PREVIOUS CONTENTS OF R0,R1,R2 & R3.
2101
2102  005026  005046                   DIVIDE: CLR     -(SP)                   ;SAVE LOC FOR SIGNS
2103  005030  012746  000021                   MOV     #17.,-(SP)              ;SET ITERATION COUNT
2104  005034  016601  000012                   MOV     12(SP),R1               ;GET LSH DIVIDEND
2105  005040  016600  000010                   MOV     10(SP),R0               ;GET MSH DIVIDEND
2106  005044  016602  000006                   MOV     6(SP),R2                ;GET DIVISOR
2107  005050  005402                           NEG     R2                      ;NEGATE DIVISOR
2108  005052  000241                           CLC                             ;CLEAR 'C' BIT IN PSW
2109  005054  000405                           BR      2$
2110  005056  006100                   1$:     ROL     R0                      ;ROTATE MSH DIVIDEND
2111  005060  010003                           MOV     R0,R3                   ;SAVE IN R3
2112  005062  060203                           ADD     R2,R3                   ;SUBTRACT DIVISOR FROM MSH DIVIDEND
2113  005064  103001                           BCC     2$                      ;BRANCH IF DIVIDEND > DIVISOR
2114  005066  010300                           MOV     R3,R0                   ;SAVE REMAINDER IN R0
2115  005070  006101                   2$:     ROL     R1                      ;ROTATE LSH DIVIDEND
2116  005072  005316                           DEC     (SP)                    ;DECREMENT ITERATION COUNT
2117  005074  001370                           BNE     1$
2118  005076  005726                           TST     (SP)+                   ;POP ITERATION COUNTER
2119  005100  005726                           TST     (SP)+                   ;POP SIGN CORRECTION
2120  005102  010166  000006                   MOV     R1,6(SP)                ;PUSH REMAINDER ON STACK
2121  005106  010066  000004                   MOV     R0,4(SP)                ;PUSH QUOTIENT ONTO STACK
2122  005112  012616                           MOV     (SP)+,(SP)
```

```
2123  005114  000207                          RTS     PC
2124
2125                                          .SBTTL  DRIVE SUBROUTINES
2126                                  ;SUBROUTINE TO CHECK IF DRIVE IS AVAILABLE
2127                                  ;CALL:  MOVB    DRIVE#,DRVNUM
2128                                  ;       JSR     PC,DRVAVA
2129                                  ;RETURN:        'C' BIT SET IF NOT AVAILABLE
2130  005116  113765  001004  000010  DRVAVA: MOVB    @#DRVNUM,CS2(R5)        ;LOAD DRIVE #
2131  005124  032765  040000  000026          BIT     #TAP,DT(R5)             ;CHECK IF TAPE UNIT
2132  005132  001003                          BNE     1$
2133  005134  004767  000034                  JSR     PC,RHINIT
2134  005140  000262                          SEV                             ;SET 'V' TO IND NOT AVAIL
2135  005142  000207                  1$:     RTS     PC                      ;RETURN
2136
2137                                  ;SUBROUTINE TO CHECK IF TE16 SLAVE IS AVAILABLE FOR TEST
2138                                  ;CALL:  MOVB    DRIVE #,@#DRVNUM        ;PASS DRIVE # VIA DRVNUM
2139                                  ;       MOVB    SLAVE #,@#SLVNUM        ;PASS SLAVE # VIA SLVNUM
2140                                  ;       JSR     PC,SLVAVA               ;CALL SUBROUTINE
2141  005144  113765  001004  000010  SLVAVA: MOVB    @#DRVNUM,CS2(R5)        ;LOAD DRIVE #
2142  005152  113765  001005  000032          MOVB    @#SLVNUM,TC(R5)         ;AND SLAVE #
2143  005160  032765  002000  000026          BIT     #SPR,DT(R5)             ;BRANCH IF SLAVE PRESENT
2144  005166  001001                          BNE     1$
2145  005170  000262                          SEV                             ;SET 'V' TO INDICATE NO SLAVE
2146  005172  000207                  1$:     RTS     PC
2147
2148                                  ;SUBROUTINE TO  INITIALIZE RH CONTROLLER
2149                                  ;CALL:  JSR PC,RHINIT
2150
2151  005174  012765  000040  000010  RHINIT: MOV     #40,CS2(R5)
2152  005202  113765  001004  000010          MOVB    @#DRVNUM,CS2(R5)
2153  005210  005046                          CLR     -(SP)
2154  005212  113716  001005                  MOVB    @#SLVNUM,(SP)
2155  005216  012665  000032                  MOV     (SP)+,TC(R5)            ;LOAD SLAVE # INTO TC REG
2156  005222  052765  000300  000032          BIS     #NORM!11,TC(R5)
2157  005230  000207                          RTS     PC
2158
2159                                  ;SUBROUTINE TO WAIT FOR DRIVE READY (DRY)
2160  005232  005027                  WAITRDY:CLR     (PC)+                   ;CLEAR WAIT TIMER
2161  005234  000000                  WAITTIM:.WORD   0
2162  005236  105765  000012          1$:     TSTB    DS(R5)                  ;WAIT FOR READY TO SET
2163  005242  100406                          BMI     2$
2164  005244  005267  177764                  INC     WAITTIM                 ;INCREMENT WAIT TIMER
2165  005250  001372                          BNE     1$                      ;BRANCH IF TIME HAS NOT EXPIRED
2166  005252  000004  014273                  TYPE,E.TIMEXP                   ;TYPE 'TIME EXPIRED WAITING FOR RDY'
2167  005256  000425                          BR      99$                     ;TAKE ERROR EXIT
2168  005260  032765  002000  000012  2$:     BIT     #EOT,DS(R5)             ;CHECK FOR END OF TAPE
2169  005266  001415                          BEQ     3$                      ;BRANCH IF NO EOT
2170  005270  000004  013330                  TYPE,M.NAM
2171  005274  000004  013656                  TYPE,M.EOT                      ;TYPE 'END OF TAPE'
2172  005300  004767  000032                  JSR     PC,.REWIND              ;REWIND SLAVE
2173  005304  102412                          BVS     99$                     ;BRANCH IF ERROR ON REWIND
2174  005306  004767  000106                  JSR     PC,WRITE                ;WRITE A RECORD
2175  005312  005215                          INC     (R5)                    ;SET 'GO' BIT
2176  005314  004767  177712                  JSR     PC,WAITRDY              ;WAIT FOR READY
2177  005320  000404                          BR      99$                     ;TAKE ERROR EXIT
2178  005322  032765  040000  000012  3$:     BIT     #ERR,DS(R5)             ;CHECK ERROR EXIT
```

```
2179  005330  001401                    BEQ     100$
2180  005332  000262            99$:    SEV
2181  005334  000207            100$:   RTS     PC
2182
```

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

```
                                        ;       MOVB    SLAVE #,@#SLVNUM
                                        ;       JSR     PC,.REWIND
                                        ;SUBROUTINE RETURNS TO CALLER WITH SELECTED SLAVE AT 'BOT', & 'V' SET IF
                                        ;AN ERROR OCCURS.

2190  005336  004767  177632            .REWIND:JSR     PC,RHINIT               ;INITIALIZE CONTROLLER
```

```
2191  005342  004367  000206            JSR     R3,TMCMD            ;GO TO TM COMMAND SUBROUTINE
2192  005346  000000                    .WORD   0                   ;BUS ADDRESS (NOT USED)
2193  005350  000000                    .WORD   0                   ;WORD COUNT (NOT USED)
2194  005352  000000                    .WORD   0                   ;FRAME COUNT (NOT USED)
2195  005354  000006                    .WORD   RWD                 ;REWIND COMMAND
2196  005356  005215                    INC     (R5)                ;SET 'GO' BIT
2197  005360  032765  000002  000012  1$:       BIT     #BOT,DS(R5)         ;BRANCH IF 'BOT' SET
2198  005366  001005                    BNE     2$
2199  005370  032765  040000  000012            BIT     #ERR,DS(R5)         ;CHECK ERROR BIT
2200  005376  001006                    BNE     99$                 ;BRANCH IF ERROR BIT SET
2201  005400  000767                    BR      1$
2202
2203  005402  032765  020000  000012  2$:       BIT     #PIP,DS(R5)         ;WAIT FOR TAPE MOTION TO STOP
2204  005410  001374                    BNE     2$
2205  005412  000401                    BR      100$
2206  005414  000262                  99$:    SEV
2207  005416  000207                  100$:   RTS     PC
2208
2209                                  ;SUBROUTINE TO WRITE 256. WORD RECORD
2210                                  ;CALL:  JSR     PC,WRITE
2211
2212  005420  004367  000130          WRITE:  JSR     R3,TMCMD            ;GO TO TM COMMAND SUBROUTINE
2213  005424  015700                    .WORD   WTBUF               ;BUS ADDRESS
2214  005426  177600                    .WORD   WRDCNT              ;WORD COUNT
2215  005430  177400                    .WORD   FRMCNT              ;FRAME COUNT
2216  005432  000060                    .WORD   WFWD                ;WRITE FORWARD COMMAND
2217  005434  000207                    RTS     PC
2218
2219                                  ;SUBROUTINE TO READ A 256. WORD RECORD.
2220                                  ;CALL:  JSR     PC,READ
2221
2222  005436  004337  005554          READ:   JSR     R3,@#TMCMD
2223  005442  015700                    .WORD   RDBUF               ;ADDRESS OF READ BUFFER
2224  005444  177600                    .WORD   WRDCNT              ;2'S COMPLEMENT OF WORD COUNT
2225  005446  177400                    .WORD   FRMCNT              ;2'S COMPLEMENT OF FRAME COUNT
2226  005450  000070                    .WORD   RDFWD               ;READ FORWARD COMMAND
2227  005452  000207                    RTS     PC
2228
2229                                  ;SUBROUTINE TO INITIATE READ REVERSE COMMAND
2230                                  ;CALL:  JSR     PC,REVRD
2231                                  ;
2232  005454  004367  000074          REVRD:  JSR     R3,TMCMD
2233  005460  016300                    .WORD   RDBUF+256.          ;ADDRESS OF READ REVERSE BUFFER
2234  005462  177600                    .WORD   WRDCNT              ;2'S COMPLEMENT OF WORD COUNT
2235  005464  177400                    .WORD   FRMCNT              ;2'S COMPLEMENT OF FRAME COUNT
2236  005466  000076                    .WORD   RDREV               ;READ REVERSE COMMAND
2237  005470  000207                    RTS     PC
2238
2239                                  ;SUBROUTINE TO SPACE FORWARD 1 RECORD
2240  005472  012765  177777  000006  FWDSPC: MOV     #-1,FC(R5)          ;LOAD RECORD COUNT
2241  005500  012715  000031            MOV     #SPCFWD+1,(R5)      ;LOAD COMMAND
2242  005504  004767  177522            JSR     PC,WAITRDY          ;WAIT FOR READY
2243  005510  000207                    RTS     PC                  ;RETURN
2244
2245                                  ;SUBROUTINE TO WRITE A RECORD AND BACK SPACE OVER  THE RECORD.
2246  005512  004767  177702          WRT.BK: JSR     PC,WRITE            ;WRITE THE RECORD
```

```
2247  005516  005215                          INC     (R5)                    ;SET 'GO' BIT
2248  005520  004767  177506                  JSR     PC,WAITRDY
2249  005524  102412                          BVS     2$
2250  005526  012765  177777  000006          MOV     #-1,FC(R5)              ;LOAD RECORD COUNT
2251  005534  012715  000033                  MOV     #SPCREV+1,(R5)          ;LOAD COMMAND
2252  005540  004767  177466                  JSR     PC,WAITRDY
2253  005544  102402                          BVS     2$
2254  005546  004767  177166          1$:     JSR     PC,DELAY                ;WAIT FOR TAPE MOTION TO STOP
2255  005552  000207                  2$:     RTS     PC
2256
2257                                          ;SUBROUTINE TO LOAD A COMMAND
2258                                          ;CALL:  JSR     R3,TMCMD
2259                                          ;       .WORD   BUS ADDRESS
2260                                          ;       .WORD   WORD COUNT (2'S COMPLEMENT)
2261                                          ;       .WORD   FRAME COUNT (2'S COMPLEMENT)
2262                                          ;       .WORD   COMMAND
2263
2264  005554  012365  000004          TMCMD:  MOV     (R3)+,BA(R5)            ;LOAD BUS ADDRESS
2265  005560  012365  000002                  MOV     (R3)+,WC(R5)            ;LOAD WORD COUNT
2266  005564  012365  000006                  MOV     (R3)+,FC(R5)            ;LOAD FRAME COUNT
2267  005570  012315                          MOV     (R3)+,(R5)              ;LOAD COMMAND
2268  005572  000203                          RTS     R3                      ;RETURN
2269
2270                                          ;SUBROUTINE TO PRINT TE16 SERIAL NUMBER
2271                                          ;JSR    PC,SNPT
2272
2273  005574  016503  000030          SNPT:   MOV     SN(R5),R3
2274  005600  012701  001144                  MOV     #ODIGITS,R1
2275  005604  000303                          SWAB    R3
2276  005606  006003                          ROR     R3
2277  005610  006003                          ROR     R3
2278  005612  006003                          ROR     R3
2279  005614  006003                          ROR     R3                      ;GET FIRST DIGIT
2280  005616  042703  177760                  BIC     #177760,R3
2281  005622  052703  000260                  BIS     #260,R3
2282  005626  110321                          MOVB    R3,(R1)+                ;FILL FIRST DIGIT
2283  005630  016503  000030                  MOV     SN(R5),R3
2284  005634  000303                          SWAB    R3
2285  005636  042703  177760                  BIC     #177760,R3
2286  005642  052703  000260                  BIS     #260,R3
2287  005646  110321                          MOVB    R3,(R1)+                ;GET SECOND DIGIT
2288  005650  016503  000030                  MOV     SN(R5),R3
2289  005654  006003                          ROR     R3
2290  005656  006003                          ROR     R3
2291  005660  006003                          ROR     R3
2292  005662  006003                          ROR     R3
2293  005664  042703  177760                  BIC     #177760,R3
2294  005670  052703  000260                  BIS     #260,R3
2295  005674  110321                          MOVB    R3,(R1)+                ;GET THIRD DIGIT
2296  005676  016503  000030                  MOV     SN(R5),R3
2297  005702  042703  177760                  BIC     #177760,R3
2298  005706  052703  000260                  BIS     #260,R3
2299  005712  110321                          MOVB    R3,(R1)+                ;GET FOURTH DIGIT
2300  005714  105011                          CLRB    (R1)
2301  005716  000004  001144                  TYPE,ODIGITS                    ;TYPE SERIAL NUMBER
2302  005722  000207                          RTS     PC                      ;RETURN
```

```
2303
2304                                            .SBTTL  PROGRAM INITIALIZATION
2305  005724  012706  000600          INIT:     MOV     #STKPTR,SP              ;SET STACK PTR
2306
2307  005730  013746  000006          SUSWR:    MOV     @#6,-(SP)               ;SAVE VECTORS
2308  005734  013746  000004                    MOV     @#4,-(SP)
2309  005740  012737  005760  000004            MOV     #61$,@#4                        ;SET UP FOR TIMEOUT
2310  005746  022777  177777  173024            CMP     #-1,@SWR                ;REFERENCE HARDWARE SWITCH REGISTER
2311  005754  001402                           BEQ     60$
2312  005756  000404                           BR      62$
2313  005760  022626                  61$:     CMP     (SP)+,(SP)+             ;ADJUST STACK
2314  005762  012767  000176  173010  60$:     MOV     #SWREG,SWR              ;POINT TO SOFTWARE SWITCH REG
2315  005770  012637  000004          62$:     MOV     (SP)+,@#4               ;RESTORE VECTORS
2316  005774  012637  000006                    MOV     (SP)+,@#6
2317  006000  022737  000176  001000            CMP     #SWREG,@#SWR
2318  006006  001002                           BNE     64$
2319  006010  004767  174014                   JSR     PC,CNTLU
2320  006014  105037  001124          64$:     CLRB    @#PRGFLG                ;CLEAR PROGRAM FLAG
2321  006020  105037  001121                   CLRB    @#ITCNT                 ;CLEAR ITERATION COUNT
2322  006024  105037  001122                   CLRB    @#TSTNUM                ;SET TEST # 0
2323  006030  105037  001123                   CLRB    @#ERFLG                 ;CLEAR ERROR FLAG
2324  006034  105067  173070                   CLRB    ASFLG                   ;CLEAR ASK FLAG
2325  006040  012737  000006  000004            MOV     #ERRVEC+2,@#ERRVEC
2326  006046  012737  000002  000006            MOV     #RTI,@#ERRVEC+2         ;CHECK IF 'LP' IS AVAILABLE
2327  006054  005037  001264          2$:      CLR     @#INBUF
2328  006060  000004  001374                   TYPE,CRLF
2329  006064  000004  013330                   TYPE,M.NAM                      ;TYPE TITLE
2330  006070  000004  013376                   TYPE,I.REG                      ;ASK USER TO TYPE CONT BASE ADRS
2331  006074  004767  175374                   JSR     PC,.INPUT               ;GET USER INPUT
2332  006100  004767  175154          4$:      JSR     PC,CNVTAO               ;CONVERT ASCII TO OCTAL
2333  006104  013737  001116  001010            MOV     @#OCTALO,@#TMBASE       ;SET NEW ADDRESS
2334  006112  013705  001010          5$:      MOV     @#TMBASE,R5
2335
2336                                  ;ROUTINE TO CHECK IF CONTROLLER (RH11) IS AVAILAABLE
2337  006116  000261                           SEC                             ;SET 'C' IN PSW
2338  006120  005715                           TST     (R5)                    ;BRANCH IF CONTROLLER AVAIL
2339  006122  103003                           BCC     6$
2340  006124  000004  013715                   TYPE,E.NCON
2341  006130  000675                           BR      INIT
2342  006132  012737  003650  000004  6$:      MOV     #ERRTRP,@#ERRVEC        ;SET ERROR TRAP VECTOR
```

```
2343                                    ;ROUTINE TO GET TM02 DRIVES USER DESIRES TO TEST
2344  006140  105037  001123  DRIVES: CLRB    @#ERFLG                 ;CLEAR ERROR FLAG
2345  006144  012701  001154          MOV     #DRVTBL,R1              ;MARK ALL DRIVES AS NOT TO
2346  006150  012700  000004          MOV     #4,R0                   ;BE TESTED. A '0' INDICATES
2347  006154  005021          1$:     CLR     (R1)+                   ;THAT A DRIVE IS NOT TO BE
2348  006156  005300                  DEC     R0                      ;TESTED
2349  006160  001375                  BNE     1$
2350  006162  000004  013443          TYPE,I.DRVS
2351  006166  004767  175302          JSR     PC,.INPUT               ;GET USER INPUT
2352  006172  012700  001264          MOV     #INBUF,R0
2353  006176  122710  000101          CMPB    #'A,(R0)                ;AN 'A' SPECIFIES ALL
2354  006202  001013                  BNE     3$                      ;DRIVES TO BE TESTED
2355  006204  110667  172714          MOVB    SP,PRGFLG               ;SET FLAG TO IND ALL DRIVES
2356  006210  012701  001154          MOV     #DRVTBL,R1              ;MARK ALL DRIVES TO BE TESTED
2357  006214  012700  000004          MOV     #4,R0                   ;A '-1' INDICATES THAT A DRIVE
2358  006220  012721  177777  2$:     MOV     #-1,(R1)+               ;IS TO BE TESTED
2359  006224  005300                  DEC     R0
2360  006226  001374                  BNE     2$
2361  006230  000417                  BR      CHKDRV                  ;GO CHECK DRIVE AVAILABILITY
2362
2363                                    ;GET USER SELECTED DRIVES AND MARK EACH DRIVE SELECTED TO BE TESTED
2364  006232  122710  000015  3$:     CMPB    #CR,(R0)
2365  006236  001414                  BEQ     CHKDRV
2366  006240  121027  000054          CMPB    (R0),#',                ;CHECK IF 'COMMA'
2367  006244  001001                  BNE     4$
2368  006246  105720                  TSTB    (R0)+                   ;STEP PTR PAST 'COMMA'
2369  006250  112001  004$:     MOVB    (R0)+,R1
2370  006252  042701  177770          BIC     #177770,R1
2371  006256  112761  177777  001154  MOVB    #-1,DRVTBL(R1)
2372  006264  000240                  NOP
2373  006266  000761                  BR      3$
2374
2375                                    ;ASCERTAIN THAT DRIVES (TM02'S) SPECIFIED ARE AVAILABLE
2376  006270  005000  CHKDRV: CLR     R0                      ;A 0/-1 INDICATES THAT THE
2377  006272  105760  001154  1$:     TSTB    DRVTBL(R0)              ;DRIVE IS NOT/IS TO BE TESTED
2378  006276  001005                  BNE     3$
2379  006300  005200          2$:     INC     R0
2380  006302  122700  000010          CMPB    #8.,R0
2381  006306  001371                  BNE     1$
2382  006310  000421                  BR      4$
2383  006312  110037  001004  3$:     MOVB    R0,@#DRVNUM
2384  006316  004737  005116          JSR     PC,@#DRVAVA             ;CHECK IF AVAILABLE
2385  006322  102366                  BVC     2$                      ;'V' BIT SET INDICATES NOT AVAIL
2386  006324  000004  013762          TYPE,E.NDRV
2387  006330  116037  001132  014014  MOVB    DIGTAB(R0),@#E.NAVA     ;SET DRIVE # IN MESSAGE
2388  006336  000004  014014          TYPE,E.NAVA
2389  006342  110637  001123          MOVB    SP,@#ERFLG              ;SET 'ERROR' FLAG
2390  006346  105060  001154          CLRB    DRVTBL(R0)              ;MARK DRIVE UNAVAILABLE
2391  006352  000752                  BR      2$                      ;CHECK NEXT DRIVE
2392  006354  105737  001123  4$:     TSTB    @#ERFLG                 ;GO GET SLAVES IF NO ERROR
2393  006360  001403                  BEQ     SLAVES
2394  006362  105737  001124          TSTB    @#PRGFLG                ;ASK USER TO RETYPE DRIVES IF
2395  006366  001664                  BEQ     DRIVES                  ;'ALL' NOT SPECIFIED
2396
2397                                    ;ROUTINE TO GET SLAVES (TE16'S) USER DESIRES TO TEST
2398  006370  105037  001123  SLAVES: CLRB    @#ERFLG                 ;CLEAR 'ERROR' FLAG
```

```
2399  006374  012701  001164          MOV     #SLVTBL,R1              ;MARK ALL SLAVES (64.) AS NOT
2400  006400  012700  000040          MOV     #32.,R0                 ;TO BE TESTED.A 0 INDICATES THAT
2401  006404  005021          1$:     CLR     (R1)+                   ;A DRIVE'S SLAVE IS NOT TO BE
2402  006406  005300                  DEC     R0                      ;TESTED
2403  006410  001375                  BNE     1$
2404  006412  005000                  CLR     R0                      ;R0 = DRIVE # FOR SLAVES
2405  006414  012701  001164          MOV     #SLVTBL,R1              ;R1 POINTS TO DRIVE'S SLAVE
2406  006420  105760  001154  2$:     TSTB    DRVTBL(R0)              ;IF DRIVE IS TO BE TESTED
2407  006424  001007                  BNE     4$                      ;GO TO 4$ OTHERWISE
2408  006426  062701  000010  3$:     ADD     #8.,R1                  ;STEP SLAVE PTR TO NEXT DRIVE'S
2409  006432  005200                  INC     R0                      ;SLAVES AND INCREMENT DRIVE #
2410  006434  122700  000010          CMPB    #8.,R0                  ;CHECK ALL DRIVES
2411  006440  001367                  BNE     2$                      ;AND WHEN ALL DRIVES CHECKED
2412  006442  000454                  BR      CHKSLV                  ;GO CHECK SLAVE AVAILABILITY
2413
2414  006444  105737  001124  4$:     TSTB    @#PRGFLG                ;BRANCH IF USER SELECTED ALL
2415  006450  001020                  BNE     5$                      ;DRIVES
2416  006452  116067  172326          MOVB    R0,DRVNUM               ;GET DRIVE #
2417  006456  116037  001132  013524  MOVB    DIGTAB(R0),@#I.DRV      ;PREPARE USER ACTION MESSAGE
2418  006464  000004  013505          TYPE,I.SLVS
2419  006470  004767  175000          JSR     PC,.INPUT               ;GET USER INPUT
2420  006474  012703  001264          MOV     #INBUF,R3               ;SET PTR TO USER INPUT
2421  006500  122710  000101          CMPB    #'A,(R0)                ;BRANCH IF USER DOES NOT WANT
2422  006504  001015                  BNE     7$                      ;'ALL' SLAVES
2423  006506  110637  001124          MOVB    SP,@#PRGFLG             ;SET 'ALL' INDICATOR
2424  006512  012701  001164  5$:     MOV     #SLVTBL,R1              ;MARK ALL SLAVES FOR ALL
2425  006516  012700  000040          MOV     #32.,R0                 ;DRIVES AS TO BE TESTED
2426  006522  012721  177777  6$:     MOV     #-1,(R1)+
2427  006526  005300                  DEC     R0
2428  006530  001374                  BNE     6$
2429  006532  105737  001124          TSTB    @#PRGFLG                ;BRANCH IF ALL WAS SELECTED
2430  006536  001016                  BNE     CHKSLV
2431
2432  006540  122713  000015  7$:     CMPB    #CR,(R3)                ;GET USER SELECTED SLAVES FOR
2433  006544  001730                  BEQ     3$                      ;DRIVE
2434  006546  121327  000054          CMPB    (R3),#',                ;STEP PTR PAST 'COMMA'
2435  006552  001001                  BNE     8$
2436  006554  105723                  TSTB    (R3)+
2437  006556  112304  8$:     MOVB    (R3)+,R4                ;AND MARK SELECED SLAVE
2438  006560  042704  177770          BIC     #177770,R4              ;AS TO BE TESTED
2439  006564  060104                  ADD     R1,R4
2440  006566  112714  177777          MOVB    #-1,(R4)
2441  006572  000762                  BR      7$
2442
2443                          ;ASCERTAIN THAT SLAVES (TE16'S) SELECTED ARE AVAILABLE
2444  006574  005000  CHKSLV: CLR     R0                      ;R0 WILL CONTAIN THE DRIVE #
2445  006576  005001                  CLR     R1                      ;AND R1 THE SLAVE #
2446  006600  012702  001164          MOV     #SLVTBL,R2              ;SET PTR TO SLAVE TABLE
2447  006604  105760  001154  1$:     TSTB    DRVTBL(R0)              ;BRANCH IF DRIVE SELECTED
2448  006610  001007                  BNE     3$                      ;& AVAILABLE FOR TEST
2449  006612  005200          2$:     INC     R0                      ;INCREMENT DRIVE #
2450  006614  062702  000010          ADD     #8.,R2                  ;STEP SLAVE PTR TO NEXT DRIVE'S
2451  006620  022700  000010          CMP     #8.,R0                  ;SLAVES. BRANCH TO 1$ IF NOT ALL
2452  006624  001367                  BNE     1$                      ;DRIVES CHECKED OTHERWISE EXIT
2453  006626  000434                  BR      7$
2454
```

```
2455  006630  005001                  3$:     CLR     R1                      ;SET SLAVE # 0
2456  006632  105712                  4$:     TSTB    (R2)                    ;BRANCH IF DRIVE'S SLAVE IS SEL-
2457  006634  001006                          BNE     6$                      ;ECTED FOR TEST
2458  006636  005201                  5$:     INC     R1                      ;INCREMENT SLAVE #
2459  006640  005202                          INC     R2                      ;STEP PTR TO NEXT SLAVE
2460  006642  022701  000010                  CMP     #8.,R1                  ;GO TO 4$ IF ALL SLAVES NOT
2461  006646  001371                          BNE     4$                      ;CHECKED
2462  006650  000760                          BR      2$                      ;OTHERWISE GO TO 2$ ABOVE
2463
2464  006652  110037  001004          6$:     MOVB    R0,@#DRVNUM             ;PASS DRIVE & SLAVE #
2465  006656  110137  001005                  MOVB    R1,@#SLVNUM
2466  006662  004737  005144                  JSR     PC,@#SLVAVA             ;AND CHECK IF AVAILABLE
2467  006666  102363                          BVC     5$                      ;'V' BIT SET ON RETURN IND-
2468  006670  116037  001132  014004        MOVB    DIGTAB(R0),@#E.DRV      ;ICATES ERROR. PREPARE ERROR
2469  006676  116137  001132  014014        MOVB    DIGTAB(R1),@#E.NAVA     ;MESSAGE
2470  006704  000004  013776                  TYPE,E.NSLV
2471  006710  110637  001123                  MOVB    SP,@#ERFLG              ;SET ERROR INDICATOR
2472  006714  105012                          CLRB    (R2)                    ;CLEAR SLAVE TABLE ENTRY
2473  006716  000747                          BR      5$                      ;GET NEXT SLAVE
2474
2475  006720  105737  001123          7$:     TSTB    @#ERFLG                 ;BRANCH IF NO ERROR
2476  006724  001403                          BEQ     100$
2477  006726  105737  001124                  TSTB    @#PRGFLG                ;BRANCH IF NOT 'ALL'
2478  006732  001616                          BEQ     SLAVES                  ;ASK USER TO RETYPE SLAVES
2479  006734  012737  003650  000004  100$:   MOV     #ERRTRP,@#ERRVEC
2480
2481                                  ;SCAN DIVE AND SLAVE TABLE FOR DRIVE/SLAVE COMBINATION TO TEST
2482  006742  105037  001004                  CLRB    @#DRVNUM                ;SET DRIVE AND SLAVE # 0
2483  006746  105037  001005                  CLRB    @#SLVNUM
2484  006752  012737  001164  001006          MOV     #SLVTBL,@#SLVPTR        ;SET PTR TO SLAVE TABLE
2485  006760  105037  001125                  CLRB    @#UNTFND                ;CLEAR 'UNIT FOUND' IND.
2486
2487  006764  113700  001004          BEGIN:  MOVB    @#DRVNUM,R0             ;GET DRIVE #
2488  006770  113701  001005                  MOVB    @#SLVNUM,R1             ;AND SLAVE #
```

```
2489  006774  013702  001006          MOV     @#SLVPTR,R2             ;GET SLAVE PTR
2490  007000  105760  001154  1$:     TSTB    DRVTBL(R0)              ;BRANCH IF DRIVE AVAIL TO TEST
2491  007004  001011                  BNE     3$
2492  007006  005001                  CLR     R1                      ;CLEAR SLAVE #
2493  007010  062702  000010          ADD     #8.,R2                  ;AND STEP PTR TO NEXT DRIVE'S
2494  007014  005200          2$:     INC     R0                      ;SLAVES AND INCREMENT DRIVE #
2495  007016  022700  000010          CMP     #8.,R0                  ;EXIT TEST IF ALL DRIVES
2496  007022  001366                  BNE     1$                      ;CHECKED OTHERWISE CONTINUE
2497  007024  000137  012732          JMP     @#END                   ;SCAN FOR NEXT 'UNIT'
2498
2499  007030  105712          3$:     TSTB    (R2)                    ;BRANCH IF SLAVE ON DRIVE IS
2500  007032  001007                  BNE     4$                      ;AVAILABLE THERWISE STEP
2501  007034  005202                  INC     R2                      ;PTR TO NEXT SLAVE
2502  007036  005201                  INC     R1                      ;INCREMENT SLAVE #
2503  007040  122701  000010          CMPB    #8.,R1                  ;UNTIL ALL SLAVES CHECKED
2504  007044  001371                  BNE     3$                      ;WHEN ALL SLAVES CHECKED
2505  007046  005001                  CLR     R1                      ;SET SLAVE # 0
2506  007050  000761                  BR      2$                      ;AND CONTINUE SCAN
2507
2508  007052  110637  001125  4$:     MOVB    SP,@#UNTFND             ;INDICATE THAT A 'UNIT' IS FOUND
2509  007056  110037  001004          MOVB    R0,@#DRVNUM             ;SET DRIVE 3
```

```
2510  007062  110137  001005          MOVB    R1,@#SLVNUM             ;SET SLAVE #
2511  007066  010237  001006          MOV     R2,@#SLVPTR             ;SAVE SLAVE PTR
2512
2513  007072  105737  001130  5$:     TSTB    @#ASFLG
2514  007076  001044                  BNE     7$
2515  007100  112767  000001  172022  MOVB    #1,ASFLG
2516
2517  007106  105037  001124          CLRB    @#PRGFLG        ;CLEAR PROGRAM INDICATOR
2518  007112  000004  013564          TYPE,I.SKEW             ;ASK USER IF HE WANTS TO RUN SKEW TESTS
2519  007116  004767  174352          JSR     PC,.INPUT               ;GET USER INPUT
2520  007122  012703  001264          MOV     #INBUF,R3       ;GET REPLY
2521  007126  122713  000060          CMPB    #'0,(R3)        ;BRANCH IF 'NO' (0)
2522  007132  001406                  BEQ     6$
2523  007134  122713  000061          CMPB    #'1,(R3)        ;CHECK IF 'YES' (1)
2524  007140  001354                  BNE     5$              ;NEITHER SO ASK AGAIN
2525  007142  111337  001124          MOVB    (R3),@#PRGFLG   ;SET INDICATOR
2526  007146  000420                  BR      7$
2527
2528  007150  105037  001127  6$:     CLRB    @#NRZFLG        ;CLEAR NRZ INDICATOR
2529  007154  000004  013625          TYPE,I.NRZ              ;ASK USER IF DRIVE 'NRZ' ONLY
2530  007160  004767  174310          JSR     PC,.INPUT               ;GET USER INPUT
2531  007164  012703  001264          MOV     #INBUF,R3       ;GET REPLY
2532  007170  122713  000060          CMPB    #'0,(R3)        ;BRANCH IF 'NO' (0)
2533  007174  001405                  BEQ     7$
2534  007176  122713  000061          CMPB    #'1,(R3)        ;CHECK IF 'YES' (1)
2535  007202  001362                  BNE     6$              ;ASK AGAIN IF NEITHER
2536  007204  111337  001127          MOVB    (R3),@#NRZFLG   ;SET INDICATOR
2537  007210                  7$:
2538
2539  007210  052737  000100  177560  TYPHDR: BIS     #100,@#TKS              ;SET KEYBOARD IE BIT
2540  007216  000004  014344          TYPE,L.HDR1
2541  007222  116037  001132  014524  MOVB    DIGTAB(R0),@#L.DRV      ;SET DRIVE #
2542  007230  116137  001132  014536  MOVB    DIGTAB(R1),@#L.SLV      ;AND SLAVE #
2543  007236  112737  000071  014541  MOVB    #'9,@#L.CHAN            ;GET SLAVES CHANNEL TYPE
2544  007244  032765  010000  000026  BIT     #CH7,DT(R5)
2545  007252  001403                  BEQ     1$
2546  007254  112737  000067  014541  MOVB    #'7,@#L.CHAN            ;SET 7 CHANNEL
2547  007262  000004  014457  1$:     TYPE,L.HDR2
2548  007266  004767  176302          JSR     PC,SNPT         ;GO PRINT SERIAL NUMBER
2549  007272  000004  014560          TYPE,L.HDR3
2550  007276  012737  007336  001002  MOV     #TST001,@#SCPADR        ;SET 'SCOPE' ADDRESS FOR FIRST TEST
2551  007304  010500                  MOV     R5,R0
2552  007306  062700  000006          ADD     #FC,R0                  ;R0 CONTAINS ADDRESS OF FC REG
2553  007312  010501                  MOV     R5,R1
2554  007314  062701  000012          ADD     #DS,R1                  ;R1 CONTAINS ADDRESS OF DS REG
2555  007320  012703  004444          MOV     #TIMER,R3               ;SET JUMP ADDRESS TO TIMER
2556  007324  105737  001124          TSTB    @#PRGFLG                ;BRANCH IF NOT SKEW TESTS
2557  007330  001402                  BEQ     TST001
2558  007332  000137  012764          JMP     @#SKEWTST
```

```
                                        .SBTTL  START OF TESTS
                                ;TEST 001 - WRITE FROM BOT
                                ;THIS TEST WILL MEASURE ACCELERATION DELAY REQUIRED TO
                                ;MOVE THE TAPE APPROXIMATELY SEVEN (7) INCHES FORWARD
                                ;FROM DEAD STOP BEFORE STARTING TO TRANSFER DATA.

                                ;THIS TEST MEASURES TIME FROM 'GO'=1 TO 'ACCL'=0.
007336  112737  000001  001122  TST001: MOVB    #1,@#TSTNUM             ;SET TEST #
007344  012702  007370                  MOV     #1$,R2                  ;SET RETURN PC FROM TIMER
007350  004767  175762                  JSR     PC,.REWIND              ;REWIND SLAVE
007354  102420                          BVS     99$                     ;BRANCH IF ERROR ON REWIND
007356  004767  176036                  JSR     PC,WRITE                ;GO SETUP WRITE COMMAND
007362  004767  174766                  JSR     PC,TIMON                ;TURN TIMER ON
007366  005215                          INC     (R5)                    ;SET 'GO' BIT

007370  005765  000032          1$:     TST     TC(R5)                  ;BRANCH WHEN 'ACCL'=0
007374  100002                          BPL     2$
007376  000163  004444                  JMP     TIMER(R3)               ;GO TO TIMER & RETURN VIA R2

007402  004767  175624          2$:     JSR     PC,WAITRDY              ;WAIT FOR COMMAND TO FINISH
007406  102403                          BVS     99$                     ;BRANCH IF ERROR
007410  004767  175066                  JSR     PC,TIMOK                ;GO CHECK TIME
007414  000401                          BR      100$
007416  104400                  99$:    HLT
007420  104000                  100$:   SCOPE

                                ;TEST 002 - WRITE START
                                ;THIS TST MEASURES TIME FROM 'GO'=1 TO 'ACCL'=0.
007422  112737  000002  001122  TST002: MOVB    #2,@#TSTNUM             ;SET TEST # 2
007430  004767  175764                  JSR     PC,WRITE                ;INITIATE WRITE COMMAND
007434  012702  007446                  MOV     #1$,R2                  ;SET RETURN PC FROM TIMER
007440  004767  174710                  JSR     PC,TIMON
007444  005215                          INC     (R5)                    ;SET 'GO' BIT

007446  005765  000032          1$:     TST     TC(R5)                  ;BRANCH WHEN 'ACCL'=0
007452  100002                          BPL     2$
007454  000163  004444                  JMP     TIMER(R3)               ;GO TO TIMER & RETURN VIA R2

007460  004767  175546          2$:     JSR     PC,WAITRDY              ;WAIT FOR READY
007464  102403                          BVS     99$                     ;BRANCH IF ERROR
007466  004767  175010                  JSR     PC,TIMOK                ;GO CHECK TIME RECORDED
007472  000401                          BR      100$                    ;EXIT VIA SCOPE

007474  104400                  99$:    HLT                             ;REPORT ERROR
007476  104000                  100$:   SCOPE

                                ;TEST 003- WRITE SHUTDOWN
                                ;THIS TEST MEASURES TIME FROM 'FC REG'=0 TO 'SWON'=1.
007500  112737  000003  001122  TST003: MOVB    #3,@#TSTNUM             ;SET TEST#3
007506  004767  175706                  JSR     PC,WRITE                ;INITIATE WRITE COMMAND
007512  005215                          INC     (R5)                    ;SET 'GO' BIT

007514  005710                  1$:     TST     (R0)                    ;BRANCH WHEN WRITING FINISHED
007516  001404                          BEQ     2$
007520  032711  040000                  BIT     #ERR,(R1)               ;MONITOR ERROR BIT
007524  001017                          BNE     99$
```

```
2615  007526  000772                          BR      1$
2616
2617  007530                          2$:
2618  007530  004767  174620                  JSR     PC,TIMON                ;TURN TIMER ON
2619  007534  010702                          MOV     PC,R2                   ;LOAD RETURN PC FROM TIMER
2620  007536  032711  000020          3$:     BIT     #SDWN,(R1)              ;BRANCH WHEN DS <SDWN> SETS
2621  007542  001002                          BNE     4$
2622  007544  000163  004444                  JMP     TIMER(R3)               ;GO TO TIMER & RETURN VIA R2
2623
2624  007550  004767  175456          4$:     JSR     PC,WAITRDY              ;WAIT FOR READY
2625  007554  102403                          BVS     99$
2626  007556  004767  174720                  JSR     PC,TIMOK                ;GO CHECK TIME RECORDED
2627  007562  000401                          BR      100$
2628  007564  104400                  99$:    HLT                             ;REPORT ERROR
2629  007566  104000                  100$:   SCOPE
2630
2631                                  ;TEST 004 - WRITE SETTLEDOWN
2632                                  ;THIS TEST MEASURES TIME FROM 'SWDN'=1 TO 'SWDN'=0.
2633  007570  112737  000004  001122  T004:  MOVB    #4,@#TSTNUM
2634  007576  004767  175616                  JSR     PC,WRITE
2635  007602  005215                          INC     (R5)                    ;SET 'GO' BIT
2636
2637  007604  005710                  1$:     TST     (R0)                    ;BRANCH WHEN WRITING FINISHED
2638  007606  001404                          BEQ     2$
2639  007610  032711  040000                  BIT     #ERR,(R1)               ;CHECK ERROR BIT
2640  007614  001026                          BNE     99$
2641  007616  000772                          BR      1$
2642
2643  007620  032711  000020          2$:     BIT     #SDWN,(R1)              ;WAIT FOR ASSERTION OF 'SDWN'
2644  007624  001004                          BNE     3$
2645  007626  032711  040000                  BIT     #ERR,(R1)               ;MONITOR ERROR BIT
2646  007632  001017                          BNE     99$
2647  007634  000771                          BR      2$
2648
2649  007636                          3$:
2650  007636  004767  174512                  JSR     PC,TIMON                ;TURN TIMER ON
2651  007642  010702                          MOV     PC,R2                   ;SET RETURN PC FROM TIMER
2652  007644  032711  000020                  BIT     #SDWN,(R1)              ;BRANCH WHEN SWDN CLEARS
2653  007650  001402                          BEQ     5$
2654  007652  000163  004444                  JMP     TIMER(R3)               ;GO TO TIMER & RETURN VIA R2
2655
2656  007656  004767  175350          5$:     JSR     PC,WAITRDY              ;WAIT FOR READY
2657  007662  102403                          BVS     99$
2658  007664  004767  174612                  JSR     PC,TIMOK
2659  007670  000401                          BR      100$
2660
2661  007672  104400                  99$:    HLT
2662  007674  104000                  100$:   SCOPE
2663
2664                                  ;TEST 005 - READ FROM BOT
2665                                  ;THIS TEST MEASURES TIME FROM 'GO'=1 TO 'ACCL'=0.
2666  007676  112737  000005  001122  T005:  MOVB    #5,@#TSTNUM             ;SET TEST #5
2667  007704  004767  175426                  JSR     PC,.REWIND              ;REWIND SLAVE
2668  007710  102422                          BVS     99$                     ;BRANCH IF ERROR ON REWIND
2669  007712  004767  175520                  JSR     PC,READ
2670  007716  012702  007730                  MOV     #1$,R2                  ;SET RETURN PC FROM TIMER
```

```
 2671  007722  004767  174426                 JSR     PC,TIMON                ;TURN TIMER ON
 2672  007726  005215                         INC     (R5)                    ;SET 'GO' BIT
 2673
 2674  007730  005765  000032          1$:    TST     TC(R5)                  ;BRANCH WHEN 'ACCL' RESETS
 2675  007734  100002                         BPL     2$
 2676  007736  000163  004444                 JMP     TIMER(R3)               ;GO TO TIMER & RETURN VIA R2
 2677
 2678  007742  004767  175264          2$:    JSR     PC,WAITRDY              ;WAIT FOR READY
 2679  007746  102403                         BVS     99$                     ;BRANCH IF ERROR
 2680  007750  004767  174526                 JSR     PC,TIMOK                ;CHECK RECORDED TIME
 2681  007754  000401                         BR      100$
 2682
 2683  007756  104400                 99$:    HLT
 2684  007760  104000                 100$:   SCOPE
 2685
 2686                                 ;TEST 006 - READ START
 2687                                 ;THIS TEST MEASURES TIME FROM 'GO'=1 TO 'ACCL'=0.
 2688  007762  112737  000006  001122 TST006: MOVB    #6,@#TSTNUM             ;SET TEST #6
 2689  007770  004767  175516                 JSR     PC,WRT.BK               ;WRITE A RECORD & BACK SPACE
 2690  007774  102422                         BVS     99$
 2691  007776  004767  175434                 JSR     PC,READ
 2692  010002  012702  010014                 MOV     #1$,R2                  ;SET RETURN PC FROM TIMER
 2693  010006  004767  174342                 JSR     PC,TIMON                ;TURN TIMER ON
 2694  010012  005215                         INC     (R5)                    ;SET 'GO' BIT
 2695
 2696  010014  005765  000032          1$:    TST     TC(R5)                  ;BRANCH WHEN 'ACCL' RESETS
 2697  010020  100002                         BPL     2$
 2698  010022  000163  004444                 JMP     TIMER(R3)               ;GO TO TIMER & RETURN VIA R2
 2699
 2700  010026  004767  175200          2$:    JSR     PC,WAITRDY
 2701  010032  102403                         BVS     99$
 2702  010034  004767  174442                 JSR     PC,TIMOK
 2703  010040  000401                         BR      100$
 2704
 2705  010042  104400                 99$:    HLT
 2706  010044  104000                 100$:   SCOPE
 2707
 2708                                 ;TEST 007 - READ SHUTDOWN
 2709                                 ;THIS TEST MEASURES TIME FROM 'FC REG'=FRAME COUNT TO 'SWDN'=1.
 2710  010046  112737  000007  001122 TST007: MOVB    #7,@#TSTNUM             ;SET TEST #7
 2711  010054  004767  175432                 JSR     PC,WRT.BK               ;WRITE A RECORD & BACK SPACE
 2712  010060  102430                         BVS     99$                     ;BRANCH IF ERROR
 2713  010062  004767  175350                 JSR     PC,READ
 2714  010066  005215                         INC     (R5)                    ;SET 'GO' BIT
 2715
 2716  010070  022710  000400          1$:    CMP     #-FRMCNT,(R0)           ;WAIT FOR FRAME COUNT TO
 2717  010074  001404                         BEQ     2$                      ;= # OF FRAMES WRITTEN
 2718  010076  032711  040000                 BIT     #ERR,(R1)               ;MONITOR ERROR BIT
 2719  010102  001017                         BNE     99$
 2720  010104  000771                         BR      1$
 2721
 2722  010106                         2$:
 2723  010106  004767  174242                 JSR     PC,TIMON                ;TURN TIMER ON
 2724  010112  010702                         MOV     PC,R2                   ;SET RETURN PC FROM TIMER
 2725  010114  032711  000020                 BIT     #SDWN,(R1)              ;BRANCH WHEN SDWN SETS
 2726  010120  001002                         BNE     3$
```

```
2727  010122  000163  004444                    JMP     TIMER(R3)               ;GO TO TIMER & RETURN VIA R2
2728
2729  010126  004767  175100          3$:       JSR     PC,WAITRDY
2730  010132  102403                            BVS     99$
2731  010134  004767  174342                    JSR     PC,TIMOK
2732  010140  000401                            BR      100$
2733
2734  010142  104400                99$:       HLT                             ;REPORT ERROR
2735  010144  104000                100$:      SCOPE
2736
2737                                ;TEST 010 - READ SETTLEDOWN
2738                                ;THIS TEST MEASURES TIME FROM 'SWDN'=1 TO 'SWDN'=0.
2739  010146  112737  000010  001122 TST010:    MOVB    #10,@#TSTNUM            ;SET TEST #10
2740  010154  012702  010232                    MOV     #4$,R2                  ;SET RETURN PC FROM TIMER
2741  010160  004767  175326                    JSR     PC,WRT.BK               ;WRITE A RECORD & BACK SPACE
2742  010164  102436                            BVS     99$
2743  010166  004767  175244                    JSR     PC,READ
2744  010172  005215                            INC     (R5)                    ;SET 'GO' BIT
2745
2746  010174  105711                1$:        TSTB    (R1)                    ;WAIT FOR READY
2747  010176  100404                            BMI     2$                      ;BRANCH WHEN SET
2748  010200  032711  040000                    BIT     #ERR,(R1)               ;CHECK ERROR BIT
2749  010204  001026                            BNE     99$
2750  010206  000772                            BR      1$
2751
2752  010210  032711  000020          2$:       BIT     #SDWN,(R1)              ;WAIT FOR ASSERTION OF 'SDWN'
2753  010214  001004                            BNE     3$
2754  010216  032711  040000                    BIT     #ERR,(R1)               ;MONITOR ERROR BIT
2755  010222  001017                            BNE     99$
2756  010224  000771                            BR      2$
2757
2758  010226                        3$:
2759  010226  004767  174122                    JSR     PC,TIMON                ;TURN TIMER ON
2760  010232  032765  000020  000012 4$:       BIT     #SDWN,DS(R5)            ;WAIT FOR NEGATION OF SDWN
2761  010240  001402                            BEQ     5$
2762  010242  000163  004444                    JMP     TIMER(R3)               ;GO TO TIMER & RETURN VIA R2
2763
2764  010246  004767  174760          5$:       JSR     PC,WAITRDY
2765  010252  102403                            BVS     99$
2766  010254  004767  174222                    JSR     PC,TIMOK
2767  010260  000401                            BR      100$
2768
2769  010262  104400                99$:       HLT
2770  010264  104000                100$:      SCOPE
2771
2772
2773                                ;TEST 011-READ REVERSE START
2774                                ;THIS TEST MEASURES TIME FROM 'GO'=1 TO 'ACCL'=0.
2775  010266  112737  000011  001122 TST011:    MOVB    #11,@#TSTNUM
2776  010274  012702  010332                    MOV     #1$,R2                  ;SET RETURN PC FROM TIMER
2777  010300  004767  175114                    JSR     PC,WRITE                ;WRITE A RECORD
2778  010304  005215                            INC     (R5)                    ;SET 'GO' BIT
2779  010306  004767  174720                    JSR     PC,WAITRDY
2780  010312  102422                            BVS     99$
2781  010314  004767  174420                    JSR     PC,DELAY                ;WAIT FOR TAPE MOTION TO STOP
2782  010320  004767  175130                    JSR     PC,REVRD
```

```
2783  010324  004767  174024          JSR     PC,TIMON                ;TURN TIMER ON
2784  010330  005215                  INC     (R5)                    ;SET 'GO' BIT
2785
2786  010332  005765  000032  1$:     TST     TC(R5)                  ;BRANCH WHEN 'ACCL' = 0
2787  010336  100002                  BPL     2$
2788  010340  000163  004444          JMP     TIMER(R3)               ;GO TO TIMER & RETURN VIA R2
2789
2790  010344  004767  174662  2$:     JSR     PC,WAITRDY
2791  010350  102403                  BVS     99$                     ;BRANCH IF ERROR
2792  010352  004767  174124          JSR     PC,TIMOK
2793  010356  000401                  BR      100$
2794
2795  010360  104400  99$:     HLT
2796  010362  104000  100$:    SCOPE
2797
2798                  ;TEST 012-READ REVERSE SHUTDOWN
2799                  ;THIS TEST MEASURES TIME FROM 'FC REG' = FRAME COUNT TO 'SDWN'=1.
2800  010364  112737  000012  001122  TST012: MOVB    #12,@#TSTNUM
2801  010372  012702  010442          MOV     #3$,R2                  ;SET RETURN PC FROM TIMER
2802  010376  004767  175016          JSR     PC,WRITE                ;WRITE A RECORD
2803  010402  005215                  INC     (R5)                    ;SET 'GO' BIT
2804  010404  004767  174622          JSR     PC,WAITRDY
2805  010410  102427                  BVS     99$
2806  010412  004767  175036          JSR     PC,REVRD
2807  010416  005215                  INC     (R5)                    ;SET 'GO' BIT
2808
2809  010420  022710  000400  1$:     CMP     #-FRMCNT,(R0)           ;BRANCH WHEN FRAME COUNT
2810  010424  001404                  BEQ     2$                      ;= # OF RECORD WRITTEN
2811  010426  032711  040000          BIT     #ERR,(R1)               ;MONITOR ERROR BIT IN 'DS' REG
2812  010432  001016                  BNE     99$
2813  010434  000771                  BR      1$
2814
2815  010436                  2$:
2816  010436  004767  173712          JSR     PC,TIMON                ;TURN TIMER ON
2817  010442  032711  000020  3$:     BIT     #SDWN,(R1)              ;BRANCH WHEN SDWN SETS
2818  010446  001002                  BNE     4$
2819  010450  000163  004444          JMP     TIMER(R3)               ;GO TO TIMER & RETURN VIA R2
2820
2821  010454  004767  174552  4$:     JSR     PC,WAITRDY              ;WAIT FOR READY
2822  010460  102403                  BVS     99$
2823  010462  004767  174014          JSR     PC,TIMOK
2824  010466  000401                  BR      100$
2825
2826  010470  104400  99$:     HLT
2827  010472  104000  100$:    SCOPE
2828
2829                  ;TEST 013-READ REVERSE SETTLEDOWN
2830                  ;THIS TEST MEASURES TIME FROM 'SDWN'=1 TO 'SDWN'=0.
2831  010474  112737  000013  001122  TST013: MOVB    #13,@#TSTNUM
2832  010502  012702  010566          MOV     #4$,R2                  ;SET RETURN PC FROM TIMER
2833  010506  004767  174706          JSR     PC,WRITE                ;WRITE A RECORD
2834  010512  005215                  INC     (R5)                    ;SET 'GO' BIT
2835  010514  004767  174512          JSR     PC,WAITRDY
2836  010520  102435                  BVS     99$
2837  010522  004767  174726          JSR     PC,REVRD
2838  010526  005215                  INC     (R5)                    ;SET 'GO' BIT
```

```
2839
2840  010530  105711                1$:     TSTB    (R1)                    ;BRANCH WHEN
2841  010532  100404                        BMI     2$                      ;READY SETS
2842  010534  032711  040000              BIT     #ERR,(R1)
2843  010540  001025                      BNE     99$
2844  010542  000772                      BR      1$
2845
2846  010544  032711  000020        2$:     BIT     #SDWN,(R1)
2847  010550  001004                      BNE     3$
2848  010552  032711  040000              BIT     #ERR,(R1)
2849  010556  001016                      BNE     99$
2850  010560  000771                      BR      2$
2851
2852  010562                        3$:
2853  010562  004767  173566              JSR     PC,TIMON                ;TURN TIMER ON
2854  010566  032711  000020        4$:     BIT     #SDWN,(R1)              ;BRANCH WHEN SWDN = 0
2855  010572  001402                      BEQ     5$
2856  010574  000163  004444              JMP     TIMER(R3)               ;GO TO TIMER & RETURN VIA R2
2857
2858  010600  004767  174426        5$:     JSR     PC,WAITRDY              ;WAIT FOR READY
2859  010604  102403                      BVS     99$
2860  010606  004767  173670              JSR     PC,TIMOK
2861  010612  000401                      BR      100$
2862
2863  010614  104400                99$:    HLT
2864  010616  104000                100$:   SCOPE
2865
2866                                ;REWIND DRIVE
2867  010620                        A:
2868  010620  004767  174512              JSR     PC,.REWIND              ;REWIND SLAVE
2869  010624  102401                      BVS     99$                     ;BRANCH IF ERROR ON REWIND
2870  010626  102002                      BVC     100$
2871  010630  104400                99$:    HLT
2872  010632  000772                      BR      A
2873  010634                        100$:
2874
2875                                ;TEST 014-TURN AROUND DELAY (FORWARD-REVERSE)
2876                                ;THIS TEST MEASURES TIME FROM 'GO'=1 (READ REVERSE) TO 'ACCL'=0
2877  010634  112737  000014  001122  TST014: MOVB    #14,@#TSTNUM
2878  010642  012702  010674              MOV     #2$,R2                  ;SET RETURN PC FROM TIMER
2879  010646  004767  174546              JSR     PC,WRITE                ;WRITE A RECORD
2880  010652  005215                      INC     (R5)                    ;SET 'GO' BIT
2881  010654  004767  174352              JSR     PC,WAITRDY
2882  010660  102420                      BVS     99$
2883
2884  010662  004767  174566        1$:     JSR     PC,REVRD                ;READ THE RECORD (REVERSE)
2885  010666  004767  173462              JSR     PC,TIMON                ;TURN TIMER ON
2886  010672  005215                      INC     (R5)                    ;SET 'GO' BIT
2887
2888  010674  005765  000032        2$:     TST     TC(R5)                  ;WAIT FOR 'ACCL' = 0
2889  010700  100002                      BPL     3$
2890  010702  000163  004444              JMP     TIMER(R3)               ;GO TO TIMER & RETURN VIA R2
2891
2892  010706  004767  174320        3$:     JSR     PC,WAITRDY
2893  010712  102403                      BVS     99$
2894  010714  004767  173562              JSR     PC,TIMOK
```

```
2895  010720  000401                          BR      100$
2896
2897  010722  104400                  99$:    HLT
2898  010724  104000                  100$:   SCOPE
2899
2900                                  ;TEST 015- TURN AROUND DELAY (REVERSE-FORWARD)
2901                                  ;THIS TEST MEASURES TIME FROM 'GO'=1 (READ) TO 'ACCL'=0.
2902  010726  112737  000015  001122  TST015: MOVB    #15,@#TSTNUM
2903  010734  012702  011002                  MOV     #2$,R2                  ;SET RETURN PC FROM TIMER
2904  010740  004767  174454                  JSR     PC,WRITE                ;WRITE A RECORD
2905  010744  005215                          INC     (R5)                    ;SET 'GO' BIT
2906  010746  004767  174260                  JSR     PC,WAITRDY              ;WAIT FOR READY
2907  010752  102426                          BVS     99$
2908  010754  004767  174474                  JSR     PC,REVRD                ;READ A RECORD IN THE
2909  010760  005215                          INC     (R5)                    ;SET 'GO' BIT
2910
2911  010762  004767  174244                  JSR     PC,WAITRDY
2912  010766  102420                          BVS     99$
2913
2914  010770  004767  174442          1$:     JSR     PC,READ                 ;READ RECORD FORWARD
2915  010774  004767  173354                  JSR     PC,TIMON                ;TURN TIMER ON
2916  011000  005215                          INC     (R5)                    ;SET 'GO' BIT
2917
2918  011002  005765  000032          2$:     TST     TC(R5)                  ;WAIT FOR 'ACCL' = 0
2919  011006  100002                          BPL     3$
2920  011010  000163  004444                  JMP     TIMER(R3)               ;GO TO TIMER & RETURN VIA R2
2921
2922  011014  004767  174212          3$:     JSR     PC,WAITRDY
2923  011020  102403                          BVS     99$
2924  011022  004767  173454                  JSR     PC,TIMOK
2925  011026  000401                          BR      100$
2926
2927  011030  104400                  99$:    HLT
2928  011032  104000                  100$:   SCOPE
2929
2930                                  ;TEST 016-GAP SIZE (STOP HALF)
2931  011034  112737  000016  001122  TST016: MOVB    #16.,@#TSTNUM
2932  011042  012702  011100                  MOV     #1$,R2                  ;SET RETURN PC FROM TIMER
2933  011046  004767  174346                  JSR     PC,WRITE                ;WRITE A RECORD
2934  011052  005215                          INC     (R5)                    ;SET 'GO' BIT
2935  011054  004767  174152                  JSR     PC,WAITRDY
2936  011060  102421                          BVS     99$
2937  011062  004767  173652                  JSR     PC,DELAY                ;DELAY 350 MS
2938  011066  004767  174362                  JSR     PC,REVRD                ;READ REVERSE RECORD
2939  011072  004767  173256                  JSR     PC,TIMON                ;TURN TIMER ON
2940  011076  005215                          INC     (R5)                    ;SET 'GO' BIT
2941
2942  011100  005710          1$:     TST     (R0)                    ;WAIT FOR FRAME COUNT > 0
2943  011102  001002                          BNE     2$
2944  011104  000163  004444                  JMP     TIMER(R3)               ;GO TO TIMER & RETURN VIA R2
2945
2946  011110  004767  174116          2$:     JSR     PC,WAITRDY              ;WAIT FOR READY BIT TO SET
2947  011114  102403                          BVS     99$
2948  011116  004767  173360                  JSR     PC,TIMOK                ;CHECK TIME
2949  011122  000401                          BR      100$
2950
```

```
2951  011124  104400                  99$:    HLT
2952  011126  104000                  100$:   SCOPE
2953
2954                                  ;TEST 017-GAP SIZE (START HALF)
2955  011130  112737  000017  001122  TST017: MOVB    #17,@#TSTNUM
2956  011136  012702  011210                  MOV     #1$,R2          ;SET RETURN PC FROM TIMER
2957  011142  004767  174252                  JSR     PC,WRITE        ;WRITE A RECORD
2958  011146  005215                          INC     (R5)            ;SET 'GO' BIT
2959  011150  004767  174056                  JSR     PC,WAITRDY      ;WAIT FOR READY
2960  011154  102427                          BVS     99$
2961  011156  004767  174272                  JSR     PC,REVRD        ;READ REVERSE THE RECORD
2962  011162  005215                          INC     (R5)            ;SET 'GO' BIT
2963  011164  004767  174042                  JSR     PC,WAITRDY      ;WAIT FOR READY
2964  011170  102421                          BVS     99$             ;BRANCH ON ERROR
2965  011172  004767  173542                  JSR     PC,DELAY        ;WAIT FOR TAPE MOTION TO STOP
2966  011176  004767  174234                  JSR     PC,READ         ;READ RECORD
2967  011202  004767  173146                  JSR     PC,TIMON        ;TURN TIMER ON
2968  011206  005215                          INC     (R5)            ;SET 'GO' BIT
2969
2970  011210  005710                  1$:     TST     (R0)            ;WAIT FOR FRAME COUNT > 0
2971  011212  001002                          BNE     2$
2972  011214  000163  004444                  JMP     TIMER(R3)       ;GO TO TIMER & RETURN VIA R2
2973
2974  011220  004767  174006          2$:     JSR     PC,WAITRDY      ;WAIT FOR READY
2975  011224  102403                          BVS     99$
2976  011226  004767  173250                  JSR     PC,TIMOK        ;CHECK TIME
2977  011232  000401                          BR      100$
2978
2979  011234  104400                  99$:    HLT
2980  011236  104000                  100$:   SCOPE
2981
2982                                  ;TEST 020- GAP SIZE (INTERRECORD)
2983                                  ;THIS TEST MEASURES TIME FROM 'GO'=1 TO 'FC REG' >0.
2984  011240  112737  000020  001122  TST020: MOVB    #20,@#TSTNUM
2985  011246  012702  011330                  MOV     #1$,R2          ;SET RETURN PC FROM TIMER
2986  011252  004767  174142                  JSR     PC,WRITE        ;WRITE A RECORD
2987  011256  005215                          INC     (R5)            ;SET 'GO' BIT
2988  011260  004767  173746                  JSR     PC,WAITRDY      ;WAIT FOR READY
2989  011264  102433                          BVS     99$
2990  011266  004767  174126                  JSR     PC,WRITE        ;WRITE SECOND RECORD
2991  011272  005215                          INC     (R5)            ;SET 'GO' BIT
2992  011274  004767  173732                  JSR     PC,WAITRDY      ;WAIT FOR READY
2993  011300  102425                          BVS     99$
2994  011302  004767  174146                  JSR     PC,REVRD        ;READ REVERSE SECOND RECORD
2995  011306  005215                          INC     (R5)            ;SET 'GO' BIT
2996  011310  004767  173716                  JSR     PC,WAITRDY      ;WAIT FOR READY
2997  011314  102417                          BVS     99$
2998  011316  004767  174132                  JSR     PC,REVRD        ;READ REVERSE FIRST RECORD
2999  011322  004767  173026                  JSR     PC,TIMON        ;TURN TIMER ON
3000  011326  005215                          INC     (R5)            ;SET 'GO' BIT
3001
3002  011330  005710                  1$:     TST     (R0)            ;WAIT FOR FRAME COUNT > 0
3003  011332  001002                          BNE     2$
3004  011334  000163  004444                  JMP     TIMER(R3)       ;GO TO TIMER & RETURN VIA R2
3005
3006  011340  004767  173666          2$:     JSR     PC,WAITRDY      ;WAIT FOR READY
```

```
3007  011344  102403                        BVS     99$
3008  011346  004767  173130                JSR     PC,TIMOK
3009  011352  000401                        BR      100$
3010
3011  011354  104400                99$:    HLT
3012  011356  104000                100$:   SCOPE
3013
3014                                ;TEST 021- GAP CONSISTANCY
3015                                ;THIS TEST MEASURES TIME FROM 'GO'=1 TO 'FC REG' > 0.
3016                                ;THE TEST REWINDS THE TAPE,WRITES 17 RECORDS WITH A DELAY FROM 1-16 MS
3017                                ;BETWEEN EACH WRITE COMMAND. AFTER THE 17. RECORDS ARE WRITTTEN THE
3018                                ;PROGRAM READ REVERSES 16 RECORDS. AT THIS POINT THE TAPE IS STOPPED BE-
3019                                ;TWEEN THE FIRST AND SECOND RECORD. A READ COMMAND IS EXECUTED TO READ
3020                                ;THE 16 RECORDS WITH THE TIME BETEWEN GO=1 TO FC > 0 STORED IN 'GAPTBL'
3021                                ;FOR EACH RECORD READ. AFTER 16 RECORDS HAVE BEEN READ THE TIME IS VER-
3022                                ;IFIED FOR EACH READ. AFTER ALL RECORD TIMES ARE VERIFIED THEY ARE AVER-
3023                                ;AGED AND PLACED IN THE 'ATIMTBL' (BY SCOPE). THE ABOVE PROCESS IS RE-
3024                                ;PEATED FOR EACH ITERATION.
3025
3026  011360  112737  000021  001122  TST021: MOVB    #21,@#TSTNUM
3027  011366  012702  011524              MOV     #4$,R2                  ;SET RETURN PC FROM TIMER
3028  011372  004767  173740              JSR     PC,.REWIND              ;REWIND SLAVE
3029  011376  102530                      BVS     99$                     ;BRANCH IF ERROR ON REWIND
3030  011400  005067  167510              CLR     DELTIM                  ;CLEAR VARIABLE DELAY TIME
3031  011404  012700  000021              MOV     #17.,R0                 ;SET # OF RECORDS TO WRITE
3032  011410  004767  174004        1$:   JSR     PC,WRITE                ;WRITE 17. RECORDS
3033  011414  005215                      INC     (R5)                    ;SET 'GO' BIT
3034  011416  004767  173610              JSR     PC,WAITRDY              ;WAIT FOR READY
3035  011422  102516                      BVS     99$
3036  011424  004767  173340              JSR     PC,DELAYV               ;DELAY BEFORE WRITING NEXT REC.
3037  011430  062767  000022  167456      ADD     #18.,DELTIM             ;SET NEXT DELAY TIME
3038  011436  005300                      DEC     R0                      ;DECREMENT RECORDS WRITTEN COUNT
3039  011440  001363                      BNE     1$
3040
3041  011442  012700  000021              MOV     #17.,R0                 ;SET # OF RECS. TO REVERSE READ
3042  011446  004767  174002        2$:   JSR     PC,REVRD                ;REVERSE READ 17. RECORDS
3043  011452  005215                      INC     (R5)                    ;SET 'GO' BIT
3044  011454  004767  173552              JSR     PC,WAITRDY              ;WAIT FOR READY
3045  011460  102477                      BVS     99$
3046  011462  005300                      DEC     R0                      ;DECREMENT RECORD COUNT
3047  011464  001370                      BNE     2$
3048
3049  011466  012700  000020              MOV     #16.,R0                 ;SET # OF RECORDS TO READ
3050  011472  012701  001054              MOV     #GAPTBL,R1              ;SET PTR TO GAP TABLE FOR TEST
3051  011476  004767  173734              JSR     PC,READ                 ;READ A RECORD
3052  011502  005215                      INC     (R5)                    ;SET 'GO' BIT
3053
3054  011504  004767  173522        3$:   JSR     PC,WAITRDY              ;WAIT FOR READY
3055  011510  102463                      BVS     99$
3056  011512  004767  173720              JSR     PC,READ                 ;READ NEXT RECORD
3057  011516  004767  172632              JSR     PC,TIMON                ;TURN TIMER ON
3058  011522  005215                      INC     (R5)                    ;SET 'GO' BIT
3059
3060  011524  005765  000006        4$:   TST     FC(R5)                  ;WAIT FOR FRAME COUNT > 0
3061  011530  001002                      BNE     5$
3062  011532  000163  004444              JMP     TIMER(R3)               ;GO TO TIMER & RETURN VIA R2
```

```
3063
3064  011536  004767  173470          5$:     JSR     PC,WAITRDY              ;WAIT FOR READY
3065  011542  102446                          BVS     99$
3066  011544  010421                          MOV     R4,(R1)+                ;STORE TIME IN GAPTBL
3067  011546  005300                          DEC     R0                      ;DECREMENT # OF RECORDS READ
3068  011550  001355                          BNE     3$
3069
3070  011552  105037  001120                  CLRB    @#GAP                   ;SET GAP # 0
3071  011556  012700  000020                  MOV     #16.,R0
3072  011562  012701  001054                  MOV     #GAPTBL,R1
3073
3074  011566  012104                  6$:     MOV     (R1)+,R4                ;GET GAP TICK COUNT
3075  011570  004767  173016                  JSR     PC,GAPOK                ;CHECK TIME
3076  011574  105237  001120                  INCB    @#GAP                   ;INCREMENT GAP #
3077  011600  122737  000020  001120          CMPB    #16.,@#GAP              ;BRANCH IF ALL GAPS NOT CHECKED
3078  011606  001367                          BNE     6$
3079
3080  011610  012700  000020                  MOV     #16.,R0                 ;SETUP TO AVERAGE GAP SIZES
3081  011614  012701  001054                  MOV     #GAPTBL,R1              ;SET PTR TO TABLE
3082  011620  005002                          CLR     R2                      ;CLEAR 'SUM' REGISTERS
3083  011622  005003                          CLR     R3
3084  011624  062102                  7$:     ADD     (R1)+,R2                ;ADD ALL GAP SIZES TOGETHER
3085  011626  005503                          ADC     R3
3086  011630  005300                          DEC     R0
3087  011632  001374                          BNE     7$
3088  011634  012700  000004                  MOV     #4,R0                   ;NOW DIVIDE BY 16.
3089  011640  006203                  8$:     ASR     R3                      ;BY SHIFTING 4 PLACES RIGHT
3090  011642  006002                          ROR     R2
3091  011644  005300                          DEC     R0
3092  011646  001374                          BNE     8$
3093  011650  010204                          MOV     R2,R4                   ;MOVE AVERAGED TIMES TO R4
3094  011652  004767  172624                  JSR     PC,TIMOK                ;CHECK AVERAGED TIMES
3095  011656  000401                          BR      100$
3096
3097  011660  104400                  99$:    HLT
3098  011662  104000                  100$:   SCOPE
3099
3100                                  ;TEST 022-DUMMY TEST
3101                                  ;THIS TEST MEASURES NOTHING
3102  011664  112737  000022  001122  TST022: MOVB    #22,@#TSTNUM
3103
3104                                  ;TEST 023-DATA TIME (2008PI)
3105                                  ;THIS TEST MEASURES TIME FROM 'FC REG' CHANGES TO 'RDY'=1.
3106  011672  112737  000023  001122  TST023: MOVB    #23,@#TSTNUM
3107  011700  012702  011752                  MOV     #3$,R2                  ;SET RETURN PC FROM TIMER
3108  011704  004767  173426                  JSR     PC,.REWIND              ;REWIND SLAVE
3109  011710  102437                          BVS     99$                     ;BRANCH IF ERROR ON REWIND
3110  011712  004367  173636                  JSR     R3,TMCMD                ;WRITE 800 WORD RECORD
3111  011716  015700                          .WORD   WTBUF                   ;SET WRITE BUFFER ADDRESS
3112  011720  176340                          .WORD   -800.                   ;WORD COUNT
3113  011722  174700                          .WORD   -1600.                  ;FRAME COUNT
3114  011724  000060                          .WORD   WFWD                    ;WRITE COMMAND
3115  011726  005215                          INC     (R5)                    ;SET 'GO' BIT
3116
3117  011730  022710  174700          1$:     CMP     #-1600.,(R0)            ;WAIT FOR FRAME COUNT TO CHANGE
3118  011734  001004                          BNE     2$
```

```
3119  011736  032711  040000          BIT     #ERR,(R1)               ;MONITOR ERROR BIT
3120  011742  001022                  BNE     99$
3121  011744  000771                  BR      1$
3122
3123  011746                  2$:
3124  011746  004767  172402          JSR     PC,TIMON                ;TURN TIMER ON
3125  011752  105711          3$:     TSTB    (R1)                    ;WAIT FOR READY TO SET
3126  011754  100402                  BMI     4$
3127  011756  000163  004444          JMP     TIMER(R3)               ;GO TO TIMER & RETURN VIA R2
3128  011762  012700  000003  4$:     MOV     #3,R0                   ;SET TO DIVIDE BY 8
3129  011766  006204          5$:     ASR     R4                      ;BY SHIFTING RIGHT 3 PLACES
3130  011770  005300                  DEC     R0
3131  011772  001375                  BNE     5$
3132  011774  004767  173232          JSR     PC,WAITRDY
3133  012000  102403                  BVS     99$
3134  012002  004767  172474          JSR     PC,TIMOK                ;CHECK TIME
3135  012006  000401                  BR      100$
3136
3137  012010  104400          99$:    HLT
3138  012012  104000          100$:   SCOPE
3139
3140                          ;TEST 024-DATA TIME (556BPI)
3141  012014  112737  000024  001122  TST024: MOVB   #24,@#TSTNUM
3142  012022  012702  012102          MOV     #3$,R2                  ;SET RETURN PC FROM TIMER
3143  012026  004767  173304          JSR     PC,.REWIND              ;REWIND SLAVE
3144  012032  102442                  BVS     99$                     ;BRANCH IF ERROR ON REWIND
3145  012034  052765  000700  000032  BIS     #BPI556+NORM11,TC(R5)   ;LOAD TAPE CONTROL REGISTER
3146  012042  004367  173506          JSR     R3,TMCMD                ;WRITE 2224. WORD RECORD
3147  012046  015700                  .WORD   WTBUF
3148  012050  173520                  .WORD   -2224.
3149  012052  167240                  .WORD   -4448.
3150  012054  000060                  .WORD   WFWD
3151  012056  005215                  INC     (R5)                    ;SET 'GO' BIT
3152
3153  012060  022710  167240  1$:     CMP     #-4448.,(R0)            ;BRANCH WHEN WRITING BEGINS
3154  012064  001004                  BNE     2$
3155  012066  032711  040000          BIT     #ERR,(R1)               ;MONITOR ERROR BIT
3156  012072  001022                  BNE     99$
3157  012074  000771                  BR      1$
3158
3159  012076                  2$:
3160  012076  004767  172252          JSR     PC,TIMON                ;TURN TIMER ON
3161  012102  105711          3$:     TSTB    (R1)                    ;BRANCH WHEN READY SETS
3162  012104  100402                  BMI     4$
3163  012106  000163  004444          JMP     TIMER(R3)               ;GO TO TIMER & RETURN VIA R2
3164
3165  012112  012700  000003  4$:     MOV     #3,R0                   ;SET SHIFT COUNT
3166  012116  006204          5$:     ASR     R4
3167  012120  005300                  DEC     R0
3168  012122  001375                  BNE     5$
3169  012124  004767  173102          JSR     PC,WAITRDY
3170  012130  102403                  BVS     99$
3171  012132  004767  172344          JSR     PC,TIMOK                ;CHECK TIME
3172  012136  000401                  BR      100$
3173
3174  012140  104400          99$:    HLT
```

```
 3175  012142  104000                  100$:   SCOPE
 3176
 3177                                  ;TEST 025-DATA TIME (800BPI)
 3178  012144  112737  000025  001122  TST025: MOVB    #025,@#TSTNUM
 3179  012152  012702  012232                  MOV     #3$,R2                  ;SET RETURN PC FROM TIMER
 3180  012156  004767  173154                  JSR     PC,.REWIND              ;REWIND SLAVE
 3181  012162  102442                          BVS     99$                     ;BRANCH IF ERROR ON REWIND
 3182  012164  052765  001300  000032          BIS     #BPI800+NORM11,TC(R5)   ;SET 800 BPI
 3183  012172  004367  173356                  JSR     R3,TMCMD                ;WRITE 3200. WORD RECORD
 3184  012176  015700                          .WORD   WTBUF
 3185  012200  171600                          .WORD   -3200.
 3186  012202  163400                          .WORD   -6400.
 3187  012204  000060                          .WORD   WFWD
 3188  012206  005215                          INC     (R5)                    ;SET 'GO' BIT
 3189
 3190  012210  022710  163400          1$:     CMP     #-6400.,(R0)            ;WAIT FOR WRITING TO START
 3191  012214  001004                          BNE     2$
 3192  012216  032711  040000                  BIT     #ERR,(R1)               ;MONITOR ERROR BIT
 3193  012222  001022                          BNE     99$
 3194  012224  000771                          BR      1$
 3195
 3196  012226                          2$:
 3197  012226  004767  172122                  JSR     PC,TIMON                ;TURN TIMER ON
 3198  012232  105711                  3$:     TSTB    (R1)                    ;BRANCH WHEN READY SETS
 3199  012234  100402                          BMI     4$
 3200  012236  000163  004444                  JMP     TIMER(R3)               ;GO TO TIMER & RETURN VIA R2
 3201
 3202  012242  012700  000003          4$:     MOV     #3,R0                   ;SET SHIFT COUNT
 3203  012246  006204                  5$:     ASR     R4
 3204  012250  005300                          DEC     R0
 3205  012252  001375                          BNE     5$
 3206  012254  004767  172752                  JSR     PC,WAITRDY
 3207  012260  102403                          BVS     99$
 3208  012262  004767  172214                  JSR     PC,TIMOK                ;CHECK TIME
 3209  012266  000401                          BR      100$
 3210
 3211  012270  104400                  99$:    HLT
 3212  012272  104000                  100$:   SCOPE
 3213
 3214                                  ;TEST 026-DATA TIME (1600BPI)
 3215  012274  112737  000026  001122  TST026: MOVB    #026,@#TSTNUM
 3216  012302  105737  001127                  TSTB    @#NRZFLG                ;BRANCH IF DRIVE 'NRZ ONLY'
 3217  012306  001046                          BNE     TST027
 3218  012310  012702  012370                  MOV     #3$,R2                  ;SET RETURN PC FROM TIMER
 3219  012314  004767  173016                  JSR     PC,.REWIND              ;REWIND SLAVE
 3220  012320  102437                          BVS     99$                     ;BRANCH IF ERROR ON REWIND
 3221  012322  052765  002300  000032          BIS     #PE1600+NORM11,TC(R5)   ;SET 1600 BPI
 3222  012330  004367  173220                  JSR     R3,TMCMD                ;WRITE 3200. WORD RECORD
 3223  012334  015700                          .WORD   WTBUF
 3224  012336  171600                          .WORD   -3200.
 3225  012340  163400                          .WORD   -6400.
 3226  012342  000060                          .WORD   WFWD
 3227  012344  005215                          INC     (R5)                    ;SET 'GO' BIT
 3228
 3229  012346  022710  163400          1$:     CMP     #-6400.,(R0)            ;BRANCH WHEN WRITING STARTS
 3230  012352  001004                          BNE     2$
```

```
 3231  012354  032711  040000            BIT     #ERR,(R1)               ;MONITOR ERROR BIT
 3232  012360  001017                    BNE     99$
 3233  012362  000771                    BR      1$
 3234
 3235  012364                    2$:
 3236  012364  004767  171764            JSR     PC,TIMON                ;TURN TIMER ON
 3237  012370  105711            3$:     TSTB    (R1)                    ;BRANCH WHEN READY SETS
 3238  012372  100402                    BMI     4$
 3239  012374  000163  004444            JMP     TIMER(R3)               ;GO TO TIMER & RETURN VIA R2
 3240
 3241  012400  006204            4$:     ASR     R4                      ;DIVIDE TIME BY 4
 3242  012402  006204                    ASR     R4
 3243  012404  004767  172622            JSR     PC,WAITRDY
 3244  012410  102403                    BVS     99$
 3245  012412  004767  172064            JSR     PC,TIMOK                ;CHECK TIME
 3246  012416  000401                    BR      100$
 3247
 3248  012420  104400            99$:    HLT
 3249  012422  104000            100$:   SCOPE
 3250
 3251                            ;TEST 027-ERASE
 3252                            ;THIS TST MEASURES TIME FROM 'GO'=1 TO 'RDY'=1.
 3253  012424  112737  000027  001122  TST027: MOVB   #27,@#TSTNUM
 3254  012432  012702  012460            MOV     #1$,R2                  ;SET RETURN PC FROM TIMER
 3255  012436  004337  005554            JSR     R3,@#TMCMD
 3256  012442  000000                    .WORD   0
 3257  012444  000000                    .WORD   0
 3258  012446  000000                    .WORD   0
 3259  012450  000024                    .WORD   ERASE
 3260  012452  004767  171676            JSR     PC,TIMON                ;TURN TIMER ON
 3261  012456  005215                    INC     (R5)                    ;SET 'GO' BIT
 3262
 3263  012460  105711            1$:     TSTB    (R1)                    ;BRANCH WHEN READY SETS
 3264  012462  100402                    BMI     2$
 3265  012464  000163  004444            JMP     TIMER(R3)               ;GO TO TIMER & RETURN VIA R2
 3266
 3267  012470  004767  172536            JSR     PC,WAITRDY
 3268  012474  102403                    BVS     99$
 3269  012476  004767  172000            JSR     PC,TIMOK
 3270  012502  000401                    BR      100$
 3271
 3272  012504  104400            99$:    HLT
 3273  012506  104000            100$:   SCOPE
 3274
 3275                            ;TEST-030 TAPE MARK
 3276                            ;THIS TEST MEASURES TIME FROM 'GO'=1 TO 'RDY'=1.
 3277  012510  112737  000030  001122  TST030: MOVB   #30,@#TSTNUM
 3278  012516  012702  012560            MOV     #1$,R2                  ;SET RETURN PC FROM TIMER
 3279  012522  004767  172672            JSR     PC,WRITE                ;WRITE A RECORD
 3280  012526  005215                    INC     (R5)                    ;SET 'GO' BIT
 3281  012530  004767  172476            JSR     PC,WAITRDY
 3282  012534  102423                    BVS     99$
 3283  012536  004337  005554            JSR     R3,@#TMCMD
 3284  012542  000000                    .WORD   0
 3285  012544  000000                    .WORD   0
 3286  012546  000000                    .WORD   0
```

```
3287  012550  000026                    .WORD   WFMK
3288  012552  004767  171576            JSR     PC,TIMON        ;TURN TIMER ON
3289  012556  005215                    INC     (R5)            ;SET 'GO' BIT
3290
3291  012560  105711            1$:     TSTB    (R1)            ;BRANCH WHEN READY SETS
3292  012562  100402                    BMI     2$
3293  012564  000163  004444            JMP     TIMER(R3)       ;GO TO TIMER & RETURN VIA R2
3294
3295  012570  004767  172436    2$:     JSR     PC,WAITRDY
3296  012574  102403                    BVS     99$
3297  012576  004767  171700            JSR     PC,TIMOK
3298  012602  000401                    BR      100$
3299
3300  012604  104400            99$:    HLT
3301  012606                    100$:
3302  012606  004767  172524            JSR     PC,.REWIND      ;REWIND SLAVE
3303  012612  102774                    BVS     99$             ;BRANCH IF ERROR ON REWIND
3304  012614  104000                    SCOPE
3305
```

```
3306  012616  012700  000012          FINISH: MOV     #10.,R0                 ;SET LINE FEED COUNT
3307  012622  000004  001374          1$:     TYPE,CRLF
3308  012626  005300                          DEC     R0
3309  012630  001374                          BNE     1$
3310  012632  032777  000100  166140        BIT     #SW06,@SWR
3311  012640  001410                          BEQ     2$
3312  012642  113700  001004                MOVB    @#DRVNUM,R0
3313  012646  113701  001005                MOVB    @#SLVNUM,R1
3314  012652  113702  001006                MOVB    @#SLVPTR,R2
3315  012656  000137  007210                JMP     @#TYPHDR
3316  012662  105237  001005          2$:     INCB    @#SLVNUM                ;SET NEXT SLAVE #
3317  012666  005237  001006                INC     @#SLVPTR                ;AND ITS POINTER
3318  012672  122737  000010  001005        CMPB    #8.,@#SLVNUM            ;BRANCH IF LAST SLAVE (7)
3319  012700  001402                          BEQ     3$
3320  012702  000137  006764                JMP     @#BEGIN                 ;BEGIN TEST ON NEXT SLAVE
3321  012706  105037  001005          3$:     CLRB    @#SLVNUM                ;SET SLAVE #0
3322  012712  105237  001004                INCB    @#DRVNUM                ;AND INCREMENT DRIVE #
3323  012716  122737  000010  001004        CMPB    #8.,@#DRVNUM            ;AND CHECK IF LAST DRIVE
3324  012724  001402                          BEQ     END
3325  012726  000137  006764                JMP     @#BEGIN
3326
3327  012732  105737  001125          END:    TSTB    @#UNTFND                ;BRANCH IF A UNIT WAS FOUND
3328  012736  001004                          BNE     1$
3329  012740  000004  014047                TYPE,E.UNIT
3330  012744  000137  005724                JMP     @#INIT
3331  012750  000000          1$:     HALT
3332  012752  004767  167012                JSR     PC,CKSWR                ;CHECK FOR CNTL G
3333  012756  000005                        RESET
3334  012760  000137  005724                JMP     @#INIT                  ;RESTART
```

```
                                        ;SKEW TAPE TIMING TESTS
                                        ;THE FOLLOWING TESTS REQUIRE A SPECIALLY WRITTEN 800 BPI SKEW TAPE
012764  012737  012772  001002  SKEWTST:MOV     #TST031,@#SCPADR        ;SET SCOPE POINTER

                                        ;TEST 031- SKEW TAPE SPEED TEST-FORWARD
                                        ;THIS TEST READS 32" OF TAPE (26400.-800. = 25600. FRAMES) THEN
                                        ;DIVIDES TIME BY 32. TO GET TIME TO READ 1" (800. FRAMES) OF TAPE.
012772  112737  000031  001122  TST031: MOVB    #31,@#TSTNUM
013000  012702  013056                  MOV     #2$,R2                  ;SET RETURN PC FROM TIMER
013004  004767  172326                  JSR     PC,.REWIND              ;REWIND SLAVE
013010  102441                          BVS     99$                     ;BRANCH IF ERROR ON REWIND
013012  052765  001300  000032          BIS     #BPI800+NORM11,TC(R5)   ;SET 800 BPI
013020  052765  000010  000010          BIS     #BAI,CS2(R5)            ;INHIBIT BUS ADDRESS INCREMENT
013026  004337  005554                  JSR     R3,@#TMCMD              ;READ 32" OF TAPE-FORWARD
013032  015700                          .WORD   RDBUF
013034  177777                          .WORD   -1.
013036  063440                  10$:     .WORD   26400.                  ;FRAME COUNT
013040  000070                          .WORD   RDFWD
013042  005215                          INC     (R5)                    ;SET 'GO' BIT

013044  022710  001440          1$:     CMP     #800.,(R0)              ;WAIT FOR FIRST 800 FRAMES
013050  101375                          BHI     1$                      ;TO BE READ

013052  004767  171276                  JSR     PC,TIMON                ;TURN TIMER ON
013056  023710  013036          2$:     CMP     @#10$,(R0)              ;WAIT FOR READING TO FINISH
013062  103402                          BLO     3$
013064  000163  004444                  JMP     TIMER(R3)               ;GO TO TIMER & RETURN VIA R2

013070  012700  000005          3$:     MOV     #5,R0                   ;DIVIDE TIME BY 32.
013074  006204                  4$:     ASR     R4
013076  005300                          DEC     R0
013100  001375                          BNE     4$
013102  004767  172066                  JSR     PC,RHINIT               ;INIT DRIVE
013106  004767  171370                  JSR     PC,TIMOK                ;CHECK TIME
013112  000401                          BR      100$

013114  104400                  99$:     HLT
013116  104000                  100$:    SCOPE

                                        ;TEST 032-SKEW TAPE SPEED TEST-REVERSE
                                        ;THIS TEST READS FORWARD 40" (32000. FRAMES) OF TAPE. THEN READS REVERSE
                                        ;32" (26400.-900. = 25600. FRAMES) OF TAPE. THE TIME IS THEN DIVIDED BY
                                        ;32. TO GET TIME TO READ 1" (800. FRAMES) OF TAPE.
013120  112737  000032  001122  TST032: MOVB    #32,@#TSTNUM
013126  012702  013254                  MOV     #3$,R2                  ;SET RETURN PC FROM TIMER
013132  004767  172200                  JSR     PC,.REWIND              ;REWIND SLAVE
013136  102465                          BVS     99$                     ;BRANCH IF ERROR ON REWIND
013140  052765  001300  000032          BIS     #BPI800+NORM11,TC(R5)
013146  052765  000010  000010          BIS     #BAI,CS2(R5)
013154  004337  005554                  JSR     R3,@#TMCMD              ;READ FORWARD 32000. FRAMES
013160  015700                          .WORD   RDBUF
013162  177777                          .WORD   -1.                     ;WORD COUNT
013164  076400                  10$:     .WORD   32000.                  ;FRAME COUNT
013166  000070                          .WORD   RDFWD                   ;READ FORWARD
013170  005215                          INC     (R5)                    ;SET 'GO' BIT
```

```
3391  013172  023710  013164          1$:     CMP     @#10$,(R0)
3392  013176  101375                          BHI     1$
3393
3394  013200  004767  171770                  JSR     PC,RHINIT               ;INIT DRIVE
3395  013204  004767  171530                  JSR     PC,DELAY                ;WAIT FOR TAPE MOTION TO STOP
3396  013210  052765  001300  000032          BIS     #8PI800+NORM11,TC(R5)   ;SET 800 BPI
3397  013216  052765  000010  000010          BIS     #BAI,CS2(R5)            ;INHIBIT BUS ADDRESS INCREMENT
3398  013224  004337  005554                  JSR     R3,@#TMCMD              ;READ REVERSE 32" OF TAPE
3399  013230  015700                          .WORD   RDBUF                   ;READ BUFFER
3400  013232  177777                          .WORD   -1.                     ;WORD COUNT
3401  013234  063440          11$:            .WORD   26400.                  ;FRAME COUNT
3402  013236  000076                          .WORD   RDREV                   ;READ REVERSE
3403  013240  005215                          INC     (R5)                    ;SET 'GO' BIT
3404
3405  013242  022710  001440          2$:     CMP     #800.,(R0)              ;WAIT FOR FIRST 800 FRAMES
3406  013246  101375                          BHI     2$                      ;TO BE READ
3407
3408  013250  004767  171100                  JSR     PC,TIMON                ;TURN TIMER ON
3409  013254  023710  013234          3$:     CMP     @#11$,(R0)              ;WAIT FOR ALL FRAMES TO BE READ
3410  013260  103402                          BLO     4$
3411  013262  000163  004444                  JMP     TIMER(R3)               ;GO TO TIMER & RETURN VIA R2
3412
3413  013266  012700  000005          4$:     MOV     #5,R0                   ;DIVIDE TIME BY 32.
3414  013272  006204                  5$:     ASR     R4
3415  013274  005300                          DEC     R0
3416  013276  001375                          BNE     5$
3417  013300  004767  171670                  JSR     PC,RHINIT
3418  013304  004767  171172                  JSR     PC,TIMOK
3419  013310  000401                          BR      100$
3420
3421  013312  104400          99$:    HLT
3422  013314                  100$:
3423  013314  004767  172016                  JSR     PC,.REWIND              ;REWIND SLAVE
3424  013320  102774                          BVS     99$                     ;BRANCH IF ERROR ON REWIND
3425  013322  104000                          SCOPE
3426
3427  013324  000137  012616                  JMP     @#FINISH
3428
3429
3430
```

```
3431                                                    .SBTTL    PROGRAM MESSAGES
3432                                                ;OPERATOR INSTRUCTIONS
3433  013330  005015  042524  033061  M.NAM:  .ASCIZ  <CR><LF>'TE16 DRIVE FUNCTION TIMER (DZTUG-B)'
3434  013336  042040  044522  042526
3435  013344  043040  047125  052103
3436  013352  047511  020116  044524
3437  013360  042515  020122  042050
3438  013366  052132  043525  041055
3439  013374  000051
3440  013376  005015  054524  042520  I.REG:  .ASCIZ  <CR><LF>'TYPE FIRST ADDRESS OF CONTROLLER  '
3441  013404  043040  051111  052123
3442  013412  040440  042104  042522
3443  013420  051523  047440  020106
3444  013426  047503  052116  047522
3445  013434  046114  051105  020040
3446  013442     000
3447  013443     124  050131  020105  I.DRVS: .ASCIZ  %TYPE TM02 DRIVE #'S TO BE TESTED %
3448  013450  046524  031060  042040
3449  013456  044522  042526  021440
3450  013464  051447  052040  020117
3451  013472  042502  052040  051505
3452  013500  042524  020104     000
3453  013505     106  051117  052040  I.SLVS: .ASCII  'FOR TM02 DRIVE '
3454  013512  030115  020062  051104
3455  013520  053111  020105
3456  013524  026460  052040  050131  I.DRV:  .ASCIZ  %0- TYPE SLAVE #'S TO BE TESTED %
3457  013532  020105  046123  053101
3458  013540  020105  023443  020123
3459  013546  047524  041040  020105
3460  013554  042524  052123  042105
3461  013562  000040
3462  013564  050123  042505  020104  I.SKEW: .ASCIZ  'SPEED TESTS ONLY? (YES/NO = 1/0)'
3463  013572  042524  052123  020123
3464  013600  047117  054514  020077
3465  013606  054450  051505  047057
3466  013614  020117  020075  027461
3467  013622  024460     000
3468  013625     116  055122  047440  I.NRZ:  .ASCIZ  'NRZ ONLY? (YES/NO = 1/0)'
3469  013632  046116  037531  024040
3470  013640  042531  027523  047516
3471  013646  036440  030440  030057
3472  013654  000051
3473  013656  005015  047105  020104  M.EOT:  .ASCIZ  <CR><LF>'END OF TAPE'<CR><LF>
3474  013664  043117  052040  050101
3475  013672  006505  000012
3476
3477                                                ;ERROR MESSAGES
3478  013676  005015  051124  050101  E.TRP4: .ASCIZ  <CR><LF>'TRAPPED TO 4'
3479  013704  042520  020104  047524
3480  013712  032040     000
3481  013715     116  020117  047503  E.NCON: .ASCIZ  'NO CONTROLLER AT ADDRESS SPECIFIED'<CR><LF>
3482  013722  052116  047522  046114
3483  013730  051105  040440  020124
3484  013736  042101  051104  051505
3485  013744  020123  050123  041505
3486  013752  043111  042511  006504
```

```
3487  013760  000012
3488  013762  046524  031060  042040  E.NDRV: .ASCIZ  'TM02 DRIVE '
3489  013770  044522  042526  000040
3490  013776  051104  053111  020105  E.NSLV: .ASCII  'DRIVE '
3491  014004  020060  046123  053101  E.DRV:  .ASCII  '0 SLAVE '
3492  014012  020105
3493  014014  020060  047516  020124  E.NAVA: .ASCIZ  '0 NOT AVAILABLE FOR TEST'<CR><LF>
3494  014022  053101  044501  040514
3495  014030  046102  020105  047506
3496  014036  020122  042524  052123
3497  014044  005015      000
3498  014047      116 020117  046524  E.UNIT: .ASCIZ  'NO TM02/TE16 UNIT FOUND TO TEST'<CR><LF>
3499  014054  031060  052057  030505
3500  014062  020066  047125  052111
3501  014070  043040  052517  042116
3502  014076  052040  020117  042524
3503  014104  052123  005015      000
3504  014111      123 043117  020124  E.SFT:  .ASCIZ  'SOFT ERROR (DATA)'<CR><LF>
3505  014116  051105  047522  020122
3506  014124  042050  052101  024501
3507  014132  005015      000
3508  014135      124 051505  020124  E.HDR:  .ASCIZ  'TEST # '
3509  014142  020043      000
3510  014145      040 042504  044526  E.HDR1: .ASCII  ' DEVICE ERROR'<CR><LF>
3511  014152  042503  042440  051122
3512  014160  051117  005015
3513  014164  051503  004461  041527          .ASCIZ  'CS1'<HT>'WC'<HT>'BA'<HT>'FC'<HT>'CS2'<HT>'DS'<HT>'ER'<HT>'TC'<CR><LF>
3514  014172  041011  004501  041506
3515  014200  041411  031123  042011
3516  014206  004523  051105  052011
3517  014214  006503  000012
3518  014220  047440  052125  047440  E.HDR2: .ASCIZ  ' OUT OF RANGE ERROR'<CR><LF>
3519  014226  020106  040522  043516
3520  014234  020105  051105  047522
3521  014242  006522  000012
3522  014246  005015  044524  042515  E.TIMOV:        .ASCIZ  <CR><LF>'TIMER OVERFLOWED'<CR><LF>
3523  014254  020122  053117  051105
3524  014262  046106  053517  042105
3525  014270  005015      000
3526  014273      015 052012  046511  E.TIMEX:        .ASCIZ  <CR><LF>'TIME EXPIRED WAITING FOR RDY'<CR><LF>
3527  014300  020105  054105  044520
3528  014306  042522  020104  040527
3529  014314  052111  047111  020107
3530  014322  047506  020122  042122
3531  014330  006531  000012
3532  014334  043440  050101  021440  E.GAP:  .ASCIZ  ' GAP # '
3533  014342  000040
3534
3535                                  ;TIME DOCUMENT LINES
3536  014344  025052  025052  025052  L.HDR1: .ASCIZ  '*********************************************************************************
3537  014352  025052  025052  025052
3538  014360  025052  025052  025052
3539  014366  025052  025052  025052
3540  014374  025052  025052  025052
3541  014402  025052  025052  025052
3542  014410  025052  025052  025052
```

```
3543  014416  025052  025052  025052
3544  014424  025052  025052  025052
3545  014432  025052  025052  025052
3546  014440  025052  025052  025052
3547  014446  025052  025052  025052
3548  014454  005015     000
3549  014457     052  052040  030115  L.HDR2: .ASCII  '* TM02 DRIVE FUNCTION TIMES- DRIVE # '
3550  014464  020062  051104  053111
3551  014472  020105  052506  041516
3552  014500  044524  047117  052040
3553  014506  046511  051505  020055
3554  014514  051104  053111  020105
3555  014522  020043
3556  014524  020060  046123  053101  L.DRV:  .ASCII  '0 SLAVE # '
3557  014532  020105  020043
3558  014536  020060     040          L.SLV:  .ASCII  '0  '
3559  014541     071  041440  040510  L.CHAN: .ASCIZ  '9 CHAN. SER # '
3560  014546  027116  051440  051105
3561  014554  021440  000040
3562  014560  006440  025012  005015  L.HDR3: .ASCII  ' '<CR><LF>'*'<CR><LF>
3563  014566  020052  052506  041516          .ASCIZ  '* FUNCTION'<HT><HT>'TIME(SPECIFICATION)'<HT>'TIME(ACTUAL)'<CR><LF>
3564  014574  044524  047117  004411
3565  014602  044524  042515  051450
3566  014610  042520  044503  044506
3567  014616  040503  044524  047117
3568  014624  004451  044524  042515
3569  014632  040450  052103  040525
3570  014640  024514  005015     000
3571
3572  014645     122  047101  042507  L.RNG:  .ASCIZ  'RANGE=<'
3573  014652  036075     000
3574  014655     101  052103  040525  L.ACT:  .ASCIZ  'ACTUAL='
3575  014662  036514     000
3576
3577                                  ;TEST DESCRIPTOR HEADERS
3578  014665     052  053440  044522  A.T001: .ASCIZ  '* WRITE FROM BOT'<HT>
3579  014672  042524  043040  047522
3580  014700  020115  047502  004524
3581  014706     000
3582  014707     052  053440  044522  A.T002: .ASCIZ  '* WRITE START'<HT><HT>
3583  014714  042524  051440  040524
3584  014722  052122  004411     000
3585  014727     052  053440  044522  A.T003: .ASCIZ  '* WRITE SHUTDOWN'<HT>
3586  014734  042524  051440  052510
3587  014742  042124  053517  004516
3588  014750     000
3589  014751     052  053440  044522  A.T004: .ASCIZ  '* WRITE SETTLEDOWN'<HT>
3590  014756  042524  051440  052105
3591  014764  046124  042105  053517
3592  014772  004516     000
3593  014775     052  051040  040505  A.T005: .ASCIZ  '* READ FROM BOT'<HT><HT>
3594  015002  020104  051106  046517
3595  015010  041040  052117  004411
3596  015016     000
3597  015017     052  051040  040505  A.T006: .ASCIZ  '* READ START'<HT><HT>
3598  015024  020104  052123  051101
```

```
3599  015032  004524  000011
3600  015036  020052  042522  042101  A.T007: .ASCIZ  '* READ SHUTDOWN'<HT><HT>
3601  015044  051440  052510  042124
3602  015052  053517  004516  000011
3603  015060  020052  042522  042101  A.T010: .ASCIZ  '* READ SETTLEDOWN'<HT>
3604  015066  051440  052105  046124
3605  015074  042105  053517  004516
3606  015102      000
3607  015103      052  051040  040505  A.T011: .ASCIZ  '* READ REV START'<HT>
3608  015110  020104  042522  020126
3609  015116  052123  051101  004524
3610  015124      000
3611  015125      052  051040  040505  A.T012: .ASCIZ  '* READ REV SHUTDOWN'<HT>
3612  015132  020104  042522  020126
3613  015140  044123  052125  047504
3614  015146  047127  000011
3615  015152  020052  042522  042101  A.T013: .ASCIZ  '* READ REV SETTLEDOWN'<HT>
3616  015160  051040  053105  051440
3617  015166  052105  046124  042105
3618  015174  053517  004516      000
3619  015201      052  052040  051125  A.T014: .ASCIZ  '* TURN AROUND DELAY F-R'<HT>
3620  015206  020116  051101  052517
3621  015214  042116  042040  046105
3622  015222  054501  043040  051055
3623  015230  000011
3624  015232  020052  052524  047122  A.T015: .ASCIZ  '* TURN AROUND DELAY R-F'<HT>
3625  015240  040440  047522  047125
3626  015246  020104  042504  040514
3627  015254  020131  026522  004506
3628  015262      000
3629  015263      052  043440  050101  A.T016: .ASCIZ  '* GAP SIZE-STOP HALF'<HT>
3630  015270  051440  055111  026505
3631  015276  052123  050117  044040
3632  015304  046101  004506      000
3633  015311      052  043440  050101  A.T017: .ASCIZ  '* GAP SIZE-START HALF'<HT>
3634  015316  051440  055111  026505
3635  015324  052123  051101  020124
3636  015332  040510  043114  000011
3637  015340  020052  040507  020120  A.T020: .ASCIZ  '* GAP SIZE-INTERRECORD'<HT>
3638  015346  044523  042532  044455
3639  015354  052116  051105  042522
3640  015362  047503  042122  000011
3641  015370  020052  040507  020120  A.T021: .ASCIZ  '* GAP CONSISTANCY'<HT>
3642  015376  047503  051516  051511
3643  015404  040524  041516  004531
3644  015412      000
3645  015413      052  042040  052101  A.T023: .ASCIZ  '* DATA TIME-200BPI'<HT>
3646  015420  020101  044524  042515
3647  015426  031055  030060  050102
3648  015434  004511      000
3649  015437      052  042040  052101  A.T024: .ASCIZ  '* DATA TIME-556BPI'<HT>
3650  015444  020101  044524  042515
3651  015452  032455  033065  050102
3652  015460  004511      000
3653  015463      052  042040  052101  A.T025: .ASCIZ  '* DATA TIME-800BPI'<HT>
3654  015470  020101  044524  042515
```

```
3655  015476  034055  030060  050102
3656  015504  004511     000
3657  015507     052  042040  052101  A.T026: .ASCIZ  '* DATA TIME-1600BPI'<HT>
3658  015514  020101  044524  042515
3659  015522  030455  030066  041060
3660  015530  044520  000011
3661  015534  020052  051105  051501  A.T027: .ASCIZ  '* ERASE GAP TIME'<HT>
3662  015542  020105  040507  020120
3663  015550  044524  042515  000011
3664  015556  020052  051127  052111  A.T030: .ASCIZ  '* WRITE FILE MARK'<HT>
3665  015564  020105  044506  042514
3666  015572  046440  051101  004513
3667  015600     000
3668  015601     052  052040  050101  A.T031: .ASCIZ   * TAPE SPEED-FWD'<HT>
3669  015606  020105  050123  042505
3670  015614  026504  053506  004504
3671  015622     000
3672  015623     052  052040  050101  A.T032: .ASCIZ  '* TAPE SPEED-REV'<HT>
3673  015630  020105  050123  042505
3674  015636  026504  042522  004526
3675  015644     000
3676
3677  015645     015  057012  000107  L.CNTG: .ASCIZ  <CR><LF>'↑G'
3678  015652  005015  053523  036522  L.SWR:  .ASCIZ  <CR><LF>'SWR='
3679  015660     000
3680  015661     040  047040  053505  L.NEW:  .ASCIZ  '  NEW= '
3681  015666  020075     000
3682  015671     015  037412  005015  L.QUEST:        .ASCIZ  <CR><LF>'?'<CR><LF>
3683  015676     000
3684          015700                          .EVEN
3685          015700                          RDBUF=.
3686          015700                          WTBUF=.
3687  015700  000200                          .BLKW   128.
3688          000001                          .END
```

```
A       010620
ACCL  = 100000
ANGTAB  001412
AS    = 000016
ASFLG   001130
ATA   = 100000
ATIME   001012
ATIMTB  001014
A.T001  014665
A.T002  014707
A.T003  014727
A.T004  014751
A.T005  014775
A.T006  015017
A.T007  015036
A.T010  015060
A.T011  015103
A.T012  015125
A.T013  015152
A.T014  015201
A.T015  015232
A.T016  015263
A.T017  015311
A.T020  015340
A.T021  015370
A.T023  015413
A.T024  015437
A.T025  015463
A.T026  015507
A.T027  015534
A.T030  015556
A.T031  015601
A.T032  015623
A16   = 000400
A17   = 001000
BA    = 000004
BAI   = 000010
BEGIN   006764
BELL    001403
BKSLSH  001377
BOT   = 000002
BPI200= 000000
BPI556= 000400
BPI800= 001000
BPTVEC= 000014
CDM11 = 000320
CHKDRV  006270
CHKSLV  006574
CH7   = 010000
CKSWR   001770
CLR   = 000040
CNTLU   002030
CNTRLC= 000003
CNTRLO= 000017
CNTRLU= 000025
CNVDEC  002730
CNVOCT  002622
CNVTAO  003260
CNVTD   002742
CNVTO   002634
COUNT   001764
CR    = 000015
CRLF    001374
CSITM = 002000
CS1   = 000000
CS2   = 000010
DASH    001405
DB    = 000022
DCONST  003036
DELAY   004740
DELAYV  004770
DELTIM  001114
DIGTAB  001132
DIVIDE  005026
DLT   = 100000
DPR   = 000400
DRIVES  006140
DRVAVA  005116
DRVNUM  001004
DRVTBL  001154
DRY   = 000200
DRYCLR= 000010
DS    = 000012
DT    = 000026
DTE   = 010000
DVA   = 004000
DVO   = 000000
DV1   = 000001
DV2   = 000002
DV3   = 000003
DV4   = 000004
DV5   = 000005
DV6   = 000006
DV7   = 000007
ECHO    001401
EMTVEC= 000030
END     012732
EOT   = 002000
ER    = 000014
ERASE = 000024
ERFLG   001123
ERR   = 040000
ERRTRP  003650
ERRVEC= 000004
E.DRV   014004
E.GAP   014334
E.HDR   014135
E.HDR1  014145
E.HDR2  014220
E.NAVA  014014
E.NCON  013715
E.NDRV  013762
E.NSLV  013776
E.SFT   014111
E.TIME  014273
E.TIMO  014246
E.TRP4  013676
E.UNIT  014047
FC    = 000006
FCE   = 001000
FINISH  012616
FMT   = 000020
FPEVEC= 000244
FRMCNT= 177400
FWDSPC  005472
GAP     001120
GAPOK   004612
GAPTBL  001054
GO    = 000001
GTIMTB  001572
HLT   = 104400
HT    = 000011
IDB   = 000010
IE    = 000100
ILF   = 000001
ILR   = 000002
INBUF   001264
INCVAE= 000100
INIT    005724
IOTVEC= 000020
IR    = 000100
ITCNT   001121
I.DRV   013524
I.DRVS  013443
I.NRZ   013625
I.REG   013376
I.SKEW  013564
I.SLVS  013505
LF    = 000012
LKS   = 177546
LKVEC = 000100
LPB   = 177516
LPS   = 177514
L.ACT   014655
L.CHAN  014541
L.CNTG  015645
L.DRV   014524
L.HDR1  014344
L.HDR2  014457
L.HDR3  014560
L.NEW   015661
L.QUES  015671
L.RNG   014645
L.SLV   014536
L.SWR   015652
MCPE  = 020000
MDPE  = 000400
MMVEC = 000250
MOL   = 010000
MR    = 000024
MXF   = 001000
M.EOT   013656
M.NAM   013330
NAMPTR  001672
NED   = 010000
NEF   = 004000
NEM   = 004000
NOP   = 000000
NORM11= 000300
NRZFLG  001127
NSG   = 000400
OCTALO  001116
ODIGIT  001144
OPI   = 020000
OR    = 000200
OSC   = 000100
OUT     002240
OUTBUF= 005724
OUTGAP  003146
OUTSPC  003052
PARVEC= 000114
PAT   = 000020
PEFLRC= 000200
PES   = 000040
PE1600= 002000
PFVEC = 000024
PGE   = 002000
PIP   = 020000
PIRQ  = 177772
PIRVEC= 000240
PLKCSR= 172540
PLKVEC= 000104
PRGFLG  001124
PSEL  = 002000
PSW   = 177776
PUBLIS  003346
RDBUF = 015700
RDFWD = 000070
RDREV = 000076
RDSW    001766
RDY   = 000200
READ    005436
RESVEC= 000010
REVRD   005454
RHINIT  005174
RMR   = 000004
RWD   = 000006
RWDOFF= 000002
R10   =%000000
R11   =%000001
R12   =%000002
R13   =%000003
R14   =%000004
R15   =%000005
SC    = 100000
SCOPE = 104000
SCPADR  001002
SDWN  = 000020
SKEWTS  012764
SLA   = 000001
SLAVES  006370
SLR   = 177774
SLVAVA  005144
SLVNUM  001005
SLVPTR  001006
SLVTBL  001164
SN    = 000030
SNPT    005574
SPACE   001410
SPACE2  001407
SPCFWD= 000030
SPCREV= 000032
SPR   = 002000
SSC   = 000100
STIMTB  001416
STKPTR= 000600
SUSWR   005730
SWR     001000
SWREG   000176
SW06  = 000100
SW07  = 000200
SW08  = 000400
SW09  = 001000
SW10  = 002000
SW11  = 004000
SW13  = 020000
SW14  = 040000
SW15  = 100000
TAP   = 040000
TBITVE= 000014
TC    = 000032
TCRLF   002360
TEMPST  001762
TIB     001760
TIMER   004444
```

DZTUGB.P11 15-JUL-77 12:52 SYMBOL TABLE

```
TIMERR  004454          TSTNUM  001122          TST023  011672          TYPOCT  002630          $CNTRL  002325
TIMER0  004470          TST001  007336          TST024  012014          UBREAK= 177770          $CRLF   002326
TIMER1  004420          TST002  007422          TST025  012144          UNS    = 040000          $FILL   002315
TIMOK   004502          TST003  007500          TST026  012274          UNTFND  001125          $HT    = 000011
TIMON   004354          TST004  007570          TST027  012424          UPE    = 020000          $NULL   002314
TKB    = 177562         TST005  007676          TST030  012510          WAITRD  005232          $TKFLG  002317
TKISR   003606          TST006  007762          TST031  012772          WAITTI  005234          $TPB    002322
TKS    = 177560         TST007  010046          TST032  013120          WC     = 000002          $TPFLG  002316
TKVEC  = 000060         TST010  010146          TTIN    002242          WCE    = 040000          $TPS    002320
TMBASE  001010          TST011  010266          TTIN1   002262          WCHKF  = 000050          .      = 016300
TMCMD   005554          TST012  010364          TTIN2   002276          WCHKR  = 000056          .HLT    003652
TMCS1  = 172440         TST013  010474          TYPDEC  002736          WFMK   = 000026          .INPUT  003474
TMK    = 000004         TST014  010634          TYPE   = 000004         WFWD   = 000060          .RESTO  002600
TPB    = 177566         TST015  010726          TYPE1   002352          WRDCNT= 177600          .REWIN  005336
TPS    = 177564         TST016  011034          TYPE2   002400          WRITE   005420          .SAVE   002556
TPVEC  = 000064         TST017  011130          TYPE3   002406          WRL    = 004000          .SCOPE  004126
TRAPVE= 000034          TST020  011240          TYPE4   002412          WRT.BK  005512          .TYPE   002332
TRE    = 040000         TST021  011360          TYPFLG  001126          WTBUF  = 015700
TRTVEC= 000014          TST022  011664          TYPHDR  007210          $CHARC  002324

. ABS.  016300      000

 ERRORS DETECTED:  0

 DZTUGB,DZTUGB.SEQ/SOL/NL:TOC/DOC=DZTUGB.P11
 RUN-TIME: 4 7 .3 SECONDS
 RUN-TIME RATIO: 179/12=14.7
 CORE USED:  7K  (13 PAGES)

 DOCUMENT PAGES:  87

EOF1DZTUGBSEQ              00010000      770804          PDP10 411
```