TM03/TE16

DRIVE FUNCTION TIMER

MD-11-DZTEE-A

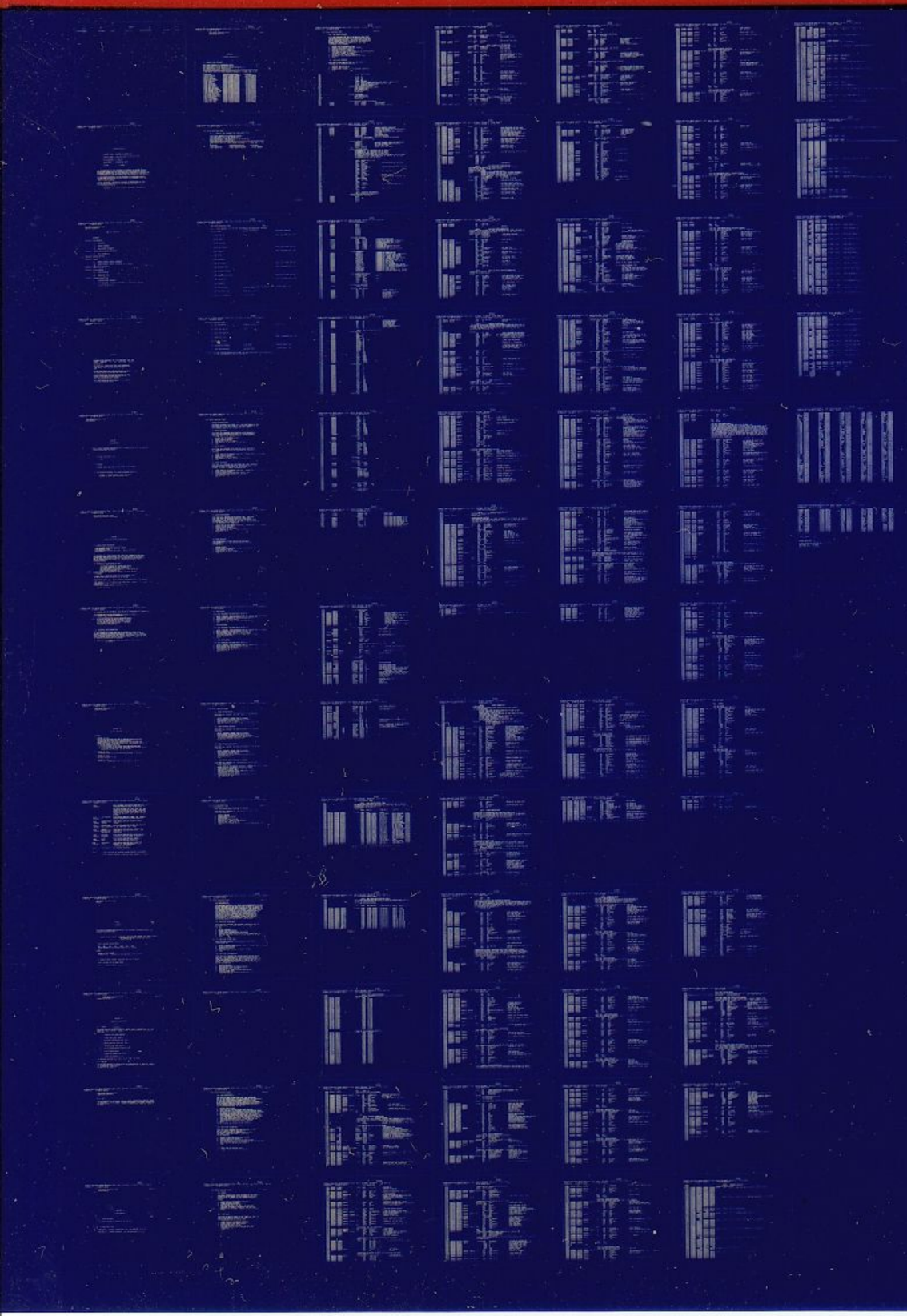
EP-DZTEE-A-DL-A

FICHE 1 OF 1

JUN 1977

digital

MADE IN USA



B01

E0F1DZTCBDSE0	00010000	770629	PDP10 411	BDHDR1DZTEEASE0	00010000	770629

```
.REM    %
```

IDENTIFICATION

PRODUCT CODE: MAINDEC-11-DZTEE-A-D
PRODUCT NAME: TM03/TE16 DRIVE FUNCTION TIMER
DATE CREATED: 15 FEB 77
MAINTAINER: DIAGNOSTIC GROUP
AUTHOR: J. G. ADAMS

TM03 DRIVE FUNCTION TIMER PAGE 2
TABLE OF CONTENTS

TABLE OF CONTENTS

ABSTRACT

PROGRAM DZTEE MEASURES THE TIME REQUIRED AND GAP SIZES PRODUCED BY THE TM03/TE16 MAGTAPE DRIVE/SLAVE.

THE TEST WILL CHECK BOTH THE LOGIC GENERATED TIME DELAYS, AND THE DISTANCES TRAVELED BY THE TAPE.

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 AN ERROR IS DATA RELATED(PARITY; ETC) THEY ARE PRINTED AS SOFT ERRORS.

IF A 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 TM03/TE16 CONTROLLER/MAGTAPE STATIONS.

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-DZTEA CONTROL LOGIC TEST(PART 1)
MAINDEC-11-DZTEC BASIC FUNCTION TEST

CHAPTER 2

LOADING AND STARTING PROCEDURE

2.0 LOAD & START PROCEDURE:

LOAD PROGRAM USING THE ABSOLUTE LOADER
LOAD ADDRESS = 200
SET OPERATING SWITCHES SEE CHAPT 3 SWITCH SETTINGS
PRESS START

THE PROGRAM WILL THEN REQUEST THE FIRST BUS ADDRESS OF THE RHXX CONTROLLER, TM03 DRIVES TO BE TESTED, TE16 SLAVES TO BE TESTED, AND IF SPEED TESTS ARE TO BE RUN. IN ADDITON TO EACH REQUEST A DEFAULT ANSWER WILL BE TYPED. TO INVOKE THE DEFAULT TYPE A CARRIAGE RETURN.

THE REQUESTS & THEIR DEFAULTS ARE:

TYPE FIRST ADDRESS OF CONTROLLER:172440
TYPE TM03 DRIVE #'S TO BE TESTED:ALL
FOR TM03 DRIVE X-TYPE SLAVE #'S TO BE TESTED:ALL
SPEED TESTS ?(YES/NO):NO

NOTES: SLAVE #'S ARE NOT REUESTED IF DEFAULT TO DRIVE REQUEST IS INVOKED.

IF MORE THAN 1 DRIVE OR SLAVE IS TO BE TESTED, TYPE BETWEEN EACH DRIVE OR SLAVE # TO BE TESTED.

SPEED TESTS CAN & WILL ONLY BE RUN BY ANSWERING YES TO THE REQUEST.

TYPE CONTROL U (↑U) TO DELETE LINE TYPED;TYPE 'RUBOUT' TO DELETE LAST CHARACTER(S).
PROGRAM WILL REPORT ERRORS, AND END OF PASS.

2.1 RESTART PROCEDURE

THE PROGRAM MAY BE RESTARTED USING START UP PARAMETERS AT ADDRESS 210.

THE PROGRAM MAY ALSO BE RESTARTED BY TYPING A CONTROL C (↑C).
A ↑C RESTART WILL REQUEST PARAMETERS.

NOTE: AFTER RESTARTING THE SWITCH REGISTER SHOULD
BE SET TO PROGRAM SWITCH SETTINGS. IF 210
IS LEFT AS THE SWITCH SETTING THE PROGRAM
WILL SELECT & RUN TEST 10 ONLY. SEE SWITCH
SETTINGS FOR EXPLANATION.

2.2 AUTOMATIC MODE OPERATION

IF THE PROGRAM IS LOADED AND RUN IN AUTOMATIC (CHAIN) MODE
DEFAULT RESPONSES TO OPERATOR REQUESTS ARE USED, AND ALL AVAIL-
ABLE TM03/TE16 COMBINATIONS ARE TESTED. ADDITIONALLY THE SOFTWARE
SWR IS INVOKED WITH A SWITCH SETTING OF 100000 IF LOADED VIA ACT11.
NO OPERATOR INTERVENTION IS REQUIRED

** EXCEPTION: IF LOADED VIA TMDP TM03 DRIVE 0 TE16 SLAVE 0 IS
NOT TESTED.

CHAPTER 3

SWITCH SETTINGS

CONTROL:

1) CONTROL G (↑G):
   SELECTS THE SOFTWARE SWR AND ALLOWS NEW SWITCH SETTINGS.
   THE MACHINE WILL THEN TYPE: SWR=XXXXXXNEW=
   WHERE: XXXXXX IS THE OCTAL CONTENTS OF THE SOFTWARE SWR.
   AFTER THE ''NEW='' HAS BEEN TYPED THEN THE OPERATOR CAN DO ONE
   OF THE FOLLOWING AT THE TTY:
     A) TYPE A NUMBER TO BE LOADED INTO THE SOFTWARE SWR.
     B) IF A (CR) IS THE FIRST KEY DEPRESSED THE SOFTWARE SWITCH
        REGISTER CONTENTS WILL NOT BE CHANGED.

2) CONTROL A (↑A):
   ALTERNATES USAGE OF THE SWR BETWEEN HARDWARE & SOFTWARE.

3) CONTROL C (↑C):
   RESTARTS PROGRAM AT 200

4) CONTROL U (↑U):
   DELETES ALL CHARACTERS TYPED IN RESPONSE TO A REQUEST.

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.
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.
SW08 (000900)	PRINT TIME	THIS SWITCH WHEN SET WILL PRINT A TIME LINE AFTER EACH ITERATION.
SW06 (000100)	CONTINUOUS CYCLE	THIS SWITCH WHEN SET WILL CAUSE THE PROGRAM TO RUN CONTINUOUSLY UNTIL STOPPED BY THE OPERATOR.
SW5-0	TEST SELECT	RUN SUBTEST SELECTED

NOTE: A TEST CAN ONLY BE SELECTED DURING STARTUP (OR RESTART).

DO NOT INHIBIT SUBTEST ITERATIONS WHEN PROGRAM IS RUNNING.

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 AS SOFT ERRORS 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>=1
3. PROVIDES CONTINUOUS LOOP <SW14>
4. MOVES FUNCTION TIME INTO TABLE
5. OUTPUTS LINE ITEM <SW10>=1
6. DELAYS 350MS BEFORE STARTING TEST
7. INIT'S DRIVE/SLAVE
8. CLEARS THE ERROR FLAG (ERFLG)
9. CHECK FOR CONTROL G (↑G)

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

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 THE TIME RECORDED BY THE SUBTEST.

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.

CHAPTER 6
MISCELLANEOUS

6.1 STACK POINTER

THE STACK POINTER IS INITAILLY SET TO 500.

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 TM03 DRIVE #'S TO BE TESTED:ALL 0
FOR TM03 DRIVE 0- TYPE SLAVE #'S TO BE TESTED:ALL 0
TAPE SPEED TESTS ONLY? (YES/NO):NO
********************************************************************************
* TM03 DRIVE FUNCTION TIMES- DRIVE # 0 SLAVE # 7 9 CHAN. SER. # 5009
*
* FUNCTION                TIME(SPECIFICATION)     TIME(ACTUAL)
* WRITE FROM BOT          RANGE=<176000-172000>       ACTUAL=174740
* WRITE START             RANGE=<009500-008700>       ACTUAL=009120
* WRITE SHUTDOWN          RANGE=<008500-007500>       ACTUAL=008840
* WRITE SETTLEDOWN        RANGE=<013500-007300>       ACTUAL=010970
* READ FROM BOT           RANGE=<045000-041000>       ACTUAL=043580
* READ START              RANGE=<003200-002400>       ACTUAL=002740
* READ SHUTDOWN           RANGE=<004100-003050>       ACTUAL=004360
* READ SETTLEDOWN         RANGE=<013500-007300>       ACTUAL=010970
* READ REV START          RANGE=<003200-002400>       ACTUAL=002740
* READ REV SHUTDOWN       RANGE=<003700-003300>       ACTUAL=003520
* READ REV SETTLEDOWN     RANGE=<013500-007300>       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=<014300-012600>       ACTUAL=014500
* GAP CONSISANCY          RANGE=<013500-011800>       ACTUAL=013040-
* DATA TIME-800BPI        RANGE=<024000-022000>       ACTUAL=023400
* DATA TIME-1600BPI       RANGE=<025100-024100>       ACTUAL=024470
* ERASE GAP TIME          RANGE=<101000-098000>       ACTUAL=099510
* WRITE FILE MARK         RANGE=<104000-102000>       ACTUAL=103990
```

7.1.1 SAMPLE TIME DOCUMENT FOR TAPE SPEED TESTS

```
TYPE FIRST ADDRESS OF CONTROLLER 172440:
TYPE TM03 DRIVE #'S TO BE TESTED:ALL 0
FOR TM03 DRIVE 0- TYPE SLAVE #'S TO BE TESTED:ALL 7
SPEED TESTS ONLY? (YES/NO):NO Y
*************************************************************************
#TM03 DRIVE FUNCTION TIMES- DRIVE # 0 SLAVE # 7 9 CHAN. SER. # 5009
#
#FUNCTION                TIME(SPECIFICATION)     TIME(ACTUAL)
#TAPE SPEED FWD          RANGE=<022700-021700>       ACTUAL=022500
#TAPE SPEED REV          RANGE=<022700-021700>       ACTUAL=022500
```

TM03 DRIVE FUNCTION TIMER PAGE 13

7.2 TEST SEQUENCE WITH RELATED ADJUSTMENTS AND ASSOCIATED HARDWARE

TEST NO./NAME	RELATED ADJUSTMENTS	ASSOCIATED HARDWARE
1. WRITE FROM BOT	*NONE	*M8911 ROM*M8903 ACCL CNTR
2. WRITE START	* "	* " * "
3. WRITE SHUTDOWN	* "	* " * "
4. WRITE SETTLEDOWN	* "	*M8910 SETTLEDOWN ONE SHOT
5. READ FROM BOT	* "	*M8911 ROM*M8903 ACCL CNTR
6. READ START	* "	* " * "
7. READ SHUTDOWN	* "	* " * "
10. READ SETTLEDOWN	* "	*M8910 SETTLEDOWN ONE SHOT
11. READ REVERSE START	* "	*M8911 ROM*M8903 ACCL CNTR
12. READ REVERSE SHUTDOWN	* "	* " * "
13. READ REVERSE SETTLEDOWN	* "	*M8910 SETTLEDOWN ONE SHOT
14. TURN AROUND F-R	* "	*M8911 ROM*M8903 ACCL CNTR
15. TURN AROUND R-F	* "	* " * "
16. GAP SIZE-STOP HALF	*FWRD/REV SPEED-START/STOP-RAMPS	*CAPSTAN SERVO LOOP
17. GAP SIZE-START HALF	*SAME AS IN TEST 16	* " " "
20. GAP SIZE INTERRECORD	*FWD/REV SPEED	* " " "

TM03 DRIVE FUNCTION TIMER PAGE 14

21. GAP CONSISTENCY	*SAME AS IN TEST 16	*WRITE CLOCK
22. DATA TIME 800 BPI	*NONE	* " "
23. DATA TIME 1600 BPI	* "	* " "
24. ERASE GAP TIME	* "	*M8911 ROM*M8903 ACCL CNTR
25. WRITE FILE MARK	* "	* " " * " " "
26. TAPE SPEED-FORWARD	*FWD SPEED	*CAPSTAN SERVO LOOP
27. TAPE SPEED-REVERSE	*REVERSE SPEED	*CAPSTAN SERVO LOOP

*****NOTE: IF TIME PROBLEMS APPEAR IN T1 THRU T25, RUN TAPE SPEED TESTS FIRST*******
TEST 26 & 27 REQUIRE AN 800 BPI SKEW TAPE

7.3 SUBTEST DESCRIPTIONS:

THE FIRST THIRTEEN (13) TESTS (T1 - T15) ARE CHECKS OF THE ROM CIRCUITS IN THE TE16 (M9811), THE ACCL COUNTER IN THE TM03 (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 INTERRECORD 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

T15. TURN AROUND DELAY-REVERSE TO FORWARD

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

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

GAP MEASUREMENTS:

THE PREVIOUS THIRTEEN (13) TESTS WERE MEASUREMENTS OF LOGIC DELAYS PERFORMED BY THE TM03 OR TE16 IN ORDER TO ALLOW FOR PROPER ACCELERATION AND DECELLERATION OF TAPE ACCORDING TO THE DESIRED INTERCORD GAP (.6 INCHES). THIS TEST, HOWEVER, WILL MEASURE THE PHYSICAL SIZE OF THE INTERCORD GAP THAT EXISTS ON TAPE AS A RESULT OF THE START/STOP TIMES OF THE CAPSTAN ITSELF. BECAUSE THE 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.

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 LISTING FOR GAP TIMES)

T22. DATA TIME AT 800 BPI:

THIS TEST WILL MEASURE THE TIME REQUIRED TO WRITE
ONE (1) INCH OF TAPE AT 800 BPI.
BY WRITING A RECORD OF ENOUGH FRAMES TO MOVE
THE TAPE 1 INCH (800 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

T23. DATA TIME AT 1600 BPI (PE):
     REPEAT STEPS 1 THRU 5 AT 1600 BPI.

T24. 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

T25. 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

T26. 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.

T27. 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.

```
    %
  981                                   .LIST   BIN,LOC,SEQ
  982
  983                                   .NLIST  MC
  984                                   .NLIST  TOC
  985                                   .LIST   ME
  986                                   .ENABLE ABS,AMA
  987                                   .MCALL  $CPVEC,$CPREG,$CATCH,$TYPE,.$ACT11,.$EOP,$CHAIN
  988                                   .TITLE  DZTEE-A TM03/TE16 DRIVE FUNCTION TIMER
  989                                   .SBTTL  STARTING INSTRUCTIONS
  990                                 ;LOADING AND STARTING PROCEDURE
  991                                 ;       LOAD PROGRAM USING ABS LOADER
  992                                 ;       LOAD ADDRESS 200
  993                                 ;       SET SWITCH OPTIONS
  994                                 ;       PRESS START
  995
  996                                 ;RESTART PROCEDURE
  997                                 ;       LOAD ADDRESS 210
  998                                 ;       SET SWITCH OPTIONS
  999                                 ;       PRESS START
 1000
 1001                                 ;SWITCH REGISTER SWITCH ASSIGNMENTS
 1002           100000                        SW15=   100000          ;HALT ON ERROR
 1003           040000                        SW14=   040000          ;LOOP SUBTEST
 1004           020000                        SW13=   020000          ;INHIBIT ERROR TYPE OUT
```

```
                                STARTING INSTRUCTIONS

004000                                  SW11=   004000          ;INHIBIT SUBTEST ITERATION
002000                                  SW10=   002000          ;INHIBIT PUBLISHING TIME SPECIFICATION
001000                                  SW09=   001000          ;RING BELL ON ERROR
000400                                  SW08=   000400          ;TYPE LINE ITEM AFTER EACH ITERATION
000200                                  SW07=   00200           ;NOT USED
000100                                  SW06=   000100          ;CONTINUOUS CYCLE
                                        SW05-SW00               ;RUN TEST SELECTED
                                ;       **NOTE: IF <SW15-SW00> = 177777 AT STARTUP USE SOFTWARE
                                ;               SWITCH REGISTER.
                                ;
                                ;CONSOLE COMMANDS
                                ;       CONTROL C               ;RESTART PROGRAM (SAME AS START @ 200)
                                ;       CONTROL G               ;SET NEW SOFTWARE SWITCH REGISTER
                                ;       CONTROL U               ;DELETE LINE TYPED
                                ;       RUBOUT (DELETE)         ;DELETE LAST CHAR TYPED
                                ;
                                ;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 CS1 (SET BY SCOPE)

                                        .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
000000                                  R0=%0
000001                                  R1=%1
000002                                  R2=%2
000003                                  R3=%3
000004                                  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

                    ;RH,TM03/TE16 REGISTERS
                        TMCS1=  172440

                    ;TM03/TE16 INDEX VALUES
                        CS1=    00                  ;CONTROL STATUS #1
                        WC=     02
                        BA=     04                  ;BUS ADDRESS REGISTER
                        FC=     06                  ;FRAME COUNT
                        CS2=    10                  ;CONTROL STATUS #2
                        DS=     12                  ;DRIVE STATUS
                        ER=     14                  ;ERROR REG #1
```

```
                                AS=     16              ;ATTENTION SUMMARY
                                DB=     22              ;DATA BUFFER REG
                                MR=     24              ;MAINTENANCE REG
                                DT=     26              ;DRIVE TYPE REG
                                SN=     30              ;SERIAL NUMBER REGISTER
                                TC=     32              ;TAPE CONTROL REG

                        .SBTTL  TM03/TE16 REGISTER BITS
                ;RHCS1-CS1(RS)
                                GO=     1
                                NOP=    0
                                RWDOFF= 2
                                RWD=    6
                                DRYCLR= 10
                                WFMK=   26
                                ERASE=  24
                                SPCFWD= 30
                                SPCREV= 32
                                WCHKF=  50
                                WCHKR=  56
                                WFWD=   60
                                RDFWD=  70
                                RDREV=  76
                                IE=     100
                                RDY=    200
                                A16=    400
                                A17=    1000
                                PSEL=   2000
                                DVA=    4000
                                MCPE=   20000
                                TRE=    40000
                                SC=     100000
                ;RHCS2-CS2(RS)
                                DV0=    0
                                DV1=    1
                                DV2=    2
                                DV3=    3
                                DV4=    4
                                DV5=    5
                                DV6=    6
                                DV7=    7
                                BAI=    10
                                PAT=    20
                                CLR=    40
                                IR=     100
                                OR=     200
                                MDPE=   400
                                MXF=    1000
                                PGE=    2000
                                NEM=    4000
                                NED=    10000
                                UPE=    20000
                                WCE=    40000
                                DLT=    100000
                ;RHDS-DS(RS)
                                SLA=    1
```

```
1173            000002                          BOT=    2
1174            000004                          TMK=    4
1175            000010                          IDB=    10
1176            000020                          SDWN=   20
1177            000040                          PES=    40
1178            000100                          SSC=    100
1179            000200                          DRY=    200
1180            000400                          DPR=    400
1181            002000                          EOT=    2000
1182            004000                          WRL=    4000
1183            010000                          MOL=    10000
1184            020000                          PIP=    20000
1185            040000                          ERR=    40000
1186            100000                          ATA=    100000
1187                                    ;RHER-ER(R5)
1188            000001                          ILF=    1
1189            000002                          ILR=    2
1190            000004                          RMR=    4
1191
1192            000020                          FMT=    20
1193            000100                          INCVAE= 100
1194            000200                          PEFLRC= 200
1195            000400                          NSG=    400
1196            001000                          FCE=    1000
1197            002000                          CSITM=  2000
1198            004000                          NEF=    4000
1199            010000                          DTE=    10000
1200            020000                          OPI=    20000
1201            040000                          UNS=    40000
1202
1203                                    ;RHMR-MR(R5)
1204            000100                          OSC=    100
1205
1206                                    ;RHDT-DT(R5)
1207            002000                          SPR=    2000
1208            010000                          CH7=    10000
1209            040000                          TAP=    40000
1210
1211                                    ;RHTC-TC(R5)
1212            001700                          NORM11= 1700
1213            000320                          CDM11=  320
1214            000000                          BPI200= 0
1215            000400                          BPI556= 000400
1216            001000                          BPI800= 001000
1217            002300                          PE1600= 002300
1218            100000                          ACCL=   100000
1219
1220
1221
1222                                    ;INSTRUCTION EQUATES
1223            104400                          HLT=    TRAP
1224            104000                          SCOPE=  EMT
1225            000004                          TYPE=   IOT
1226
1227                                    ;MISCELLANEOUS EQUATES
1228            005620                          OUTBUF=INIT             ;OUTPUT BUFFER STARTS AT BEG OF PROGRAM
```

```
1229   177400                  FRMCNT= -256.                   ;FRAME COUNT
1230   177600                  WRDCNT= -128.                   ;WORD COUNT
1231                   ;ASCII EQUATES
1232   000001                  CNTRLA= 1                       ;ASCII CODE FOR CONTROL A (↑A)
1233   000003                  CNTRLC= 3                       ;ASCII CODE FOR CONTROL C (↑C)
1234   000007                  CNTRLG= 7                       ;ASCII CODE FOR CONTROL G (↑G)
1235   000011                  HT=     11                      ;ASCII CODE FOR HORIZONTAL TAB
1236   000012                  LF=     12                      ;ASCII CODE FOR LINE FEED
1237   000015                  CR=     15                      ;ASCII CODE FOR CARRIAGE RETURN
1238   000017                  CNTRLO= 17                      ;ASCII CODE FOR CONTROL O (↑O)
1239   000025                  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  002130                  .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  004010                  .WORD   .SCOPE          ;SET EMT TRAP TO .SCOPE ROUTINE
000032  000340                  .WORD   340             ;PRIORITY LEVEL 7
000034  003544                  .WORD   .HLT            ;SET TRAP TRAP TO .HLT ROUTINE
000036  000340                  .WORD   340             ;PRIORITY LEVEL 7

                                ;ACT11 HOOK ***********
                        000040          $SVPC=.                 ;SAVE CURRENT LOCATION CTR
                        000046          .=46
000046  012642                  .WORD   $ENDAD          ;SET LOCATION 46
                        000052          .=52
000052  000000                  .WORD   0               ;SET LOCATION 52 = 0
                        000040          .=$SVPC                 ;RESTORE LOCATION CTR

                        000060          .=TKVEC
000060  003420                  .WORD   TKISR
000062  000200                  .WORD   200

                                ;SOFTWARE SWITCH REGISTER LOC. 176
                        000176          .=176
000176  000100          SWREG:  .WORD   SW06            ;SOFTWARE SWITCH REGISTER

                        000200          .=200
000200  000137  005620          JMP     @#INIT                  ;GO TO START OF PROGRAM
                        000210          .=210
000210  000137  006726          JMP     @#RSTRT         ;RESTART ADDRESS

                        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               ;TM03 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 TM03/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          OCTALO: .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
```

```
1296  001125  000                    UNTFND: .BYTE   0               ;UNIT FOUND INDICATOR
1297  001126  000                    TYPFLG: .BYTE   0
1298  001127  000                    PSCNT:  .BYTE   0               ;CONTAINS PASS COUNT
1299  001130  000                    ASFLG:  .BYTE   0               ;1/0 = YES/NO.
1300          001132                         .EVEN
1301  001132  030460                 DIGTAB: "01
1302  001134  031462                         "23
1303  001136  032464                         "45
1304  001140  033466                         "67
1305  001142  034470                         "89
1306  001144  000006                 ODIGITS:.BLKB   6               ;RESERVE SPACE FOR CONVERTED DIGITS
1307  001152  000                            .BYTE   0               ;TERMINATOR
1308          001154                         .EVEN
1309  001154  000010                 DRVTBL: .BLKB   8.              ;A 0/-1 = DRIVE NOT TO BE/TO BE TESTED
1310  001164  000100                 SLVTBL: .BLKB   64.             ;A 0/-1 = SLAVE NOT TO BE/TO BE TESTED
1311  001264  000110                 INBUF:  .BLKB   72.             ;TELETYPE INPUT BUFFER
1312  001374  005015  000            CRLF:   .ASCIZ  <CR><LF>        ;MISCELLANEOUS ASCII CHARACTERS
1313  001377  134     000            BKSLSH: .ASCIZ  '\'
1314  001401  060     000            ECHO:   .ASCIZ  '0'
1315  001403  007     000            BELL:   .ASCIZ  <7>
1316  001405  055     000            DASH:   .ASCIZ  '-'
1317  001407  040                    SPACE2: .ASCII  '  '
1318  001410  000040                 SPACE:  .ASCIZ  ' '
1319  001412  004476  000            ANGTAB: .ASCIZ  '>'<HT>
1320          001416                         .EVEN
```

TIME SPECIFICATION TABLE

```
1321                                            .SBTTL  TIME SPECIFICATION TABLE
1322                                    ;THE BELOW TABLE CONTAINS THE SPECIFIED FUNCTION TIMES IN TENS OF
1323                                    ;MICROCESONDS. NOTE THAT WHEN TIMES ARE TYPED THAT THEY ARE TYPED IN
1324                                    ;MICROSECONDS (BY APPENDING A 0).
1325                                    ;FORMAT IS
1326                                    ;       .WORD   MAX,MIN         ;TIME IN MS     FUNCTION        TEST #
1327
1328  001416  000000  000000           STIMTBL:.WORD   0,0             ;SPARE
1329  001422  036050  035230                   .WORD   15400.,15000.   ;154.0-150.0    WRITE FROM BOT  TST001
1330  001426  001666  001546                   .WORD   00950.,00870.   ;9.5-8.7        WRITE START     TST002
1331  001432  001522  001356                   .WORD   00850.,00750.   ;8.9-8.5        WRITE SHUTDOWN  TST003
1332  001436  002506  001332                   .WORD   01350.,00730.   ;13.5-7.3        WRITE STLDOWN   TST004
1333  001442  007164  006344                   .WORD   03700.,03300.   ;37.0-33.0       READ FROM BOT   TST005
1334  001446  000500  000360                   .WORD   00320.,00240.   ;3.2-2.4        READ START      TST006
1335  001452  000632  000454                   .WORD   00410.,00300.   ;4.1-3.00       READ SHUTDOWN   TST007
1336  001456  002506  001332                   .WORD   01350.,00730.   ;13.5-7.3        READ SETTLEDOWN TST010
1337  001462  000500  000360                   .WORD   00320.,00240.   ;3.2-2.4        RD REV START    TST011
1338  001466  000562  000512                   .WORD   00370.,00330.   ;3.7-3.3        RD REV SHTDWN   TST012
1339  001472  002506  001332                   .WORD   01350.,00730.   ;13.5-7.3        RD REV STLDWN   TST013
1340  001476  003206  002056                   .WORD   01670.,01070.   ;16.7-10.7      TRN RND DLY F-R TST014
1341  001502  003206  002056                   .WORD   01670.,01070.   ;16.7-10.7      TRN RND DLY R-F TST015
1342  001506  002412  001666                   .WORD   01290.,00950.   ;12.9-9.5        GAP SIZE STOP   TST016
1343  001512  002234  001522                   .WORD   01180.,00850.   ;11.8-8.5        GAP SIZE STRT   TST017
1344  001516  002626  002354                   .WORD   01430.,01260.   ;14.3-12.6       GAP SIZE INTER  TST020
1345  001522  002506  002234                   .WORD   01350.,01180.   ;13.5-11.8       GAP CONSISANCY  TST021
1346  001526  004540  004230                   .WORD   02400.,02200.   ;24.0-22.0       DAT TIME 800BPI TST022
1347  001532  004716  004552                   .WORD   02510.,02410.   ;25.1-24.1       DAT TIME 1600PE TST023
1348  001536  023564  023110                   .WORD   10100.,09800.   ;101.0-98.0      ERASE           TST024
1349  001542  024240  023730                   .WORD   10400.,10200.   ;104.0-102.0     WRT FILE MARK   TST025
1350  001546  004336  004172                   .WORD   02270.,02170.   ;22.7-21.7       TAPE SPEED FWD  TST026
1351  001552  004336  004172                   .WORD   02270.,02170.   ;22.7-21.7       TAPE SPEED REV  TST027
1352
1353                                    ;NOTE: TEST 26 AND 27 REQUIRE PRERECORDED 800BPI SKEW TAPE.
```

GAP TIME SPECIFICATION TABLE

```
1354                                    
1355                                    
1356                                    
1357                                    
1358                                    
1359                                    
1360                                    
1361  001556  002544  002354            
1362  001562  002652  002506            
1363  001566  002722  002506            
1364  001572  002710  002474            
1365  001576  002570  002260            
1366  001602  002556  002114            
1367  001606  002532  002114            
1368  001612  002532  002114            
1369  001616  002506  002176            
1370  001622  002474  002176            
1371  001626  002474  002176            
1372  001632  002474  002176            
1373  001636  002474  002176            
1374  001642  002474  002176            
1375  001646  002474  002176            
1376  001652  002474  002176            
```

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

;               .WORD   MAX,MIN(10)     ;TIME IN MS(10) GAP #   DELAY IN MS(10)

GTIMTBL:.WORD   01380.,01260.   ;13.8-12.6      GAP-0   0 MS
        .WORD   01450.,01350.   ;14.5-13.5      GAP-1   1.0 MS
        .WORD   01490.,01350.   ;14.9-13.5      GAP-2   2.0 MS
        .WORD   01480.,01340.   ;14.8-13.4      GAP-3   3.0 MS
        .WORD   01400.,01200.   ;14.0-12.0      GAP-4   4.0 MS
        .WORD   01390.,01100.   ;13.9-11.0      GAP-5   5.0 MS
        .WORD   01370.,01100.   ;13.7-11.0      GAP-6   6.0 MS
        .WORD   01370.,01100.   ;13.7-11.0      GAP-7   7.0 MS
        .WORD   01350.,01150.   ;13.5-11.5      GAP-10  8.0 MS
        .WORD   01340.,01150.   ;13.4-11.5      GAP-11  9.0 MS
        .WORD   01340.,01150.   ;13.4-11.5      GAP-12  10.0 MS
        .WORD   01340.,01150.   ;13.4-11.5      GAP-13  11.0 MS
        .WORD   01340.,01150.   ;13.4-11.5      GAP-14  12.0 MS
        .WORD   01340.,01150.   ;13.4-11.5      GAP-15  13.1 MS
        .WORD   01340.,01150.   ;13.4-11.5      GAP-16  14.1 MS
        .WORD   01340.,01150.   ;13.4-11.5      GAP-17  15.1 MS
```

```
                        .SBTTL  TEST HEADER POINTERS
                ;THE BELOW TABLE CONTAINS POINTERS TO EACH TEST'S DESCRIPTOR
001656  000000  NAMPTR: .WORD   0
001660  014765          .WORD   A.T001
001662  015007          .WORD   A.T002
001664  015027          .WORD   A.T003
001666  015051          .WORD   A.T004
001670  015075          .WORD   A.T005
001672  015117          .WORD   A.T006
001674  015136          .WORD   A.T007
001676  015160          .WORD   A.T010
001700  015203          .WORD   A.T011
001702  015225          .WORD   A.T012
001704  015252          .WORD   A.T013
001706  015301          .WORD   A.T014
001710  015332          .WORD   A.T015
001712  015363          .WORD   A.T016
001714  015411          .WORD   A.T017
001716  015440          .WORD   A.T020
001720  015470          .WORD   A.T021
001722  015513          .WORD   A.T022
001724  015537          .WORD   A.T023
001726  015564          .WORD   A.T024
001730  015606          .WORD   A.T025
001732  015631          .WORD   A.T026
001734  015653          .WORD   A.T027

                ;TABLE OF TEST STARTING ADDRESSES
001736  007166  TSTTBL: .WORD   TST000
001740  007432          .WORD   TST001
001742  007516          .WORD   TST002
001744  007574          .WORD   TST003
001746  007664          .WORD   TST004
001750  007772          .WORD   TST005
001752  010056          .WORD   TST006
001754  010142          .WORD   TST007
001756  010242          .WORD   TST010
001760  010362          .WORD   TST011
001762  010460          .WORD   TST012
001764  010570          .WORD   TST013
001766  010730          .WORD   TST014
001770  011022          .WORD   TST015
001772  011130          .WORD   TST016
001774  011224          .WORD   TST017
001776  011334          .WORD   TST020
002000  011454          .WORD   TST021
002002  011760          .WORD   TST022
002004  012110          .WORD   TST023
002006  012240          .WORD   TST024
002010  012360          .WORD   TST025
002012  012714          .WORD   TST026
002014  013042          .WORD   TST027
```

```
1429  002016  000000                    TIB:    .WORD   0
1430                                    ;ROUTINE TO LOAD SSOFTWARE SWR
1431
1432  002020  022737  000176  001000    GTSWR:  CMP     #SWREG,SWR      ;BRANCH IF SOFTWARE SWR
1433  002026  001027                    BNE     2$              ;NOT INVOKED
1434  002030  004737  002354            JSR     PC,.SAVE        ;SAVE REGISTERS ON THE STACK
1435  002034  000004  015702            TYPE,L.SWR
1436  002040  017702  176734            MOV     @SWR,R2
1437  002044  004737  002426            JSR     PC,TYPOCT
1438  002050  000004  015711            TYPE,L.NEW
1439  002054  004737  003272            JSR     PC,.INPUT               ;GET USER INPUT
1440  002060  122737  000015  001264    CMPB    #CR,@#INBUF             ;EXIT IF FIRST CHAR IS <CR>
1441  002066  001405                    BEQ     1$
1442  002070  004737  003056            JSR     PC,CNVTAO               ;CONERT ASCII TO OCTAL
1443  002074  013777  001116  176676    MOV     @#OCTALO,@SWR           ;SET NEW SWITCH REG CONTENTS
1444  002102  004737  002376    1$:     JSR     PC,.RESTORE
1445  002106  000207            2$:     RTS     PC
1446
1447
1448                                            .SBTTL  PROGRAM SUBROUTINES
1449                                            .SBTTL          TYPE SUBROUTINE
1450                                    ;;ROUTINE TO TYPE ASCII MESSAGE. MESSAGE MUST TERMINATE WITH A 0 BYTE.
1451                                    ;;THE ROUTINE WILL INSERT A NUMBER OF NULL CHARACTERS AFTER A LINE FEED.
1452                                    ;;CALL: TYPE                    ;;A TRAP TYPE INSTRUCTION
1453                                    ;;      MESADR                  ;;MESADR IS FIRST ADDRESS OF ASCIZ STRING
1454
1455                                    ;;TAGS USED BY THE TYPE ROUTINE BELOW
1456          000011                            $HT=11                  ;;HORIZONTAL TAB
1457  002110      000                   $NULL:  .BYTE   0               ;;CONTAINS NULL CHARACTER
1458  002111      002                   $FILL:  .BYTE   2               ;;CONTAINS # OF FILLER CHARACTERS
1459  002112      000                   $TPFLG: .BYTE   0               ;;CONTAINS TELEPRINTER AVAILABLE FLAG
1460                                                                    ;;0/377 = AVAIL/NOT AVAIL
1461  002113      000                   $TKFLG: .BYTE   0               ;;CONTAINS KEYBOARD AVAILABLE FLAG
1462  002114  177564                    $TPS:   .WORD   177564          ;;ADDRESS OF TLEPRINTER STATUS REGISTER
1463  002116  177566                    $TPB:   .WORD   177566          ;;ADDRESS OF TELEPRINTER DATA BUFFER
1464  002120      000                   $CHARCNT:.BYTE  0               ;;CONTAINS # OF CHARS TYPED
1465  002121      000                   $CNTRLO:.BYTE   0               ;;CONTAINS CONTROL O CHAR (IF TYPED)
1466  002122  005015      000           $CRLF:  .ASCIZ  <15><12>
1467          002126                            .EVEN
1468  002126  000000                    RDSW:   .WORD   0
1469
1470  002130  010046                    .TYPE:  MOV     R0,-(SP)        ;;SAVE R0
1471  002132  017600  000002            MOV     @2(SP),R0       ;;GET MESSAGE ADDRESS
1472  002136  062766  000002  000002    ADD     #2,2(SP)        ;;ADJUST RETURN PC
1473  002144  105037  002121            CLRB    $CNTRLO
1474
1475  002150  105737  002121    TYPE1:  TSTB    $CNTRLO         ;;BRANCH IF CONTROL O(^O) WASN'T TYPED
1476  002154  001410                    BEQ     TYPE2
1477  002156  000004  002122    TCRLF:  TYPE,$CRLF              ;;TYPE <CR><LF>
1478  002162  105737  002126            TSTB    RDSW
1479  002166  100006                    BPL     TYPE3
1480  002170  005037  002126            CLR     RDSW
1481  002174  000207                    RTS     PC
1482  002176  112046    TYPE2:  MOVB    (R0)+,-(SP)     ;;PUSH CHARACTER TO BE TYPED ONTO STACK
1483  002200  001003                    BNE     TYPE4           ;;BRANCH IF NOT THE TERMINATOR
1484  002202  005726                    TST     (SP)+           ;;POP TERMINATOR CHAR OFF THE STACK
```

```
1485  002204  012600                TYPE3:  MOV     (SP)+,R0         ;;RESTORE R0
1486  002206  000002                        RTI                      ;;RETURN TO CALLER
1487
1488  002210  122716  000011        TYPE4:  CMPB    #$HT,(SP)        ;;BRANCH IF HORIZONTAL TAB <HT>
1489  002214  001445                        BEQ     9$
1490  002216  004737  002250                JSR     PC,5$            ;;TYPE CHARACTER
1491  002222  122726  000012        3$:     CMPB    #12,(SP)+        ;;CHECK IF CHARACTER WAS A LINE FEED
1492  002226  001350                        BNE     TYPE1            ;;BRANCH IF NOT LINE FEED
1493  002230  013746  002110                MOV     $NULL,-(SP)      ;;GET # OF FILLERS REQUIRED AND FILLER
1494                                                                 ;;CHARACTER.
1495
1496  002234  105366  000001        4$:     DECB    1(SP)            ;;DECREMENT FILLERS REQ. COUNT
1497  002240  002770                        BLT     3$               ;;BRANCH IF NO MORE FILLERS ARE REQUIRED
1498  002242  004737  002250                JSR     PC,5$            ;;TYPE FILLER CHARACTER
1499  002246  000772                        BR      4$
1500
1501  002250  105777  177640        5$:     TSTB    @$TPS            ;;WAIT FOR OUTPUT DEVICE
1502  002254  100375                        BPL     .-4
1503  002256  122737  000017  002121        CMPB    #17,@#$CNTRLO    ;;CHECK IF CONTROL O WAS TYPED
1504  002264  001403                        BEQ     6$               ;;STOP TYPING MESSAGE IF ↑O WAS TYPED
1505  002266  116677  000002  177622        MOVB    2(SP),@$TPB      ;;OUTPUT CHARACTER
1506  002274  122766  000015  000002 6$:     CMPB    #15,2(SP)        ;;BRANCH IF NOT <CR>
1507  002302  001003                        BNE     7$
1508  002304  105037  002120                CLRB    $CHARCNT         ;;CLEAR CHARACTERS TYPED COUNT
1509  002310  000406                        BR      8$
1510  002312  122766  000012  000002 7$:     CMPB    #12,2(SP)        ;;BRANCH IF <LF> OR 'NULL'
1511  002320  002002                        BGE     8$
1512  002322  105237  002120                INCB    $CHARCNT         ;;INCREMENT CHARACTER TYPED COUNT
1513  002326  000207                8$:     RTS     PC
1514
1515                                        ;;HORIZONTAL TAB <HT> PROCESSER
1516  002330  112716  000040        9$:     MOVB    #40,(SP)         ;;LOAD 'SPACE'
1517  002334  004737  002250        10$:    JSR     PC,5$            ;;TYPE 'SPACE'
1518  002340  132737  000007  002120        BITB    #7,$CHARCNT      ;;TYPE SPACES UNTIL A MULTIPLE
1519  002346  001372                        BNE     10$              ;;OF 8 CHARACTERS HAVE BEEN TYPED
1520  002350  105726                        TSTB    (SP)+            ;;POP SPACE
1521  002352  000676                        BR      TYPE1            ;;GET NEXT CHARACTER
1522
1523                                        ;SUBROUTINE TO SAVE GENERAL REGISTERS ON THE STACK
1524                                        ;CALL:  SAVE
1525  002354  010546                .SAVE:  MOV     R5,-(SP)                 ;SAVE REGISTERS ON THE STACK
1526  002356  010446                        MOV     R4,-(SP)
1527  002360  010346                        MOV     R3,-(SP)
1528  002362  010246                        MOV     R2,-(SP)
1529  002364  010146                        MOV     R1,-(SP)
1530  002366  010046                        MOV     R0,-(SP)
1531  002370  016646  000014                MOV     14(SP),-(SP)             ;GET RETURN PC
1532  002374  000207                        RTS     PC                       ;RETURN
1533
1534                                        ;SUBROUTINE TO RESTORE GENERAL REGISTERS FROM THE STACK
1535                                        ;CALL:  RESTORE
1536  002376  012666  000014        .RESTORE:MOV    (SP)+,14(SP)             ;MOVE RETURN PC
1537  002402  012600                        MOV     (SP)+,R0                 ;RESTORE REGISTERS
1538  002404  012601                        MOV     (SP)+,R1
1539  002406  012602                        MOV     (SP)+,R2
1540  002410  012603                        MOV     (SP)+,R3
```

```
1541  002412  012604                          MOV     (SP)+,R4
1542  002414  012605                          MOV     (SP)+,R5
1543  002416  000207                          RTS     PC                      ;RETURN
1544
1545                                  ;SUBROUTINE TO CONVERT OCTAL DATA TO ASCII
1546                                  ;CALL:  MOV     NUMBER,R2               ;MOVE NUMBER TO R2
1547                                  ;       JSR     PC,CNVOCT
1548
1549  002420  110637  001126          CNVOCT: MOVB    SP,TYPFLG               ;SET DO NOT TYPE FLAG
1550  002424  000402                          BR      CNVTO
1551
1552                                          .SBTTL          OCTAL TO ASCII & TYPE ROUTINE
1553                                  ;SUBROUTINE TO CONVERT OCTAL NUMBER TO ASCII AND TYPE IT OUT
1554                                  ;CALL:  MOV     NUMBER,R2               ;PUT # IN R2
1555                                  ;       JSR     PC,TYPOCT               ;CALL ROUTINE
1556
1557  002426  105037  001126          TYPOCT: CLRB    @#TYPFLG                ;SET TYPE FLAG
1558  002432                          CNVTO:
1559  002432  004737  002354                  JSR     PC,.SAVE                ;SAVE REGISTERS ON THE STACK
1560  002436  012704  001144                  MOV     #ODIGITS,R4             ;SET PTR TO OUTPUT
1561  002442  005003                          CLR     R3                      ;R3 WILL CONTAIN OCTAL DIGIT
1562  002444  010201                          MOV     R2,R1                   ;GET # TO BE TYPED
1563  002446  006302                  1$:     ASL     R2                      ;SHIFT #
1564  002450  006103                          ROL     R3
1565  002452  012700  000006                  MOV     #6,R0                   ;SET DIGIT COUNTER
1566  002456  000404                          BR      3$
1567
1568  002460  006302                  2$:     ASL     R2                      ;SHIFT # 3 PLACES LEFT
1569  002462  006103                          ROL     R3
1570  002464  005301                          DEC     R1
1571  002466  001374                          BNE     2$
1572  002470  012701  000003          3$:     MOV     #3,R1                   ;SET SHIFT COUNTER
1573  002474  116324  001132                  MOVB    DIGTAB(R3),(R4)+        ;MOVE ASCII EQUIV TO OUTPUT
1574  002500  005003                          CLR     R3
1575  002502  005300                          DEC     R0                      ;DECREMENT DIGIT COUNT
1576  002504  001365                          BNE     2$                      ;GET NEXT DIGIT
1577  002506  105737  001126                  TSTB    @#TYPFLG                ;BRANCH IF ASCII IS
1578  002512  001002                          BNE     4$                      ;NOT TO BE TYPED
1579  002514  000004  001144                  TYPE,ODIGITS
1580  002520                          4$:
1581  002520  004737  002376                  JSR     PC,.RESTORE             ;RESTORE REGISTERS FROM THE STACK
1582  002524  000207                          RTS     PC
1583
1584
1585                                  ;SUBROUTINE TO CONVERT OCTAL DATA TO DECIMAL ASCII
1586                                  ;CALL:  MOV     NUMBER,R2               ;MOVE NUMBER TO R2
1587                                  ;       JSR     PC,CNVDEC
1588
1589  002526  110637  001126          CNVDEC: MOVB    SP,@#TYPFLG             ;SET DO NOT TYPE FLAG
1590  002532  000402                          BR      CNVTD
1591                                          .SBTTL          OCTAL TO DECIMAL & TYPE ROUTINE
1592                                  ;THIS ROUTINE CONVERTS AN OCTAL # TO DECIMAL ASCII AND TYPES IT OUT
1593                                  ;CALL:  MOV     NUMBER,R2               ;PUT # IN R2
1594                                  ;       JSR     PC,TYPDEC               ;CALL ROUTINE
1595
1596  002534  105037  001126          TYPDEC: CLRB    @#TYPFLG                ;SET TYPE FLAG
```

```
                                        OCTAL TO DECIMAL & TYPE ROUTINE

002540                          CNVTD:
002540  004737  002354                  JSR     PC,.SAVE                ;SAVE REGISTERS ON THE STACK
002544  005000                          CLR     R0                      ;R0 IS INDEX TO DECIMAL CONSTANT
002546  012704  001144                  MOV     #ODIGITS,R4             ;SET OUTPUT PTR
002552  005003                  1$:     CLR     R3                      ;R3 CONTAINS DECIMAL DIGIT
002554  166002  002634          2$:     SUB     DCONST(R0),R2           ;SUBTRACT DECIMAL CONSTANT UNTIL
002560  103402                          BLO     3$                      ;INPUT # GOES NEGATIVE
002562  005203                          INC     R3                      ;KEEPING TRACK OF SUBTRACTIONS
002564  000773                          BR      2$
002566  066002  002634          3$:     ADD     DCONST(R0),R2           ;ADD BACK CONSTANT WHEN NEGATIVE
002572  116324  001132                  MOVB    DIGTAB(R3),(R4)+        ;MOVE ASCII EQUIVILENT
002576  062700  000002                  ADD     #2,R0                   ;NEXT CONSTANT
002602  005760  002634                  TST     DCONST(R0)              ;UNTIL ALL CONSTANTS DONE
002606  001361                          BNE     1$
002610  112724  000060                  MOVB    #'0,(R4)+               ;LAST DIGIT IS 0
002614  105737  001126                  TSTB    @#TYPFLG                ;BRANCH IF ASCII IS
002620  001002                          BNE     4$                      ;NOT TO BE TYPED
002622  000004  001144                  TYPE,ODIGITS
002626                          4$:
002626  004737  002376                  JSR     PC,.RESTORE             ;RESTORE REGISTERS FROM THE STACK
002632  000207                          RTS     PC

002634  023420                  DCONST: .WORD   10000.
002636  001750                          .WORD   1000.
002640  000144                          .WORD   100.
002642  000012                          .WORD   10.
002644  000001                          .WORD   1.
002646  000000                          .WORD   0                       ;TERMINATOR

                                        .SBTTL          TYPE SPECIFIED TIMES ROUTINE
                                ;THIS SUBROUTINE OUTPUTS THE TIME SPECIFICATIONS FOR THE TEST
                                ;AND ALSO THE ACTUAL TIME RECORDED (ATIME)
                                ;FORMAT OF LINE TYPED
                                ;RANGE=<AAAAAA-BBBBBB>                  ACTUAL=CCCCCC
                                ;WHERE:         AAAAAA IS MAXIMUM TIME FOR TEST (STIMTBL(TSTNUMX4)).
                                ;               BBBBBB IS MINIMUM TIME FOR TEST (STIMTBL(TSTNUMX4+2)).
                                ;               CCCCCC IS ACTUAL TIME RECORDED BY TEST (ATIME).
                                ;CALL:  MOVB    TEST NUMBER,R2  ;LOAD TEST NUMBER
                                ;       MOV     #TIME,@#ATIME           ;MOVE TIME TO ATIME
                                ;       JSR     PC,OUTSPC
002650  010246                  OUTSPC: MOV     R2,-(SP)                ;SAVE R2 & R3 ON THE STACK
002652  010346                          MOV     R3,-(SP)
002654  006302                          ASL     R2                      ;MULTIPLY TEST # TIMES 4
002656  006302                          ASL     R2                      ;TO FORM INDEX INTO STIMTBL
002660  010203                          MOV     R2,R3                   ;R3 CONTAINS INDEX INTO TABLE
002662  000004  014745                  TYPE,L.RNG
002666  016302  001416                  MOV     STIMTBL(R3),R2          ;GET MAXIMUM SPEC TIME
002672  004737  002534                  JSR     PC,TYPDEC               ;CONVERT TO DECIMAL & TYPE
002676  000004  001405                  TYPE,DASH
002702  016302  001420                  MOV     STIMTBL+2(R3),R2        ;GET MINIMUM TIME
002706  004737  002534                  JSR     PC,TYPDEC               ;CONVERT TO DECIMAL & TYPE
002712  000004  001412                  TYPE,ANGTAB
002716  000004  014755                  TYPE,L.ACT
002722  013702  001012                  MOV     @#ATIME,R2              ;GET ACTUAL TIME
002726  004737  002534                  JSR     PC,TYPDEC               ;CONVERT TO DECIMAL & TYPE
002732  000004  001374                  TYPE,CRLF
```

```
                                        TYPE SPECIFIED TIMES ROUTINE

1653  002736  012603                    MOV     (SP)+,R3
1654  002740  012602                    MOV     (SP)+,R2
1655  002742  000207                    RTS     PC                      ;RETURN
1656
1657                                        .SBTTL          TYPE GAP TIMES SUBROUTINE
1658                                ;THIS SUBROUTINE IS USED TO TYPE THE SPECIFIED GAP SIZES (RECORDED IN
1659                                ;TST021). IT IS CALLED BY THE GAPOK ROUTINE IF THE GAP SIZE IS OUT OF
1660                                ;RANGE VIA THE HLT ROUTINE (HLT+2).
1661                                ;CALL:  MOVB    #GAP,GAP                ;LOAD GAP # INTO GAP
1662                                ;       MOV     #TIME,ATIME             ;LOAD ACTUAL TIME INTO ATIME
1663                                ;       JSR     PC,OUTGAP
1664
1665  002744  010246                OUTGAP: MOV     R2,-(SP)                ;SAVE R2 AND R3
1666  002746  010346                        MOV     R3,-(SP)
1667  002750  113703  001120                MOVB    GAP,R3                  ;GET GAP #
1668  002754  006303                        ASL     R3
1669  002756  006303                        ASL     R3
1670  002760  000004  014745                TYPE,L.RNG
1671  002764  016302  001556                MOV     GTIMTBL(R3),R2          ;GET MAX TIME
1672  002770  004737  002534                JSR     PC,TYPDEC               ;CONVERT TO DECIMAL & TYPE
1673  002774  000004  001405                TYPE,DASH
1674  003000  016302  001560                MOV     GTIMTBL+2(R3),R2        ;GET MIN TIME
1675  003004  004737  002534                JSR     PC,TYPDEC               ;CONVERT TO DECIMAL & TYPE
1676  003010  000004  001412                TYPE,ANGTAB                     ;TYPE <
1677  003014  000004  014755                TYPE,L.ACT
1678  003020  013702  001012                MOV     @#ATIME,R2              ;GET ACTUAL TIME
1679  003024  004737  002534                JSR     PC,TYPDEC               ;CONVERT TO DECIMAL & TYPE
1680  003030  000004  014434                TYPE,E.GAP
1681  003034  113702  001120                MOVB    @#GAP,R2                ;GET GAP #
1682  003040  004737  002426                JSR     PC,TYPOCT               ;TYPE GAP #
1683  003044  000004  001374                TYPE,CRLF
1684  003050  012603                        MOV     (SP)+,R3                ;RESTORE R3 AND R2
1685  003052  012602                        MOV     (SP)+,R2
1686  003054  000207                        RTS     PC
1687
1688                                        .SBTTL          ASCII TO OCTAL CONVERT SUBROUTINE
1689                                ;SUBROUTINE TO CONVERT ASCII DATA TO OCTAL. CONVERTED OCTAL DATA
1690                                ;IS LEFT IN OCTALO <15-00>.
1691  003056                        CNVTAO:
1692  003056  004737  002354                JSR     PC,.SAVE                ;SAVE REGISTERS ON THE STACK
1693  003062  012700  001264                MOV     #INBUF,R0               ;SET PTR TO ASCII DATA
1694  003066  012701  001116                MOV     #OCTALO,R1              ;GET ADDRESS OF OCTAL DATA
1695  003072  005011                        CLR     (R1)                    ;CLEAR OUT OLD OCTAL DATA
1696  003074  005061  000002                CLR     2(R1)
1697  003100  122710  000015        1$:     CMPB    #CR,(R0)                ;<CR> TERMINATES INPUT
1698  003104  001414                        BEQ     3$
1699  003106  112002                        MOVB    (R0)+,R2                ;GET 'OCTAL' DATA
1700  003110  042702  177770                BIC     #177770,R2              ;STRIP UNUSED BITS
1701  003114  012703  000003                MOV     #3,R3                   ;SET SHIFT COUNT
1702  003120  006311                2$:     ASL     (R1)                    ;SHIFT LAST
1703  003122  006161  000002                ROL     2(R1)                   ;OCTAL DIGIT
1704  003126  005303                        DEC     R3
1705  003130  001373                        BNE     2$
1706  003132  050211                        BIS     R2,(R1)                 ;AND INSERT THIS DIGIT
1707  003134  000761                        BR      1$                      ;GO GET NEXT DIGIT
1708  003136                        3$:
```

```
                                        ASCII TO OCTAL CONVERT SUBROUTINE

1709  003136  004737  002376                JSR     PC,.RESTORE             ;RESTORE REGISTERS FROM THE STACK
1710  003142  000207                        RTS     PC                      ;RETURN
1711
1712                                            .SBTTL          PUBLISH SUBROUTINE
1713                                    ;THE PUBLISH SUBROUTINE AVERAGES THE RECORDED TIMES FOR EACH TEST IT-
1714                                    ;ERATION (IF 16. ITERATIONS) AND PLACES THE AVERAGE RESULT IN 'ATIME'.
1715                                    ;IT TYPES THE NAME OF THE FUNCTION THAT WAS TIMED,THE TIME SPEC-
1716                                    ;IFICATION AND THE ACTUAL TIME .
1717
1718  003144                            PUBLISH:
1719  003144  004737  002354                   JSR     PC,.SAVE                ;SAVE REGISTERS ON THE STACK
1720  003150  012700  001014                   MOV     #ATIMTBL,R0     ;GET TABLE ADDRESS CONTAINING TIMES
1721  003154  113701  001121                   MOVB    @#ITCNT,R1      ;GET # OF ENTRIES (GIVEN BY ITERATION COUNT)
1722  003160  122701  000001                   CMPB    #1,R1                   ;BRANCH IF SINGLE ITERATION
1723  003164  001423                           BEQ     4$
1724  003166  005002                           CLR     R2                      ;CLEAR 'SUM' REGISTERS
1725  003170  005003                           CLR     R3
1726  003172  122701  000020                   CMPB    #16.,R1                 ;BRANCH IF 16. ITERATIONS
1727  003176  001402                           BEQ     1$
1728  003200  000000                           HALT                            ;ITERATION COUNT MUST BE 1 OR 16.
1729  003202  000777                           BR      .                       ;DO NOT CHANGE POSIT OF SW11
1730                                                                           ;WHEN TEST IS RUNNING.
1731
1732  003204  062002                    1$:    ADD     (R0)+,R2                ;SUM INDIVIDUAL TIMES
1733  003206  005503                           ADC     R3
1734  003210  005301                           DEC     R1
1735  003212  001374                           BNE     1$
1736
1737  003214  012700  000004           2$:    MOV     #4,R0
1738  003220  006203                    3$:    ASR     R3                      ;SHIFT TIME IN R3 & R2 4 PLACES
1739  003222  006002                           ROR     R2                      ;RIGHT = DIVIDE BY 16.
1740  003224  005300                           DEC     R0
1741  003226  001374                           BNE     3$
1742  003230  010237  001012                   MOV     R2,@#ATIME              ;MOVE AVERAGED TIMES
1743
1744  003234  113700  001122           4$:    MOVB    @#TSTNUM,R0             ;GET TEST #
1745  003240  006300                           ASL     R0
1746  003242  016037  001656  003252           MOV     NAMPTR(R0),5$           ;GET TEST NAME STRING ADDRESS
1747  003250  000004                           TYPE
1748  003252  000000                    5$:    .WORD   0
1749  003254  113702  001122                   MOVB    @#TSTNUM,R2             ;GET TEST #
1750  003260  004737  002650                   JSR     PC,OUTSPC               ;OUTPUT TIMES
1751  003264  004737  002376                   JSR     PC,.RESTORE             ;RESTORE REGISTERS FROM THE STACK
1752  003270  000207                           RTS     PC
1753
1754                                            .SBTTL          INPUT SUBROUTINE
1755                                    ;SUBROUTINE TO GET TTY INPUT
1756                                    ;CALL:  JSR     PC,.INPUT
1757                                    ;INPUT DATA IS RETUNED IN BUFFER BEGINNING AT INBUF.
1758
1759  003272  010046                    .INPUT: MOV     R0,-(SP)        ;SAVE R0 ON THE STACK
1760  003274  012700  001264           1$:     MOV     #INBUF,R0
1761  003300  105737  177560           2$:     TSTB    @#TKS
1762  003304  100375                           BPL     2$
1763
1764  003306  113746  177562                   MOVB    @#TKB,-(SP)     ;GET CHARACTER
```

```
                                        INPUT SUBROUTINE

 1765  003312  042716  000200                BIC     #200,(SP)
 1766  003316  122716  000177                CMPB    #177,(SP)       ;CHECK RUBOUT
 1767  003322  001004                        BNE     3$
 1768  003324  124026                        CMPB    -(R0),(SP)+     ;REMOVE CHARACTER FROM INPUT
 1769  003326  000004  001377                TYPE,BKSLSH
 1770  003332  000762                        BR      2$              ;WAIT FOR NEXT CHARACTER
 1771  003334  122716  000025        3$:     CMPB    #CNTRLU,(SP)    ;CHECK CONTROL U (↑U)
 1772  003340  001004                        BNE     4$
 1773  003342  005726                        TST     (SP)+
 1774  003344  000004  001374                TYPE,CRLF
 1775  003350  000751                        BR      1$
 1776  003352  122716  000003        4$:     CMPB    #CNTRLC,(SP)    ;BRANCH IF NOT CONTROL C
 1777  003356  001003                        BNE     40$
 1778  003360  000005                        RESET                   ;RESET I/O
 1779  003362  000137  005620                JMP     @#INIT          ;RESTART PROGRAM
 1780  003366  111637  001401        40$:    MOVB    (SP),@#ECHO
 1781  003372  111620                        MOVB    (SP),(R0)+
 1782  003374  122726  000015                CMPB    #CR,(SP)+
 1783  003400  001403                        BEQ     5$
 1784  003402  000004  001401                TYPE,ECHO
 1785  003406  000734                        BR      2$
 1786  003410  000004  001374        5$:     TYPE,CRLF
 1787  003414  012600                        MOV     (SP)+,R0
 1788  003416  000207                        RTS     PC
 1789
 1790                                ;KEYBOARD INTERRUPT SERVICE ROUTINE
 1791  003420  113746  177562        ↑KISR:  MOVB    @#TKB,-(SP)     ;GET TYPED CHARACTER
 1792  003424  042716  000200                BIC     #200,(SP)       ;STRIP PARITY BIT
 1793  003430  122716  000017                CMPB    #CNTRL0,(SP)    ;BRANCH IF NOT CONTROL O (↑O)
 1794  003434  001002                        BNE     1$
 1795  003436  111637  002121                MOVB    (SP),$CNTRL0    ;SET CONTROL O INDICATOR IN TYPE ROUTINE
 1796
 1797  003442  122716  000003        1$:     CMPB    #3,(SP)         ;BRANCH IF NOT CONTROL C (↑C)
 1798  003446  001007                        BNE     2$
 1799  003450  023727  000042  012642        CMP     @#42,#SENDAD    ;INHIBIT ↑C IF ACT11 QV OR AA
 1800  003456  001403                        BEQ     2$
 1801  003460  000005                        RESET
 1802  003462  000137  005620                JMP     @#INIT          ;RESTART PROGRAM
 1803
 1804  003466  122716  000001        2$:     CMPB    #CNTRLA,(SP)    ;BRANCH IF NOT ↑A
 1805  003472  001011                        BNE     3$
 1806  003474  022737  000176  001000        CMP     #SWREG,SWR      ;BRANCH IF HARDWARE SWR IS INVOKED
 1807  003502  001010                        BNE     4$
 1808  003504  012737  177570  001000        MOV     #177570,SWR     ;INVOKE HARDWARE SWR
 1809  003512  000004  013425                TYPE,M.HSWR
 1810  003516  122716  000007        3$:     CMPB    #CNTRLG,(SP)    ;BRANCH IF NOT ↑G
 1811  003522  001005                        BNE     5$
 1812  003524  012737  000176  001000        MOV     #SWREG,SWR      ;INVOKE SOFTWARE SWR
 1813  003532  004737  002020                JSR     PC,GTSWR        ;GET NEW SWITCH REGISTER
 1814  003536  005726                5$:     TST     (SP)+           ;POP CHARACTER OFF THE STACK
 1815  003540  000002                        RTI                     ;RETURN TO CALLER
```

```
                                        .SBTTL          ERROR SERVICE ROUTINES
                                ;ROUTINE TO PROCESS ERROR TRAPS (TRAPS TO 4)
003542  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>.

003544  004737  002354          .HLT:   JSR     PC,.SAVE                ;SAVE REGISTERS ON THE STACK
003550  110637  001123          1$:     MOVB    SP,@#ERFLG              ;SET ERROR FLAG
003554  032777  020000  175216          BIT     #SW13,@SWR              ;BRANCH IF NO TYPOUT
003562  001075                        BNE     4$
003564  000004  014235                TYPE,E.HDR
003570  113702  001122                MOVB    @#TSTNUM,R2             ;GET TEST #
003574  004737  002426                JSR     PC,TYPOCT               ;AND TYPE IT
003600  016600  000016                MOV     16(SP),R0               ;GET RETURN PC
003604  162700  000002                SUB     #2,R0                   ;NOW PC OF HLT CALL
003610  111000                        MOVB    (R0),R0                 ;NOW HLT CALL ITSELF
003612  001417                        BEQ     2$                      ;BRANCH IF HLT
003614  000004  014320                TYPE,E.HDR2
003620  122700  000002                CMPB    #2,R0                   ;BRANCH IF NOT HLT+2
003624  001005                        BNE     10$
003626  004737  002744                JSR     PC,OUTGAP               ;TYPE GAP SPECIFIED TIMES
003632  000004  001374                TYPE,CRLF
003636  000447                        BR      4$
003640  004737  002650          10$:    JSR     PC,OUTSPC               ;TYPE SPECIFIED TIMES
003644  000004  001374                TYPE,CRLF
003650  000442                        BR      4$
003652  016500  000014          2$:     MOV     ER(R5),R0
003656  032765  002300  000032        BIT     #PE1600,TC(R5)
003664  001403                        BEQ     20$
003666  042700  102100                BIC     #102100,R0
003672  000402                        BR      21$
003674  042700  102300          20$:    BIC     #102300,R0
003700  005700                  21$:    TST     R0
003702  001003                        BNE     22$
003704  000004  014211                TYPE,E.SFT              ;TYPE SOFT ERROR MESSAGE
003710  000434                        BR      6$

003712  000004  014245          22$:    TYPE,E.HDR1
003716  010500                        MOV     R5,R0                   ;GET FIRST ADDRESS OF REGS.
003720  012701  000007                MOV     #7.,R1                  ;TYPE FIRST 7 REGS.
003724  012002                  3$:     MOV     (R0)+,R2                ;GET REG CONTENTS
003726  004737  002426                JSR     PC,TYPOCT               ;AND TYPE IT
003732  000004  001407                TYPE,SPACE2
003736  005301                        DEC     R1
003740  001371                        BNE     3$
003742  016502  000032                MOV     TC(R5),R2               ;GET CONTENTS OF TC REGISTER
003746  004737  002426                JSR     PC,TYPOCT
003752  000004  001374                TYPE,CRLF

003756  032777  001000  175014  4$:     BIT     #SW09,@SWR              ;BRANCH IF NO RING THE BELL
003764  001402                        BEQ     5$
003766  000004  001403                TYPE,BELL
003772  005777  175002          5$:     TST     @SWR                    ;HALT ON ERROR?
```

```
1872  003776  100001                        BPL     6$
1873  004000  000000                        HALT
1874  004002                          6$:
1875  004002  004737  002376                JSR     PC,.RESTORE             ;RESTORE REGISTERS FROM THE STACK
1876  004006  000002                        RTI                             ;RETURN
1877
1878
```

```
 1879                                           .SBTTL          SCOPE SUBROUTINE
 1880                                   ;SCOPE ROUTINE
 1881                                   ;THIS ROUTINE IS ENTERED UPON COMPLETION OF EACH SUBTEST
 1882                                   ;THE SCOPE ROUTINE:
 1883                                   ;       OUTPUTS TIME SPEC ON EACH ITERATION IF SW08 IS SET
 1884                                   ;       REPEATS TEST IF SW14 IS SET
 1885                                   ;       STORES ACTUAL TIME FOR FUNCTION IN TIME TABLE (ATIMTBL)
 1886                                   ;       PUBLISHES TIME IF SW10=0
 1887                                   ;       UPDATES ITERATION COUNT AND IF ITERATIONS COMPLETE CONTINUES
 1888                                   ;       TO NEXT TEST, OTHERWISE REPEATS TEST.
 1889                                   ;       DELAYS BEFORE CONTINUING OR REPEATING TEST.
 1890                                   ;       INITIALIZES DRIVE
 1891                                   ;RETURNS:       R5=BASE ADDRESS OF TM03 REGISTERS (ADDRESS OF CS1)
 1892                                   ;               R1='DS' REG ADDRESS
 1893                                   ;               R0='FC' REG ADDRESS
 1894
 1895  004010  013705  001010          .SCOPE: MOV     @#TMBASE,R5             ;SET R5 TO FIRST TM REG
 1896  004014  032777  000400  174756          BIT     #SW08,@SWR              ;BRANCH IF SPECIFICATION LINE
 1897  004022  001404                          BEQ     10$                     ;NOT DESIRED ON EACH ITERATION
 1898  004024  113702  001122                  MOVB    @#TSTNUM,R2             ;GET TEST NUMBER
 1899  004030  004737  002650                  JSR     PC,OUTSPC               ;OUTPUT TIME RECORDED
 1900  004034  032777  040000  174736  10$:    BIT     #SW14,@SWR              ;BRANCH IF CONTINUOS LOOP
 1901  004042  001432                          BEQ     2$                      ;NOT DESIRED
 1902  004044  017701  174730          1$:     MOV     @SWR,R1                 ;GET SWITCHES
 1903  004050  042701  177740                  BIC     #177740,R1              ;CLEAR ALL BUT TEST #
 1904  004054  001406                          BEQ     11$                     ;BRANCH IF ALL SELECTED
 1905  004056  020137  001122                  CMP     R1,@#TSTNUM             ;BRANCH IF RUNNING SELECTED TEST
 1906  004062  001403                          BEQ     11$
 1907  004064  012737  007166  001002          MOV     #TST000,SCPADR          ;RESTART AT TST000
 1908  004072  004737  004634          11$:    JSR     PC,DELAY                ;DELAY 350 MS
 1909  004076  004737  005070                  JSR     PC,RHINIT               ;INIT
 1910  004102  105037  001123                  CLRB    @#ERFLG                 ;CLEAR ERROR FLAG
 1911  004106  013716  001002                  MOV     SCPADR,(SP)
 1912  004112  010501                          MOV     R5,R1
 1913  004114  062701  000012                  ADD     #DS,R1                  ;ADDRESS OF 'DS' REG IS IN R1
 1914  004120  010500                          MOV     R5,R0
 1915  004122  062700  000006                  ADD     #FC,R0                  ;ADDRESS OF 'FC' REG IS IN R0
 1916  004126  000002                          RTI
 1917
 1918  004130  105737  001123          2$:     TSTB    @#ERFLG                 ;BRANCH IF ERROR FLAG IS SET
 1919  004134  001006                          BNE     3$
 1920  004136  113700  001121                  MOVB    @#ITCNT,R0              ;GET ITERATION COUNT
 1921  004142  006300                          ASL     R0                      ;STORE TIME IN TABLE
 1922  004144  013760  001012  001014          MOV     @#ATIME,ATIMTBL(R0)
 1923  004152  105237  001121          3$:     INCB    @#ITCNT                 ;INCREMENT ITERATION COUNT
 1924  004156  105737  001127                  TSTB    @#PSCNT                 ;INHIBIT ITERATIONS ON
 1925  004162  001410                          BEQ     4$                      ;ON FIRST PASS
 1926  004164  032777  004000  174606          BIT     #SW11,@SWR              ;BRANCH IF SINGLE ITERATION DESIRED
 1927  004172  001004                          BNE     4$
 1928  004174  122737  000020  001121          CMPB    #16.,@#ITCNT            ;BRANCH IF ITERATIONS INCOMPLETE
 1929  004202  001320                          BNE     1$
 1930  004204  032777  000037  174566  4$:     BIT     #37,@SWR        ;IF TEST SELECTED IS TEST 0
 1931  004212  001002                          BNE     42$             ;TREAT AS ALL TESTS
 1932  004214  011637  001002          40$:    MOV     (SP),@#SCPADR   ;SET SCOPE ADDRESS TO NEXT TEST
 1933  004220  032777  002000  174552  42$:    BIT     #SW10,@SWR      ;BRANCH IF NO PUBLICATION DESIRED
 1934  004226  001005                          BNE     5$
```

```
                                        SCOPE SUBROUTINE

1935  004230  005737  005716            TST     CHNFLG                  ;BRANCH IF IN CHAIN MODE
1936  004234  001002                    BNE     5$
1937  004236  004737  003144            JSR     PC,PUBLISH              ;GO PUBLISH TEST DATA
1938  004242  105037  001121    5$:     CLRB    @#ITCNT                 ;RESET ITERATION COUNT
1939  004246  000676                    BR      1$
1940
1941                                    .SBTTL  TIMER SUBROUTINES
1942
1943                                    ;SUBROUTINE TO SYNCHRONIZE THE TIMER AND TURN IT ON.
1944                                    ;REGISTER 4 IS CLEARED, AND THE OSCILLATOR POLARITY IS MONITORED
1945                                    ;THE ROUTINE IS EXITED WHEN THE OSCILLATOR POLARITY CHANGES WITH R3
1946                                    ;SET TO INDICATE THE POLARITY OF THE OSCILLATOR.
1947                                    ;CALL:  JSR     PC,TIMON
1948                                    ;RETURNS:       R3 SET TO INDICATE LAST POLARITY (+24/-24=0/1)
1949                                    ;               R4 = 0
1950
1951  004250  005004            TIMON:  CLR     R4                      ;CLEAR TIME COUNT
1952  004252  012703  000024            MOV     #24,R3                  ;SET POLARITY TO '0' STATE
1953  004256  032765  000100  000024    BIT     #OSC,MR(R5)             ;BRANCH IF POLARITY IS '0'
1954  004264  001405                    BEQ     2$
1955  004266  032765  000100  000024    BIT     #OSC,MR(R5)     1$:     ;WAIT FOR OSCILLATOR TO RETURN
1956  004274  001374                    BNE     1$
1957  004276  000405                    BR      4$
1958
1959  004300  005403            2$:     NEG     R3                      ;NEGATE PREV POLARITY INDICATOR
1960  004302  032765  000100  000024    BIT     #OSC,MR(R5)     3$:     ;WAIT FOR OSCILLATOR TO RETURN
1961  004310  001774                    BEQ     3$                      ;TO '1' STATE
1962  004312  000207            4$:     RTS     PC
1963
1964                                    ;SUBROUTINE TO COUNT TIME
1965                                    ;EACH TIME THE OSCILLATOR TOGGLES (BIT <06> IN MR REG) REGISTER
1966                                    ;R4 IS INCREMENTED, AND THE REGISTER R3 IS NEGATED TO INDICATE
1967                                    ;THE LAST STATE OF THE OSCILLATOR.
1968                                    ;CALL   JMP     TIMER(R3)               ;R3 IS SET BY TIMON ROUTINE
1969                                    ;       R2=RETURN ADDRESS TO CALLER
1970                                    ;NOTE: THE TIME TO EXECUTE THIS ROUTINE IS VERY CRITICAL. IT MUST BE
1971                                    ;LESS THAN 40 US.
1972
1973                                    ;ENTER HERE VIA JMP  TIMER(R3) WHEN R3=-24 (PREV STATE=1)
1974  004314  032765  000100  000024    TIMER1: BIT     #OSC,MR(R5)             ;BRANCH IF CURRENT STATE IS '0'
1975  004322  001406                    BEQ     TIMER                   ;GO INCREMENT TIME
1976  004324  000112                    JMP     (R2)                    ;RETURN TO TEST
1977
1978          004340                    .=TIMER1+24
1979  004340  005403            TIMER:  NEG     R3                      ;NEGATE PREV STATE INDICATOR
1980  004342  005204                    INC     R4                      ;INCREMENT 'TICK' COUNT
1981  004344  100401                    BMI     TIMERR                  ;BRANCH ON OVERFLOW
1982  004346  000112                    JMP     (R2)                    ;RETURN TO TEST
1983  004350  000004  014346    TIMERR: TYPE,E.TIMOV                    ;TYPE 'TIMER OVERFLOWED'
1984  004354  104400                    HLT                             ;REPORT HARDWARE ERROR
1985  004356  000177  174420            JMP     @SCPADR                 ;RETURN TO BEGINNING OF TEST
1986
1987          004364                    .=TIMER+24
1988                                    ;ENTER HERE VIA JMP  TIMER(R3) WHEN R3=+24 (PREV STATE=0)
1989  004364  032765  000100  000024    TIMER0: BIT     #OSC,MR(R5)             ;BRANCH IF CURRENT STATE = '1'
1990  004372  001362                    BNE     TIMER
```

```
                        JMP     (R2)

                ;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

004376          TIMOK:
004376  004737  002354          JSR     PC,.SAVE                ;SAVE REGISTERS ON THE STACK
004402  012700  000070          MOV     #56.,R0                 ;GET TIME PER TICK
004406  010401                  MOV     R4,R1                   ;GET TICKS COUNT
004410  005002                  CLR     R2                      ;CLEAR SUMMING REGISTERS
004412  005003                  CLR     R3
004414  060002          1$:     ADD     R0,R2                   ;MULTIPLY TIME PER TICK
004416  005503                  ADC     R3                      ;BY TICK COUNT
004420  005301                  DEC     R1
004422  001374                  BNE     1$
004424  010246                  MOV     R2,-(SP)                ;DIVIDE COUNT BY 10.

004426  010346                  MOV     R3,-(SP)
004430  012746  000012          MOV     #10.,-(SP)
004434  004737  004722          JSR     PC,DIVIDE
004440  005726                  TST     (SP)+                   ;DISCARD REMAINDER
004442  012637  001012          MOV     (SP)+,@#ATIME           ;STORE QUOTIENT
004446  113700  001122          MOVB    @#TSTNUM,R0             ;GET TEST #
004452  006300                  ASL     R0
004454  006300                  ASL     R0
004456  023760  001012  001416  CMP     @#ATIME,STIMTBL(R0)     ;CHECK THAT TIME IS WITHIN
004464  101004                  BHI     2$                      ;LIMITS SPECIFIED
004466  023760  001012  001420  CMP     @#ATIME,STIMTBL+2(R0)
004474  101001                  BHI     3$
004476  104401          2$:     HLT+1                           ;CALL ERROR ROUTINE
004500                  3$:
004500  004737  002376          JSR     PC,.RESTORE             ;RESTORE REGISTERS FROM THE STACK
004504  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

004506          GAPOK:
004506  004737  002354          JSR     PC,.SAVE                ;SAVE REGISTERS ON THE STACK
004512  012700  000070          MOV     #56.,R0                 ;GET TIME PER TICK
004516  010401                  MOV     R4,R1                   ;GET TICK COUNT
004520  005002                  CLR     R2                      ;CLEAR SUMMING REGISTERS
004522  005003                  CLR     R3
004524  060002          1$:     ADD     R0,R2                   ;MULTIPLY TICK COUNT
004526  005503                  ADC     R3                      ;BY TIME PER TICK
```

```
                        DEC     R1
                        BNE     1$

                        MOV     R2,-(SP)                ;DIVIDE TIME BY 10.
                        MOV     R3,-(SP)
                        MOV     #10.,-(SP)
                        JSR     PC,DIVIDE
                        TST     (SP)+                   ;DISCARD REMAINDER
                        MOV     (SP)+,@#ATIME           ;STORE QUOTIENT
                        MOVB    @#GAP,R3                ;GET GAP #
                        ASL     R3                      ;MULTPLY BY 4
                        ASL     R3                      ;TO GET AT TABLE ENTRY
                        CMP     @#ATIME,GTIMTBL(R3)     ;CHECK TIME (MAX)
                        BHI     2$
                        CMP     @#ATIME,GTIMTBL+2(R3)   ;CHECK TIME (MIN)
                        BHI     3$
2$:                     HLT+2                           ;REPORT OUT OF RANGE ERROR
                        BR      100$
3$:                     BIT     #SW08,@SWR              ;BRANCH IF TIMES NOT WANTED
                        BEQ     100$
                        JSR     PC,OUTGAP               ;TYPE GAP TIMES

100$:
                        JSR     PC,.RESTORE             ;RESTORE REGISTERS FROM THE STACK
                        RTS     PC                              ;RETURN TO TEST

                .SBTTL          DELAY SUBROUTINES
;THIS SUBROUTINE CAUSES A DELAY OF 350 MS.
DELAY:  JSR     PC,TIMON
                MOV     R2,-(SP)                ;SAVE R2 ON THE STACK
                MOV     #2$,R2                  ;SET RETURN ADDRESS FOR TIMER
1$:
                JMP     TIMER(R3)               ;GO TO TIMER & RETURN VIA R2
2$:     BIT     #4000,R4
                BEQ     1$
                MOV     (SP)+,R2                ;RESTORE R2
                RTS     PC

;THIS SUBROUTINE ALLOWS A CALLER SPECIFIED DELAY (UP TO 65MS.)
;CALL:  MOV     DELAY TIME,DELTIM       ;LOAD DELAY TIME (IN US)
;       JSR     PC,DELAYV
DELAYV: TST     DELTIM                  ;BRANCH IF 0 DELAY
                BEQ     3$
                JSR     PC,TIMON                ;TURN TIMER ON
                MOV     R2,-(SP)                ;SAVE R2 ON THE STACK
                MOV     #2$,R2                  ;SET RETURN ADDRESS FROM TIMER
1$:
                JMP     TIMER(R3)               ;GO TO TIMER & RETURN VIA R2
2$:     CMP     @#DELTIM,R4
                BHI     1$
                MOV     (SP)+,R2                ;RESTORE R2
3$:     RTS     PC

                .SBTTL  DIVIDE SUBROUTINE
;THIS SUBROUTINE DIVIDES A DOUBLE PRECISION # AND RETURNS THE RESULT
;TO THE CALLER ON THE STACK. BOTH DIVIDEND & DIVISOR MUST BE POSITIVE.
```

```
2103                                    ;CALL:  MOV     LEAST SIGNIFICANT HALF DIVIDEND,-(SP)
2104                                    ;       MOV     #MOST SIGNIFICANT HALF DIVIDEND,-(SP)
2105                                    ;       MOV     #DIVISOR,-(SP)
2106                                    ;       JSR     PC,DIVIDE
2107                                    ;RETURN
2108                                    ;       (SP)=REMAINDER ON STACK
2109                                    ;       2(SP)=QUOTIENT
2110
2111                                    ;NOTE:  THIS SUBROUTINE DESTROYS PREVIOUS CONTENTS OF R0,R1,R2 & R3.
2112
2113  004722  005046                    DIVIDE: CLR     -(SP)                   ;SAVE LOC FOR SIGNS
2114  004724  012746  000021                    MOV     #17.,-(SP)              ;SET ITERATION COUNT
2115  004730  016601  000012                    MOV     12(SP),R1               ;GET LSH DIVIDEND
2116  004734  016600  000010                    MOV     10(SP),R0               ;GET MSH DIVIDEND
2117  004740  016602  000006                    MOV     6(SP),R2                ;GET DIVISOR
2118  004744  005402                            NEG     R2                      ;NEGATE DIVISOR
2119  004746  000241                            CLC                             ;CLEAR 'C' BIT IN PSW
2120  004750  000405                            BR      2$
2121  004752  006100                    1$:     ROL     R0                      ;ROTATE MSH DIVIDEND
2122  004754  010003                            MOV     R0,R3                   ;SAVE IN R3
2123  004756  060203                            ADD     R2,R3                   ;SUBTRACT DIVISOR FROM MSH DIVIDEND
2124  004760  103001                            BCC     2$                      ;BRANCH IF DIVIDEND > DIVISOR
2125  004762  010300                            MOV     R3,R0                   ;SAVE REMAINDER IN R0
2126  004764  006101                    2$:     ROL     R1                      ;ROTATE LSH DIVIDEND
2127  004766  005316                            DEC     (SP)                    ;DECREMENT ITERATION COUNT
2128  004770  001370                            BNE     1$
2129  004772  005726                            TST     (SP)+                   ;POP ITERATION COUNTER
2130  004774  005726                            TST     (SP)+                   ;POP SIGN CORRECTION
2131  004776  010166  000006                    MOV     R1,6(SP)                ;PUSH REMAINDER ON STACK
2132  005002  010066  000004                    MOV     R0,4(SP)                ;PUSH QUOTIENT ONTO STACK
2133  005006  012616                            MOV     (SP)+,(SP)
2134  005010  000207                            RTS     PC
2135
2136                                            .SBTTL  DRIVE SUBROUTINES
2137                                    ;SUBROUTINE TO CHECK IF DRIVE IS AVAILABLE
2138                                    ;CALL:  MOVB    DRIVE#,DRVNUM
2139                                    ;       JSR     PC,DRVAVA
2140                                    ;RETURN:        'C' BIT SET IF NOT AVAILABLE
2141  005012  113765  001004  000010    DRVAVA: MOVB    @#DRVNUM,CS2(R5)        ;LOAD DRIVE #
2142  005020  032765  040000  000026            BIT     #TAP,DT(R5)             ;CHECK IF TAPE UNIT
2143  005026  001003                            BNE     1$
2144  005030  004737  005070                    JSR     PC,RHINIT
2145  005034  000262                            SEV                             ;SET 'V' TO IND NOT AVAIL
2146  005036  000207                    1$:     RTS     PC                      ;RETURN
2147
2148                                    ;SUBROUTINE TO CHECK IF TE16 SLAVE IS AVAILABLE FOR TEST
2149                                    ;CALL:  MOVB    DRIVE #,@#DRVNUM        ;PASS DRIVE # VIA DRVNUM
2150                                    ;       MOVB    SLAVE #,@#SLVNUM        ;PASS SLAVE # VIA SLVNUM
2151                                    ;       JSR     PC,SLVAVA               ;CALL SUBROUTINE
2152  005040  113765  001004  000010    SLVAVA: MOVB    @#DRVNUM,CS2(R5)        ;LOAD DRIVE #
2153  005046  113765  001005  000032            MOVB    @#SLVNUM,TC(R5)         ;AND SLAVE #
2154  005054  032765  002000  000026            BIT     #SPR,DT(R5)             ;BRANCH IF SLAVE PRESENT
2155  005062  001001                            BNE     1$
2156  005064  000262                            SEV                             ;SET 'V' TO INDICATE NO SLAVE
2157  005066  000207                    1$:     RTS     PC
2158
```

```
                                        ;SUBROUTINE TO  INITIALIZE RH CONTROLLER
                                        ;CALL:  JSR PC,RHINIT
                                        
005070  012765  000040  000010  RHINIT: MOV     #40,CS2(R5)
005076  113765  001004  000010          MOVB    @#DRVNUM,CS2(R5)
005104  005046                         CLR     -(SP)
005106  113716  001005                 MOVB    @#SLVNUM,(SP)
005112  012665  000032                 MOV     (SP)+,TC(R5)            ;LOAD SLAVE # INTO TC REG
005116  052765  001700  000032         BIS     #NORM11,TC(R5)
005124  000207                         RTS     PC
                                        
                                        ;SUBROUTINE TO WAIT FOR DRIVE READY (DRY)
005126  005027                 WAITRDY:CLR     (PC)+                   ;CLEAR WAIT TIMER
005130  000000                 WAITTIM:.WORD   0
005132  105765  000012         1$:     TSTB    DS(R5)                  ;WAIT FOR READY TO SET
005136  100406                         BMI     2$
005140  005237  005130                 INC     WAITTIM                 ;INCREMENT WAIT TIMER
005144  001372                         BNE     1$                      ;BRANCH IF TIME HAS NOT EXPIRED
005146  000004  014373                 TYPE,E.TIMEXP                   ;TYPE 'TIME EXPIRED WAITING FOR RDY'
005152  000425                         BR      99$                     ;TAKE ERROR EXIT
005154  032765  002000  000012  2$:     BIT     #EOT,DS(R5)             ;CHECK FOR END OF TAPE
005162  001415                         BEQ     3$                      ;BRANCH IF NO EOT
005164  000004  013252                 TYPE,M.NAM
005170  000004  013756                 TYPE,M.EOT                      ;TYPE 'END OF TAPE'
005174  004737  005232                 JSR     PC,.REWIND              ;REWIND SLAVE
005200  102412                         BVS     99$                     ;BRANCH IF ERROR ON REWIND
005202  004737  005314                 JSR     PC,WRITE                ;WRITE A RECORD
005206  005215                         INC     (R5)                    ;SET 'GO' BIT
005210  004737  005126                 JSR     PC,WAITRDY              ;WAIT FOR READY
005214  000404                         BR      99$                     ;TAKE ERROR EXIT
005216  032765  040000  000012  3$:     BIT     #ERR,DS(R5)             ;CHECK ERROR EXIT
005224  001401                         BEQ     100$
005226  000262                 99$:    SEV
005230  000207                 100$:   RTS     PC
                                        ;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.
                                        
005232  004737  005070         .REWIND:JSR     PC,RHINIT               ;INITIALIZE CONTROLLER
005236  004337  005450                 JSR     R3,TMCMD                ;GO TO TM COMMAND SUBROUTINE
005242  000000                         .WORD   0                       ;BUS ADDRESS (NOT USED)
005244  000000                         .WORD   0                       ;WORD COUNT (NOT USED)
005246  000000                         .WORD   0                       ;FRAME COUNT (NOT USED)
005250  000006                         .WORD   RWD                     ;REWIND COMMAND
005252  005215                         INC     (R5)                    ;SET 'GO' BIT
005254  032765  000002  000012  1$:     BIT     #BOT,DS(R5)             ;BRANCH IF 'BOT' SET
005262  001005                         BNE     2$
005264  032765  040000  000012         BIT     #ERR,DS(R5)             ;CHECK ERROR BIT
005272  001006                         BNE     99$                     ;BRANCH IF ERROR BIT SET
005274  000767                         BR      1$
                                        
005276  032765  020000  000012  2$:     BIT     #PIP,DS(R5)             ;WAIT FOR TAPE MOTION TO STOP
005304  001374                         BNE     2$
```

```
2215  005306  000401                    BR      100$
2216  005310  000262            99$:    SEV
2217  005312  000207            100$:   RTS     PC
2218
2219                            ;SUBROUTINE TO WRITE 256. WORD RECORD
2220                            ;CALL:  JSR     PC,WRITE
2221
2222  005314  004337  005450    WRITE:  JSR     R3,TMCMD                ;GO TO TM COMMAND SUBROUTINE
2223  005320  015730                    .WORD   WTBUF                   ;BUS ADDRESS
2224  005322  177600                    .WORD   WRDCNT                  ;WORD COUNT
2225  005324  177400                    .WORD   FRMCNT                  ;FRAME COUNT
2226  005326  000060                    .WORD   WFWD                    ;WRITE FORWARD COMMAND
2227  005330  000207                    RTS     PC
2228
2229                            ;SUBROUTINE TO READ A 256. WORD RECORD.
2230                            ;CALL:  JSR     PC,READ
2231
2232  005332  004337  005450    READ:   JSR     R3,@#TMCMD
2233  005336  015730                    .WORD   RDBUF                   ;ADDRESS OF READ BUFFER
2234  005340  177600                    .WORD   WRDCNT                  ;2'S COMPLEMENT OF WORD COUNT
2235  005342  177400                    .WORD   FRMCNT                  ;2'S COMPLEMENT OF FRAME COUNT
2236  005344  000070                    .WORD   RDFWD                   ;READ FORWARD COMMAND
2237  005346  000207                    RTS     PC
2238
2239                            ;SUBROUTINE TO INITIATE READ REVERSE COMMAND
2240                            ;CALL:  JSR     PC,REVRD
2241                            ;
2242  005350  004337  005450    REVRD:  JSR     R3,TMCMD
2243  005354  016330                    .WORD   RDBUF+256.              ;ADDRESS OF READ REVERSE BUFFER
2244  005356  177600                    .WORD   WRDCNT                  ;2'S COMPLEMENT OF WORD COUNT
2245  005360  177400                    .WORD   FRMCNT                  ;2'S COMPLEMENT OF FRAME COUNT
2246  005362  000076                    .WORD   RDREV                   ;READ REVERSE COMMAND
2247  005364  000207                    RTS     PC
2248
2249                            ;SUBROUTINE TO SPACE FORWARD 1 RECORD
2250  005366  012765  177777  000006    FWDSPC: MOV     #-1,FC(R5)      ;LOAD RECORD COUNT
2251  005374  012715  000031            MOV     #SPCFWD+1,(R5)          ;LOAD COMMAND
2252  005400  004737  005126            JSR     PC,WAITRDY              ;WAIT FOR READY
2253  005404  000207                    RTS     PC                      ;RETURN
2254
2255                            ;SUBROUTINE TO WRITE A RECORD AND BACK SPACE OVER  THE RECORD.
2256  005406  004737  005314    WRT.BK: JSR     PC,WRITE                ;WRITE THE RECORD
2257  005412  005215                    INC     (R5)                    ;SET 'GO' BIT
2258  005414  004737  005126            JSR     PC,WAITRDY
2259  005420  102412                    BVS     2$
2260  005422  012765  177777  000006    MOV     #-1,FC(R5)              ;LOAD RECORD COUNT
2261  005430  012715  000033            MOV     #SPCREV+1,(R5)          ;LOAD COMMAND
2262  005434  004737  005126            JSR     PC,WAITRDY
2263  005440  102402                    BVS     2$
2264  005442  004737  004634    1$:     JSR     PC,DELAY                ;WAIT FOR TAPE MOTION TO STOP
2265  005446  000207            2$:     RTS     PC
2266
2267                            ;SUBROUTINE TO LOAD A COMMAND
2268                            ;CALL:  JSR     R3,TMCMD
2269                            ;       .WORD   BUS ADDRESS
2270                            ;       .WORD   WORD COUNT (2'S COMPLEMENT)
```

```
                                        ;       .WORD   FRAME COUNT (2'S COMPLEMENT)
                                        ;       .WORD   COMMAND
                                        TMCMD:  MOV     (R3)+,BA(R5)            ;LOAD BUS ADDRESS
005450  012365  000004                          MOV     (R3)+,WC(R5)            ;LOAD WORD COUNT
005454  012365  000002                          MOV     (R3)+,FC(R5)            ;LOAD FRAME COUNT
005460  012365  000006                          MOV     (R3)+,(R5)              ;LOAD COMMAND
005464  012315                                  RTS     R3                      ;RETURN
005466  000203
                                                ;SUBROUTINE TO PRINT TE16 SERIAL NUMBER
                                                ;JSR     PC,SNPT
                                        SNPT:   MOV     SN(R5),R3
005470  016503  000030                          MOV     #ODIGITS,R1
005474  012701  001144                          SWAB    R3
005500  000303                                  ROR     R3
005502  006003                                  ROR     R3
005504  006003                                  ROR     R3
005506  006003                                  ROR     R3
005510  006003                                  BIC     #177760,R3              ;GET FIRST DIGIT
005512  042703  177760                          BIS     #260,R3
005516  052703  000260                          MOVB    R3,(R1)+                ;FILL FIRST DIGIT
005522  110321                                  MOV     SN(R5),R3
005524  016503  000030                          SWAB    R3
005530  000303                                  BIC     #177760,R3
005532  042703  177760                          BIS     #260,R3
005536  052703  000260                          MOVB    R3,(R1)+                ;GET SECOND DIGIT
005542  110321                                  MOV     SN(R5),R3
005544  016503  000030                          ROR     R3
005550  006003                                  ROR     R3
005552  006003                                  ROR     R3
005554  006003                                  ROR     R3
005556  006003                                  BIC     #177760,R3
005560  042703  177760                          BIS     #260,R3
005564  052703  000260                          MOVB    R3,(R1)+                ;GET THIRD DIGIT
005570  110321                                  MOV     SN(R5),R3
005572  016503  000030                          BIC     #177760,R3
005576  042703  177760                          BIS     #260,R3
005602  052703  000260                          MOVB    R3,(R1)+                ;GET FOURTH DIGIT
005606  110321                                  CLRB    (R1)
005610  105011                                  TYPE,ODIGITS                    ;TYPE SERIAL NUMBER
005612  000004  001144                          RTS     PC                      ;RETURN
005616  000207
```

```
                                .SBTTL  PROGRAM INITIALIZATION
005620  012706  000600          INIT:   MOV     #STKPTR,SP              ;SET STACK PTR
005624  005037  001264                  CLR     @#INBUF
                                
005630  013746  000006                  MOV     @#6,-(SP)               ;SAVE VECTORS
005634  013746  000004                  MOV     @#4,-(SP)
005640  012737  005660  000004          MOV     #61$,@#4                        ;SET UP FOR TIMEOUT
005646  022777  177777  173124          CMP     #-1,@SWR                ;REFERENCE HARDWARE SWITCH REGISTER
005654  001402                  BEQ     60$
005656  000404                  BR      62$
005660  022626          61$:    CMP     (SP)+,(SP)+             ;ADJUST STACK
005662  012737  000176  001000  60$:    MOV     #SWREG,SWR              ;POINT TO SOFTWARE SWITCH REG
005670  012637  000004          62$:    MOV     (SP)+,@#4               ;RESTORE VECTORS
005674  012637  000006                  MOV     (SP)+,@#6
005700  105037  001124                  CLRB    @#PRGFLG                ;CLEAR PROGRAM FLAG
005704  105037  001130                  CLRB    @#ASFLG                 ;CLEAR ASK FLAG
005710  105037  001127                  CLRB    @#PSCNT         ;SET PASS COUNT = 0
005714  005027                  CLR     (PC)+           ;;CLEAR CHAIN INDICATOR
005716  000000          CHNFLG: .WORD   0               ;;CHAIN MODE INDICATOR
                                                        ;;1/0 = CHAIN/NOT CHAIN MODE
005720  022737  012642  000042          CMP     #SENDAD,@#42    ;;BRANCH IF LOADED VIA ACT11 CHAIN MODE
005726  001404                  BEQ     50$
005730  005737  000042                  TST     @#42            ;;BRANCH IF IN DUMP MODE
005734  001413                  BEQ     52$
005736  000406                  BR      51$
005740  012737  000176  001000  50$:    MOV     #SWREG,SWR      ;;INVOKE SOFTWARE SWR
005746  012777  100000  173024          MOV     #100000,@SWR    ;;WITH HALT ON ERROR SET
005754  005237  005716          51$:    INC     CHNFLG          ;;SET CHNFLG = CHAIN MODE
005760  000137  006056                  JMP     5$              ;;GO TO CHAIN ADDRESS
005764                          52$:
005764  122737  000006  000041          CMPB    #6,@#41         ;BRANCH IF NOT LOADED VIA TMDP
005772  001002                  BNE     1$
005774  000004  013455                  TYPE,I.REM              ;ADVISE USER TO REMOVE TMDP
006000  000004  013252          1$:     TYPE,M.NAM                      ;TYPE TITLE
006004  105037  013252                  CLRB    M.NAM                   ;DO NOT TYPE TITLE ON RESTART
006010  000004  013522                  TYPE,I.REG                      ;ASK USER TO TYPE CONT BASE ADRS
006014  013702  001010                  MOV     @#TMBASE,R2             ;GET CURRENT CONT BASE ADDRESS
006020  004737  002426                  JSR     PC,TYPOCT               ;AND TYPE IT
006024  000004  001410                  TYPE,SPACE
006030  004737  003272                  JSR     PC,.INPUT               ;GET USER INPUT
006034  122737  000015  001264          CMPB    #CR,@#INBUF             ;DO NOT CHANGE CURRENT VALUE
006042  001405                  BEQ     5$                      ;IF USER TYPES <CR>
006044  004737  003056          4$:     JSR     PC,CNVTAO               ;CONVERT ASCII TO OCTAL
006050  013737  001116  001010          MOV     @#OCTALO,@#TMBASE       ;SET NEW ADDRESS
006056  013705  001010          5$:     MOV     @#TMBASE,R5

                        ;ROUTINE TO CHECK IF CONTROLLER (RH11) IS AVAILAABLE
006062  000261                  SEC                             ;SET 'C' IN PSW
006064  005715                  TST     (R5)                    ;BRANCH IF CONTROLLER AVAIL
006066  103003                  BCC     6$
006070  000004  014015                  TYPE,E.NCON
006074  000651                  BR      INIT
006076  012737  003542  000004  6$:     MOV     #ERRTRP,@#ERRVEC        ;SET ERROR TRAP VECTOR
```

```
2367                                    ;ROUTINE TO GET TM03 DRIVES USER DESIRES TO TEST
2368  006104  105037  001123  DRIVES: CLRB    @#ERFLG                 ;CLEAR ERROR FLAG
2369  006110  012701  001154          MOV     #DRVTBL,R1              ;MARK ALL DRIVES AS NOT TO
2370  006114  012700  000004          MOV     #4,R0                   ;BE TESTED. A '0' INDICATES
2371  006120  005021          1$:     CLR     (R1)+                   ;THAT A DRIVE IS NOT TO BE
2372  006122  005300                  DEC     R0                      ;TESTED
2373  006124  001375                  BNE     1$
2374  006126  005737  005716          TST     CHNFLG                  ;BRANCH IF IN CHAIN MODE
2375  006132  001014                  BNE     2$
2376  006134  000004  013567          TYPE,I.DRVS
2377  006140  004737  003272          JSR     PC,.INPUT               ;GET USER INPUT
2378  006144  012700  001264          MOV     #INBUF,R0
2379  006150  122710  000101          CMPB    #'A,(R0)                ;IF USER RESPONDS WITH 'A' OR
2380  006154  001403                  BEQ     2$                      ;<CR> THEN ALL AVAILABLE DRIVES
2381  006156  122710  000015          CMPB    #CR,(R0)                ;ARE TO BE TESTED
2382  006162  001013                  BNE     4$
2383  006164  110637  001124  2$:     MOVB    SP,PRGFLG               ;SET FLAG TO IND ALL DRIVES
2384  006170  012701  001154          MOV     #DRVTBL,R1              ;MARK ALL DRIVES TO BE TESTED
2385  006174  012700  000004          MOV     #4,R0                   ;A '-1' INDICATES THAT A DRIVE
2386  006200  012721  177777  3$:     MOV     #-1,(R1)+               ;IS TO BE TESTED
2387  006204  005300                  DEC     R0
2388  006206  001374                  BNE     3$
2389  006210  000417                  BR      CHKDRV                  ;GO CHECK DRIVE AVAILABILITY
2390
2391                                    ;GET USER SELECTED DRIVES AND MARK EACH DRIVE SELECTED TO BE TESTED
2392  006212  122710  000015  4$:     CMPB    #CR,(R0)
2393  006216  001414                  BEQ     CHKDRV
2394  006220  121027  000054          CMPB    (R0),#',                ;CHECK IF 'COMMA'
2395  006224  001001                  BNE     5$
2396  006226  105720                  TSTB    (R0)+                   ;STEP PTR PAST 'COMMA'
2397  006230  112001          5$:     MOVB    (R0)+,R1
2398  006232  042701  177770          BIC     #177770,R1
2399  006236  112761  177777  001154  MOVB    #-1,DRVTBL(R1)
2400  006244  000240                  NOP
2401  006246  000761                  BR      4$
2402
2403                                    ;ASCERTAIN THAT DRIVES (TM03'S) SPECIFIED ARE AVAILABLE
2404  006250  005000  CHKDRV: CLR     R0                              ;A (0) IN DRVTBL(R0) INDICATES
2405  006252  105760  001154  1$:     TSTB    DRVTBL(R0)              ;THE DRIVE IS NOT TO BE TESTED
2406  006256  001005                  BNE     3$                      ;A '1' INDICATES TO BE TESTED
2407  006260  005200          2$:     INC     R0
2408  006262  122700  000010          CMPB    #8.,R0
2409  006266  001371                  BNE     1$
2410  006270  000424                  BR      5$
2411  006272  110037  001004  3$:     MOVB    R0,@#DRVNUM             ;GET DRIVE #
2412  006276  004737  005012          JSR     PC,@#DRVAVA             ;AND CHECK IF AVAILABLE
2413  006302  102366                  BVC     2$                      ;'V' BIT SET INDICATES NOT AVAIL
2414  006304  105737  001124          TSTB    @#PRGFLG                ;DO NOT TYPE NOT AVAILABLE
2415  006310  001011                  BNE     4$                      ;MESSAGE IF ALL SELECTED
2416  006312  000004  014062          TYPE,E.NDRV
2417  006316  116037  001132  014114  MOVB    DIGTAB(R0),@#E.NAVA     ;SET DRIVE # IN MESSAGE
2418  006324  000004  014114          TYPE,E.NAVA
2419  006330  110637  001123          MOVB    SP,@#ERFLG              ;SET 'ERROR' FLAG
2420  006334  105060  001154  4$:     CLRB    DRVTBL(R0)              ;MARK DRIVE UNAVAILABLE
2421  006340  000747                  BR      2$                      ;CHECK NEXT DRIVE
2422  006342  105737  001123  5$:     TSTB    @#ERFLG                 ;GO GET SLAVES IF NO ERROR
```

```
2423  006346  001256                        BNE     DRIVES                  ;ELSE ASK USER TO RETYPE DRIVES
2424
2425                                        ;ROUTINE TO GET SLAVES (TE16'S) USER DESIRES TO TEST
2426  006350  105037  001123          SLAVES: CLRB    @#ERFLG                 ;CLEAR ERROR INDICATOR
2427  006354  012701  001164                  MOV     #SLVTBL,R1              ;MARK ALL SLAVES (64.) AS NOT
2428  006360  012700  000040                  MOV     #32.,R0                 ;TO BE TESTED.A 0 INDICATES THAT
2429  006364  005021                  1$:     CLR     (R1)+                   ;A DRIVE'S SLAVE IS NOT TO BE
2430  006366  005300                          DEC     R0                      ;TESTED
2431  006370  001375                          BNE     1$
2432  006372  012701  001164                  MOV     #SLVTBL,R1              ;R1 POINTS TO DRIVE'S SLAVE
2433  006376  105760  001154          2$:     TSTB    DRVTBL(R0)              ;BRANCH IF DRIVE IS TO BE TESTED
2434  006402  001007                          BNE     4$                      ;& IS AVAILABLE
2435  006404  062701  000010          3$:     ADD     #8.,R1                  ;STEP SLAVE PTR TO NEXT DRIVE'S
2436  006410  005200                          INC     R0                      ;SLAVES AND INCREMENT DRIVE #
2437  006412  122700  0000[illegible]          CMPB    #8.,R0                  ;CHECK ALL DRIVES
2438  006416  001367                          BNE     2$                      ;AND WHEN ALL DRIVES CHECKED
2439  006420  000457                          BR      CHKSLV                  ;GO CHECK SLAVE AVAILABILITY
2440
2441  006422  105737  001124          4$:     TSTB    @#PRGFLG                ;BRANCH IF USER SELECTED ALL
2442  006426  001021                          BNE     5$                      ;DRIVES
2443  006430  110037  001004                  MOVB    R0,DRVNUM               ;GET DRIVE #
2444  006434  116037  001132  013656          MOVB    DIGTAB(R0),@#I.DRV      ;PREPARE USER ACTION MESSAGE
2445  006442  000004  013637                  TYPE,I.SLVS
2446  006446  004737  003272                  JSR     PC,.INPUT               ;GET USER INPUT
2447  006452  012703  001264                  MOV     #INBUF,R3               ;SET PTR TO USER INPUT
2448  006456  122713  000101                  CMPB    #'A,(R3)                ;AN 'A' OR <CR> AS FIRST CHAR
2449  006462  001403                          BEQ     5$                      ;INDICATES TEST ALL SLAVES
2450  006464  122713  000015                  CMPB    #CR,(R3)
2451  006470  001015                          BNE     7$
2452  006472  110637  001124          5$:     MOVB    SP,@#PRGFLG             ;SET 'ALL' INDICATOR
2453  006476  012701  001164                  MOV     #SLVTBL,R1              ;MARK ALL SLAVES FOR ALL
2454  006502  012700  000040                  MOV     #32.,R0                 ;DRIVES AS TO BE TESTED
2455  006506  012721  177777          6$:     MOV     #-1,(R1)+
2456  006512  005300                          DEC     R0
2457  006514  001374                          BNE     6$
2458  006516  105737  001124                  TSTB    @#PRGFLG                ;BRANCH IF ALL WAS SELECTED
2459  006522  001016                          BNE     CHKSLV
2460
2461  006524  122713  000015          7$:     CMPB    #CR,(R3)                ;GET USER SELECTED SLAVES FOR
2462  006530  001725                          BEQ     3$                      ;DRIVE
2463  006532  121327  000054                  CMPB    (R3),#',                ;STEP PTR PAST 'COMMA'
2464  006536  001001                          BNE     8$
2465  006540  105723                          TSTB    (R3)+
2466  006542  112304                  8$:     MOVB    (R3)+,R4                ;AND MARK SELECED SLAVE
2467  006544  042704  177770                  BIC     #177770,R4              ;AS TO BE TESTED
2468  006550  060104                          ADD     R1,R4
2469  006552  112714  177777                  MOVB    #-1,(R4)
2470  006556  000762                          BR      7$
2471
2472                                        ;ASCERTAIN THAT SLAVES (TE16'S) SELECTED ARE AVAILABLE
2473  006560  005000          CHKSLV: CLR     R0                              ;R0 WILL CONTAIN THE DRIVE #
2474  006562  005001                  CLR     R1                              ;AND R1 THE SLAVE #
2475  006564  012702  001164          MOV     #SLVTBL,R2                      ;SET PTR TO SLAVE TABLE
2476  006570  105760  001154  1$:     TSTB    DRVTBL(R0)                      ;BRANCH IF DRIVE SELECTED
2477  006574  001007                  BNE     3$                              ;& AVAILABLE FOR TEST
2478  006576  005200          2$:     INC     R0                              ;INCREMENT DRIVE #
```

```
2479  006600  062702  000010            ADD     #8.,R2                  ;STEP SLAVE PTR TO NEXT DRIVE'S
2480  006604  022700  000010            CMP     #8.,R0                  ;SLAVES. BRANCH TO 1$ IF NOT ALL
2481  006610  001367                    BNE     1$                      ;DRIVES CHECKED OTHERWISE EXIT
2482  006612  000437                    BR      8$
2483
2484  006614  005001            3$:     CLR     R1                      ;SET SLAVE # 0
2485  006616  105712            4$:     TSTB    (R2)                    ;BRANCH IF DRIVE'S SLAVE IS SEL-
2486  006620  001006                    BNE     6$                      ;ECTED FOR TEST
2487  006622  005201            5$:     INC     R1                      ;INCREMENT SLAVE #
2488  006624  005202                    INC     R2                      ;STEP PTR TO NEXT SLAVE
2489  006626  022701  000010            CMP     #8.,R1                  ;GO TO 4$ IF ALL SLAVES NOT
2490  006632  001371                    BNE     4$                      ;CHECKED
2491  006634  000760                    BR      2$                      ;OTHERWISE GO TO 2$ ABOVE
2492
2493  006636  110037  001004    6$:     MOVB    R0,@#DRVNUM             ;PASS DRIVE & SLAVE #
2494  006642  110137  001005            MOVB    R1,@#SLVNUM
2495  006646  004737  005040            JSR     PC,@#SLVAVA             ;AND CHECK IF AVAILABLE
2496  006652  102363                    BVC     5$                      ;'V' SET INDCATES ERROR
2497  006654  105737  001124            TSTB    @#PRGFLG                ;DO NOT TYPE ERROR MSG IF ALL
2498  006660  001012                    BNE     7$                      ;SLAVES SELECTED
2499  006662  116037  001132  014104    MOVB    DIGTAB(R0),@#E.DRV      ;ICATES ERROR. PREPARE ERROR
2500  006670  116137  001132  014114    MOVB    DIGTAB(R1),@#E.NAVA     ;MESSAGE
2501  006676  000004  014076            TYPE,E.NSLV
2502  006702  110637  001123            MOVB    SP,@#ERFLG              ;SET ERROR INDICATOR
2503  006706  105012            7$:     CLRB    (R2)                    ;CLEAR SLAVE TABLE ENTRY
2504  006710  000744                    BR      5$                      ;GET NEXT SLAVE
2505
2506  006712  105737  001123    8$:     TSTB    @#ERFLG                 ;BRANCH IF ERROR
2507  006716  001214                    BNE     SLAVES                  ;ASK USER TO RETYPE SLAVES
2508  006720  012737  003542  000004  100$:   MOV     #ERRTRP,@#ERRVEC
2509
2510                                    ;SCAN DRIVE AND SLAVE TABLE FOR DRIVE/SLAVE COMBINATION TO TEST.
2511                                    ;RESTART ADDRESS--PROGRAM STARTS HERE WHEN START ADDRESS = 210 AND
2512                                    ;AFTER EACH PASS
2513  006726  105037  001004    RSTRT:  CLRB    @#DRVNUM                ;SET DRIVE AND SLAVE # 0
2514  006732  105037  001005            CLRB    @#SLVNUM
2515  006736  012737  001164  001006    MOV     #SLVTBL,@#SLVPTR        ;SET PTR TO SLAVE TABLE
2516  006744  105037  001125            CLRB    @#UNTFND                ;CLEAR 'UNIT FOUND' IND.
2517
2518                                    ;PROGRAM RESTARTS HERE AFTER EACH DRIVE/SLAVE HAS BEEN TESTED.
2519  006750  113700  001004    BEGIN:  MOVB    @#DRVNUM,R0             ;GET DRIVE #
2520  006754  113701  001005            MOVB    @#SLVNUM,R1             ;AND SLAVE #
2521  006760  013702  001006            MOV     @#SLVPTR,R2             ;GET SLAVE PTR
2522  006764  122737  000006  000041    CMPB    #6,@#41                 ;BRANCH IF LOADED VIA TMDP
2523  006772  001001                    BNE     1$
2524  006774  105012                    CLRB    (R2)                    ;SET DRIVE #0,SLAVE #0 NOT TO
2525                                                                    ;BE TESTED.
2526  006776  105760  001154    1$:     TSTB    DRVTBL(R0)              ;BRANCH IF DRIVE AVAIL TO TEST
2527  007002  001011                    BNE     3$
2528  007004  005001                    CLR     R1                      ;CLEAR SLAVE #
2529  007006  062702  000010            ADD     #8.,R2                  ;AND STEP PTR TO NEXT DRIVE'S
2530  007012  005200            2$:     INC     R0                      ;SLAVES AND INCREMENT DRIVE #
2531  007014  022700  000010            CMP     #8.,R0                  ;EXIT TEST IF ALL DRIVES
2532  007020  001366                    BNE     1$                      ;CHECKED OTHERWISE CONTINUE
2533  007022  000137  012570            JMP     @#END                   ;SCAN FOR NEXT 'UNIT'
2534
```

```
2535  007026  105712                  3$:     TSTB    (R2)            ;BRANCH IF SLAVE ON DRIVE IS
2536  007030  001007                          BNE     4$              ;AVAILABLE THERWISE STEP
2537  007032  005202                          INC     R2              ;PTR TO NEXT SLAVE
2538  007034  005201                          INC     R1              ;INCREMENT SLAVE #
2539  007036  122701  000010                  CMPB    #8.,R1          ;UNTIL ALL SLAVES CHECKED
2540  007042  001371                          BNE     3$              ;WHEN ALL SLAVES CHECKED
2541  007044  005001                          CLR     R1              ;SET SLAVE # 0
2542  007046  000761                          BR      2$              ;AND CONTINUE SCAN
2543
2544  007050  110637  001125          4$:     MOVB    SP,@#UNTFND     ;INDICATE THAT A 'UNIT' IS FOUND
2545  007054  110037  001004                  MOVB    R0,@#DRVNUM     ;SET DRIVE 3
```

```
2546  007060  110137  001005          MOVB    R1,@#SLVNUM             ;SET SLAVE #
2547  007064  010237  001006          MOV     R2,@#SLVPTR             ;SAVE SLAVE PTR
2548
2549  007070  105737  001130  5$:     TSTB    @#ASFLG
2550  007074  001034                  BNE     7$
2551  007076  112737  000001  001130  MOVB    #1,ASFLG
2552  007104  005737  005716          TST     CHNFLG                  ;BRANCH IF IN CHAIN MODE
2553  007110  001026                  BNE     7$
2554  007112  105037  001124          CLRB    @#PRGFLG        ;CLEAR PROGRAM INDICATOR
2555  007116  000004  013723          TYPE,I.SPD              ;ASK USER IF HE WANTS TO RUN SPEED TESTS
2556  007122  004737  003272          JSR     PC,.INPUT               ;GET USER INPUT
2557  007126  012703  001264          MOV     #INBUF,R3       ;GET REPLY
2558  007132  122713  000015          CMPB    #CR,(R3)        ;DO NOT DO SKEW TESTS IF <CR> IS FIRST
2559  007136  001405                  BEQ     6$
2560  007140  132713  000001          BITB    #1,(R3)         ;BRANCH IF 'N'
2561  007144  001402                  BEQ     6$
2562  007146  111337  001124          MOVB    (R3),@#PRGFLG   ;SET INDICATOR
2563  007152  022737  000176  001000  6$:     CMP     #SWREG,SWR              ;BRANCH IF SOFTWARE SWR
2564  007160  001002                  BNE     7$              ;NOT INVOKED
2565  007162  004737  002020          JSR     PC,GTSWR                ;GET SWITCH REGISTER
2566  007166                  7$:
2567
2568                          ;NOTE THIS IS NOT A TEST
2569                          ;INITIALIZE PROGRAM FLAGS
2570  007166  013705  001010  TSTOOO: MOV     @#TMBASE,R5             ;SET ADDRESS OF FIRST TM03 REG
2571  007172  010500                  MOV     R5,R0
2572  007174  062700  000006          ADD     #FC,R0                  ;R0 CONTAINS ADDRESS OF FC REG
2573  007200  010501                  MOV     R5,R1
2574  007202  062701  000012          ADD     #DS,R1                  ;R1 CONTAINS ADDRESS OF DS REG
2575  007206  012703  004340          MOV     #TIMER,R3               ;SET JUMP ADDRESS TO TIMER
2576  007212  105037  001121          CLRB    @#ITCNT                 ;CLEAR SUBTEST ITERATION COUNT
2577  007216  052737  000100  177560  BIS     #100,@#TKS              ;SET KEYBOARD IE BIT
2578
2579                          ;GET USER RUN PROCEDURE
2580                          ;IF SWR <05::00> IS NOT 0 THEN RUN TEST IN SWR<05::00>
2581                          ;OTHERWISE RUN ALL TESTS
2582
2583  007224  004737  005232          JSR     PC,.REWIND              ;REWIND SLAVE
2584  007230  102474                  BVS     99$                     ;BRANCH IF ERROR ON REWIND
2585  007232  004737  005314          JSR     PC,WRITE                ;WRITE A RECORD
2586  007236  005215                  INC     (R5)                    ;SET 'GO' BIT
2587  007240  004737  005126          JSR     PC,WAITRDY              ;WAIT FOR READY
2588  007244  102466                  BVS     99$
2589  007246  005737  005716          TST     CHNFLG                  ;BRANCH IF IN CHAIN MODE
2590  007252  001064                  BNE     100$
2591  007254  117702  171520          MOVB    @SWR,R2                 ;GET SWITCHES
2592  007260  042702  177740          BIC     #177740,R2              ;CLEAR ALL BUT TEST #
2593  007264  001421                  BEQ     2$                      ;& BRANCH IF TEST 0 WAS SELECTED
2594  007266  000004  014235          TYPE,E.HDR              ;TYPE TEST #
2595  007272  004737  002426          JSR     PC,TYPOCT
2596  007276  006302                  ASL     R2                      ;FORM INDEX VALUE
2597  007300  016237  001656  007310  MOV     NAMPTR(R2),1$           ;GET ADDRESS OF TEST'S NAME
2598  007306  000004                  TYPE                            ;AND TYPE IT
2599  007310  000000          1$:     .WORD   0
2600  007312  000004  001374          TYPE,CRLF
2601  007316  016237  001736  001002  MOV     TSTTBL(R2),@#SCPADR     ;SET SCOPE ADDRESS FOR TEST
```

```
 2602  007324  000172  001736                JMP     @TSTTBL(R2)             ;GO TO TEST
 2603  007330  000004  014444          2$:   TYPE,L.HDR1
 2604  007334  113702  001004                MOVB    DRVNUM,R2               ;GET DRIVE #
 2605  007340  113704  001005                MOVB    SLVNUM,R4               ;AND SLAVE #
 2606  007344  116237  001132  014624        MOVB    DIGTAB(R2),@#L.DRV      ;SET DRIVE AND SLAVE #'S
 2607  007352  116437  001132  014636        MOVB    DIGTAB(R4),@#L.SLV      ;INTO L.HDR2 MESSAGE
 2608  007360  112737  000071  014641        MOVB    #'9,@#L.CHAN            ;GET SLAVES CHANNEL TYPE
 2609  007366  000004  014557                TYPE,L.HDR2
 2610  007372  004737  005470                JSR     PC,SNPT                 ;PRINT SLAVE SERIAL #
 2611  007376  000004  014660                TYPE,L.HDR3
 2612  007402  105737  001124                TSTB    @#PRGFLG                ;BRANCH IF SPEED TESTS NOT
 2613  007406  001406                        BEQ     100$                    ;SELECTED
 2614  007410  012737  012714  001002        MOV     #TST026,@#SCPADR        ;SET SCOPE LOOP ADDRESS
 2615  007416  000137  012706                JMP     @#SKEWTST               ;GO DO SPEED TESTS
 2616  007422  104400                  99$:  HLT
 2617  007424  012737  007432  001002  100$:  MOV     #TST001,@#SCPADR        ;SET SCOPE LOOP ADDRESS
 2618
```

```
                                        .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.
007432  112737  000001  001122  TST001: MOVB    #1,@#TSTNUM             ;SET TEST #
007440  012702  007464                   MOV     #1$,R2                  ;SET RETURN PC FROM TIMER
007444  004737  005232                   JSR     PC,.REWIND              ;REWIND SLAVE
007450  102420                           BVS     99$                     ;BRANCH IF ERROR ON REWIND
007452  004737  005314                   JSR     PC,WRITE                ;GO SETUP WRITE COMMAND
007456  004737  004250                   JSR     PC,TIMON                ;TURN TIMER ON
007462  005215                           INC     (R5)                    ;SET 'GO' BIT

007464  005765  000032           1$:     TST     TC(R5)                  ;BRANCH WHEN 'ACCL'=0
007470  100002                           BPL     2$
007472  000163  004340                   JMP     TIMER(R3)               ;GO TO TIMER & RETURN VIA R2

007476  004737  005126           2$:     JSR     PC,WAITRDY              ;WAIT FOR COMMAND TO FINISH
007502  102403                           BVS     99$                     ;BRANCH IF ERROR
007504  004737  004376                   JSR     PC,TIMOK                ;GO CHECK TIME
007510  000401                           BR      100$
007512  104400           99$:    HLT
007514  104000           100$:   SCOPE

                                ;TEST 002 - WRITE START
                                ;THIS TST MEASURES TIME FROM 'GO'=1 TO 'ACCL'=0.
007516  112737  000002  001122  TST002: MOVB    #2,@#TSTNUM             ;SET TEST # 2
007524  004737  005314                   JSR     PC,WRITE                ;INITIATE WRITE COMMAND
007530  012702  007542                   MOV     #1$,R2                  ;SET RETURN PC FROM TIMER
007534  004737  004250                   JSR     PC,TIMON
007540  005215                           INC     (R5)                    ;SET 'GO' BIT

007542  005765  000032           1$:     TST     TC(R5)                  ;BRANCH WHEN 'ACCL'=0
007546  100002                           BPL     2$
007550  000163  004340                   JMP     TIMER(R3)               ;GO TO TIMER & RETURN VIA R2

007554  004737  005126           2$:     JSR     PC,WAITRDY              ;WAIT FOR READY
007560  102403                           BVS     99$                     ;BRANCH IF ERROR
007562  004737  004376                   JSR     PC,TIMOK                ;GO CHECK TIME RECORDED
007566  000401                           BR      100$                    ;EXIT VIA SCOPE

007570  104400           99$:    HLT                                     ;REPORT ERROR
007572  104000           100$:   SCOPE

                                ;TEST 003- WRITE SHUTDOWN
                                ;THIS TEST MEASURES TIME FROM 'FC REG'=0 TO 'SWDN'=1.
007574  112737  000003  001122  TST003: MOVB    #3,@#TSTNUM             ;SET TEST#3
007602  004737  005314                   JSR     PC,WRITE                ;INITIATE WRITE COMMAND
007606  005215                           INC     (R5)                    ;SET 'GO' BIT

007610  005710           1$:     TST     (R0)                    ;BRANCH WHEN WRITING FINISHED
007612  001404                           BEQ     2$
007614  032711  040000                   BIT     #ERR,(R1)               ;MONITOR ERROR BIT
007620  001017                           BNE     99$
```

```
                        000772                  BR      1$
                                        2$:
                        004737  004250          JSR     PC,TIMON        ;TURN TIMER ON
                        010702                  MOV     PC,R2           ;LOAD RETURN PC FROM TIMER
                        032711  000020  3$:     BIT     #SDWN,(R1)      ;BRANCH WHEN DS <SDWN> SETS
                        001002                  BNE     4$
                        000163  004340          JMP     TIMER(R3)       ;GO TO TIMER & RETURN VIA R2
                        004737  005126  4$:     JSR     PC,WAITRDY      ;WAIT FOR READY
                        102403                  BVS     99$
                        004737  004376          JSR     PC,TIMOK        ;GO CHECK TIME RECORDED
                        000401                  BR      100$
                        104400          99$:    HLT                     ;REPORT ERROR
                        104000          100$:   SCOPE

                                        ;TEST 004 - WRITE SETTLEDOWN
                                        ;THIS TEST MEASURES TIME FROM 'SWDN'=1 TO 'SWDN'=0.
                        112737  000004  001122  TST004: MOVB    #4,@#TSTNUM
                        004737  005314          JSR     PC,WRITE
                        005215                  INC     (R5)            ;SET 'GO' BIT

                        005710  1$:     TST     (R0)                    ;BRANCH WHEN WRITING FINISHED
                        001404                  BEQ     2$
                        032711  040000          BIT     #ERR,(R1)       ;CHECK ERROR BIT
                        001026                  BNE     99$
                        000772                  BR      1$

                        032711  000020  2$:     BIT     #SDWN,(R1)      ;WAIT FOR ASSERTION OF 'SDWN'
                        001004                  BNE     3$
                        032711  040000          BIT     #ERR,(R1)       ;MONITOR ERROR BIT
                        001017                  BNE     99$
                        000771                  BR      2$

                                        3$:
                        004737  004250          JSR     PC,TIMON        ;TURN TIMER ON
                        010702                  MOV     PC,R2           ;SET RETURN PC FROM TIMER
                        032711  000020          BIT     #SDWN,(R1)      ;BRANCH WHEN SWDN CLEARS
                        001402                  BEQ     5$
                        000163  004340          JMP     TIMER(R3)       ;GO TO TIMER & RETURN VIA R2

                        004737  005126  5$:     JSR     PC,WAITRDY      ;WAIT FOR READY
                        102403                  BVS     99$
                        004737  004376          JSR     PC,TIMOK
                        000401                  BR      100$

                        104400          99$:    HLT
                        104000          100$:   SCOPE

                                        ;TEST 005 - READ FROM BOT
                                        ;THIS TEST MEASURES TIME FROM 'GO'=1 TO 'ACCL'=0.
                        112737  000005  001122  TST005: MOVB    #5,@#TSTNUM     ;SET TEST #5
                        004737  005232          JSR     PC,.REWIND      ;REWIND SLAVE
                        102422                  BVS     99$             ;BRANCH IF ERROR ON REWIND
                        004737  005332          JSR     PC,READ
                        012702  010024          MOV     #1$,R2          ;SET RETURN PC FROM TIMER
```

```
2731  010016  004737  004250          JSR     PC,TIMON                ;TURN TIMER ON
2732  010022  005215                  INC     (R5)                    ;SET 'GO' BIT
2733
2734  010024  005765  000032  1$:     TST     TC(R5)                  ;BRANCH WHEN 'ACCL' RESETS
2735  010030  100002                  BPL     2$
2736  010032  000163  004340          JMP     TIMER(R3)               ;GO TO TIMER & RETURN VIA R2
2737
2738  010036  004737  005126  2$:     JSR     PC,WAITRDY              ;WAIT FOR READY
2739  010042  102403                  BVS     99$                     ;BRANCH IF ERROR
2740  010044  004737  004376          JSR     PC,TIMOK                ;CHECK RECORDED TIME
2741  010050  000401                  BR      100$
2742
2743  010052  104400  99$:    HLT
2744  010054  104000  100$:   SCOPE
2745
2746                  ;TEST 006 - READ START
2747                  ;THIS TEST MEASURES TIME FROM 'GO'=1 TO 'ACCL'=0.
2748  010056  112737  000006  001122  TST006: MOVB    #6,@#TSTNUM             ;SET TEST #6
2749  010064  004737  005406          JSR     PC,WRT.BK               ;WRITE A RECORD & BACK SPACE
2750  010070  102422                  BVS     99$
2751  010072  004737  005332          JSR     PC,READ
2752  010076  012702  010110          MOV     #1$,R2                  ;SET RETURN PC FROM TIMER
2753  010102  004737  004250          JSR     PC,TIMON                ;TURN TIMER ON
2754  010106  005215                  INC     (R5)                    ;SET 'GO' BIT
2755
2756  010110  005765  000032  1$:     TST     TC(R5)                  ;BRANCH WHEN 'ACCL' RESETS
2757  010114  100002                  BPL     2$
2758  010116  000163  004340          JMP     TIMER(R3)               ;GO TO TIMER & RETURN VIA R2
2759
2760  010122  004737  005126  2$:     JSR     PC,WAITRDY
2761  010126  102403                  BVS     99$
2762  010130  004737  004376          JSR     PC,TIMOK
2763  010134  000401                  BR      100$
2764
2765  010136  104400  99$:    HLT
2766  010140  104000  100$:   SCOPE
2767
2768                  ;TEST 007 - READ SHUTDOWN
2769                  ;THIS TEST MEASURES TIME FROM 'FC REG'=FRAME COUNT TO 'SWDN'=1.
2770  010142  112737  000007  001122  TST007: MOVB    #7,@#TSTNUM             ;SET TEST #7
2771  010150  004737  005406          JSR     PC,WRT.BK               ;WRITE A RECORD & BACK SPACE
2772  010154  102430                  BVS     99$                     ;BRANCH IF ERROR
2773  010156  004737  005332          JSR     PC,READ
2774  010162  005215                  INC     (R5)                    ;SET 'GO' BIT
2775
2776  010164  022710  000400  1$:     CMP     #-FRMCNT,(R0)           ;WAIT FOR FRAME COUNT TO
2777  010170  001404                  BEQ     2$                      ;= # OF FRAMES WRITTEN
2778  010172  032711  040000          BIT     #ERR,(R1)               ;MONITOR ERROR BIT
2779  010176  001017                  BNE     99$
2780  010200  000771                  BR      1$
2781
2782  010202          2$:
2783  010202  004737  004250          JSR     PC,TIMON                ;TURN TIMER ON
2784  010206  010702                  MOV     PC,R2                   ;SET RETURN PC FROM TIMER
2785  010210  032711  000020          BIT     #SDWN,(R1)              ;BRANCH WHEN SDWN SETS
2786  010214  001002                  BNE     3$
```

```
2787  010216  000163  004340                  JMP     TIMER(R3)               ;GO TO TIMER & RETURN VIA R2
2788
2789  010222  004737  005126          3$:     JSR     PC,WAITRDY
2790  010226  102403                          BVS     99$
2791  010230  004737  004376                  JSR     PC,TIMOK
2792  010234  000401                          BR      100$
2793
2794  010236  104400                  99$:    HLT                             ;REPORT ERROR
2795  010240  104000                  100$:   SCOPE
2796
2797                                  ;TEST 010 - READ SETTLEDOWN
2798                                  ;THIS TEST MEASURES TIME FROM 'SWDN'=1 TO 'SWDN'=0.
2799  010242  112737  000010  001122  TST010: MOVB    #10,@#TSTNUM            ;SET TEST #10
2800  010250  012702  010326                  MOV     #4$,R2                  ;SET RETURN PC FROM TIMER
2801  010254  004737  005406                  JSR     PC,WRT.BK               ;WRITE A RECORD & BACK SPACE
2802  010260  102436                          BVS     99$
2803  010262  004737  005332                  JSR     PC,READ
2804  010266  005215                          INC     (R5)                    ;SET 'GO' BIT
2805
2806  010270  105711                  1$:     TSTB    (R1)                    ;WAIT FOR READY
2807  010272  100404                          BMI     2$                      ;BRANCH WHEN SET
2808  010274  032711  040000                  BIT     #ERR,(R1)               ;CHECK ERROR BIT
2809  010300  001026                          BNE     99$
2810  010302  000772                          BR      1$
2811
2812  010304  032711  000020          2$:     BIT     #SDWN,(R1)              ;WAIT FOR ASSERTION OF 'SDWN'
2813  010310  001004                          BNE     3$
2814  010312  032711  040000                  BIT     #ERR,(R1)               ;MONITOR ERROR BIT
2815  010316  001017                          BNE     99$
2816  010320  000771                          BR      2$
2817
2818  010322                          3$:
2819  010322  004737  004250                  JSR     PC,TIMON                ;TURN TIMER ON
2820  010326  032765  000020  000012  4$:     BIT     #SDWN,DS(R5)            ;WAIT FOR NEGATION OF SDWN
2821  010334  001402                          BEQ     5$
2822  010336  000163  004340                  JMP     TIMER(R3)               ;GO TO TIMER & RETURN VIA R2
2823
2824  010342  004737  005126          5$:     JSR     PC,WAITRDY
2825  010346  102403                          BVS     99$
2826  010350  004737  004376                  JSR     PC,TIMOK
2827  010354  000401                          BR      100$
2828
2829  010356  104400                  99$:    HLT
2830  010360  104000                  100$:   SCOPE
2831
2832
2833                                  ;TEST 011-READ REVERSE START
2834                                  ;THIS TEST MEASURES TIME FROM 'GO'=1 TO 'ACCL'=0.
2835  010362  112737  000011  001122  TST011: MOVB    #11,@#TSTNUM
2836  010370  012702  010426                  MOV     #1$,R2                  ;SET RETURN PC FROM TIMER
2837  010374  004737  005314                  JSR     PC,WRITE                ;WRITE A RECORD
2838  010400  005215                          INC     (R5)                    ;SET 'GO' BIT
2839  010402  004737  005126                  JSR     PC,WAITRDY
2840  010406  102422                          BVS     99$
2841  010410  004737  004634                  JSR     PC,DELAY                ;WAIT FOR TAPE MOTION TO STOP
2842  010414  004737  005350                  JSR     PC,REVRD
```

```
2843  010420  004737  004250            JSR     PC,TIMON                ;TURN TIMER ON
2844  010424  005215                    INC     (R5)                    ;SET 'GO' BIT
2845
2846  010426  005765  000032  1$:       TST     TC(R5)                  ;BRANCH WHEN 'ACCL' = 0
2847  010432  100002                    BPL     2$
2848  010434  000163  004340            JMP     TIMER(R3)               ;GO TO TIMER & RETURN VIA R2
2849
2850  010440  004737  005126  2$:       JSR     PC,WAITRDY
2851  010444  102403                    BVS     99$                     ;BRANCH IF ERROR
2852  010446  004737  004376            JSR     PC,TIMOK
2853  010452  000401                    BR      100$
2854
2855  010454  104400  99$:     HLT
2856  010456  104000  100$:    SCOPE
2857
2858                  ;TEST 012-READ REVERSE SHUTDOWN
2859                  ;THIS TEST MEASURES TIME FROM 'FC REG' = FRAME COUNT TO 'SDWN'=1.
2860  010460  112737  000012  001122  TST012: MOVB    #12,@#TSTNUM
2861  010466  012702  010536            MOV     #3$,R2                  ;SET RETURN PC FROM TIMER
2862  010472  004737  005314            JSR     PC,WRITE                ;WRITE A RECORD
2863  010476  005215                    INC     (R5)                    ;SET 'GO' BIT
2864  010500  004737  005126            JSR     PC,WAITRDY
2865  010504  102427                    BVS     99$
2866  010506  004737  005350            JSR     PC,REVRD
2867  010512  005215                    INC     (R5)                    ;SET 'GO' BIT
2868
2869  010514  022710  000400  1$:       CMP     #-FRMCNT,(R0)           ;BRANCH WHEN FRAME COUNT
2870  010520  001404                    BEQ     2$                      ;= # OF RECORD WRITTEN
2871  010522  032711  040000            BIT     #ERR,(R1)               ;MONITOR ERROR BIT IN 'DS' REG
2872  010526  001016                    BNE     99$
2873  010530  000771                    BR      1$
2874
2875  010532          2$:
2876  010532  004737  004250            JSR     PC,TIMON                ;TURN TIMER ON
2877  010536  032711  000020  3$:       BIT     #SDWN,(R1)              ;BRANCH WHEN SDWN SETS
2878  010542  001002                    BNE     4$
2879  010544  000163  004340            JMP     TIMER(R3)               ;GO TO TIMER & RETURN VIA R2
2880
2881  010550  004737  005126  4$:       JSR     PC,WAITRDY              ;WAIT FOR READY
2882  010554  102403                    BVS     99$
2883  010556  004737  004376            JSR     PC,TIMOK
2884  010562  000401                    BR      100$
2885
2886  010564  104400  99$:     HLT
2887  010566  104000  100$:    SCOPE
2888
2889                  ;TEST 013-READ REVERSE SETTLEDOWN
2890                  ;THIS TEST MEASURES TIME FROM 'SDWN'=1 TO 'SDWN'=0.
2891  010570  112737  000013  001122  TST013: MOVB    #13,@#TSTNUM
2892  010576  012702  010662            MOV     #4$,R2                  ;SET RETURN PC FROM TIMER
2893  010602  004737  005314            JSR     PC,WRITE                ;WRITE A RECORD
2894  010606  005215                    INC     (R5)                    ;SET 'GO' BIT
2895  010610  004737  005126            JSR     PC,WAITRDY
2896  010614  102435                    BVS     99$
2897  010616  004737  005350            JSR     PC,REVRD
2898  010622  005215                    INC     (R5)                    ;SET 'GO' BIT
```

```
                                                1$:     TSTB    (R1)                    ;BRANCH WHEN
010624  105711                                          BMI     2$                      ;READY SETS
010626  100404
010630  032711  040000                                 BIT     #ERR,(R1)
010634  001025                                         BNE     99$
010636  000772                                         BR      1$

010640  032711  000020                        2$:     BIT     #SDWN,(R1)
010644  001004                                         BNE     3$
010646  032711  040000                                 BIT     #ERR,(R1)
010652  001016                                         BNE     99$
010654  000771                                         BR      2$

010656                                          3$:
010656  004737  004250                                 JSR     PC,TIMON                ;TURN TIMER ON
010662  032711  000020                        4$:     BIT     #SDWN,(R1)              ;BRANCH WHEN SWDN = 0
010666  001402                                         BEQ     5$
010670  000163  004340                                 JMP     TIMER(R3)               ;GO TO TIMER & RETURN VIA R2

010674  004737  005126                        5$:     JSR     PC,WAITRDY              ;WAIT FOR READY
010700  102403                                         BVS     99$
010702  004737  004376                                 JSR     PC,TIMOK
010706  000401                                         BR      100$

010710  104400                                99$:    HLT
010712  104000                                100$:   SCOPE

                                                ;REWIND DRIVE
010714                                          A:
010714  004737  005232                                 JSR     PC,.REWIND              ;REWIND SLAVE
010720  102401                                         BVS     99$                     ;BRANCH IF ERROR ON REWIND
010722  102002                                         BVC     100$
010724  104400                                99$:    HLT
010726  000772                                         BR      A
010730                                          100$:

                                                ;TEST 014-TURN AROUND DELAY (FORWARD-REVERSE)
                                                ;THIS TEST MEASURES TIME FROM 'GO'=1 (READ REVERSE) TO 'ACCL'=0
010730  112737  000014  001122                TST014: MOVB    #14,@#TSTNUM
010736  012702  010770                                MOV     #2$,R2                  ;SET RETURN PC FROM TIMER
010742  004737  005314                                JSR     PC,WRITE                ;WRITE A RECORD
010746  005215                                        INC     (R5)                    ;SET 'GO' BIT
010750  004737  005126                                JSR     PC,WAITRDY
010754  102420                                        BVS     99$

010756  004737  005350                        1$:     JSR     PC,REVRD                ;READ THE RECORD (REVERSE)
010762  004737  004250                                JSR     PC,TIMON                ;TURN TIMER ON
010766  005215                                        INC     (R5)                    ;SET 'GO' BIT

010770  005765  000032                        2$:     TST     TC(R5)                  ;WAIT FOR 'ACCL' = 0
010774  100002                                        BPL     3$
010776  000163  004340                                JMP     TIMER(R3)               ;GO TO TIMER & RETURN VIA R2

011002  004737  005126                        3$:     JSR     PC,WAITRDY
011006  102403                                        BVS     99$
011010  004737  004376                                JSR     PC,TIMOK
```

```
                                        BR      100$

                                99$:    HLT
                                100$:   SCOPE

                                ;TEST 015- TURN AROUND DELAY (REVERSE-FORWARD)
                                ;THIS TEST MEASURES TIME FROM 'GO'=1 (READ) TO 'ACCL'=0.
                                TST015: MOVB    #15,@#TSTNUM
                                        MOV     #2$,R2                  ;SET RETURN PC FROM TIMER
                                        JSR     PC,WRITE                ;WRITE A RECORD
                                        INC     (R5)                    ;SET 'GO' BIT
                                        JSR     PC,WAITRDY              ;WAIT FOR READY
                                        BVS     99$
                                        JSR     PC,REVRD                ;READ A RECORD IN THE
                                        INC     (R5)                    ;SET 'GO' BIT

                                        JSR     PC,WAITRDY
                                        BVS     99$

                                1$:     JSR     PC,READ                 ;READ RECORD FORWARD
                                        JSR     PC,TIMON                ;TURN TIMER ON
                                        INC     (R5)                    ;SET 'GO' BIT

                                2$:     TST     TC(R5)                  ;WAIT FOR 'ACCL' = 0
                                        BPL     3$
                                        JMP     TIMER(R3)               ;GO TO TIMER & RETURN VIA R2

                                3$:     JSR     PC,WAITRDY
                                        BVS     99$
                                        JSR     PC,TIMOK
                                        BR      100$

                                99$:    HLT
                                100$:   SCOPE

                                ;TEST 016-GAP SIZE (STOP HALF)
                                TST016: MOVB    #16,@#TSTNUM
                                        MOV     #1$,R2                  ;SET RETURN PC FROM TIMER
                                        JSR     PC,WRITE                ;WRITE A RECORD
                                        INC     (R5)                    ;SET 'GO' BIT
                                        JSR     PC,WAITRDY
                                        BVS     99$
                                        JSR     PC,DELAY                ;DELAY 350 MS
                                        JSR     PC,REVRD                ;READ REVERSE RECORD
                                        JSR     PC,TIMON                ;TURN TIMER ON
                                        INC     (R5)                    ;SET 'GO' BIT

                                1$:     TST     (R0)                    ;WAIT FOR FRAME COUNT > 0
                                        BNE     2$
                                        JMP     TIMER(R3)               ;GO TO TIMER & RETURN VIA R2

                                2$:     JSR     PC,WAITRDY              ;WAIT FOR READY BIT TO SET
                                        BVS     99$
                                        JSR     PC,TIMOK                ;CHECK TIME
                                        BR      100$
```

```
3011  011220  104400                  99$:    HLT
3012  011222  104000                  100$:   SCOPE
3013
3014                                  ;TEST 017-GAP SIZE (START HALF)
3015  011224  112737  000017  001122  TST017: MOVB    #17,@#TSTNUM
3016  011232  012702  011304                  MOV     #1$,R2          ;SET RETURN PC FROM TIMER
3017  011236  004737  005314                  JSR     PC,WRITE        ;WRITE A RECORD
3018  011242  005215                          INC     (R5)            ;SET 'GO' BIT
3019  011244  004737  005126                  JSR     PC,WAITRDY      ;WAIT FOR READY
3020  011250  102427                          BVS     99$
3021  011252  004737  005350                  JSR     PC,REVRD        ;READ REVERSE THE RECORD
3022  011256  005215                          INC     (R5)            ;SET 'GO' BIT
3023  011260  004737  005126                  JSR     PC,WAITRDY      ;WAIT FOR READY
3024  011264  102421                          BVS     99$             ;BRANCH ON ERROR
3025  011266  004737  004634                  JSR     PC,DELAY        ;WAIT FOR TAPE MOTION TO STOP
3026  011272  004737  005332                  JSR     PC,READ         ;READ RECORD
3027  011276  004737  004250                  JSR     PC,TIMON        ;TURN TIMER ON
3028  011302  005215                          INC     (R5)            ;SET 'GO' BIT
3029
3030  011304  005710                  1$:     TST     (R0)            ;WAIT FOR FRAME COUNT > 0
3031  011306  001002                          BNE     2$
3032  011310  000163  004340                  JMP     TIMER(R3)       ;GO TO TIMER & RETURN VIA R2
3033
3034  011314  004737  005126          2$:     JSR     PC,WAITRDY      ;WAIT FOR READY
3035  011320  102403                          BVS     99$
3036  011322  004737  004376                  JSR     PC,TIMOK        ;CHECK TIME
3037  011326  000401                          BR      100$
3038
3039  011330  104400                  99$:    HLT
3040  011332  104000                  100$:   SCOPE
3041
3042                                  ;TEST 020- GAP SIZE (INTERRECORD)
3043                                  ;THIS TEST MEASURES TIME FROM 'GO'=1 TO 'FC REG' >0.
3044  011334  112737  000020  001122  TST020: MOVB    #20,@#TSTNUM
3045  011342  012702  011424                  MOV     #1$,R2          ;SET RETURN PC FROM TIMER
3046  011346  004737  005314                  JSR     PC,WRITE        ;WRITE A RECORD
3047  011352  005215                          INC     (R5)            ;SET 'GO' BIT
3048  011354  004737  005126                  JSR     PC,WAITRDY      ;WAIT FOR READY
3049  011360  102433                          BVS     99$
3050  011362  004737  005314                  JSR     PC,WRITE        ;WRITE SECOND RECORD
3051  011366  005215                          INC     (R5)            ;SET 'GO' BIT
3052  011370  004737  005126                  JSR     PC,WAITRDY      ;WAIT FOR READY
3053  011374  102425                          BVS     99$
3054  011376  004737  005350                  JSR     PC,REVRD        ;READ REVERSE SECOND RECORD
3055  011402  005215                          INC     (R5)            ;SET 'GO' BIT
3056  011404  004737  005126                  JSR     PC,WAITRDY      ;WAIT FOR READY
3057  011410  102417                          BVS     99$
3058  011412  004737  005350                  JSR     PC,REVRD        ;READ REVERSE FIRST RECORD
3059  011416  004737  004250                  JSR     PC,TIMON        ;TURN TIMER ON
3060  011422  005215                          INC     (R5)            ;SET 'GO' BIT
3061
3062  011424  005710                  1$:     TST     (R0)            ;WAIT FOR FRAME COUNT > 0
3063  011426  001002                          BNE     2$
3064  011430  000163  004340                  JMP     TIMER(R3)       ;GO TO TIMER & RETURN VIA R2
3065
3066  011434  004737  005126          2$:     JSR     PC,WAITRDY      ;WAIT FOR READY
```

```
3067  011440  102403                        BVS     99$
3068  011442  004737  004376                JSR     PC,TIMOK
3069  011446  000401                        BR      100$
3070
3071  011450  104400                99$:    HLT
3072  011452  104000                100$:   SCOPE
3073
3074                                ;TEST 021- GAP CONSISTANCY
3075                                ;THIS TEST MEASURES TIME FROM 'GO'=1 TO 'FC REG' > 0.
3076                                ;THE TEST REWINDS THE TAPE,WRITES 17 RECORDS WITH A DELAY FROM 1-16 MS
3077                                ;BETWEEN EACH WRITE COMMAND. AFTER THE 17. RECORDS ARE WRITTTEN THE
3078                                ;PROGRAM READ REVERSES 16 RECORDS. AT THIS POINT THE TAPE IS STOPPED BE-
3079                                ;TWEEN THE FIRST AND SECOND RECORD. A READ COMMAND IS EXECUTED TO READ
3080                                ;THE 16 RECORDS WITH THE TIME BETEWEN GO=1 TO FC > 0 STORED IN 'GAPTBL'
3081                                ;FOR EACH RECORD READ. AFTER 16 RECORDS HAVE BEEN READ THE TIME IS VER-
3082                                ;IFIED FOR EACH READ. AFTER ALL RECORD TIMES ARE VERIFIED THEY ARE AVER-
3083                                ;AGED AND PLACED IN THE 'ATIMTBL' (BY SCOPE). THE ABOVE PROCESS IS RE-
3084                                ;PEATED FOR EACH ITERATION.
3085
3086  011454  112737  000021  001122  TST021: MOVB    #21,@#TSTNUM
3087  011462  012702  011620                MOV     #4$,R2          ;SET RETURN PC FROM TIMER
3088  011466  004737  005232                JSR     PC,.REWIND      ;REWIND SLAVE
3089  011472  102530                        BVS     99$             ;BRANCH IF ERROR ON REWIND
3090  011474  005037  001114                CLR     DELTIM          ;CLEAR VARIABLE DELAY TIME
3091  011500  012700  000021                MOV     #17.,R0         ;SET # OF RECORDS TO WRITE
3092  011504  004737  005314        1$:     JSR     PC,WRITE        ;WRITE 17. RECORDS
3093  011510  005215                        INC     (R5)            ;SET 'GO' BIT
3094  011512  004737  005126                JSR     PC,WAITRDY      ;WAIT FOR READY
3095  011516  102516                        BVS     99$
3096  011520  004737  004664                JSR     PC,DELAYV       ;DELAY BEFORE WRITING NEXT REC.
3097  011524  062737  000022  001114        ADD     #18.,DELTIM     ;SET NEXT DELAY TIME
3098  011532  005300                        DEC     R0              ;DECREMENT RECORDS WRITTEN COUNT
3099  011534  001363                        BNE     1$
3100
3101  011536  012700  000021                MOV     #17.,R0         ;SET # OF RECS. TO REVERSE READ
3102  011542  004737  005350        2$:     JSR     PC,REVRD        ;REVERSE READ 17. RECORDS
3103  011546  005215                        INC     (R5)            ;SET 'GO' BIT
3104  011550  004737  005126                JSR     PC,WAITRDY      ;WAIT FOR READY
3105  011554  102477                        BVS     99$
3106  011556  005300                        DEC     R0              ;DECREMENT RECORD COUNT
3107  011560  001370                        BNE     2$
3108
3109  011562  012700  000020                MOV     #16.,R0         ;SET # OF RECORDS TO READ
3110  011566  012701  001054                MOV     #GAPTBL,R1      ;SET PTR TO GAP TABLE FOR TEST
3111  011572  004737  005332                JSR     PC,READ         ;READ A RECORD
3112  011576  005215                        INC     (R5)            ;SET 'GO' BIT
3113
3114  011600  004737  005126        3$:     JSR     PC,WAITRDY      ;WAIT FOR READY
3115  011604  102463                        BVS     99$
3116  011606  004737  005332                JSR     PC,READ         ;READ NEXT RECORD
3117  011612  004737  004250                JSR     PC,TIMON        ;TURN TIMER ON
3118  011616  005215                        INC     (R5)            ;SET 'GO' BIT
3119
3120  011620  005765  000006        4$:     TST     FC(R5)          ;WAIT FOR FRAME COUNT > 0
3121  011624  001002                        BNE     5$
3122  011626  000163  004340                JMP     TIMER(R3)       ;GO TO TIMER & RETURN VIA R2
```

```
3123
3124  011632  004737  005126          5$:     JSR     PC,WAITRDY              ;WAIT FOR READY
3125  011636  102446                          BVS     99$
3126  011640  010421                          MOV     R4,(R1)+                ;STORE TIME IN GAPTBL
3127  011642  005300                          DEC     R0                      ;DECREMENT # OF RECORDS READ
3128  011644  001355                          BNE     3$
3129
3130  011646  105037  001120                  CLRB    @#GAP                   ;SET GAP # 0
3131  011652  012700  000020                  MOV     #16.,R0
3132  011656  012701  001054                  MOV     #GAPTBL,R1
3133
3134  011662  012104                  6$:     MOV     (R1)+,R4                ;GET GAP TICK COUNT
3135  011664  004737  004506                  JSR     PC,GAPOK                ;CHECK TIME
3136  011670  105237  001120                  INCB    @#GAP                   ;INCREMENT GAP #
3137  011674  122737  000020  001120          CMPB    #16.,@#GAP              ;BRANCH IF ALL GAPS NOT CHECKED
3138  011702  001367                          BNE     6$
3139
3140  011704  012700  000020                  MOV     #16.,R0                 ;SETUP TO AVERAGE GAP SIZES
3141  011710  012701  001054                  MOV     #GAPTBL,R1              ;SET PTR TO TABLE
3142  011714  005002                          CLR     R2                      ;CLEAR 'SUM' REGISTERS
3143  011716  005003                          CLR     R3
3144  011720  062102                  7$:     ADD     (R1)+,R2                ;ADD ALL GAP SIZES TOGETHER
3145  011722  005503                          ADC     R3
3146  011724  005300                          DEC     R0
3147  011726  001374                          BNE     7$
3148  011730  012700  000004                  MOV     #4,R0                   ;NOW DIVIDE BY 16.
3149  011734  006203                  8$:     ASR     R3                      ;BY SHIFTING 4 PLACES RIGHT
3150  011736  006002                          ROR     R2
3151  011740  005300                          DEC     R0
3152  011742  001374                          BNE     8$
3153  011744  010204                          MOV     R2,R4                   ;MOVE AVERAGED TIMES TO R4
3154  011746  004737  004376                  JSR     PC,TIMOK                ;CHECK AVERAGED TIMES
3155  011752  000401                          BR      100$
3156
3157  011754  104400                  99$:    HLT
3158  011756  104000                  100$:   SCOPE
3159
3160
3161
3162                                  ;TEST 022-DATA TIME (800BPI)
3163                                  ;THIS TEST MEASURES THE TIME FROM FC REG >-6400 TO 'RDY' = 1.
3164  011760  112737  000022  001122  TST022: MOVB    #022,@#TSTNUM
3165  011766  012702  012046                  MOV     #3$,R2                  ;SET RETURN PC FROM TIMER
3166  011772  004737  005232                  JSR     PC,.REWIND              ;REWIND SLAVE
3167  011776  102442                          BVS     99$                     ;BRANCH IF ERROR ON REWIND
3168  012000  052765  001700  000032          BIS     #NORM11,TC(R5)          ;SET 800 BPI
3169  012006  004337  005450                  JSR     R3,TMCMD                ;WRITE 3200. WORD RECORD
3170  012012  015730                          .WORD   WTBUF
3171  012014  171600                          .WORD   -3200.
3172  012016  163400                          .WORD   -6400.
3173  012020  000060                          .WORD   WFWD
3174  012022  005215                          INC     (R5)                    ;SET 'GO' BIT
3175
3176  012024  022710  163400          1$:     CMP     #-6400.,(R0)            ;WAIT FOR WRITING TO START
3177  012030  001004                          BNE     2$
3178  012032  032711  040000                  BIT     #ERR,(R1)               ;MONITOR ERROR BIT
```

```
                        BNE     99$
                        BR      1$

                2$:
                        JSR     PC,TIMON                ;TURN TIMER ON
                3$:     TSTB    (R1)                    ;BRANCH WHEN READY SETS
                        BMI     4$
                        JMP     TIMER(R3)               ;GO TO TIMER & RETURN VIA R2

                4$:     MOV     #3,R0                   ;SET SHIFT COUNT
                5$:     ASR     R4
                        DEC     R0
                        BNE     5$
                        JSR     PC,WAITRDY
                        BVS     99$
                        JSR     PC,TIMOK                ;CHECK TIME
                        BR      100$

                99$:    HLT
                100$:   SCOPE

                ;TEST 023-DATA TIME (1600BPI)
                ;THIS TEST MEASURES THE TIME FROM FC REG >-6400 TO 'RDY' = 1.
001122          TST023: MOVB    #023,@#TSTNUM
                        MOV     #3$,R2                  ;SET RETURN PC FROM TIMER
                        JSR     PC,.REWIND              ;REWIND SLAVE
                        BVS     99$                     ;BRANCH IF ERROR ON REWIND
000032                  BIC     #3700,TC(R5)            ;CLEAR CURRENT DENSITY
000032                  BIS     #PE1600,TC(R5)          ;SET 1600 BPI
                        JSR     R3,TMCMD                ;WRITE 3200. WORD RECORD
                        .WORD   WTBUF
                        .WORD   -3200.
                        .WORD   -6400.
                        .WORD   WFWD
                        INC     (R5)                    ;SET 'GO' BIT

                1$:     CMP     #-6400.,(R0)            ;BRANCH WHEN WRITING STARTS
                        BNE     2$
                        BIT     #ERR,(R1)               ;MONITOR ERROR BIT
                        BNE     99$
                        BR      1$

                2$:
                        JSR     PC,TIMON                ;TURN TIMER ON
                3$:     TSTB    (R1)                    ;BRANCH WHEN READY SETS
                        BMI     4$
                        JMP     TIMER(R3)               ;GO TO TIMER & RETURN VIA R2

                4$:     ASR     R4                      ;DIVIDE TIME BY 4
                        ASR     R4
                        JSR     PC,WAITRDY
                        BVS     99$
                        JSR     PC,TIMOK                ;CHECK TIME
                        BR      100$

                99$:    HLT
```

```
3235  012236  104000                  100$:   SCOPE
3236
3237                                  ;TEST 024-ERASE
3238                                  ;THIS TEST MEASURES TIME FROM 'GO'=1 TO 'RDY'=1.
3239  012240  112737  000024  001122  TST024: MOVB    #24,@#TSTNUM
3240  012246  012702  012330                  MOV     #2$,R2                  ;SET RETURN PC FROM TIMER
3241  012252  004737  005232                  JSR     PC,.REWIND              ;REWIND SLAVE
3242  012256  102436                         BVS     99$                     ;BRANCH IF ERROR ON REWIND
3243  012260  004737  005070                  JSR     PC,RHINIT               ;SET NRZ
3244  012264  004737  005314                  JSR     PC,WRITE                ;WRITE A RECORD
3245  012270  005215                         INC     (R5)                    ;SET 'GO' BIT
3246  012272  004737  005126                  JSR     PC,WAITRDY
3247  012276  102426                         BVS     99$
3248  012300  012737  012306  001002          MOV     #1$,@#SCPADR
3249  012306  004337  005450          1$:     JSR     R3,@#TMCMD
3250  012312  000000                         .WORD   0
3251  012314  000000                         .WORD   0
3252  012316  000000                         .WORD   0
3253  012320  000024                         .WORD   ERASE
3254  012322  004737  004250                  JSR     PC,TIMON                ;TURN TIMER ON
3255  012326  005215                         INC     (R5)                    ;SET 'GO' BIT
3256
3257  012330  105711                  2$:     TSTB    (R1)                    ;BRANCH WHEN READY SETS
3258  012332  100402                         BMI     3$
3259  012334  000163  004340                  JMP     TIMER(R3)               ;GO TO TIMER & RETURN VIA R2
3260
3261  012340  004737  005126          3$:     JSR     PC,WAITRDY
3262  012344  102403                         BVS     99$
3263  012346  004737  004376                  JSR     PC,TIMOK
3264  012352  000401                         BR      100$
3265
3266  012354  104400                  99$:    HLT
3267  012356  104000                  100$:   SCOPE
3268
3269                                  ;TEST 025 TAPE MARK
3270                                  ;THIS TEST MEASURES TIME FROM 'GO'=1 TO 'RDY'=1.
3271  012360  112737  000025  001122  TST025: MOVB    #25,@#TSTNUM
3272  012366  012702  012430                  MOV     #1$,R2                  ;SET RETURN PC FROM TIMER
3273  012372  004737  005314                  JSR     PC,WRITE                ;WRITE A RECORD
3274  012376  005215                         INC     (R5)                    ;SET 'GO' BIT
3275  012400  004737  005126                  JSR     PC,WAITRDY
3276  012404  102423                         BVS     99$
3277  012406  004337  005450                  JSR     R3,@#TMCMD
3278  012412  000000                         .WORD   0
3279  012414  000000                         .WORD   0
3280  012416  000000                         .WORD   0
3281  012420  000026                         .WORD   WFMK
3282  012422  004737  004250                  JSR     PC,TIMON                ;TURN TIMER ON
3283  012426  005215                         INC     (R5)                    ;SET 'GO' BIT
3284
3285  012430  105711                  1$:     TSTB    (R1)                    ;BRANCH WHEN READY SETS
3286  012432  100402                         BMI     2$
3287  012434  000163  004340                  JMP     TIMER(R3)               ;GO TO TIMER & RETURN VIA R2
3288
3289  012440  004737  005126          2$:     JSR     PC,WAITRDY
3290  012444  102403                         BVS     99$
```

```
3291  012446  004737  004376                JSR     PC,TIMOK
3292  012452  000401                        BR      100$
3293
3294  012454  104400              99$:      HLT
3295  012456                    100$:
3296  012456  004737  005232                JSR     PC,.REWIND              ;REWIND SLAVE
3297  012462  102774                        BVS     99$                     ;BRANCH IF ERROR ON REWIND
3298  012464  104000                        SCOPE
3299
```

```
3300  012466  032777  002000  166304  FINISH: BIT     #SW10,@SWR              ;DO NOT SPACE PAPER
3301  012474  001011                          BNE     2$                      ;IF USER SELECTED NO OUTPUT
3302  012476  005737  005716                  TST     CHNFLG                  ;OR IF IN CHAIN MODE
3303  012502  001006                          BNE     2$
3304  012504  012700  000012                  MOV     #10.,R0                 ;SET LINE FEED COUNT
3305  012510  000004  001374          1$:     TYPE,CRLF
3306  012514  005300                          DEC     R0
3307  012516  001374                          BNE     1$
3308
3309
3310  012520  105237  001005          2$:     INCB    @#SLVNUM                ;SET NEXT SLAVE #
3311  012524  005237  001006                  INC     @#SLVPTR                ;AND ITS POINTER
3312  012530  122737  000010  001005          CMPB    #8.,@#SLVNUM            ;BRANCH IF LAST SLAVE (7)
3313  012536  001402                          BEQ     3$
3314  012540  000137  006750                  JMP     @#BEGIN                 ;BEGIN TEST ON NEXT SLAVE
3315  012544  105037  001005          3$:     CLRB    @#SLVNUM                ;SET SLAVE #0
3316  012550  105237  001004                  INCB    @#DRVNUM                ;AND INCREMENT DRIVE #
3317  012554  122737  000010  001004          CMPB    #8.,@#DRVNUM            ;AND CHECK IF LAST DRIVE
3318  012562  001402                          BEQ     END
3319  012564  000137  006750                  JMP     @#BEGIN
3320
3321  012570  105737  001125          END:    TSTB    @#UNTFND                ;BRANCH IF A UNIT WAS FOUND
3322  012574  001004                          BNE     1$
3323  012576  000004  014147                  TYPE,E.UNIT
3324  012602  000137  005620                  JMP     @#INIT
3325  012606  105237  001127          1$:     INCB    @#PSCNT         ;INCREMENT PASS COUNT
3326  012612  000004  013406                  TYPE,M.EOP
3327  012616  113702  001127                  MOVB    @#PSCNT,R2      ;GET PASSCOUNT
3328  012622  004737  002426                  JSR     PC,TYPOCT       ;AND TYPE IT
3329  012626  000004  001374                  TYPE,CRLF
3330  012632  013700  000042                  MOV     @#42,R0         ;GET ACT11 RETURN ADDRESS
3331  012636  001405                          BEQ     HERE            ;BRANCH IF NOT ACT11
3332  012640  000005                          RESET
3333  012642  004710                  SENDAD: JSR     PC,(R0)
3334  012644  000240                          NOP
3335  012646  000240                          NOP
3336  012650  000240                          NOP
3337  012652  000240                  HERE:   NOP
3338  012654  005737  005716                  TST     CHNFLG          ;BRANCH IF CHAIN MODE
3339  012660  001004                          BNE     1$
3340  012662  032777  000100  166110          BIT     #SW06,@SWR              ;BRANCH IF NOT CONTINOUS LOOP
3341  012670  001402                          BEQ     2$
3342  012672  000137  006726          1$:     JMP     @#RSTRT                 ;RESTART
3343  012676  000000                  2$:     HALT
3344  012700  000005                          RESET
3345  012702  000137  005620                  JMP     @#INIT                  ;RESTART
```

```
3346                                    ;SKEW TAPE TIMING TESTS
3347                                    ;THE FOLLOWING TESTS REQUIRE A SPECIALLY WRITTEN 800 BPI SKEW TAPE
3348  012706  012737  012714  001002    SKEWTST:MOV     #TST026,@#SCPADR        ;SET SCOPE POINTER
3349
3350                                    ;TEST 026- SKEW TAPE SPEED TEST-FORWARD
3351                                    ;THIS TEST READS 32" OF TAPE (26400.-800. = 25600. FRAMES) THEN
3352                                    ;DIVIDES TIME BY 32. TO GET TIME TO READ 1" (800. FRAMES) OF TAPE.
3353  012714  112737  000026  001122    TST026: MOVB    #26,@#TSTNUM
3354  012722  012702  013000                    MOV     #2$,R2                  ;SET RETURN PC FROM TIMER
3355  012726  004737  005232                    JSR     PC,.REWIND              ;REWIND SLAVE
3356  012732  102441                            BVS     99$                     ;BRANCH IF ERROR ON REWIND
3357  012734  052765  001700  000032            BIS     #NORM11,TC(R5)  ;SET 800 BPI
3358  012742  052765  000010  000010            BIS     #BAI,CS2(R5)            ;INHIBIT BUS ADDRESS INCREMENT
3359  012750  004337  005450                    JSR     R3,@#TMCMD              ;READ 32" OF TAPE-FORWARD
3360  012754  015730                            .WORD   RDBUF
3361  012756  177777                            .WORD   -1.
3362  012760  063440                    10$:    .WORD   26400.                  ;FRAME COUNT
3363  012762  000070                            .WORD   RDFWD
3364  012764  005215                            INC     (R5)                    ;SET 'GO' BIT
3365
3366  012766  022710  001440            1$:     CMP     #800.,(R0)              ;WAIT FOR FIRST 800 FRAMES
3367  012772  101375                            BHI     1$                      ;TO BE READ
3368
3369  012774  004737  004250                    JSR     PC,TIMON                ;TURN TIMER ON
3370  013000  023710  012760            2$:     CMP     @#10$,(R0)              ;WAIT FOR READING TO FINISH
3371  013004  103402                            BLO     3$
3372  013006  000163  004340                    JMP     TIMER(R3)               ;GO TO TIMER & RETURN VIA R2
3373
3374  013012  012700  000005            3$:     MOV     #5,R0                   ;DIVIDE TIME BY 32.
3375  013016  006204                    4$:     ASR     R4
3376  013020  005300                            DEC     R0
3377  013022  001375                            BNE     4$
3378  013024  004737  005070                    JSR     PC,RHINIT               ;INIT DRIVE
3379  013030  004737  004376                    JSR     PC,TIMOK                ;CHECK TIME
3380  013034  000401                            BR      100$
3381
3382  013036  104400                    99$:    HLT
3383  013040  104000                    100$:   SCOPE
3384
3385                                    ;TEST 027-SKEW TAPE SPEED TEST-REVERSE
3386                                    ;THIS TEST READS FORWARD 40" (32000. FRAMES) OF TAPE, THEN READS REVERSE
3387                                    ;32" (26400.-800. = 25600. FRAMES) OF TAPE. THE TIME IS THEN DIVIDED BY
3388                                    ;32. TO GET TIME TO READ 1" (800. FRAMES) OF TAPE.
3389  013042  112737  000027  001122    TST027: MOVB    #27,@#TSTNUM
3390  013050  012702  013176                    MOV     #3$,R2                  ;SET RETURN PC FROM TIMER
3391  013054  004737  005232                    JSR     PC,.REWIND              ;REWIND SLAVE
3392  013060  102465                            BVS     99$                     ;BRANCH IF ERROR ON REWIND
3393  013062  052765  001700  000032            BIS     #NORM11,TC(R5)
3394  013070  052765  000010  000010            BIS     #BAI,CS2(R5)
3395  013076  004337  005450                    JSR     R3,@#TMCMD              ;READ FORWARD 32000. FRAMES
3396  013102  015730                            .WORD   RDBUF
3397  013104  177777                            .WORD   -1.                     ;WORD COUNT
3398  013106  076400                    10$:    .WORD   32000.                  ;FRAME COUNT
3399  013110  000070                            .WORD   RDFWD                   ;READ FORWARD
3400  013112  005215                            INC     (R5)                    ;SET 'GO' BIT
3401
```

```
3402  013114  023710  013106          1$:     CMP     @#10$,(R0)
3403  013120  101375                          BHI     1$
3404
3405  013122  004737  005070                  JSR     PC,RHINIT               ;INIT DRIVE
3406  013126  004737  004634                  JSR     PC,DELAY                ;WAIT FOR TAPE MOTION TO STOP
3407  013132  052765  001700  000032          BIS     #NORM11,TC(R5)          ;SET 800 BPI
3408  013140  052765  000010  000010          BIS     #BAI,CS2(R5)            ;INHIBIT BUS ADDRESS INCRENENT
3409  013146  004337  005450                  JSR     R3,@#TMCMD              ;READ REVERSE 32" OF TAPE
3410  013152  015730                          .WORD   RDBUF                   ;READ BUFFER
3411  013154  177777                          .WORD   -1.                     ;WORD COUNT
3412  013156  063440          11$:    .WORD   26400.                  ;FRAME COUNT
3413  013160  000076                          .WORD   RDREV                   ;READ REVERSE
3414  013162  005215                          INC     (R5)                    ;SET 'GO' BIT
3415
3416  013164  022710  001440          2$:     CMP     #800.,(R0)              ;WAIT FOR FIRST 800 FRAMES
3417  013170  101375                          BHI     2$                      ;TO BE READ
3418
3419  013172  004737  004250                  JSR     PC,TIMON                ;TURN TIMER ON
3420  013176  023710  013156          3$:     CMP     @#11$,(R0)              ;WAIT FOR ALL FRAMES TO BE READ
3421  013202  103402                          BLO     4$
3422  013204  000163  004340                  JMP     TIMER(R3)               ;GO TO TIMER & RETURN VIA R2
3423
3424  013210  012700  000005          4$:     MOV     #5,R0                   ;DIVIDE TIME BY 32.
3425  013214  006204          5$:     ASR     R4
3426  013216  005300                          DEC     R0
3427  013220  001375                          BNE     5$
3428  013222  004737  005070                  JSR     PC,RHINIT
3429  013226  004737  004376                  JSR     PC,TIMOK
3430  013232  000401                          BR      100$
3431
3432  013234  104400          99$:    HLT
3433  013236                  100$:
3434  013236  004737  005232                  JSR     PC,.REWIND              ;REWIND SLAVE
3435  013242  102774                          BVS     99$                     ;BRANCH IF ERROR ON REWIND
3436  013244  104000                          SCOPE
3437
3438  013246  000137  012466                  JMP     @#FINISH
3439
3440
3441
```

```
                                                        .SBTTL    PROGRAM MESSAGES
                                                ;OPERATOR INSTRUCTIONS
013252  005015  046524  031460  M.NAM:   .ASCII  <CR><LF>'TM03/TE16 DRIVE FUNCTION TIMER (DZTEE-A)'
013260  052057  030505  020066
013266  051104  053111  020105
013274  052506  041516  044524
013302  047117  052040  046511
013310  051105  024040  055104
013316  042524  026505  024501
013324  005015  054524  042520          .ASCIZ  <CR><LF>'TYPE <CR> TO TERMINATE RESPONSE & ↑C TO RESTART'
013332  036040  051103  020076
013340  047524  052040  051105
013346  044515  040516  042524
013354  051040  051505  047520
013362  051516  020105  020046
013370  041536  052040  020117
013376  042522  052123  051101
013404  000124
013406  005015  047105  020104  M.EOP:   .ASCIZ  <CR><LF>'END OF PASS '
013414  043117  050040  051501
013422  020123      000
013425      015  044012  051101  M.HSWR: .ASCIZ  <CR><LF>'HARDWARE SWR IN USE'<CR><LF>
013432  053504  051101  020105
013440  053523  020122  047111
013446  052440  042523  005015
013454      000
013455      015  051012  046505  I.REM:   .ASCIZ  <CR><LF>'REMOVE TMDP FROM TE16 TO BE TESTED'
013462  053117  020105  046524
013470  050104  043040  047522
013476  020115  042524  033061
013504  052040  020117  042502
013512  052040  051505  042524
013520  000104
013522  005015  054524  042520  I.REG:   .ASCIZ  <CR><LF>'TYPE FIRST ADDRESS OF CONTROLLER  '
013530  043040  051111  052123
013536  040440  042104  042522
013544  051523  047440  020106
013552  047503  052116  047522
013560  046114  051105  020040
013566      000
013567      124  050131  020105  I.DRVS: .ASCIZ  %TYPE TM03 DRIVE #'S TO BE TESTED:  ALL %
013574  046524  031460  042040
013602  044522  042526  021440
013610  051447  052040  020117
013616  042502  052040  051505
013624  042524  035104  020040
013632  046101  020114      000
013637      106  051117  052040  I.SLVS: .ASCII  'FOR TM03 DRIVE '
013644  030115  020063  051104
013652  053111  020105
013656  026460  052040  050131  I.DRV:   .ASCIZ  %D- TYPE SLAVE #'S TO BE TESTED: ALL %
013664  020105  046123  053101
013672  020105  023443  020123
013700  047524  041040  020105
013706  042524  052123  042105
013714  020072  046101  020114
```

```
3498  013722  000
3499  013723  123  042520  042105  I.SPD:  .ASCIZ  'SPEED TESTS? (YES/NO): NO '
3500  013730  052040  051505  051524
3501  013736  020077  054450  051505
3502  013744  047057  024517  020072
3503  013752  047516  000040
3504  013756  005015  047105  020104  M.EOT:  .ASCIZ  <CR><LF>'END OF TAPE'<CR><LF>
3505  013764  043117  052040  050101
3506  013772  006505  000012
3507
3508                            ;ERROR MESSAGES
3509  013776  005015  051124  050101  E.TRP4: .ASCIZ  <CR><LF>'TRAPPED TO 4'
3510  014004  042520  020104  047524
3511  014012  032040  000
3512  014015  116  020117  047503  E.NCON: .ASCIZ  'NO CONTROLLER AT ADDRESS SPECIFIED'<CR><LF>
3513  014022  052116  047522  046114
3514  014030  051105  040440  020124
3515  014036  042101  051104  051505
3516  014044  020123  050123  041505
3517  014052  043111  042511  006504
3518  014060  000012
3519  014062  046524  031460  042040  E.NDRV: .ASCIZ  'TM03 DRIVE '
3520  014070  044522  042526  000040
3521  014076  051104  053111  020105  E.NSLV: .ASCII  'DRIVE '
3522  014104  020060  046123  053101  E.DRV:  .ASCII  '0 SLAVE '
3523  014112  020105
3524  014114  020060  047516  020124  E.NAVA: .ASCIZ  '0 NOT AVAILABLE FOR TEST'<CR><LF>
3525  014122  053101  044501  040514
3526  014130  046102  020105  047506
3527  014136  020122  042524  052123
3528  014144  005015  000
3529  014147  116  020117  046524  E.UNIT: .ASCIZ  'NO TM03/TE16 UNIT FOUND TO TEST'<CR><LF>
3530  014154  031460  052057  030505
3531  014162  020066  047125  052111
3532  014170  043040  052517  042116
3533  014176  052040  020117  042524
3534  014204  052123  005015  000
3535  014211  123  043117  020124  E.SFT:  .ASCIZ  'SOFT ERROR (DATA)'<CR><LF>
3536  014216  051105  047522  020122
3537  014224  042050  052101  024501
3538  014232  005015  000
3539  014235  124  051505  020124  E.HDR:  .ASCIZ  'TEST # '
3540  014242  020043  000
3541  014245  040  042504  044526  E.HDR1: .ASCII  ' DEVICE ERROR'<CR><LF>
3542  014252  042503  042440  051122
3543  014260  051117  005015
3544  014264  051503  004461  041527          .ASCIZ  'CS1'<HT>'WC'<HT>'BA'<HT>'FC'<HT>'CS2'<HT>'DS'<HT>'ER'<HT>'TC'<CR><LF>
3545  014272  041011  004501  041506
3546  014300  041411  031123  042011
3547  014306  004523  051105  052011
3548  014314  006503  000012
3549  014320  047440  052125  047440  E.HDR2: .ASCIZ  ' OUT OF RANGE ERROR'<CR><LF>
3550  014326  020106  040522  043516
3551  014334  020105  051105  047522
3552  014342  006522  000012
3553  014346  005015  044524  042515  E.TIMOV:        .ASCIZ  <CR><LF>'TIMER OVERFLOWED'<CR><LF>
```

```
3554  014354  020122  053117  051105
3555  014362  046106  053517  042105
3556  014370  005015     000
3557  014373     015  052012  046511  E.TIMEX:        .ASCIZ  <CR><LF>'TIME EXPIRED WAITING FOR RDY'<CR><LF>
3558  014406  020105  054105  044520
3559  014414  042522  020104  040527
3560  014422  052111  047111  020107
3561  014430  047506  020122  042122
3562  014436  006531  000012
3563  014444  043440  050101  021440  E.GAP:  .ASCIZ  ' GAP # '
3564  014452  000040
3565
3566                                  ;TIME DOCUMENT LINES
3567  014454  025052  025052  025052  L.HDR1: .ASCIZ  '*************************************************************************
3568  014462  025052  025052  025052
3569  014470  025052  025052  025052
3570  014476  025052  025052  025052
3571  014504  025052  025052  025052
3572  014512  025052  025052  025052
3573  014520  025052  025052  025052
3574  014526  025052  025052  025052
3575  014534  025052  025052  025052
3576  014542  025052  025052  025052
3577  014550  025052  025052  025052
3578  014556  025052  025052  025052
3579  014564  005015     000
3580  014567     052  052040  030115  L.HDR2: .ASCII  '* TM03 DRIVE FUNCTION TIMES- DRIVE # '
3581  014574  020063  051104  053111
3582  014602  020105  052506  041516
3583  014610  044524  047117  052040
3584  014616  046511  051505  020055
3585  014624  051104  053111  020105
3586  014632  020043
3587  014634  020060  046123  053101  L.DRV:  .ASCII  '0 SLAVE # '
3588  014642  020105  020043
3589  014646  020060     040          L.SLV:  .ASCII  '0  '
3590  014651     071  041440  040510  L.CHAN: .ASCIZ  '9 CHAN. SER # '
3591  014656  027116  051440  051105
3592  014664  021440  000040
3593  014670  006440  025012  005015  L.HDR3: .ASCII  ' '<CR><LF>'*'<CR><LF>
3594  014676  020052  052506  041516          .ASCIZ  '* FUNCTION'<HT><HT>'TIME(SPECIFICATION)'<HT>'TIME(ACTUAL)'<CR><LF>
3595  014704  044524  047117  004411
3596  014712  044524  042515  051450
3597  014720  042520  044503  044506
3598  014726  040103  044524  047117
3599  014734  004451  044524  042515
3600  014742  040450  052103  040525
3601  014750  024514  005015     000
3602
3603  014755     122  047101  042507  L.RNG:  .ASCIZ  'RANGE=<'
3604  014762  036075     000
3605  014765     101  052103  040525  L.ACT:  .ASCIZ  'ACTUAL='
3606  014772  036514     000
3607
3608                                  ;TEST DESCRIPTOR HEADERS
3609  014775     052  053440  044522  A.TOO1: .ASCIZ  '* WRITE FROM BOT'<HT>
```

```
3610  014772  042524  043040  047522
3611  015000  020115  047502  004524
3612  015006     000
3613  015007     052  053440  044522  A.T002: .ASCIZ  '* WRITE START'<HT><HT>
3614  015014  042524  051440  040524
3615  015022  052122  004411     000
3616  015027     052  053440  044522  A.T003: .ASCIZ  '* WRITE SHUTDOWN'<HT>
3617  015034  042524  051440  052510
3618  015042  042124  053517  004516
3619  015050     000
3620  015051     052  053440  044522  A.T004: .ASCIZ  '* WRITE SETTLEDOWN'<HT>
3621  015056  042524  051440  052105
3622  015064  046124  042105  053517
3623  015072  004516     000
3624  015075     052  051040  040505  A.T005: .ASCIZ  '* READ FROM BOT'<HT><HT>
3625  015102  020104  051106  046517
3626  015110  041040  052117  004411
3627  015116     000
3628  015117     052  051040  040505  A.T006: .ASCIZ  '* READ START'<HT><HT>
3629  015124  020104  052123  051101
3630  015132  004524  000011
3631  015136  020052  042522  042101  A.T007: .ASCIZ  '* READ SHUTDOWN'<HT><HT>
3632  015144  051440  052510  042124
3633  015152  053517  004516  000011
3634  015160  020052  042522  042101  A.T010: .ASCIZ  '* READ SETTLEDOWN'<HT>
3635  015166  051440  052105  046124
3636  015174  042105  053517  004516
3637  015202     000
3638  015203     052  051040  040505  A.T011: .ASCIZ  '* READ REV START'<HT>
3639  015210  020104  042522  020126
3640  015216  052123  051101  004524
3641  015224     000
3642  015225     052  051040  040505  A.T012: .ASCIZ  '* READ REV SHUTDOWN'<HT>
3643  015232  020104  042522  020126
3644  015240  044123  052125  047504
3645  015246  047127  000011
3646  015252  020052  042522  042101  A.T013: .ASCIZ  '* READ REV SETTLEDOWN'<HT>
3647  015260  051040  053105  051440
3648  015266  052105  046124  042105
3649  015274  053517  004516     000
3650  015301     052  052040  051125  A.T014: .ASCIZ  '* TURN AROUND DELAY F-R'<HT>
3651  015306  020116  051101  052517
3652  015314  042116  042040  046105
3653  015322  054501  043040  051055
3654  015330  000011
3655  015332  020052  052524  047122  A.T015: .ASCIZ  '* TURN AROUND DELAY R-F'<HT>
3656  015340  040440  047522  047125
3657  015346  020104  042504  040514
3658  015354  020131  026522  004506
3659  015362     000
3660  015363     052  043440  050101  A.T016: .ASCIZ  '* GAP SIZE-STOP HALF'<HT>
3661  015370  051440  055111  026505
3662  015376  052123  050117  044040
3663  015404  046101  004506     000
3664  015411     052  043440  050101  A.T017: .ASCIZ  '* GAP SIZE-START HALF'<HT>
3665  015416  051440  055111  026505
```

```
3666  015424  052123  051101  020124
3667  015432  040510  043114  000011
3668  015440  020052  040507  020120  A.TO20: .ASCIZ  '* GAP SIZE-INTERRECORD'<HT>
3669  015446  044523  042532  044455
3670  015454  052116  051105  042522
3671  015462  047503  042123  000011
3672  015470  020052  040507  020120  A.TO21: .ASCIZ  '* GAP CONSISTANCY'<HT>
3673  015476  047503  051516  051511
3674  015504  040524  041516  004531
3675  015512     000
3676  015513     052  042040  052101  A.TO22: .ASCIZ  '* DATA TIME-800BPI'<HT>
3677  015520  020101  044524  042515
3678  015526  034055  030060  050102
3679  015534  004511     000
3680  015537     052  042040  052101  A.TO23: .ASCIZ  '* DATA TIME-1600BPI'<HT>
3681  015544  020101  044524  042515
3682  015552  030455  030066  041060
3683  015560  044520  000011
3684  015564  020052  051105  051501  A.TO24: .ASCIZ  '* ERASE GAP TIME'<HT>
3685  015572  020105  040507  020120
3686  015600  044524  042515  000011
3687  015606  020052  051127  052111  A.TO25: .ASCIZ  '* WRITE FILE MARK'<HT>
3688  015614  020105  044506  042514
3689  015622  046440  051101  004513
3690  015630     000
3691  015631     052  052040  050101  A.TO26: .ASCIZ  '* TAPE SPEED-FWD'<HT>
3692  015636  020105  050123  042505
3693  015644  026504  053506  004504
3694  015652     000
3695  015653     052  052040  050101  A.TO27: .ASCIZ  '* TAPE SPEED-REV'<HT>
3696  015660  020105  050123  042505
3697  015666  026504  042522  004526
3698  015674     000
3699
3700  015675     015  057012  000107  L.CNTG: .ASCIZ  <CR><LF>'↑G'
3701  015702  005015  053523  036522  L.SWR:  .ASCIZ  <CR><LF>'SWR='
3702  015710     000
3703  015711     040  047040  053505  L.NEW:  .ASCIZ  '  NEW= '
3704  015716  020075     000
3705  015721     015  037412  005015  L.QUEST:        .ASCIZ  <CR><LF>'?'<CR><LF>
3706  015726     000
3707          015730                          .EVEN
3708          015730                  RDBUF=.
3709          015730                  WTBUF=.
3710  015730  000200                          .BLKW   128.
3711          000001                          .END
```

```
A       010714
ACCL  = 100000
ANGTAB  001412
AS    = 000016
ASFLG   001130
ATA   = 100000
ATIME   001012
ATIMTB  001014
A.T001  014765
A.T002  015007
A.T003  015027
A.T004  015051
A.T005  015075
A.T006  015117
A.T007  015136
A.T010  015160
A.T011  015203
A.T012  015225
A.T013  015252
A.T014  015301
A.T015  015332
A.T016  015363
A.T017  015411
A.T020  015440
A.T021  015470
A.T022  015513
A.T023  015537
A.T024  015564
A.T025  015606
A.T026  015631
A.T027  015653
A16   = 000400
A17   = 001000
BA    = 000004
BAI   = 000010
BEGIN   006750
BELL    001403
BKSLSH  001377
BOT   = 000002
BPI200= 000000
BPI556= 000400
BPI800= 001000
BPTVEC= 000014
CDM11 = 000320
CHKDRV  006250
CHKSLV  006560
CHNFLG  005716
CH7   = 010000
CLR   = 000040
CNTRLA= 000001
CNTRLC= 000003
CNTRLG= 000007
CNTRLO= 000017
CNTRLU= 000025
CNVDEC  002526
CNVOCT  002420
CNVTAO  003056
CNVTD   002540
CNVTO   002432
CR    = 000015
CRLF    001374
CSITM = 002000
CSI   = 000000
CS2   = 000010
DASH    001405
DB    = 000022
DCONST  002634
DELAY   004634
DELAYV  004664
DELTIM  001114
DIGTAB  001132
DIVIDE  004722
DLT   = 100000
DPR   = 000400
DRIVES  006104
DRVAVA  005012
DRVNUM  001004
DRVTBL  001154
DRY   = 000200
DRYCLR= 000010
DS    = 000012
DT    = 000026
DTE   = 010000
DVA   = 004000
DV0   = 000000
DV1   = 000001
DV2   = 000002
DV3   = 000003
DV4   = 000004
DV5   = 000005
DV6   = 000006
DV7   = 000007
ECHO    001401
EMTVEC= 000030
END     012570
EOT   = 002000
ER    = 000014
ERASE = 000024
ERFLG   001123
ERR   = 040000
ERRTRP  003542
ERRVEC= 000004
E.DRV   014104
E.GAP   014434
E.HDR   014235
E.HDR1  014245
E.HDR2  014320
E.NAVA  014114
E.NCON  014015
E.NDRV  014062
E.NSLV  014076
E.SFT   014211
E.TIME  014373
E.TIMO  014346
E.TRP4  013776
E.UNIT  014147
FC    = 000006
FCE   = 001000
FINISH  012466
FMT   = 000020
FPEVEC= 000244
FRMCNT= 177400
FWDSPC  005366
GAP     001120
GAPOK   004506
GAPTBL  001054
GO    = 000001
GTIMTB  001556
GTSWR   002020
HERE    012652
HLT   = 104400
HT    = 000011
IDB   = 000010
IE    = 000100
ILF   = 000001
ILR   = 000002
INBUF   001264
INCVAE= 000100
INIT    005620
IOTVEC= 000020
IR    = 000100
ITCNT   001121
I.DRV   013656
I.DRVS  013567
I.REG   013522
I.REM   013455
I.SLVS  013637
I.SPD   013723
LF    = 000012
LKS   = 177546
LKVEC = 000100
LPB   = 177516
LPS   = 177514
L.ACT   014755
L.CHAN  014641
L.CNTG  015675
L.DRV   014624
L.HDR1  014444
L.HDR2  014557
L.HDR3  014660
L.NEW   015711
L.QUES  015721
L.RNG   014745
L.SLV   014636
L.SWR   015702
MCPE  = 020000
MDPE  = 000400
MMVEC = 000250
MOL   = 010000
MR    = 000024
MXF   = 001000
M.EOP   013406
M.EOT   013756
M.HSWR  013425
M.NAM   013252
NAMPTR  001656
NED   = 010000
NEF   = 004000
NEM   = 004000
NOP   = 000000
NORM11= 001700
NSG   = 000400
OCTALO  001116
ODIGIT  001144
OPI   = 020000
OR    = 000200
OSC   = 000100
OUTBUF= 005620
OUTGAP  002744
OUTSPC  002650
PARVEC= 000114
PAT   = 000020
PEFLRC= 000200
PES   = 000040
PE1600= 002300
PFVEC = 000024
PGE   = 002000
PIP   = 020000
PIRQ  = 177772
PIRVEC= 000240
PLKCSR= 172540
PLKVEC= 000104
PRGFLG  001124
PSCNT   001127
PSEL  = 002000
PSW   = 177776
PUBLIS  003144
RDBUF = 015730
RDFWD = 000070
RDREV = 000076
RDSW    002126
RDY   = 000200
READ    005332
RESVEC= 000010
REVRD   005350
RHINIT  005070
RMR   = 000004
RSTRT   006726
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  012706
SLA   = 000001
SLAVES  006350
SLR   = 177774
SLVAVA  005040
SLVNUM  001005
SLVPTR  001006
SLVTBL  001164
SN    = 000030
SNPT    005470
SPACE   001410
SPACE2  001407
SPCFWD= 000030
SPCREV= 000032
SPR   = 002000
SSC   = 000100
STIMTB  001416
STKPTR= 000600
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   002156
TIB     002016
TIMER   004340
```

```
TIMERR  004350          TRTVEC= 000014          TST017  011224          UNS    = 040000          $CRLF   002122
TIMER0  004364          TSTNUM  001122          TST020  011334          UNTFND  001125         $ENDAD  012642
TIMER1  004314          TSTTBL  001736          TST021  011454          UPE    = 020000          $FILL   002111
TIMOK   004376          TST000  007166          TST022  011760          WAITRD  005126         $HT    = 000011
TIMON   004250          TST001  007432          TST023  012110          WAITTI  005130         $NULL   002110
TKB    = 177562         TST002  007516          TST024  012240          WC     = 000002         $SVPC  = 000040
TKISR   003420          TST003  007574          TST025  012360          WCE    = 040000         $TKFLG  002113
TKS    = 177560         TST004  007664          TST026  012714          WCHKF  = 000050         $TPB    002116
TKVEC  = 000060         TST005  007772          TST027  013042          WCHKR  = 000056         $TPFLG  002112
TMBASE  001010          TST006  010056          TYPDEC  002534          WFMK   = 000026         $TPS    002114
TMCMD   005450          TST007  010142          TYPE   = 000004         WFWD   = 000060         .      = 016330
TMCS1  = 172440         TST010  010242          TYPE1   002150          WRDCNT= 177600          .HLT    003544
TMK    = 000004         TST011  010362          TYPE2   002176          WRITE   005314          .INPUT  003272
TPB    = 177566         TST012  010460          TYPE3   002204          WRL    = 004000         .RESTO  002376
TPS    = 177564         TST013  010570          TYPE4   002210          WRT.BK  005406          .REWIN  005232
TPVEC  = 000064         TST014  010730          TYPFLG  001126          WTBUF  = 015730         .SAVE   002354
TRAPVE= 000034          TST015  011022          TYPOCT  002426          $CHARC  002120          .SCOPE  004010
TRE    = 040000         TST016  011130          UBREAK= 177770          $CNTRL  002121          .TYPE   002130

. ABS.  016330     000
```

ERRORS DETECTED: 0

DZTEEA,DZTEEA/SOL←DZTEAA.SML/ML,DZTEEA.P11
RUN-TIME: 4 6 .3 SECONDS
RUN-TIME RATIO: 147/11=13.1
CORE USED: 7K (13 PAGES)