TM03/TU45

DRIVE FUNCTION TIMER

CZTUSA0

AH-E500A-MC

FICHE 1 OF 1

JUL 1978

digital

MADE IN USA

```
                .REM    %
```

IDENTIFICATION

PRODUCT CODE: AC-E499A-MC

PRODUCT NAME: CZTUSA0 TM03/TU45 DRIVE FUNCTION TIMER

DATE CREATED: 25 MAY 1978

MAINTAINER: COMPUTER SPECIAL SYSTEMS

AUTHOR: CSS DIAGNOSTICS

TM03 DRIVE FUNCTION TIMER
TABLE OF CONTENTS

TABLE OF CONTENTS

ABSTRACT

CHAPTER 1 REQUIREMENTS

1.1 EQUIPMENT

1.2 MEMORY STORAGE

1.3 PRELIMINARY PROGRAMS

CHAPTER 2 LOADING AND STARTING PROCEDURE

2.1 ACT11 OPERATION

CHAPTER 3 SWITCH SETTINGS

CHAPTER 4 ERRORS

4.1 ERROR TYPEOUT FORMAT (HARDWARE)

4.2 ERROR TYPEOUT FORMAT (FUNCTION OUT OF RANGE)

CHAPTER 5 SUBROUTINE ABSTRACTS

CHAPTER 6 MISCELLANEOUS

6.1 STACK POINTER

6.2 EXECUTION TIME

CHAPTER 7 PROGRAM DESCRIPTION

7.1 FUNCTION TIME DOCUMENT

7.2 TEST SEQUENCE / RELATED ADJUSTMENTS / ASSOCIATED HARDWARE

7.3 SUBTEST DESCRIPTIONS

ABSTRACT

PROGRAM CZTUSAO MEASURES THE TIME REQUIRED AND GAP SIZES PRODUCED BY THE TM03/TU45 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/TU45 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)

CZTUOA CONTROL LOGIC TEST(PART 1)
CZTUPA0 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, TU45 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/TU45 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 TU45 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=<063000-059000>       ACTUAL=061000
* WRITE START             RANGE=<004500-003700>       ACTUAL=004100
* WRITE SHUTDOWN          RANGE=<003600-002600>       ACTUAL=003100
* WRITE SETTLEDOWN        RANGE=<007600-001400>       ACTUAL=4500
* READ FROM BOT           RANGE=<009200-005200>       ACTUAL=007200
* READ START              RANGE=<001700-000900>       ACTUAL=001300
* READ SHUTDOWN           RANGE=<001450-000350>       ACTUAL=000900
* READ SETTLEDOWN         RANGE=<007600-001400>       ACTUAL=004500
* READ REV START          RANGE=<001700-000900>       ACTUAL=001300
* READ REV SHUTDOWN       RANGE=<001900-001500>       ACTUAL=001700
* READ REV SETTLEDOWN     RANGE=<007600-001400>       ACTUAL=004500
* TURN AROUND DELAY F-R   RANGE=<008800-002800>       ACTUAL=005800
* TURN AROUND DELAY R-F   RANGE=<008800-002800>       ACTUAL=005800
* GAP SIZE-STOP HALF      RANGE=<007900-004500>       ACTUAL=006200
* GAP SIZE-START HALF     RANGE=<008550-005250>       ACTUAL=006900
* GAP SIZE-INTERRECORD    RANGE=<006150-004450>       ACTUAL=005300
* GAP CONSISANCY          RANGE=<007050-005350>       ACTUAL=006200
* DATA TIME-800BPI        RANGE=<012000-011000>       ACTUAL=011500
* DATA TIME-1600BPI       RANGE=<012500-011500>       ACTUAL=012000
* ERASE GAP TIME          RANGE=<049900-046900>       ACTUAL=048400
* WRITE FILE MARK         RANGE=<051200-049200>       ACTUAL=050200
```

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=<011300-010300>       ACTUAL=010800
*TAPE SPEED REV          RANGE=<011300-010300>       ACTUAL=010800
```

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	*M8928 ROM*M8903 ACCL CNTR
2. WRITE START	* ''	* '' * ''
3. WRITE SHUTDOWN	* ''	* '' * ''
4. WRITE SETTLEDOWN	* ''	*M8928 SETTLEDOWN ONE SHOT
5. READ FROM BOT	* ''	*M8928 ROM*M8903 ACCL CNTR
6. READ START	* ''	* '' * ''
7. READ SHUTDOWN	* ''	* '' * ''
10. READ SETTLEDOWN	* ''	*M8928 SETTLEDOWN ONE SHOT
11. READ REVERSE START	* ''	*M8928 ROM*M8903 ACCL CNTR
12. READ REVERSE SHUTDOWN	* ''	* '' * ''
13. READ REVERSE SETTLEDOWN	* ''	*M8928 SETTLEDOWN ONE SHOT
14. TURN AROUND F-R	* ''	*M8928 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	* "	*M8928 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

TM03 DRIVE FUNCTION TIMER PAGE 15

7.3 SUBTEST DESCRIPTIONS:

THE FIRST THIRTEEN (13) TESTS (T1 - T15) ARE CHECKS OF THE ROM CIRCUITS IN THE TU45 (M9811), THE ACCL COUNTER IN THE TM03 (M8903), AND THE SETTLEDOWN ONE SHOT (M8928).

T1. WRITE FROM BOT:

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

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

T2. WRITE START:

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

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

T3. WRITE SHUTDOWN:

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

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

T4. WRITE SETTLEDOWN:

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

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

T5. READ FROM BOT

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

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

T6. READ START

THIS TEST MEASURES THE SAME DELAY AS IN T2.

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

T7. READ SHUTDOWN:

THIS TEST MEASURES THE SAME DELAY AS IN T3.

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

T10. READ SETTLEDOWN:

THIS TEST MEASURES THE SAME DELAY AS IN T4.

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

T11. READ REVERSE START:

THIS TEST WILL MEASURE THE START DELAY IN THE REVERSE
DIRECTION.

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

T12. READ REVERSE SHUTDOWN

THIS TEST WILL MEASURE THE READ SHUTDOWN IN THE REVERSE
DIRECTION.

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

T13. READ REVERSE SETTLEDOWN:

THIS TEST WILL MEASURE THE READ SETTLEDOWN IN THE REVERSE
DIRECTION.

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

T14. TURN AROUND DELAY-FORWARD TO REVERSE

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

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

TM03 DRIVE FUNCTION TIMER PAGE 18

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 TU45 IN ORDER TO ALLOW FOR
PROPER ACCELERATION AND DECELLERATION OF TAPE ACCORDING TO
THE DESIRED INTEROCORD GAP (.6 INCHES). THIS TEST, HOWEVER,
WILL MEASURE THE PHYSICAL SIZE OF THE INTERCORD GAP THAT
EXISTS ON TAPE AS A RESULT OF THE START/STOP TIMES OF
THE CAPSTAN ITSELF. BECAUSE THE INTERRCORD GAP IS
CREATED BY TWO ACTIONS, THE START OF MOTION AND THE
STOP OF MOTION IT IS NECESSARY TO MAKE TWO SEPERATE
MEASUREMENTS. A THIRD MEASUREMENT, MADE ON THE FLY, OF
THE ENTIRE LENGTH OF THE GAP WILL ALSO BE MADE.

T16. GAP SIZE (STOP HALF)

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

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

T17. GAP SIZE (START HALF)

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

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

T20. GAP SIZE (INTERRECORD)

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

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

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

TM03 DRIVE FUNCTION TIMER PAGE 22

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.

```
%
                                .LIST   BIN,LOC,SEQ

                                .NLIST  MC
                                .NLIST  TOC
                                .LIST   ME
                                .ENABLE ABS,AMA
                                .MCALL  $CPVEC,$CPREG,$CATCH,$TYPE,.$ACT11,.$EOP,$CHAIN
                                .TITLE  CZTUSA0 TM03/TU45 DRIVE FUNCTION TIMER
                                .SBTTL  STARTING INSTRUCTIONS
                        ;LOADING AND STARTING PROCEDURE
                        ;       LOAD PROGRAM USING ABS LOADER
                        ;       LOAD ADDRESS 200
                        ;       SET SWITCH OPTIONS
                        ;       PRESS START

                        ;RESTART PROCEDURE
                        ;       LOAD ADDRESS 210
                        ;       SET SWITCH OPTIONS
                        ;       PRESS START

                        ;SWITCH REGISTER SWITCH ASSIGNMENTS
100000                          SW15=   100000          ;HALT ON ERROR
040000                          SW14=   040000          ;LOOP SUBTEST
020000                          SW13=   020000          ;INHIBIT ERROR TYPE OUT
```

```
1006            004000                          SW11=   004000          ;INHIBIT SUBTEST ITERATION
1007            002000                          SW10=   002000          ;INHIBIT PUBLISHING TIME SPECIFICATION
1008            001000                          SW09=   001000          ;RING BELL ON ERROR
1009            000400                          SW08=   000400          ;TYPE LINE ITEM AFTER EACH ITERATION
1010            000200                          SW07=   00200           ;NOT USED
1011            000100                          SW06=   000100          ;CONTINUOUS CYCLE
1012                                    ;       SW05-SW00               ;RUN TEST SELECTED
1013                                    ;       **NOTE: IF <SW15-SW00> = 177777 AT STARTUP USE SOFTWARE
1014                                    ;               SWITCH REGISTER.
1015
1016                                    ;CONSOLE COMMANDS
1017                                    ;       CONTROL C               ;RESTART PROGRAM (SAME AS START @ 200)
1018                                    ;       CONTROL G               ;SET NEW SOFTWARE SWITCH REGISTER
1019                                    ;       CONTROL U               ;DELETE LINE TYPED
1020                                    ;       RUBOUT (DELETE)         ;DELETE LAST CHAR TYPED
1021
1022                                    ;GENERAL REGISTER USAGE:
1023                                    ;       R0=ADDRESS OF 'FC' REGISTER (SET BY SCOPE)
1024                                    ;       R1=ADDRESS OF 'DS' REGISTER (SET BY SCOPE)
1025                                    ;       R2=RETURN PC FROM TIMER (SET BY EACH TEST)
1026                                    ;       R3=INDEX INDICATING PREVIOUS OSCILLATOR POLARITY (SET BY TIMER)
1027                                    ;       R4=CONTAINS 'TICK' COUNT WHEN TIMER IS RUNNING (SET BY TIMER)
1028                                    ;       R5=ADDRESS OF CS1 (SET BY SCOPE)
1029
1030                                            .SBTTL  MACRO DEFINITIONS
1031                                            .MACRO  SAVE
1032                                            JSR     PC,..SAVE               ;SAVE REGISTERS ON THE STACK
1033                                            .ENDM   SAVE
1034                                            .MACRO  RESTORE
1035                                            JSR     PC,..RESTORE            ;RESTORE REGISTERS FROM THE STACK
1036                                            .ENDM   RESTORE
1037                                            .MACRO  INPUT
1038                                            JSR     PC,..INPUT              ;GET USER INPUT
1039                                            .ENDM   INPUT
1040                                            .MACRO  REWIND
1041                                            JSR     PC,..REWIND             ;REWIND SLAVE
1042                                            BVS     99$                     ;BRANCH IF ERROR ON REWIND
1043                                            .ENDM   REWIND
1044                                            .MACRO  TIMEON
1045                                            JSR     PC,TIMON                ;TURN TIMER ON
1046                                            .ENDM
1047                                            .MACRO  TIMCHK
1048                                            JMP     TIMER(R3)               ;GO TO TIMER & RETURN VIA R2
1049                                            .ENDM
1050                                            .MACRO  SETGO
1051                                            INC     (R5)                    ;SET 'GO' BIT
1052                                            .ENDM
1053
1054                                            .SBTTL  REGISTER ASSIGNMENTS
1055                                    ;;DEFINITIONS AND REGISTER ASSIGNMENTS
1056                                    ;;GENERAL REGISTER ASSIGNMENTS
1057            000000                          R0=%0
1058            000001                          R1=%1
1059            000002                          R2=%2
1060            000003                          R3=%3
1061            000004                          R4=%4
```

```
                        R5=%5
                        SP=%6
                        PC=%7
                        R10=%0
                        R11=%1
                        R12=%2
                        R13=%3
                        R14=%4
                        R15=%5

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

                ;;VECTOR ADDRESSES
000004                  ERRVEC=4                ;;ADDRESS OF ERROR VECTOR
000010                  RESVEC=10               ;;ADDRESS OF RESERVED INST. TRAP VECTOR
000014                  TBITVEC=14              ;;ADDRESS OF 'T' BIT TRAP VECTOR
000014                  TRTVEC=14               ;;ADDRESS OF 'TRACE' TRAP VECTOR
000014                  BPTVEC=14               ;;ADDRESS OF 'BREAKPOINT' TRAP VECTOR
000020                  IOTVEC=20               ;;ADDRESS OF IOT TRAP VECTOR
000024                  PFVEC=24                ;;ADDRESS OF POWER FAIL TRAP VECTOR
000030                  EMTVEC=30               ;;ADDRESS OF EMT VECTOR
000034                  TRAPVEC=34              ;;ADDRESS OF TRAP VECTOR
000060                  TKVEC=  60              ;;ADDRESS OF TTY KEYBOARD INT. VECTOR
000064                  TPVEC=64                ;;ADDRESS OF TTY PRINTER INTERRUPT VECTOR
000114                  PARVEC= 114             ;;ADDRESS OF MA/MF PARITY ERROR VECTOR
000240                  PIRVEC=240              ;;ADDRESS OF PIRQ VECTOR
000244                  FPEVEC=244              ;;ADDRESS OF FLOATING POINT INT. VECTOR
000250                  MMVEC=250               ;;ADDRESS OF MEM MGMT ERROR TRAP VECTOR

                ;CLOCK ADDRESS AND VECTORS
172540                  PLKCSR= 172540          ;KW11-P
000104                  PLKVEC= 104
177546                  LKS=    177546          ;KW11-L
000100                  LKVEC=  100
177514                  LPS=    177514          ;LP11
177516                  LPB=    177516

                ;RH,TM03/TU45 REGISTERS
172440                  TMCS1=  172440

                ;TM03/TU45 INDEX VALUES
000000                  CS1=    00                      ;CONTROL STATUS #1
000002                  WC=     02
000004                  BA=     04                      ;BUS ADDRESS REGISTER
000006                  FC=     06                      ;FRAME COUNT
000010                  CS2=    10                      ;CONTROL STATUS #2
000012                  DS=     12                      ;DRIVE STATUS
000014                  ER=     14                      ;ERROR REG #1
```

```
 1118          000016                          AS=     16                      ;ATTENTION SUMMARY
 1119          000022                          DB=     22                      ;DATA BUFFER REG
 1120          000024                          MR=     24                      ;MAINTENANCE REG
 1121          000026                          DT=     26                      ;DRIVE TYPE REG
 1122          000030                          SN=     30                      ;SERIAL NUMBER REGISTER
 1123          000032                          TC=     32                      ;TAPE CONTROL REG
 1124
 1125                                  .SBTTL  TM03/TU45 REGISTER BITS
 1126                                ;RHCS1-CS1(R5)
 1127          000001                          GO=     1
 1128          000000                          NOP=    0
 1129          000002                          RWDOFF= 2
 1130          000006                          RWD=    6
 1131          000010                          DRYCLR= 10
 1132          000026                          WFMK=   26
 1133          000024                          ERASE=  24
 1134          000030                          SPCFWD= 30
 1135          000032                          SPCREV= 32
 1136          000050                          WCHKF=  50
 1137          000056                          WCHKR=  56
 1138          000060                          WFWD=   60
 1139          000070                          RDFWD=  70
 1140          000076                          RDREV=  76
 1141          000100                          IE=     100
 1142          000200                          RDY=    200
 1143          000400                          A16=    400
 1144          001000                          A17=    1000
 1145          002000                          PSEL=   2000
 1146          004000                          DVA=    4000
 1147          020000                          MCPE=   20000
 1148          040000                          TRE=    40000
 1149          100000                          SC=     100000
 1150                                ;RHCS2-CS2(R5)
 1151          000000                          DV0=    0
 1152          000001                          DV1=    1
 1153          000002                          DV2=    2
 1154          000003                          DV3=    3
 1155          000004                          DV4=    4
 1156          000005                          DV5=    5
 1157          000006                          DV6=    6
 1158          000007                          DV7=    7
 1159          000010                          BAI=    10
 1160          000020                          PAT=    20
 1161          000040                          CLR=    40
 1162          000100                          IR=     100
 1163          000200                          OR=     200
 1164          000400                          MDPE=   400
 1165          001000                          MXF=    1000
 1166          002000                          PGE=    2000
 1167          004000                          NEM=    4000
 1168          010000                          NED=    10000
 1169          020000                          UPE=    20000
 1170          040000                          WCE=    40000
 1171          100000                          DLT=    100000
 1172                                ;RHDS-DS(R5)
 1173          000001                          SLA=    1
```

```
1174            000002                          BOT=    2
1175            000004                          TMK=    4
1176            000010                          IDB=    10
1177            000020                          SDWN=   20
1178            000040                          PES=    40
1179            000100                          SSC=    100
1180            000200                          DRY=    200
1181            000400                          DPR=    400
1182            002000                          EOT=    2000
1183            004000                          WRL=    4000
1184            010000                          MOL=    10000
1185            020000                          PIP=    20000
1186            040000                          ERR=    40000
1187            100000                          ATA=    100000
1188                                    ;RHER-Ek(R5)
1189            000001                          ILF=    1
1190            000002                          ILR=    2
1191            000004                          RMR=    4
1192
1193            000020                          FMT=    20
1194            000100                          INCVAE= 100
1195            000200                          PEFLRC= 200
1196            000400                          NSG=    400
1197            001000                          FCE=    1000
1198            002000                          CSITM=  2000
1199            004000                          NEF=    4000
1200            010000                          DTE=    10000
1201            020000                          OPI=    20000
1202            040000                          UNS=    40000
1203
1204                                    ;RHMR-MR(R5)
1205            000100                          OSC=    100
1206
1207                                    ;RHDT-DT(R5)
1208            002000                          SPR=    2000
1209            010000                          CH7=    10000
1210            040000                          TAP=    40000
1211
1212                                    ;RHTC-TC(R5)
1213            001700                          NORM11= 1700
1214            000320                          CDM11=  320
1215            000000                          BPI200= 0
1216            000400                          BPI556= 000400
1217            001000                          BPI800= 001000
1218            002300                          PE1600= 002300
1219            100000                          ACCL=   100000
1220
1221
1222
1223                                    ;INSTRUCTION EQUATES
1224            104400                          HLT=    TRAP
1225            104000                          SCOPE=  EMT
1226            000004                          TYPE=   IOT
1227
1228                                    ;MISCELLANEOUS EQUATES
1229            005620                          OUTBUF=INIT             ;OUTPUT BUFFER STARTS AT BEG OF PROGRAM
```

```
1230          177400                    FRMCNT= -256.                  ;FRAME COUNT
1231          177600                    WRDCNT= -128.                  ;WORD COUNT
1232                                 ;ASCII EQUATES
1233          000001                    CNTRLA= 1                      ;ASCII CODE FOR CONTROL A ( A)
1234          000003                    CNTRLC= 3                      ;ASCII CODE FOR CONTROL C ( C)
1235          000007                    CNTRLG= 7                      ;ASCII CODE FOR CONTROL G ( G)
1236          000011                    HT=     11                     ;ASCII CODE FOR HORIZONTAL TAB
1237          000012                    LF=     12                     ;ASCII CODE FOR LINE FEED
1238          000015                    CR=     15                     ;ASCII CODE FOR CARRIAGE RETURN
1239          000017                    CNTRLO= 17                     ;ASCII CODE FOR CONTROL O ( O)
1240          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                   ;TU45 SLAVE UNDER TEST
001006      000000                      SLVPTR: .WORD   0                   ;POINTER TO SLAVE TABLE (SLVTBL) BELOW
001010      172440                      TMBASE: .WORD   TMCS1               ;BASE ADDRESS OF TM03/TU45 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                      OCTAL0: .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
```

```
 1297  001125     000                UNTFND: .BYTE   0               ;UNIT FOUND INDICATOR
 1298  001126     000                TYPFLG: .BYTE   0
 1299  001127     000                PSCNT:  .BYTE   0               ;CONTAINS PASS COUNT
 1300  001130     000                ASFLG:  .BYTE   0               ;1/0 = YES/NO.
 1301             001132                     .EVEN
 1302  001132  030460                DIGTAB: ''01
 1303  001134  031462                        ''23
 1304  001136  032464                        ''45
 1305  001140  033466                        ''67
 1306  001142  034470                        ''89
 1307  001144  000006                ODIGITS:.BLKB   6               ;RESERVE SPACE FOR CONVERTED DIGITS
 1308  001152     000                        .BYTE   0               ;TERMINATOR
 1309             001154                     .EVEN
 1310  001154  000010                DRVTBL: .BLKB   8.              ;A 0/-1 = DRIVE NOT TO BE/TO BE TESTED
 1311  001164  000100                SLVTBL: .BLKB   64.             ;A 0/-1 = SLAVE NOT TO BE/TO BE TESTED
 1312  001264  000110                INBUF:  .BLKB   72.             ;TELETYPE INPUT BUFFER
 1313  001374  005015     000        CRLF:   .ASCIZ  <CR><LF>        ;MISCELLANEOUS ASCII CHARACTERS
 1314  001377     134     000        BKSLSH: .ASCIZ  '\'
 1315  001401     060     000        ECHO:   .ASCIZ  '0'
 1316  001403     007     000        BELL:   .ASCIZ  <7>
 1317  001405     055     000        DASH:   .ASCIZ  '-'
 1318  001407     040                SPACE2: .ASCII  ' '
 1319  001410  000040                SPACE:  .ASCIZ  ' '
 1320  001412  004476     000        ANGTAB: .ASCIZ  '>'<HT>
 1321             001416                     .EVEN
```

```
                                        .SBTTL  TIME SPECIFICATION TABLE
                                ;THE BELOW TABLE CONTAINS THE SPECIFIED FUNCTION TIMES IN TENS OF
                                ;MICROCESONDS. NOTE THAT WHEN TIMES ARE TYPED THAT THEY ARE TYPED IN
                                ;MICROSECONDS (BY APPENDING A 0).
                                ;FORMAT IS
                                ;       .WORD   MAX,MIN         ;TIME IN MS     FUNCTION        TEST #

001416  000000  000000        STIMTBL:.WORD   0,0             ;SPARE
001422  014234  013414                .WORD   6300.,5900.     ;63.0-59.0      WRITE FROM BOT  TST001
001426  000702  000562                .WORD   00450.,00370.   ;4.5-3.7        WRITE START     TST002
001432  000550  000404                .WORD   00360.,00260.   ;3.6-2.6        WRITE SHUTDOWN  TST003
001436  001370  000214                .WORD   00760.,00140.   ;7.6-1.4        WRITE STLDOWN   TST004
001442  001630  001010                .WORD   00920.,00520.   ;9.2-5.2        READ FROM BOT   TST005
001446  000252  000132                .WORD   00170.,00090.   ;1.7-0.9        READ START      TST006
001452  000221  000043                .WORD   00145.,00035.   ;1.45-0.35      READ SHUTDOWN   TST007
001456  001370  000214                .WORD   00760.,00140.   ;7.6-1.4        READ SETTLEDOWN TST010
001462  000252  000132                .WORD   00170.,00090.   ;1.7-0.9        RD REV START    TST011
001466  000276  000226                .WORD   00190.,00150.   ;1.9-1.5        RD REV SHTDWN   TST012
001472  001370  000214                .WORD   00760.,00140.   ;7.6-1.4        RD REV STLDWN   TST013
001476  001560  000430                .WORD   00880.,00280.   ;8.8-2.8        TRN RND DLY F-R TST014
001502  001560  000430                .WORD   00880.,00280.   ;8.8-2.8        TRN RND DLY R-F TST015
001506  001426  000702                .WORD   00790.,00450.   ;7.9-4.5        GAP SIZE STOP   TST016
001512  001527  001015                .WORD   00855.,00525.   ;8.55-5.25      GAP SIZE STRT   TST017
001516  001147  000675                .WORD   00615.,00445.   ;6.15-4.45      GAP SIZE INTER  TST020
001522  001301  001027                .WORD   00705.,00535.   ;7.05-5.35      GAP CONSISANCY  TST021
001526  002260  002114                .WORD   01200.,01100.   ;12.0-11.0      DAT TIME 800BPI TST022
001532  002342  002176                .WORD   01250.,01150.   ;12.5-11.5      DAT TIME 1600PE TST023
001536  011576  011122                .WORD   04990.,04690.   ;49.9-46.9      ERASE           TST024
001542  012000  011470                .WORD   05120.,04920.   ;51.2-49.2      WRT FILE MARK   TST025
001546  002152  002006                .WORD   01130.,01030.   ;11.3-10.3      TAPE SPEED FWD  TST026
001552  002152  002006                .WORD   01130.,01030.   ;11.3-10.3      TAPE SPEED REV  TST027

                                ;NOTE: TEST 26 AND 27 REQUIRE PRERECORDED 800BPI SKEW TAPE.
```

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

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

001556  001274  001041       GTIMTBL:.WORD   00700.,00545.   ;7.00-5.45      GAP-0   0 MS
001562  001344  001200               .WORD   00740.,00640.   ;7.4-6.4        GAP-1   1.0 MS
001566  001370  001154               .WORD   00760.,00620.   ;7.6-6.2        GAP-2   2.0 MS
001572  001274  001060               .WORD   00700.,00560.   ;7.0-5.6        GAP-3   3.0 MS
001576  001306  000776               .WORD   00710.,00510.   ;7.1-5.1        GAP-4   4.0 MS
001602  001351  000707               .WORD   00745.,00455.   ;7.45-4.55       GAP-5   5.0 MS
001606  001351  000733               .WORD   00745.,00475.   ;7.45-4.75       GAP-6   6.0 MS
001612  001351  000733               .WORD   00745.,00475.   ;7.45-4.75       GAP-7   7.0 MS
001616  001274  000764               .WORD   00700.,00500.   ;7.0-5.0         GAP-10  8.0 MS
001622  001262  000764               .WORD   00690.,00500.   ;6.9-5.0         GAP-11  9.0 MS
001626  001262  000764               .WORD   00690.,00500.   ;6.9-5.0         GAP-12  10.0 MS
001632  001262  000764               .WORD   00690.,00500.   ;6.9-5.0         GAP-13  11.0 MS
001636  001262  000764               .WORD   00690.,00500.   ;6.9-5.0         GAP-14  12.0 MS
001642  001262  000764               .WORD   00690.,00500.   ;6.9-5.0         GAP-15  13.1 MS
001646  001262  000764               .WORD   00690.,00500.   ;6.9-5.0         GAP-16  14.1 MS
001652  001262  000764               .WORD   00690.,00500.   ;6.9-5.0         GAP-17  15.1 MS
```

```
1378                                    .SBTTL  TEST HEADER POINTERS
1379                                ;THE BELOW TABLE CONTAINS POINTERS TO EACH TEST'S DESCRIPTOR
1380  001656  000000                NAMPTR: .WORD   0
1381  001660  014765                        .WORD   A.T001
1382  001662  015007                        .WORD   A.T002
1383  001664  015027                        .WORD   A.T003
1384  001666  015051                        .WORD   A.T004
1385  001670  015075                        .WORD   A.T005
1386  001672  015117                        .WORD   A.T006
1387  001674  015136                        .WORD   A.T007
1388  001676  015160                        .WORD   A.T010
1389  001700  015203                        .WORD   A.T011
1390  001702  015225                        .WORD   A.T012
1391  001704  015252                        .WORD   A.T013
1392  001706  015301                        .WORD   A.T014
1393  001710  015332                        .WORD   A.T015
1394  001712  015363                        .WORD   A.T016
1395  001714  015411                        .WORD   A.T017
1396  001716  015440                        .WORD   A.T020
1397  001720  015470                        .WORD   A.T021
1398  001722  015513                        .WORD   A.T022
1399  001724  015537                        .WORD   A.T023
1400  001726  015564                        .WORD   A.T024
1401  001730  015606                        .WORD   A.T025
1402  001732  015631                        .WORD   A.T026
1403  001734  015653                        .WORD   A.T027
1404
1405                                ;TABLE OF TEST STARTING ADDRESSES
1406  001736  007166                TSTTBL: .WORD   TST000
1407  001740  007432                        .WORD   TST001
1408  001742  007516                        .WORD   TST002
1409  001744  007574                        .WORD   TST003
1410  001746  007664                        .WORD   TST004
1411  001750  007772                        .WORD   TST005
1412  001752  010056                        .WORD   TST006
1413  001754  010142                        .WORD   TST007
1414  001756  010242                        .WORD   TST010
1415  001760  010362                        .WORD   TST011
1416  001762  010460                        .WORD   TST012
1417  001764  010570                        .WORD   TST013
1418  001766  010730                        .WORD   TST014
1419  001770  011022                        .WORD   TST015
1420  001772  011130                        .WORD   TST016
1421  001774  011224                        .WORD   TST017
1422  001776  011334                        .WORD   TST020
1423  002000  011454                        .WORD   TST021
1424  002002  011760                        .WORD   TST022
1425  002004  012110                        .WORD   TST023
1426  002006  012240                        .WORD   TST024
1427  002010  012360                        .WORD   TST025
1428  002012  012714                        .WORD   TST026
1429  002014  013042                        .WORD   TST027
```

```
                                TIB:    .WORD   0
                                ;ROUTINE TO LOAD SSOFTWARE SWR

002020  022737  000176  001000  GTSWR:  CMP     #SWREG,SWR      ;BRANCH IF SOFTWARE SWR
002026  001027                  BNE     2$              ;NOT INVOKED
002030  004737  002354                  JSR     PC,..SAVE       ;SAVE REGISTERS ON THE STACK
002034  000004  015702                  TYPE,L.SWR
002040  017702  176734                  MOV     @SWR,R2
002044  004737  002426                  JSR     PC,TYPOCT
002050  000004  015711                  TYPE,L.NEW
002054  004737  003272                  JSR     PC,..INPUT              ;GET USER INPUT
002060  122737  000015  001264          CMPB    #CR,@#INBUF             ;EXIT IF FIRST CHAR IS <CR>
002066  001405                  BEQ     1$
002070  004737  003056                  JSR     PC,CNVTAO               ;CONERT ASCII TO OCTAL
002074  013777  001116  176676          MOV     @#OCTALO,@SWR           ;SET NEW SWITCH REG CONTENTS
002102  004737  002376          1$:     JSR     PC,..RESTORE
002106  000207                  2$:     RTS     PC

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

                                ;;TAGS USED BY THE TYPE ROUTINE BELOW
000011                                  $HT=11                  ;;HORIZONTAL TAB
002110     000          $NULL:  .BYTE   0               ;;CONTAINS NULL CHARACTER
002111     002          $FILL:  .BYTE   2               ;;CONTAINS # OF FILLER CHARACTERS
002112     000          $TPFLG: .BYTE   0               ;;CONTAINS TELEPRINTER AVAILABLE FLAG
                                                        ;;0/377 = AVAIL/NOT AVAIL
002113     000          $TKFLG: .BYTE   0               ;;CONTAINS KEYBOARD AVAILABLE FLAG
002114  177564          $TPS:   .WORD   177564          ;;ADDRESS OF TLEPRINTER STATUS REGISTER
002116  177566          $TPB:   .WORD   177566          ;;ADDRESS OF TELEPRINTER DATA BUFFER
002120     000          $CHARCNT:.BYTE  0               ;;CONTAINS # OF CHARS TYPED
002121     000          $CNTRLO:.BYTE   0               ;;CONTAINS CONTROL O CHAR (IF TYPED)
002122  005015     000  $CRLF:  .ASCIZ  <15><12>
002126                          .EVEN
002126  000000          RDSW:   .WORD   0

002130  010046          .TYPE:  MOV     R0,-(SP)        ;;SAVE R0
002132  017600  000002          MOV     @2(SP),R0       ;;GET MESSAGE ADDRESS
002136  062766  000002  000002  ADD     #2,2(SP)        ;;ADJUST RETURN PC
002144  105037  002121          CLRB    $CNTRLO

002150  105737  002121  TYPE1:  TSTB    $CNTRLO         ;;BRANCH IF CONTROL O( O) WASN'T TYPED
002154  001410                  BEQ     TYPE2
002156  000004  002122  TCRLF:  TYPE,$CRLF               ;;TYPE <CR><LF>
002162  105737  002126          TSTB    RDSW
002166  100006                  BPL     TYPE3
002170  005037  002126          CLR     RDSW
002174  000207                  RTS     PC
002176  112046  TYPE2:  MOVB    (R0)+,-(SP)     ;;PUSH CHARACTER TO BE TYPED ONTO STACK
002200  001003                  BNE     TYPE4           ;;BRANCH IF NOT THE TERMINATOR
002202  005726                  TST     (SP)+           ;;POP TERMINATOR CHAR OFF THE STACK
```

```
1486  002204  012600                TYPE3:  MOV     (SP)+,R0        ;;RESTORE R0
1487  002206  000002                        RTI                     ;;RETURN TO CALLER
1488
1489  002210  122716  000011        TYPE4:  CMPB    #$HT,(SP)       ;;BRANCH IF HORIZONTAL TAB <HT>
1490  002214  001445                        BEQ     9$
1491  002216  004737  002250                JSR     PC,5$           ;;TYPE CHARACTER
1492  002222  122726  000012        3$:     CMPB    #12,(SP)+       ;;CHECK IF CHARACTER WAS A LINE FEED
1493  002226  001350                        BNE     TYPE1           ;;BRANCH IF NOT LINE FEED
1494  002230  013746  002110                MOV     $NULL,-(SP)     ;;GET # OF FILLERS REQUIRED AND FILLER
1495                                                                ;;CHARACTER.
1496
1497  002234  105366  000001        4$:     DECB    1(SP)           ;;DECREMENT FILLERS REQ. COUNT
1498  002240  002770                        BLT     3$              ;;BRANCH IF NO MORE FILLERS ARE REQUIRED
1499  002242  004737  002250                JSR     PC,5$           ;;TYPE FILLER CHARACTER
1500  002246  000772                        BR      4$
1501
1502  002250  105777  177640        5$:     TSTB    @$TPS           ;;WAIT FOR OUTPUT DEVICE
1503  002254  100375                        BPL     .-4
1504  002256  122737  000017  002121        CMPB    #17,@#$CNTRLO   ;;CHECK IF CONTROL O WAS TYPED
1505  002264  001403                        BEQ     6$              ;;STOP TYPING MESSAGE IF  O WAS TYPED
1506  002266  116677  000002  177622        MOVB    2(SP),@$TPB     ;;OUTPUT CHARACTER
1507  002274  122766  000015  000002 6$:     CMPB    #15,2(SP)       ;;BRANCH IF NOT <CR>
1508  002302  001003                        BNE     7$
1509  002304  105037  002120                CLRB    $CHARCNT        ;;CLEAR CHARACTERS TYPED COUNT
1510  002310  000406                        BR      8$
1511  002312  122766  000012  000002 7$:     CMPB    #12,2(SP)       ;;BRANCH IF <LF> OR 'NULL'
1512  002320  002002                        BGE     8$
1513  002322  105237  002120                INCB    $CHARCNT        ;;INCREMENT CHARACTER TYPED COUNT
1514  002326  000207                8$:     RTS     PC
1515
1516                                ;;HORIZONTAL TAB <HT> PROCESSER
1517  002330  112716  000040        9$:     MOVB    #40,(SP)        ;;LOAD 'SPACE'
1518  002334  004737  002250        10$:    JSR     PC,5$           ;;TYPE 'SPACE'
1519  002340  132737  000007  002120        BITB    #7,$CHARCNT     ;;TYPE SPACES UNTIL A MULTIPLE
1520  002346  001372                        BNE     10$             ;;OF 8 CHARACTERS HAVE BEEN TYPED
1521  002350  105726                        TSTB    (SP)+           ;;POP SPACE
1522  002352  000676                        BR      TYPE1           ;;GET NEXT CHARACTER
1523
1524                                ;SUBROUTINE TO SAVE GENERAL REGISTERS ON THE STACK
1525                                ;CALL:  SAVE
1526  002354  010546                .SAVE:  MOV     R5,-(SP)                ;SAVE REGISTERS ON THE STACK
1527  002356  010446                        MOV     R4,-(SP)
1528  002360  010346                        MOV     R3,-(SP)
1529  002362  010246                        MOV     R2,-(SP)
1530  002364  010146                        MOV     R1,-(SP)
1531  002366  010046                        MOV     R0,-(SP)
1532  002370  016646  000014                MOV     14(SP),-(SP)            ;GET RETURN PC
1533  002374  000207                        RTS     PC                      ;RETURN
1534
1535                                ;SUBROUTINE TO RESTORE GENERAL REGISTERS FROM THE STACK
1536                                ;CALL:  RESTORE
1537  002376  012666  000014        .RESTORE:MOV    (SP)+,14(SP)            ;MOVE RETURN PC
1538  002402  012600                        MOV     (SP)+,R0                ;RESTORE REGISTERS
1539  002404  012601                        MOV     (SP)+,R1
1540  002406  012602                        MOV     (SP)+,R2
1541  002410  012603                        MOV     (SP)+,R3
```

```
                                        MOV     (SP)+,R4
                                        MOV     (SP)+,R5
                                        RTS     PC                      ;RETURN

                                ;SUBROUTINE TO CONVERT OCTAL DATA TO ASCII
                                ;CALL:  MOV     NUMBER,R2               ;MOVE NUMBER TO R2
                                ;       JSR     PC,CNVOCT
                                
002420  110637  001126          CNVOCT: MOVB    SP,TYPFLG               ;SET DO NOT TYPE FLAG
002424  000402                          BR      CNVTO
                                
                                        .SBTTL          OCTAL TO ASCII & TYPE ROUTINE
                                ;SUBROUTINE TO CONVERT OCTAL NUMBER TO ASCII AND TYPE IT OUT
                                ;CALL:  MOV     NUMBER,R2               ;PUT # IN R2
                                ;       JSR     PC,TYPOCT               ;CALL ROUTINE
                                
002426  105037  001126          TYPOCT: CLRB    @#TYPFLG                ;SET TYPE FLAG
002432                          CNVTO:
002432  004737  002354                  JSR     PC,..SAVE               ;SAVE REGISTERS ON THE STACK
002436  012704  001144                  MOV     #ODIGITS,R4             ;SET PTR TO OUTPUT
002442  005003                          CLR     R3                      ;R3 WILL CONTAIN OCTAL DIGIT
002444  010201                          MOV     R2,R1                   ;GET # TO BE TYPED
002446  006302                  1$:     ASL     R2                      ;SHIFT #
002450  006103                          ROL     R3
002452  012700  000006                  MOV     #6,R0                   ;SET DIGIT COUNTER
002456  000404                          BR      3$
                                
002460  006302                  2$:     ASL     R2                      ;SHIFT # 3 PLACES LEFT
002462  006103                          ROL     R3
002464  005301                          DEC     R1
002466  001374                          BNE     2$
002470  012701  000003          3$:     MOV     #3,R1                   ;SET SHIFT COUNTER
002474  116324  001132                  MOVB    DIGTAB(R3),(R4)+        ;MOVE ASCII EQUIV TO OUTPUT
002500  005003                          CLR     R3
002502  005300                          DEC     R0                      ;DECREMENT DIGIT COUNT
002504  001365                          BNE     2$                      ;GET NEXT DIGIT
002506  105737  001126                  TSTB    @#TYPFLG                ;BRANCH IF ASCII IS
002512  001002                          BNE     4$                      ;NOT TO BE TYPED
002514  000004  001144                  TYPE,ODIGITS
002520                          4$:
002520  004737  002376                  JSR     PC,..RESTORE            ;RESTORE REGISTERS FROM THE STACK
002524  000207                          RTS     PC
                                
                                ;SUBROUTINE TO CONVERT OCTAL DATA TO DECIMAL ASCII
                                ;CALL:  MOV     NUMBER,R2               ;MOVE NUMBER TO R2
                                ;       JSR     PC,CNVDEC
                                
002526  110637  001126          CNVDEC: MOVB    SP,@#TYPFLG             ;SET DO NOT TYPE FLAG
002532  000402                          BR      CNVTD
                                        .SBTTL          OCTAL TO DECIMAL & TYPE ROUTINE
                                ;THIS ROUTINE CONVERTS AN OCTAL # TO DECIMAL ASCII AND TYPES IT OUT
                                ;CALL:  MOV     NUMBER,R2               ;PUT # IN R2
                                ;       JSR     PC,TYPDEC               ;CALL ROUTINE
                                
002534  105037  001126          TYPDEC: CLRB    @#TYPFLG                ;SET TYPE FLAG
```

```
1598  002540                          CNVTD:
1599  002540  004737  002354                JSR     PC,..SAVE               ;SAVE REGISTERS ON THE STACK
1600  002544  005000                          CLR     R0                      ;R0 IS INDEX TO DECIMAL CONSTANT
1601  002546  012704  001144                MOV     #ODIGITS,R4             ;SET OUTPUT PTR
1602  002552  005003                1$:       CLR     R3                      ;R3 CONTAINS DECIMAL DIGIT
1603  002554  166002  002634        2$:       SUB     DCONST(R0),R2           ;SUBTRACT DECIMAL CONSTANT UNTIL
1604  002560  103402                          BLO     3$                      ;INPUT # GOES NEGATIVE
1605  002562  005203                          INC     R3                      ;KEEPING TRACK OF SUBTRACTIONS
1606  002564  000773                          BR      2$
1607  002566  066002  002634        3$:       ADD     DCONST(R0),R2           ;ADD BACK CONSTANT WHEN NEGATIVE
1608  002572  116324  001132                MOVB    DIGTAB(R3),(R4)+        ;MOVE ASCII EQUIVILENT
1609  002576  062700  000002                ADD     #2,R0                   ;NEXT CONSTANT
1610  002602  005760  002634                TST     DCONST(R0)              ;UNTIL ALL CONSTANTS DONE
1611  002606  001361                          BNE     1$
1612  002610  112724  000060                MOVB    #'0,(R4)+               ;LAST DIGIT IS 0
1613  002614  105737  001126                TSTB    @#TYPFLG                ;BRANCH IF ASCII IS
1614  002620  001002                          BNE     4$                      ;NOT TO BE TYPED
1615  002622  000004  001144                TYPE,ODIGITS
1616  002626                          4$:
1617  002626  004737  002376                JSR     PC,..RESTORE            ;RESTORE REGISTERS FROM THE STACK
1618  002632  000207                          RTS     PC
1619
1620  002634  023420                DCONST:   .WORD   10000.
1621  002636  001750                          .WORD   1000.
1622  002640  000144                          .WORD   100.
1623  002642  000012                          .WORD   10.
1624  002644  000001                          .WORD   1.
1625  002646  000000                          .WORD   0                       ;TERMINATOR
1626
1627                                          .SBTTL          TYPE SPECIFIED TIMES ROUTINE
1628                                  ;THIS SUBROUTINE OUTPUTS THE TIME SPECIFICATIONS FOR THE TEST
1629                                  ;AND ALSO THE ACTUAL TIME RECORDED (ATIME)
1630                                  ;FORMAT OF LINE TYPED
1631                                  ;RANGE=<AAAAAA-BBBBBB>          ACTUAL=CCCCCC
1632                                  ;WHERE:         AAAAAA IS MAXIMUM TIME FOR TEST (STIMTBL(TSTNUMX4)).
1633                                  ;               BBBBBB IS MINIMUM TIME FOR TEST (STIMTBL(TSTNUMX4+2)).
1634                                  ;               CCCCCC IS ACTUAL TIME RECORDED BY TEST (ATIME).
1635                                  ;CALL:  MOVB    TEST NUMBER,R2  ;LOAD TEST NUMBER
1636                                  ;       MOV     #TIME,@#ATIME           ;MOVE TIME TO ATIME
1637                                  ;       JSR     PC,OUTSPC
1638  002650  010246                OUTSPC:   MOV     R2,-(SP)                ;SAVE R2 & R3 ON THE STACK
1639  002652  010346                          MOV     R3,-(SP)
1640  002654  006302                          ASL     R2                      ;MULTIPLY TEST # TIMES 4
1641  002656  006302                          ASL     R2                      ;TO FORM INDEX INTO STIMTBL
1642  002660  010203                          MOV     R2,R3                   ;R3 CONTAINS INDEX INTO TABLE
1643  002662  000004  014745                TYPE,L.RNG
1644  002666  016302  001416                MOV     STIMTBL(R3),R2          ;GET MAXIMUM SPEC TIME
1645  002672  004737  002534                JSR     PC,TYPDEC               ;CONVERT TO DECIMAL & TYPE
1646  002676  000004  001405                TYPE,DASH
1647  002702  016302  001420                MOV     STIMTBL+2(R3),R2        ;GET MINIMUM TIME
1648  002706  004737  002534                JSR     PC,TYPDEC               ;CONVERT TO DECIMAL & TYPE
1649  002712  000004  001412                TYPE,ANGTAB
1650  002716  000004  014755                TYPE,L.ACT
1651  002722  013702  001012                MOV     @#ATIME,R2              ;GET ACTUAL TIME
1652  002726  004737  002534                JSR     PC,TYPDEC               ;CONVERT TO DECIMAL & TYPE
1653  002732  000004  001374                TYPE,CRLF
```

```
                        MOV     (SP)+,R3
                        MOV     (SP)+,R2
                        RTS     PC                      ;RETURN

                        .SBTTL          TYPE GAP TIMES SUBROUTINE
                ;THIS SUBROUTINE IS USED TO TYPE THE SPECIFIED GAP SIZES (RECORDED IN
                ;TST021). IT IS CALLED BY THE GAPOK ROUTINE IF THE GAP SIZE IS OUT OF
                ;RANGE VIA THE HLT ROUTINE (HLT+2).
                ;CALL:  MOVB    #GAP,GAP                ;LOAD GAP # INTO GAP
                ;       MOV     #TIME,ATIME             ;LOAD ACTUAL TIME INTO ATIME
                ;       JSR     PC,OUTGAP

                OUTGAP: MOV     R2,-(SP)                ;SAVE R2 AND R3
                        MOV     R3,-(SP)
                        MOVB    GAP,R3                  ;GET GAP #
                        ASL     R3
                        ASL     R3
                        TYPE,L.RNG
                        MOV     GTIMTBL(R3),R2          ;GET MAX TIME
                        JSR     PC,TYPDEC               ;CONVERT TO DECIMAL & TYPE
                        TYPE,DASH
                        MOV     GTIMTBL+2(R3),R2        ;GET MIN TIME
                        JSR     PC,TYPDEC               ;CONVERT TO DECIMAL & TYPE
                        TYPE,ANGTAB                     ;TYPE <
                        TYPE,L.ACT
                        MOV     @#ATIME,R2              ;GET ACTUAL TIME
                        JSR     PC,TYPDEC               ;CONVERT TO DECIMAL & TYPE
                        TYPE,E.GAP
                        MOVB    @#GAP,R2                ;GET GAP #
                        JSR     PC,TYPOCT               ;TYPE GAP #
                        TYPE,CRLF
                        MOV     (SP)+,R3                ;RESTORE R3 AND R2
                        MOV     (SP)+,R2
                        RTS     PC

                        .SBTTL          ASCII TO OCTAL CONVERT SUBROUTINE
                ;SUBROUTINE TO CONVERT ASCII DATA TO OCTAL. CONVERTED OCTAL DATA
                ;IS LEFT IN OCTAL0 <15-00>.
                CNVTAO:
                        JSR     PC,.SAVE                ;SAVE REGISTERS ON THE STACK
                        MOV     #INBUF,R0               ;SET PTR TO ASCII DATA
                        MOV     #OCTAL0,R1              ;GET ADDRESS OF OCTAL DATA
                        CLR     (R1)                    ;CLEAR OUT OLD OCTAL DATA
                        CLR     2(R1)
                1$:     CMPB    #CR,(R0)                ;<CR> TERMINATES INPUT
                        BEQ     3$
                        MOVB    (R0)+,R2                ;GET 'OCTAL' DATA
                        BIC     #177770,R2              ;STRIP UNUSED BITS
                        MOV     #3,R3                   ;SET SHIFT COUNT
                2$:     ASL     (R1)                    ;SHIFT LAST
                        ROL     2(R1)                   ;OCTAL DIGIT
                        DEC     R3
                        BNE     2$
                        BIS     R2,(R1)                 ;AND INSERT THIS DIGIT
                        BR      1$                      ;GO GET NEXT DIGIT
                3$:
```

```
1710  003136  004737  002376          JSR     PC,..RESTORE            ;RESTORE REGISTERS FROM THE STACK
1711  003142  000207                  RTS     PC                      ;RETURN
1712
1713                                  .SBTTL          PUBLISH SUBROUTINE
1714                                ;THE PUBLISH SUBROUTINE AVERAGES THE RECORDED TIMES FOR EACH TEST IT-
1715                                ;ERATION (IF 16. ITERATIONS) AND PLACES THE AVERAGE RESULT IN 'ATIME'.
1716                                ;IT TYPES THE NAME OF THE FUNCTION THAT WAS TIMED,THE TIME SPEC-
1717                                ;IFICATION AND THE ACTUAL TIME .
1718
1719  003144                        PUBLISH:
1720  003144  004737  002354          JSR     PC,..SAVE               ;SAVE REGISTERS ON THE STACK
1721  003150  012700  001014          MOV     #ATIMTBL,R0     ;GET TABLE ADDRESS CONTAINING TIMES
1722  003154  113701  001121          MOVB    @#ITCNT,R1      ;GET # OF ENTRIES (GIVEN BY ITERATION COUNT)
1723  003160  122701  000001          CMPB    #1,R1                   ;BRANCH IF SINGLE ITERATION
1724  003164  001423                  BEQ     4$
1725  003166  005002                  CLR     R2                      ;CLEAR 'SUM' REGISTERS
1726  003170  005003                  CLR     R3
1727  003172  122701  000020          CMPB    #16.,R1                 ;BRANCH IF 16. ITERATIONS
1728  003176  001402                  BEQ     1$
1729  003200  000000                  HALT                            ;ITERATION COUNT MUST BE 1 OR 16.
1730  003202  000777                  BR      .                       ;DO NOT CHANGE POSIT OF SW11
1731                                                                  ;WHEN TEST IS RUNNING.
1732
1733  003204  062002          1$:     ADD     (R0)+,R2                ;SUM INDIVIDUAL TIMES
1734  003206  005503                  ADC     R3
1735  003210  005301                  DEC     R1
1736  003212  001374                  BNE     1$
1737
1738  003214  012700  000004  2$:     MOV     #4,R0
1739  003220  006203          3$:     ASR     R3                      ;SHIFT TIME IN R3 & R2 4 PLACES
1740  003222  006002                  ROR     R2                      ;RIGHT = DIVIDE BY 16.
1741  003224  005300                  DEC     R0
1742  003226  001374                  BNE     3$
1743  003230  010237  001012          MOV     R2,@#ATIME              ;MOVE AVERAGED TIMES
1744
1745  003234  113700  001122  4$:     MOVB    @#TSTNUM,R0             ;GET TEST #
1746  003240  006300                  ASL     R0
1747  003242  016037  001656  003252  MOV     NAMPTR(R0),5$           ;GET TEST NAME STRING ADDRESS
1748  003250  000004                  TYPE
1749  003252  000000          5$:     .WORD   0
1750  003254  113702  001122          MOVB    @#TSTNUM,R2             ;GET TEST #
1751  003260  004737  002650          JSR     PC,OUTSPC               ;OUTPUT TIMES
1752  003264  004737  002376          JSR     PC,..RESTORE            ;RESTORE REGISTERS FROM THE STACK
1753  003270  000207                  RTS     PC
1754
1755                                  .SBTTL          INPUT SUBROUTINE
1756                                ;SUBROUTINE TO GET TTY INPUT
1757                                ;CALL:  JSR     PC,..INPUT
1758                                ;INPUT DATA IS RETUNED IN BUFFER BEGINNING AT INBUF.
1759
1760  003272  010046          .INPUT: MOV     R0,-(SP)        ;SAVE R0 ON THE STACK
1761  003274  012700  001264  1$:     MOV     #INBUF,R0
1762  003300  105737  177560  2$:     TSTB    @#TKS
1763  003304  100375                  BPL     2$
1764
1765  003306  113746  177562          MOVB    @#TKB,-(SP)     ;GET CHARACTER
```

```
1766  003312  042716  000200              BIC     #200,(SP)
1767  003316  122716  000177              CMPB    #177,(SP)       ;CHECK RUBOUT
1768  003322  001004                      BNE     3$
1769  003324  124026                      CMPB    -(R0),(SP)+     ;REMOVE CHARACTER FROM INPUT
1770  003326  000004  001377              TYPE,BKSLSH
1771  003332  000762                      BR      2$              ;WAIT FOR NEXT CHARACTER
1772  003334  122716  000025      3$:     CMPB    #CNTRLU,(SP)    ;CHECK CONTROL U ( U)
1773  003340  001004                      BNE     4$
1774  003342  005726                      TST     (SP)+
1775  003344  000004  001374              TYPE,CRLF
1776  003350  000751                      BR      1$
1777  003352  122716  000003      4$:     CMPB    #CNTRLC,(SP)    ;BRANCH IF NOT CONTROL C
1778  003356  001003                      BNE     40$
1779  003360  000005                      RESET                   ;RESET I/O
1780  003362  000137  005620              JMP     @#INIT          ;RESTART PROGRAM
1781  003366  111637  001401      40$:    MOVB    (SP),@#ECHO
1782  003372  111620                      MOVB    (SP),(R0)+
1783  003374  122726  000015              CMPB    #CR,(SP)+
1784  003400  001403                      BEQ     5$
1785  003402  000004  001401              TYPE,ECHO
1786  003406  000734                      BR      2$
1787  003410  000004  001374      5$:     TYPE,CRLF
1788  003414  012600                      MOV     (SP)+,R0
1789  003416  000207                      RTS     PC
1790
1791                              ;KEYBOARD INTERRUPT SERVICE ROUTINE
1792  003420  113746  177562      TKISR:  MOVB    @#TKB,-(SP)     ;GET TYPED CHARACTER
1793  003424  042716  000200              BIC     #200,(SP)       ;STRIP PARITY BIT
1794  003430  122716  000017              CMPB    #CNTRLO,(SP)    ;BRANCH IF NOT CONTROL O ( O)
1795  003434  001002                      BNE     1$
1796  003436  111637  002121              MOVB    (SP),$CNTRLO    ;SET CONTROL O INDICATOR IN TYPE ROUTINE
1797
1798  003442  122716  000003      1$:     CMPB    #3,(SP)         ;BRANCH IF NOT CONTROL C ( C)
1799  003446  001007                      BNE     2$
1800  003450  023727  000042  012642      CMP     @#42,#$ENDAD    ;INHIBIT  C IF ACT11 QV OR AA
1801  003456  001403                      BEQ     2$
1802  003460  000005                      RESET
1803  003462  000137  005620              JMP     @#INIT          ;RESTART PROGRAM
1804
1805  003466  122716  000001      2$:     CMPB    #CNTRLA,(SP)    ;BRANCH IF NOT  A
1806  003472  001011                      BNE     3$
1807  003474  022737  000176  001000      CMP     #SWREG,SWR      ;BRANCH IF HARDWARE SWR IS INVOKED
1808  003502  001010                      BNE     4$
1809  003504  012737  177570  001000      MOV     #177570,SWR     ;INVOKE HARDWARE SWR
1810  003512  000004  013425              TYPE,M.HSWR
1811  003516  122716  000007      3$:     CMPB    #CNTRLG,(SP)    ;BRANCH IF NOT  G
1812  003522  001005                      BNE     5$
1813  003524  012737  000176  001000  4$: MOV     #SWREG,SWR      ;INVOKE SOFTWARE SWR
1814  003532  004737  002020              JSR     PC,GTSWR        ;GET NEW SWITCH REGISTER
1815  003536  005726              5$:     TST     (SP)+           ;POP CHARACTER OFF THE STACK
1816  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?
```

```
 1873  003776  100001                        BPL     6$
 1874  004000  000000                        HALT
 1875  004002                          6$:
 1876  004002  004737  002376                JSR     PC,.RESTORE             ;RESTORE REGISTERS FROM THE STACK
 1877  004006  000002                        RTI                             ;RETURN
 1878
 1879
```

```
                                        .SBTTL          SCOPE SUBROUTINE
                                ;SCOPE ROUTINE
                                ;THIS ROUTINE IS ENTERED UPON COMPLETION OF EACH SUBTEST
                                ;THE SCOPE ROUTINE:
                                ;       OUTPUTS TIME SPEC ON EACH ITERATION IF SW08 IS SET
                                ;       REPEATS TEST IF SW14 IS SET
                                ;       STORES ACTUAL TIME FOR FUNCTION IN TIME TABLE (ATIMTBL)
                                ;       PUBLISHES TIME IF SW10=0
                                ;       UPDATES ITERATION COUNT AND IF ITERATIONS COMPLETE CONTINUES
                                ;       TO NEXT TEST, OTHERWISE REPEATS TEST.
                                ;       DELAYS BEFORE CONTINUING OR REPEATING TEST.
                                ;       INITIALIZES DRIVE
                                ;RETURNS:       R5=BASE ADDRESS OF TM03 REGISTERS (ADDRESS OF CS1)
                                ;               R1='DS' REG ADDRESS
                                ;               R0='FC' REG ADDRESS

004010  013705  001010          .SCOPE: MOV     @#TMBASE,R5             ;SET R5 TO FIRST TM REG
004014  032777  000400  174756          BIT     #SW08,@SWR              ;BRANCH IF SPECIFICATION LINE
004022  001404                          BEQ     10$                     ;NOT DESIRED ON EACH ITERATION
004024  113702  001122                  MOVB    @#TSTNUM,R2             ;GET TEST NUMBER
004030  004737  002650                  JSR     PC,OUTSPC               ;OUTPUT TIME  RECORDED
004034  032777  040000  174736  10$:    BIT     #SW14,@SWR              ;BRANCH IF CONTINUOS LOOP
004042  001432                          BEQ     2$                      ;NOT DESIRED
004044  017701  174730          1$:     MOV     @SWR,R1                 ;GET SWITCHES
004050  042701  177740                  BIC     #177740,R1              ;CLEAR ALL BUT TEST #
004054  001406                          BEQ     11$                     ;BRANCH IF ALL SELECTED
004056  020137  001122                  CMP     R1,@#TSTNUM             ;BRANCH IF RUNNING SELECTED TEST
004062  001403                          BEQ     11$
004064  012737  007166  001002          MOV     #TST000,SCPADR          ;RESTART AT TST000
004072  004737  004634          11$:    JSR     PC,DELAY                ;DELAY 350 MS
004076  004737  005070                  JSR     PC,RHINIT               ;INIT
004102  105037  001123                  CLRB    @#ERFLG                 ;CLEAR ERROR FLAG
004106  013716  001002                  MOV     SCPADR,(SP)
004112  010501                          MOV     R5,R1
004114  062701  000012                  ADD     #DS,R1                  ;ADDRESS OF 'DS' REG IS IN R1
004120  010500                          MOV     R5,R0
004122  062700  000006                  ADD     #FC,R0                  ;ADDRESS OF 'FC' REG IS IN R0
004126  000002                          RTI

004130  105737  001123          2$:     TSTB    @#ERFLG                 ;BRANCH IF ERROR FLAG IS SET
004134  001006                          BNE     3$
004136  113700  001121                  MOVB    @#ITCNT,R0              ;GET ITERATION COUNT
004142  006300                          ASL     R0                      ;STORE TIME IN TABLE
004144  013760  001012  001014          MOV     @#ATIME,ATIMTBL(R0)
004152  105237  001121          3$:     INCB    @#ITCNT                 ;INCREMENT ITERATION COUNT
004156  105737  001127                  TSTB    @#PSCNT                 ;INHIBIT ITERATIONS ON
004162  001410                          BEQ     4$                      ;ON FIRST PASS
004164  032777  004000  174606          BIT     #SW11,@SWR              ;BRANCH IF SINGLE ITERATION DESIRED
004172  001004                          BNE     4$
004174  122737  000020  001121          CMPB    #16.,@#ITCNT            ;BRANCH IF ITERATIONS INCOMPLETE
004202  001320                          BNE     1$
004204  032777  000037  174566  4$:     BIT     #37,@SWR        ;IF TEST SELECTED IS TEST 0
004212  001002                          BNE     42$             ;TREAT AS ALL TESTS
004214  011637  001002          40$:    MOV     (SP),@#SCPADR   ;SET SCOPE ADDRESS TO NEXT TEST
004220  032777  002000  174552  42$:    BIT     #SW10,@SWR      ;BRANCH IF NO PUBLICATION DESIRED
004226  001005                          BNE     5$
```

```
1936  004230  005737  005716            TST     CHNFLG                  ;BRANCH IF IN CHAIN MODE
1937  004234  001002                    BNE     5$
1938  004236  004737  003144            JSR     PC,PUBLISH              ;GO PUBLISH TEST DATA
1939  004242  105037  001121    5$:     CLRB    @#ITCNT                 ;RESET ITERATION COUNT
1940  004246  000676                    BR      1$
1941
1942                                    .SBTTL  TIMER SUBROUTINES
1943
1944                            ;SUBROUTINE TO SYNCHRONIZE THE TIMER AND TURN IT ON.
1945                            ;REGISTER 4 IS CLEARED, AND THE OSCILLATOR POLARITY IS MONITORED
1946                            ;THE ROUTINE IS EXITED WHEN THE OSCILLATOR POLARITY CHANGES WITH R3
1947                            ;SET TO INDICATE THE POLARITY OF THE OSCILLATOR.
1948                            ;CALL:  JSR     PC,TIMON
1949                            ;RETURNS:       R3 SET TO INDICATE LAST POLARITY (+24/-24=0/1)
1950                            ;               R4 = 0
1951
1952  004250  005004            TIMON:  CLR     R4                      ;CLEAR TIME COUNT
1953  004252  012703  000024            MOV     #24,R3                  ;SET POLARITY TO '0' STATE
1954  004256  032765  000100  000024    BIT     #OSC,MR(R5)             ;BRANCH IF POLARITY IS '0'
1955  004264  001405                    BEQ     2$
1956  004266  032765  000100  000024    1$:     BIT     #OSC,MR(R5)     ;WAIT FOR OSCILLATOR TO RETURN
1957  004274  001374                    BNE     1$
1958  004276  000405                    BR      4$
1959
1960  004300  005403            2$:     NEG     R3                      ;NEGATE PREV POLARITY INDICATOR
1961  004302  032765  000100  000024    3$:     BIT     #OSC,MR(R5)     ;WAIT FOR OSCILLATOR TO RETURN
1962  004310  001774                    BEQ     3$                      ;TO '1' STATE
1963  004312  000207            4$:     RTS     PC
1964
1965                            ;SUBROUTINE TO COUNT TIME
1966                            ;EACH TIME THE OSCILLATOR TOGGLES (BIT <06> IN MR REG) REGISTER
1967                            ;R4 IS INCREMENTED, AND THE REGISTER R3 IS NEGATED TO INDICATE
1968                            ;THE LAST STATE OF THE OSCILLATOR.
1969                            ;CALL   JMP     TIMER(R3)               ;R3 IS SET BY TIMON ROUTINE
1970                            ;       R2=RETURN ADDRESS TO CALLER
1971                            ;NOTE: THE TIME TO EXECUTE THIS ROUTINE IS VERY CRITICAL. IT MUST BE
1972                            ;LESS THAN 40 US.
1973
1974                            ;ENTER HERE VIA JMP  TIMER(R3) WHEN R3=-24 (PREV STATE=1)
1975  004314  032765  000100  000024    TIMER1: BIT     #OSC,MR(R5)     ;BRANCH IF CURRENT STATE IS '0'
1976  004322  001406                            BEQ     TIMER           ;GO INCREMENT TIME
1977  004324  000112                            JMP     (R2)            ;RETURN TO TEST
1978
1979          004340                            .=TIMER1+24
1980  004340  005403            TIMER:  NEG     R3                      ;NEGATE PREV STATE INDICATOR
1981  004342  005204                    INC     R4                      ;INCREMENT 'TICK' COUNT
1982  004344  100401                    BMI     TIMERR                  ;BRANCH ON OVERFLOW
1983  004346  000112                    JMP     (R2)                    ;RETURN TO TEST
1984  004350  000004  014346    TIMERR: TYPE,E.TIMOV                    ;TYPE 'TIMER OVERFLOWED'
1985  004354  104400                    HLT                             ;REPORT HARDWARE ERROR
1986  004356  000177  174420            JMP     @SCPADR                 ;RETURN TO BEGINNING OF TEST
1987
1988          004364                    .=TIMER+24
1989                            ;ENTER HERE VIA JMP  TIMER(R3) WHEN R3=+24 (PREV STATE=0)
1990  004364  032765  000100  000024    TIMER0: BIT     #OSC,MR(R5)     ;BRANCH IF CURRENT STATE = '1'
1991  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
```

```
2048  004530  005301                      DEC     R1
2049  004532  001374                      BNE     1$
2050
2051  004534  010246                      MOV     R2,-(SP)              ;DIVIDE TIME BY 10.
2052  004536  010346                      MOV     R3,-(SP)
2053  004540  012746  000012              MOV     #10.,-(SP)
2054  004544  004737  004722              JSR     PC,DIVIDE
2055  004550  005726                      TST     (SP)+                 ;DISCARD REMAINDER
2056  004552  012637  001012              MOV     (SP)+,@#ATIME         ;STORE QUOTIENT
2057  004556  113703  001120              MOVB    @#GAP,R3              ;GET GAP #
2058  004562  006303                      ASL     R3                    ;MULTPLY BY 4
2059  004564  006303                      ASL     R3                    ;TO GET AT TABLE ENTRY
2060  004566  023763  001012  001556      CMP     @#ATIME,GTIMTBL(R3)   ;CHECK TIME (MAX)
2061  004574  101004                      BHI     2$
2062  004576  023763  001012  001560      CMP     @#ATIME,GTIMTBL+2(R3) ;CHECK TIME (MIN)
2063  004604  101002                      BHI     3$
2064  004606  104402              2$:     HLT+2                         ;REPORT OUT OF RANGE ERROR
2065  004610  000406                      BR      100$
2066  004612  032777  000400  174160  3$: BIT     #SW08,@SWR            ;BRANCH IF TIMES NOT WANTED
2067  004620  001402                      BEQ     100$
2068  004622  004737  002744              JSR     PC,OUTGAP             ;TYPE GAP TIMES
2069
2070  004626                      100$:
2071  004626  004737  002376              JSR     PC,.RESTORE           ;RESTORE REGISTERS FROM THE STACK
2072  004632  000207                      RTS     PC                            ;RETURN TO TEST
2073
2074                                      .SBTTL          DELAY SUBROUTINES
2075                              ;THIS SUBROUTINE CAUSES A DELAY OF 350 MS.
2076  004634  004737  004250      DELAY:  JSR     PC,TIMON
2077  004640  010246                      MOV     R2,-(SP)              ;SAVE R2 ON THE STACK
2078  004642  012702  004652              MOV     #2$,R2                ;SET RETURN ADDRESS FOR TIMER
2079  004646                      1$:
2080  004646  000163  004340              JMP     TIMER(R3)             ;GO TO TIMER & RETURN VIA R2
2081  004652  032704  004000      2$:     BIT     #4000,R4
2082  004656  001773                      BEQ     1$
2083  004660  012602                      MOV     (SP)+,R2              ;RESTORE R2
2084  004662  000207                      RTS     PC
2085
2086                              ;THIS SUBROUTINE ALLOWS A CALLER SPECIFIED DELAY (UP TO 65MS.)
2087                              ;CALL:  MOV     DELAY TIME,DELTIM     ;LOAD DELAY TIME (IN US)
2088                              ;       JSR     PC,DELAYV
2089  004664  005737  001114      DELAYV: TST     DELTIM                ;BRANCH IF 0 DELAY
2090  004670  001413                      BEQ     3$
2091  004672  004737  004250              JSR     PC,TIMON              ;TURN TIMER ON
2092  004676  010246                      MOV     R2,-(SP)              ;SAVE R2 ON THE STACK
2093  004700  012702  004710              MOV     #2$,R2                ;SET RETURN ADDRESS FROM TIMER
2094  004704                      1$:
2095  004704  000163  004340              JMP     TIMER(R3)             ;GO TO TIMER & RETURN VIA R2
2096  004710  023704  001114      2$:     CMP     @#DELTIM,R4
2097  004714  101373                      BHI     1$
2098  004716  012602                      MOV     (SP)+,R2              ;RESTORE R2
2099  004720  000207              3$:     RTS     PC
2100
2101                                      .SBTTL  DIVIDE SUBROUTINE
2102                              ;THIS SUBROUTINE DIVIDES A DOUBLE PRECISION # AND RETURNS THE RESULT
2103                              ;TO THE CALLER ON THE STACK. BOTH DIVIDEND & DIVISOR MUST BE POSITIVE.
```

```
                                ;CALL:  MOV     LEAST SIGNIFICANT HALF DIVIDEND,-(SP)
                                ;       MOV     #MOST SIGNIFICANT HALF DIVIDEND,-(SP)
                                ;       MOV     #DIVISOR,-(SP)
                                ;       JSR     PC,DIVIDE
                                ;RETURN
                                ;       (SP)=REMAINDER ON STACK
                                ;       2(SP)=QUOTIENT

                                ;NOTE:  THIS SUBROUTINE DESTROYS PREVIOUS CONTENTS OF R0,R1,R2 & R3.

004722  005046                  DIVIDE: CLR     -(SP)                   ;SAVE LOC FOR SIGNS
004724  012746  000021                  MOV     #17.,-(SP)              ;SET ITERATION COUNT
004730  016601  000012                  MOV     12(SP),R1               ;GET LSH DIVIDEND
004734  016600  000010                  MOV     10(SP),R0               ;GET MSH DIVIDEND
004740  016602  000006                  MOV     6(SP),R2                ;GET DIVISOR
004744  005402                          NEG     R2                      ;NEGATE DIVISOR
004746  000241                          CLC                             ;CLEAR 'C' BIT IN PSW
004750  000405                          BR      2$
004752  006100                  1$:     ROL     R0                      ;ROTATE MSH DIVIDEND
004754  010003                          MOV     R0,R3                   ;SAVE IN R3
004756  060203                          ADD     R2,R3                   ;SUBTRACT DIVISOR FROM MSH DIVIDEND
004760  103001                          BCC     2$                      ;BRANCH IF DIVIDEND > DIVISOR
004762  010300                          MOV     R3,R0                   ;SAVE REMAINDER IN R0
004764  006101                  2$:     ROL     R1                      ;ROTATE LSH DIVIDEND
004766  005316                          DEC     (SP)                    ;DECREMENT ITERATION COUNT
004770  001370                          BNE     1$
004772  005726                          TST     (SP)+                   ;POP ITERATION COUNTER
004774  005726                          TST     (SP)+                   ;POP SIGN CORRECTION
004776  010166  000006                  MOV     R1,6(SP)                ;PUSH REMAINDER ON STACK
005002  010066  000004                  MOV     R0,4(SP)                ;PUSH QUOTIENT ONTO STACK
005006  012616                          MOV     (SP)+,(SP)
005010  000207                          RTS     PC

                                        .SBTTL  DRIVE SUBROUTINES
                                ;SUBROUTINE TO CHECK IF DRIVE IS AVAILABLE
                                ;CALL:  MOVB    DRIVE#,DRVNUM
                                ;       JSR     PC,DRVAVA
                                ;RETURN:        'C' BIT SET IF NOT AVAILABLE
005012  113765  001004  000010  DRVAVA: MOVB    @#DRVNUM,CS2(R5)        ;LOAD DRIVE #
005020  032765  040000  000026          BIT     #TAP,DT(R5)             ;CHECK IF TAPE UNIT
005026  001003                          BNE     1$
005030  004737  005070                  JSR     PC,RHINIT
005034  000262                          SEV                             ;SET 'V' TO IND NOT AVAIL
005036  000207                  1$:     RTS     PC                      ;RETURN

                                ;SUBROUTINE TO CHECK IF TU45 SLAVE IS AVAILABLE FOR TEST
                                ;CALL:  MOVB    DRIVE #,@#DRVNUM        ;PASS DRIVE # VIA DRVNUM
                                ;       MOVB    SLAVE #,@#SLVNUM        ;PASS SLAVE # VIA SLVNUM
                                ;       JSR     PC,SLVAVA               ;CALL SUBROUTINE
005040  113765  001004  000010  SLVAVA: MOVB    @#DRVNUM,CS2(R5)        ;LOAD DRIVE #
005046  113765  001005  000032          MOVB    @#SLVNUM,TC(R5)         ;AND SLAVE #
005054  032765  002000  000026          BIT     #SPR,DT(R5)             ;BRANCH IF SLAVE PRESENT
005062  001001                          BNE     1$
005064  000262                          SEV                             ;SET 'V' TO INDICATE NO SLAVE
005066  000207                  1$:     RTS     PC
```

```
                                        ;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$:     BIT     #PIP,DS(R5)             ;WAIT FOR TAPE MOTION TO STOP
005304  001374                         BNE     2$
```

```
2216  005306  000401                    BR      100$
2217  005310  000262            99$:    SEV
2218  005312  000207            100$:   RTS     PC
2219
2220                            ;SUBROUTINE TO WRITE 256. WORD RECORD
2221                            ;CALL:  JSR     PC,WRITE
2222
2223  005314  004337  005450    WRITE:  JSR     R3,TMCMD                ;GO TO TM COMMAND SUBROUTINE
2224  005320  015730                    .WORD   WTBUF                   ;BUS ADDRESS
2225  005322  177600                    .WORD   WRDCNT                  ;WORD COUNT
2226  005324  177400                    .WORD   FRMCNT                  ;FRAME COUNT
2227  005326  000060                    .WORD   WFWD                    ;WRITE FORWARD COMMAND
2228  005330  000207                    RTS     PC
2229
2230                            ;SUBROUTINE TO READ A 256. WORD RECORD.
2231                            ;CALL:  JSR     PC,READ
2232
2233  005332  004337  005450    READ:   JSR     R3,a#TMCMD
2234  005336  015730                    .WORD   RDBUF                   ;ADDRESS OF READ BUFFER
2235  005340  177600                    .WORD   WRDCNT                  ;2'S COMPLEMENT OF WORD COUNT
2236  005342  177400                    .WORD   FRMCNT                  ;2'S COMPLEMENT OF FRAME COUNT
2237  005344  000070                    .WORD   RDFWD                   ;READ FORWARD COMMAND
2238  005346  000207                    RTS     PC
2239
2240                            ;SUBROUTINE TO INITIATE READ REVERSE COMMAND
2241                            ;CALL:  JSR     PC,REVRD
2242                            ;
2243  005350  004337  005450    REVRD:  JSR     R3,TMCMD
2244  005354  016330                    .WORD   RDBUF+256.              ;ADDRESS OF READ REVERSE BUFFER
2245  005356  177600                    .WORD   WRDCNT                  ;2'S COMPLEMENT OF WORD COUNT
2246  005360  177400                    .WORD   FRMCNT                  ;2'S COMPLEMENT OF FRAME COUNT
2247  005362  000076                    .WORD   RDREV                   ;READ REVERSE COMMAND
2248  005364  000207                    RTS     PC
2249
2250                            ;SUBROUTINE TO SPACE FORWARD 1 RECORD
2251  005366  012765  177777  000006  FWDSPC: MOV     #-1,FC(R5)              ;LOAD RECORD COUNT
2252  005374  012715  000031            MOV     #SPCFWD+1,(R5)          ;LOAD COMMAND
2253  005400  004737  005126            JSR     PC,WAITRDY              ;WAIT FOR READY
2254  005404  000207                    RTS     PC                      ;RETURN
2255
2256                            ;SUBROUTINE TO WRITE A RECORD AND BACK SPACE OVER  THE RECORD.
2257  005406  004737  005314    WRT.BK: JSR     PC,WRITE                ;WRITE THE RECORD
2258  005412  005215                    INC     (R5)                    ;SET 'GO' BIT
2259  005414  004737  005126            JSR     PC,WAITRDY
2260  005420  102412                    BVS     2$
2261  005422  012765  177777  000006      MOV     #-1,FC(R5)              ;LOAD RECORD COUNT
2262  005430  012715  000033            MOV     #SPCREV+1,(R5)          ;LOAD COMMAND
2263  005434  004737  005126            JSR     PC,WAITRDY
2264  005440  102402                    BVS     2$
2265  005442  004737  004634        1$:     JSR     PC,DELAY                ;WAIT FOR TAPE MOTION TO STOP
2266  005446  000207                2$:     RTS     PC
2267
2268                            ;SUBROUTINE TO LOAD A COMMAND
2269                            ;CALL:  JSR     R3,TMCMD
2270                            ;       .WORD   BUS ADDRESS
2271                            ;       .WORD   WORD COUNT (2'S COMPLEMENT)
```

```
                                        ;       .WORD   FRAME COUNT (2'S COMPLEMENT)
                                        ;       .WORD   COMMAND

005450  012365  000004         TMCMD:  MOV     (R3)+,BA(R5)            ;LOAD BUS ADDRESS
005454  012365  000002                 MOV     (R3)+,WC(R5)            ;LOAD WORD COUNT
005460  012365  000006                 MOV     (R3)+,FC(R5)            ;LOAD FRAME COUNT
005464  012315                         MOV     (R3)+,(R5)              ;LOAD COMMAND
005466  000203                         RTS     R3                      ;RETURN

                                        ;SUBROUTINE TO PRINT TU45 SERIAL NUMBER
                                        ;JSR     PC,SNPT

005470  016503  000030         SNPT:   MOV     SN(R5),R3
005474  012701  001144                 MOV     #ODIGITS,R1
005500  000303                         SWAB    R3
005502  006003                         ROR     R3
005504  006003                         ROR     R3
005506  006003                         ROR     R3
005510  006003                         ROR     R3                      ;GET FIRST DIGIT
005512  042703  177760                 BIC     #177760,R3
005516  052703  000260                 BIS     #260,R3
005522  110321                         MOVB    R3,(R1)+                ;FILL FIRST DIGIT
005524  016503  000030                 MOV     SN(R5),R3
005530  000303                         SWAB    R3
005532  042703  177760                 BIC     #177760,R3
005536  052703  000260                 BIS     #260,R3
005542  110321                         MOVB    R3,(R1)+                ;GET SECOND DIGIT
005544  016503  000030                 MOV     SN(R5),R3
005550  006003                         ROR     R3
005552  006003                         ROR     R3
005554  006003                         ROR     R3
005556  006003                         ROR     R3
005560  042703  177760                 BIC     #177760,R3
005564  052703  000260                 BIS     #260,R3
005570  110321                         MOVB    R3,(R1)+                ;GET THIRD DIGIT
005572  016503  000030                 MOV     SN(R5),R3
005576  042703  177760                 BIC     #177760,R3
005602  052703  000260                 BIS     #260,R3
005606  110321                         MOVB    R3,(R1)+                ;GET FOURTH DIGIT
005610  105011                         CLRB    (R1)
005612  000004  001144                 TYPE,ODIGITS                    ;TYPE SERIAL NUMBER
005616  000207                         RTS     PC                      ;RETURN
```

```
 2315                                                .SBTTL  PROGRAM INITIALIZATION
 2316  005620  012706  000600          INIT:   MOV     #STKPTR,SP              ;SET STACK PTR
 2317  005624  005037  001264                  CLR     @#INBUF
 2318
 2319  005630  013746  000006                  MOV     @#6,-(SP)               ;SAVE VECTORS
 2320  005634  013746  000004                  MOV     @#4,-(SP)
 2321  005640  012737  005660  000004          MOV     #61$,@#4                        ;SET UP FOR TIMEOUT
 2322  005646  022777  177777  173124          CMP     #-1,@SWR                ;REFERENCE HARDWARE SWITCH REGISTER
 2323  005654  001402                          BEQ     60$
 2324  005656  000404                          BR      62$
 2325  005660  022626                  61$:    CMP     (SP)+,(SP)+             ;ADJUST STACK
 2326  005662  012737  000176  001000  60$:    MOV     #SWREG,SWR              ;POINT TO SOFTWARE SWITCH REG
 2327  005670  012637  000004          62$:    MOV     (SP)+,@#4               ;RESTORE VECTORS
 2328  005674  012637  000006                  MOV     (SP)+,@#6
 2329  005700  105037  001124                  CLRB    @#PRGFLG                ;CLEAR PROGRAM FLAG
 2330  005704  105037  001130                  CLRB    @#ASFLG                 ;CLEAR ASK FLAG
 2331  005710  105037  001127                  CLRB    @#PSCNT         ;SET PASS COUNT = 0
 2332  005714  005027                          CLR     (PC)+           ;;CLEAR CHAIN INDICATOR
 2333  005716  000000                  CHNFLG: .WORD   0               ;;CHAIN MODE INDICATOR
 2334                                                                  ;;1/0 = CHAIN/NOT CHAIN MODE
 2335  005720  022737  012642  000042          CMP     #$ENDAD,@#42    ;;BRANCH IF LOADED VIA ACT11 CHAIN MODE
 2336  005726  001404                          BEQ     50$
 2337  005730  005737  000042                  TST     @#42            ;;BRANCH IF IN DUMP MODE
 2338  005734  001413                          BEQ     52$
 2339  005736  000406                          BR      51$
 2340  005740  012737  000176  001000  50$:    MOV     #SWREG,SWR      ;;INVOKE SOFTWARE SWR
 2341  005746  012777  100000  173024          MOV     #100000,@SWR    ;;WITH HALT ON ERROR SET
 2342  005754  005237  005716          51$:    INC     CHNFLG          ;;SET CHNFLG = CHAIN MODE
 2343  005760  000137  006056                  JMP     5$              ;;GO TO CHAIN ADDRESS
 2344  005764                          52$:
 2345  005764  122737  000006  000041          CMPB    #6,@#41         ;BRANCH IF NOT LOADED VIA TMDP
 2346  005772  001002                          BNE     1$
 2347  005774  000004  013455                  TYPE,I.REM              ;ADVISE USER TO REMOVE TMDP
 2348  006000  000004  013252          1$:     TYPE,M.NAM                      ;TYPE TITLE
 2349  006004  105037  013252                  CLRB    M.NAM                   ;DO NOT TYPE TITLE ON RESTART
 2350  006010  000004  013522                  TYPE,I.REG                      ;ASK USER TO TYPE CONT BASE ADRS
 2351  006014  013702  001010                  MOV     @#TMBASE,R2             ;GET CURRENT CONT BASE ADDRESS
 2352  006020  004737  002426                  JSR     PC,TYPOCT               ;AND TYPE IT
 2353  006024  000004  001410                  TYPE,SPACE
 2354  006030  004737  003272                  JSR     PC,.INPUT               ;GET USER INPUT
 2355  006034  122737  000015  001264          CMPB    #CR,@#INBUF             ;DO NOT CHANGE CURRENT VALUE
 2356  006042  001405                          BEQ     5$                      ;IF USER TYPES <CR>
 2357  006044  004737  003056          4$:     JSR     PC,CNVTAO               ;CONVERT ASCII TO OCTAL
 2358  006050  013737  001116  001010          MOV     @#OCTAL0,@#TMBASE       ;SET NEW ADDRESS
 2359  006056  013705  001010          5$:     MOV     @#TMBASE,R5
 2360
 2361                                  ;ROUTINE TO CHECK IF CONTROLLER (RH11) IS AVAILAABLE
 2362  006062  000261                          SEC                             ;SET 'C' IN PSW
 2363  006064  005715                          TST     (R5)                    ;BRANCH IF CONTROLLER AVAIL
 2364  006066  103003                          BCC     6$
 2365  006070  000004  014015                  TYPE,E.NCON
 2366  006074  000651                          BR      INIT
 2367  006076  012737  003542  000004  6$:     MOV     #ERRTRP,@#ERRVEC        ;SET ERROR TRAP VECTOR
```

```
2368
2369  006104  105037  001123          ;ROUTINE TO GET TM03 DRIVES USER DESIRES TO TEST
DRIVES: CLRB    @#ERFLG                 ;CLEAR ERROR FLAG
2370  006110  012701  001154          MOV     #DRVTBL,R1              ;MARK ALL DRIVES AS NOT TO
2371  006114  012700  000004          MOV     #4,R0                   ;BE TESTED. A '0' INDICATES
2372  006120  005021                  1$:     CLR     (R1)+           ;THAT A DRIVE IS NOT TO BE
2373  006122  005300                  DEC     R0                      ;TESTED
2374  006124  001375                  BNE     1$
2375  006126  005737  005716          TST     CHNFLG                  ;BRANCH IF IN CHAIN MODE
2376  006132  001014                  BNE     2$
2377  006134  000004  013567          TYPE,I.DRVS
2378  006140  004737  003272          JSR     PC,.INPUT               ;GET USER INPUT
2379  006144  012700  001264          MOV     #INBUF,R0
2380  006150  122710  000101          CMPB    #'A,(R0)                ;IF USER RESPONDS WITH 'A' OR
2381  006154  001403                  BEQ     2$                      ;<CR> THEN ALL AVAILABLE DRIVES
2382  006156  122710  000015          CMPB    #CR,(R0)                ;ARE TO BE TESTED
2383  006162  001013                  BNE     4$
2384  006164  110637  001124          2$:     MOVB    SP,PRGFLG       ;SET FLAG TO IND ALL DRIVES
2385  006170  012701  001154          MOV     #DRVTBL,R1              ;MARK ALL DRIVES TO BE TESTED
2386  006174  012700  000004          MOV     #4,R0                   ;A '-1' INDICATES THAT A DRIVE
2387  006200  012721  177777          3$:     MOV     #-1,(R1)+       ;IS TO BE TESTED
2388  006204  005300                  DEC     R0
2389  006206  001374                  BNE     3$
2390  006210  000417                  BR      CHKDRV                  ;GO CHECK DRIVE AVAILABILITY
2391
2392                                  ;GET USER SELECTED DRIVES AND MARK EACH DRIVE SELECTED TO BE TESTED
2393  006212  122710  000015          4$:     CMPB    #CR,(R0)
2394  006216  001414                  BEQ     CHKDRV
2395  006220  121027  000054          CMPB    (R0),#',                ;CHECK IF 'COMMA'
2396  006224  001001                  BNE     5$
2397  006226  105720                  TSTB    (R0)+                   ;STEP PTR PAST 'COMMA'
2398  006230  112001                  5$:     MOVB    (R0)+,R1
2399  006232  042701  177770          BIC     #177770,R1
2400  006236  112761  177777  001154  MOVB    #-1,DRVTBL(R1)
2401  006244  000240                  NOP
2402  006246  000761                  BR      4$
2403
2404                                  ;ASCERTAIN THAT DRIVES (TM03'S) SPECIFIED ARE AVAILABLE
2405  006250  005000                  CHKDRV: CLR     R0              ;A (0) IN DRVTBL(R0) INDICATES
2406  006252  105760  001154          1$:     TSTB    DRVTBL(R0)      ;THE DRIVE IS NOT TO BE TESTED
2407  006256  001005                  BNE     3$                      ;A '1' INDICATES TO BE TESTED
2408  006260  005200                  2$:     INC     R0
2409  006262  122700  000010          CMPB    #8.,R0
2410  006266  001371                  BNE     1$
2411  006270  000424                  BR      5$
2412  006272  110037  001004          3$:     MOVB    R0,@#DRVNUM     ;GET DRIVE #
2413  006276  004737  005012          JSR     PC,@#DRVAVA             ;AND CHECK IF AVAILABLE
2414  006302  102366                  BVC     2$                      ;'V' BIT SET INDICATES NOT AVAIL
2415  006304  105737  001124          TSTB    @#PRGFLG                ;DO NOT TYPE NOT AVAILABLE
2416  006310  001011                  BNE     4$                      ;MESSAGE IF ALL SELECTED
2417  006312  000004  014062          TYPE,E.NDRV
2418  006316  116037  001132  014114  MOVB    DIGTAB(R0),@#E.NAVA     ;SET DRIVE # IN MESSAGE
2419  006324  000004  014114          TYPE,E.NAVA
2420  006330  110637  001123          MOVB    SP,@#ERFLG              ;SET 'ERROR' FLAG
2421  006334  105060  001154          4$:     CLRB    DRVTBL(R0)      ;MARK DRIVE UNAVAILABLE
2422  006340  000747                  BR      2$                      ;CHECK NEXT DRIVE
2423  006342  105737  001123          5$:     TSTB    @#ERFLG         ;GO GET SLAVES IF NO ERROR
```

```
2424  006346  001256                    BNE     DRIVES                  ;ELSE ASK USER TO RETYPE DRIVES
2425
2426                                    ;ROUTINE TO GET SLAVES (TU45'S) USER DESIRES TO TEST
2427  006350  105037  001123            SLAVES: CLRB    @#ERFLG         ;CLEAR ERROR INDICATOR
2428  006354  012701  001164                    MOV     #SLVTBL,R1              ;MARK ALL SLAVES (64.) AS NOT
2429  006360  012700  000040                    MOV     #32.,R0                 ;TO BE TESTED.A 0 INDICATES THAT
2430  006364  005021                    1$:     CLR     (R1)+                   ;A DRIVE'S SLAVE IS NOT TO BE
2431  006366  005300                            DEC     R0                      ;TESTED
2432  006370  001375                            BNE     1$
2433  006372  012701  001164                    MOV     #SLVTBL,R1              ;R1 POINTS TO DRIVE'S SLAVE
2434  006376  105760  001154            2$:     TSTB    DRVTBL(R0)              ;BRANCH IF DRIVE IS TO BE TESTED
2435  006402  001007                            BNE     4$                      ;& IS AVAILABLE
2436  006404  062701  000010            3$:     ADD     #8.,R1                  ;STEP SLAVE PTR TO NEXT DRIVE'S
2437  006410  005200                            INC     R0                      ;SLAVES AND INCREMENT DRIVE #
2438  006412  122700  000010                    CMPB    #8.,R0                  ;CHECK ALL DRIVES
2439  006416  001367                            BNE     2$                      ;AND WHEN ALL DRIVES CHECKED
2440  006420  000457                            BR      CHKSLV                  ;GO CHECK SLAVE AVAILABILITY
2441
2442  006422  105737  001124            4$:     TSTB    @#PRGFLG                ;BRANCH IF USER SELECTED ALL
2443  006426  001021                            BNE     5$                      ;DRIVES
2444  006430  110037  001004                    MOVB    R0,DRVNUM               ;GET DRIVE #
2445  006434  116037  001132  013656            MOVB    DIGTAB(R0),@#I.DRV      ;PREPARE USER ACTION MESSAGE
2446  006442  000004  013637                    TYPE,I.SLVS
2447  006446  004737  003272                    JSR     PC,.INPUT               ;GET USER INPUT
2448  006452  012703  001264                    MOV     #INBUF,R3               ;SET PTR TO USER INPUT
2449  006456  122713  000101                    CMPB    #'A,(R3)                ;AN 'A' OR <CR> AS FIRST CHAR
2450  006462  001403                            BEQ     5$                      ;INDICATES TEST ALL SLAVES
2451  006464  122713  000015                    CMPB    #CR,(R3)
2452  006470  001015                            BNE     7$
2453  006472  110637  001124            5$:     MOVB    SP,@#PRGFLG             ;SET 'ALL' INDICATOR
2454  006476  012701  001164                    MOV     #SLVTBL,R1              ;MARK ALL SLAVES FOR ALL
2455  006502  012700  000040                    MOV     #32.,R0                 ;DRIVES AS TO BE TESTED
2456  006506  012721  177777            6$:     MOV     #-1,(R1)+
2457  006512  005300                            DEC     R0
2458  006514  001374                            BNE     6$
2459  006516  105737  001124                    TSTB    @#PRGFLG                ;BRANCH IF ALL WAS SELECTED
2460  006522  001016                            BNE     CHKSLV
2461
2462  006524  122713  000015            7$:     CMPB    #CR,(R3)                ;GET USER SELECTED SLAVES FOR
2463  006530  001725                            BEQ     3$                      ;DRIVE
2464  006532  121327  000054                    CMPB    (R3),#',                ;STEP PTR PAST 'COMMA'
2465  006536  001001                            BNE     8$
2466  006540  105723                            TSTB    (R3)+
2467  006542  112304                    8$:     MOVB    (R3)+,R4                ;AND MARK SELECED SLAVE
2468  006544  042704  177770                    BIC     #177770,R4              ;AS TO BE TESTED
2469  006550  060104                            ADD     R1,R4
2470  006552  112714  177777                    MOVB    #-1,(R4)
2471  006556  000762                            BR      7$
2472
2473                                    ;ASCERTAIN THAT SLAVES (TU45'S) SELECTED ARE AVAILABLE
2474  006560  005000            CHKSLV: CLR     R0                      ;R0 WILL CONTAIN THE DRIVE #
2475  006562  005001                            CLR     R1                      ;AND R1 THE SLAVE #
2476  006564  012702  001164                    MOV     #SLVTBL,R2              ;SET PTR TO SLAVE TABLE
2477  006570  105760  001154            1$:     TSTB    DRVTBL(R0)              ;BRANCH IF DRIVE SELECTED
2478  006574  001007                            BNE     3$                      ;& AVAILABLE FOR TEST
2479  006576  005200            2$:     INC     R0                      ;INCREMENT DRIVE #
```

```
2480  006600  062702  000010            ADD     #8.,R2                  ;STEP SLAVE PTR TO NEXT DRIVE'S
2481  006604  022700  000010            CMP     #8.,R0                  ;SLAVES. BRANCH TO 1$ IF NOT ALL
2482  006610  001367                    BNE     1$                      ;DRIVES CHECKED OTHERWISE EXIT
2483  006612  000437                    BR      8$
2484
2485  006614  005001            3$:     CLR     R1                      ;SET SLAVE # 0
2486  006616  105712            4$:     TSTB    (R2)                    ;BRANCH IF DRIVE'S SLAVE IS SEL-
2487  006620  001006                    BNE     6$                      ;ECTED FOR TEST
2488  006622  005201            5$:     INC     R1                      ;INCREMENT SLAVE #
2489  006624  005202                    INC     R2                      ;STEP PTR TO NEXT SLAVE
2490  006626  022701  000010            CMP     #8.,R1                  ;GO TO 4$ IF ALL SLAVES NOT
2491  006632  001371                    BNE     4$                      ;CHECKED
2492  006634  000760                    BR      2$                      ;OTHERWISE GO TO 2$ ABOVE
2493
2494  006636  110037  001004    6$:     MOVB    R0,@#DRVNUM             ;PASS DRIVE & SLAVE #
2495  006642  110137  001005            MOVB    R1,@#SLVNUM
2496  006646  004737  005040            JSR     PC,@#SLVAVA             ;AND CHECK IF AVAILABLE
2497  006652  102363                    BVC     5$                      ;'V' SET INDCATES ERROR
2498  006654  105737  001124            TSTB    @#PRGFLG                ;DO NOT TYPE ERROR MSG IF ALL
2499  006660  001012                    BNE     7$                      ;SLAVES SELECTED
2500  006662  116037  001132  014104    MOVB    DIGTAB(R0),@#E.DRV      ;ICATES ERROR. PREPARE ERROR
2501  006670  116137  001132  014114    MOVB    DIGTAB(R1),@#E.NAVA     ;MESSAGE
2502  006676  000004  014076            TYPE,E.NSLV
2503  006702  110637  001123            MOVB    SP,@#ERFLG              ;SET ERROR INDICATOR
2504  006706  105012            7$:     CLRB    (R2)                    ;CLEAR SLAVE TABLE ENTRY
2505  006710  000744                    BR      5$                      ;GET NEXT SLAVE
2506
2507  006712  105737  001123    8$:     TSTB    @#ERFLG                 ;BRANCH IF ERROR
2508  006716  001214                    BNE     SLAVES                  ;ASK USER TO RETYPE SLAVES
2509  006720  012737  003542  000004  100$:   MOV     #ERRTRP,@#ERRVEC
2510
2511                                    ;SCAN DRIVE AND SLAVE TABLE FOR DRIVE/SLAVE COMBINATION TO TEST.
2512                                    ;RESTART ADDRESS--PROGRAM STARTS HERE WHEN START ADDRESS = 210 AND
2513                                    ;AFTER EACH PASS
2514  006726  105037  001004    RSTRT:  CLRB    @#DRVNUM                ;SET DRIVE AND SLAVE # 0
2515  006732  105037  001005            CLRB    @#SLVNUM
2516  006736  012737  001164  001006    MOV     #SLVTBL,@#SLVPTR        ;SET PTR TO SLAVE TABLE
2517  006744  105037  001125            CLRB    @#UNTFND                ;CLEAR 'UNIT FOUND' IND.
2518
2519                                    ;PROGRAM RESTARTS HERE AFTER EACH DRIVE/SLAVE HAS BEEN TESTED.
2520  006750  113700  001004    BEGIN:  MOVB    @#DRVNUM,R0             ;GET DRIVE #
2521  006754  113701  001005            MOVB    @#SLVNUM,R1             ;AND SLAVE #
2522  006760  013702  001006            MOV     @#SLVPTR,R2             ;GET SLAVE PTR
2523  006764  122737  000006  000041    CMPB    #6,@#41                 ;BRANCH IF LOADED VIA TMDP
2524  006772  001001                    BNE     1$
2525  006774  105012                    CLRB    (R2)                    ;SET DRIVE #0,SLAVE #0 NOT TO
2526                                                                    ;BE TESTED.
2527  006776  105760  001154    1$:     TSTB    DRVTBL(R0)              ;BRANCH IF DRIVE AVAIL TO TEST
2528  007002  001011                    BNE     3$
2529  007004  005001                    CLR     R1                      ;CLEAR SLAVE #
2530  007006  062702  000010            ADD     #8.,R2                  ;AND STEP PTR TO NEXT DRIVE'S
2531  007012  005200            2$:     INC     R0                      ;SLAVES AND INCREMENT DRIVE #
2532  007014  022700  000010            CMP     #8.,R0                  ;EXIT TEST IF ALL DRIVES
2533  007020  001366                    BNE     1$                      ;CHECKED OTHERWISE CONTINUE
2534  007022  000137  012570            JMP     @#END                   ;SCAN FOR NEXT 'UNIT'
2535
```

```
 2536  007026  105712                3$:     TSTB    (R2)                ;BRANCH IF SLAVE ON DRIVE IS
 2537  007030  001007                        BNE     4$                  ;AVAILABLE THERWISE STEP
 2538  007032  005202                        INC     R2                  ;PTR TO NEXT SLAVE
 2539  007034  005201                        INC     R1                  ;INCREMENT SLAVE #
 2540  007036  122701  000010              CMPB    #8.,R1              ;UNTIL ALL SLAVES CHECKED
 2541  007042  001371                        BNE     3$                  ;WHEN ALL SLAVES CHECKED
 2542  007044  005001                        CLR     R1                  ;SET SLAVE # 0
 2543  007046  000761                        BR      2$                  ;AND CONTINUE SCAN
 2544
 2545  007050  110637  001125          4$:     MOVB    SP,@#UNTFND         ;INDICATE THAT A 'UNIT' IS FOUND
 2546  007054  110037  001004                MOVB    R0,@#DRVNUM         ;SET DRIVE 3
```

```
2547  007060  110137  001005           MOVB    R1,@#SLVNUM             ;SET SLAVE #
2548  007064  010237  001006           MOV     R2,@#SLVPTR             ;SAVE SLAVE PTR
2549
2550  007070  105737  001130    5$:    TSTB    @#ASFLG
2551  007074  001034                   BNE     7$
2552  007076  112737  000001  001130   MOVB    #1,ASFLG
2553  007104  005737  005716           TST     CHNFLG                  ;BRANCH IF IN CHAIN MODE
2554  007110  001026                   BNE     7$
2555  007112  105037  001124           CLRB    @#PRGFLG        ;CLEAR PROGRAM INDICATOR
2556  007116  000004  013723           TYPE,I.SPD              ;ASK USER IF HE WANTS TO RUN SPEED TESTS
2557  007122  004737  003272           JSR     PC,..INPUT              ;GET USER INPUT
2558  007126  012703  001264           MOV     #INBUF,R3       ;GET REPLY
2559  007132  122713  000015           CMPB    #CR,(R3)        ;DO NOT DO SKEW TESTS IF <CR> IS FIRST
2560  007136  001405                   BEQ     6$
2561  007140  132713  000001           BITB    #1,(R3)         ;BRANCH IF 'N'
2562  007144  001402                   BEQ     6$
2563  007146  111337  001124           MOVB    (R3),@#PRGFLG   ;SET INDICATOR
2564  007152  022737  000176  001000   6$:     CMP     #SWREG,SWR              ;BRANCH IF SOFTWARE SWR
2565  007160  001002                   BNE     7$                      ;NOT INVOKED
2566  007162  004737  002020           JSR     PC,GTSWR                ;GET SWITCH REGISTER
2567  007166                   7$:
2568
2569                           ;NOTE THIS IS NOT A TEST
2570                           ;INITIALIZE PROGRAM FLAGS
2571  007166  013705  001010   TST000: MOV     @#TMBASE,R5             ;SET ADDRESS OF FIRST TM03 REG
2572  007172  010500                   MOV     R5,R0
2573  007174  062700  000006           ADD     #FC,R0                  ;R0 CONTAINS ADDRESS OF FC REG
2574  007200  010501                   MOV     R5,R1
2575  007202  062701  000012           ADD     #DS,R1                  ;R1 CONTAINS ADDRESS OF DS REG
2576  007206  012703  004340           MOV     #TIMER,R3               ;SET JUMP ADDRESS TO TIMER
2577  007212  105037  001121           CLRB    @#ITCNT                 ;CLEAR SUBTEST ITERATION COUNT
2578  007216  052737  000100  177560   BIS     #100,@#TKS              ;SET KEYBOARD IE BIT
2579
2580                           ;GET USER RUN PROCEDURE
2581                           ;IF SWR <05::00> IS NOT 0 THEN RUN TEST IN SWR<05::00>
2582                           ;OTHERWISE RUN ALL TESTS
2583
2584  007224  004737  005232           JSR     PC,.REWIND              ;REWIND SLAVE
2585  007230  102474                   BVS     99$                     ;BRANCH IF ERROR ON REWIND
2586  007232  004737  005314           JSR     PC,WRITE                ;WRITE A RECORD
2587  007236  005215                   INC     (R5)                    ;SET 'GO' BIT
2588  007240  004737  005126           JSR     PC,WAITRDY              ;WAIT FOR READY
2589  007244  102466                   BVS     99$
2590  007246  005737  005716           TST     CHNFLG                  ;BRANCH IF IN CHAIN MODE
2591  007252  001064                   BNE     100$
2592  007254  117702  171520           MOVB    @SWR,R2                 ;GET SWITCHES
2593  007260  042702  177740           BIC     #177740,R2              ;CLEAR ALL BUT TEST #
2594  007264  001421                   BEQ     2$                      ;& BRANCH IF TEST 0 WAS SELECTED
2595  007266  000004  014235           TYPE,E.HDR              ;TYPE TEST #
2596  007272  004737  002426           JSR     PC,TYPOCT
2597  007276  006302                   ASL     R2                      ;FORM INDEX VALUE
2598  007300  016237  001656  007310   MOV     NAMPTR(R2),1$           ;GET ADDRESS OF TEST'S NAME
2599  007306  000004                   TYPE                            ;AND TYPE IT
2600  007310  000000           1$:     .WORD   0
2601  007312  000004  001374           TYPE,CRLF
2602  007316  016237  001736  001002   MOV     TSTTBL(R2),@#SCPADR     ;SET SCOPE ADDRESS FOR TEST
```

```
 2603  007324  000172  001736                  JMP     @TSTTBL(R2)             ;GO TO TEST
 2604  007330  000004  014444          2$:     TYPE,L.HDR1
 2605  007334  113702  001004                  MOVB    DRVNUM,R2               ;GET DRIVE #
 2606  007340  113704  001005                  MOVB    SLVNUM,R4               ;AND SLAVE #
 2607  007344  116237  001132  014624          MOVB    DIGTAB(R2),@#L.DRV      ;SET DRIVE AND SLAVE #'S
 2608  007352  116437  001132  014636          MOVB    DIGTAB(R4),@#L.SLV      ;INTO L.HDR2 MESSAGE
 2609  007360  112737  000071  014641          MOVB    #'9,@#L.CHAN            ;GET SLAVES CHANNEL TYPE
 2610  007366  000004  014557                  TYPE,L.HDR2
 2611  007372  004737  005470                  JSR     PC,SNPT                 ;PRINT SLAVE SERIAL #
 2612  007376  000004  014660                  TYPE,L.HDR3
 2613  007402  105737  001124                  TSTB    @#PRGFLG                ;BRANCH IF SPEED TESTS NOT
 2614  007406  001406                          BEQ     100$                    ;SELECTED
 2615  007410  012737  012714  001002          MOV     #TST026,@#SCPADR        ;SET SCOPE LOOP ADDRESS
 2616  007416  000137  012706                  JMP     @#SKEWTST               ;GO DO SPEED TESTS
 2617  007422  104400                  99$:    HLT
 2618  007424  012737  007432  001002  100$:   MOV     #TST001,@#SCPADR        ;SET SCOPE LOOP ADDRESS
 2619
```

```
                                        .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$
```

```
2676  007622  000772                    BR      1$
2677
2678  007624                          2$:
2679  007624  004737  004250            JSR     PC,TIMON                ;TURN TIMER ON
2680  007630  010702                    MOV     PC,R2                   ;LOAD RETURN PC FROM TIMER
2681  007632  032711  000020          3$:       BIT     #SDWN,(R1)              ;BRANCH WHEN DS <SDWN> SETS
2682  007636  001002                    BNE     4$
2683  007640  000163  004340            JMP     TIMER(R3)               ;GO TO TIMER & RETURN VIA R2
2684
2685  007644  004737  005126          4$:       JSR     PC,WAITRDY              ;WAIT FOR READY
2686  007650  102403                    BVS     99$
2687  007652  004737  004376            JSR     PC,TIMOK                ;GO CHECK TIME RECORDED
2688  007656  000401                    BR      100$
2689  007660  104400                  99$:      HLT                             ;REPORT ERROR
2690  007662  104000                  100$:     SCOPE
2691
2692                                  ;TEST 004 - WRITE SETTLEDOWN
2693                                  ;THIS TEST MEASURES TIME FROM 'SWDN'=1 TO 'SWDN'=0.
2694  007664  112737  000004  001122  TST004: MOVB    #4,@#TSTNUM
2695  007672  004737  005314            JSR     PC,WRITE
2696  007676  005215                    INC     (R5)                    ;SET 'GO' BIT
2697
2698  007700  005710                  1$:       TST     (R0)                    ;BRANCH WHEN WRITING FINISHED
2699  007702  001404                    BEQ     2$
2700  007704  032711  040000            BIT     #ERR,(R1)               ;CHECK ERROR BIT
2701  007710  001026                    BNE     99$
2702  007712  000772                    BR      1$
2703
2704  007714  032711  000020          2$:       BIT     #SDWN,(R1)              ;WAIT FOR ASSERTION OF 'SDWN'
2705  007720  001004                    BNE     3$
2706  007722  032711  040000            BIT     #ERR,(R1)               ;MONITOR ERROR BIT
2707  007726  001017                    BNE     99$
2708  007730  000771                    BR      2$
2709
2710  007732                          3$:
2711  007732  004737  004250            JSR     PC,TIMON                ;TURN TIMER ON
2712  007736  010702                    MOV     PC,R2                   ;SET RETURN PC FROM TIMER
2713  007740  032711  000020            BIT     #SDWN,(R1)              ;BRANCH WHEN SWDN CLEARS
2714  007744  001402                    BEQ     5$
2715  007746  000163  004340            JMP     TIMER(R3)               ;GO TO TIMER & RETURN VIA R2
2716
2717  007752  004737  005126          5$:       JSR     PC,WAITRDY              ;WAIT FOR READY
2718  007756  102403                    BVS     99$
2719  007760  004737  004376            JSR     PC,TIMOK
2720  007764  000401                    BR      100$
2721
2722  007766  104400                  99$:      HLT
2723  007770  104000                  100$:     SCOPE
2724
2725                                  ;TEST 005 - READ FROM BOT
2726                                  ;THIS TEST MEASURES TIME FROM 'GO'=1 TO 'ACCL'=0.
2727  007772  112737  000005  001122  TST005: MOVB    #5,@#TSTNUM             ;SET TEST #5
2728  010000  004737  005232            JSR     PC,.REWIND              ;REWIND SLAVE
2729  010004  102422                    BVS     99$                     ;BRANCH IF ERROR ON REWIND
2730  010006  004737  005332            JSR     PC,READ
2731  010012  012702  010024            MOV     #1$,R2                  ;SET RETURN PC FROM TIMER
```

```
2732  010016  004737  004250            JSR     PC,TIMON                ;TURN TIMER ON
2733  010022  005215                    INC     (R5)                    ;SET 'GO' BIT
2734
2735  010024  005765  000032    1$:     TST     TC(R5)                  ;BRANCH WHEN 'ACCL' RESETS
2736  010030  100002                    BPL     2$
2737  010032  000163  004340            JMP     TIMER(R3)               ;GO TO TIMER & RETURN VIA R2
2738
2739  010036  004737  005126    2$:     JSR     PC,WAITRDY              ;WAIT FOR READY
2740  010042  102403                    BVS     99$                     ;BRANCH IF ERROR
2741  010044  004737  004376            JSR     PC,TIMOK                ;CHECK RECORDED TIME
2742  010050  000401                    BR      100$
2743
2744  010052  104400    99$:    HLT
2745  010054  104000    100$:   SCOPE
2746
2747                    ;TEST 006 - READ START
2748                    ;THIS TEST MEASURES TIME FROM 'GO'=1 TO 'ACCL'=0.
2749  010056  112737  000006  001122  TST006: MOVB    #6,@#TSTNUM             ;SET TEST #6
2750  010064  004737  005406            JSR     PC,WRT.BK               ;WRITE A RECORD & BACK SPACE
2751  010070  102422                    BVS     99$
2752  010072  004737  005332            JSR     PC,READ
2753  010076  012702  010110            MOV     #1$,R2                  ;SET RETURN PC FROM TIMER
2754  010102  004737  004250            JSR     PC,TIMON                ;TURN TIMER ON
2755  010106  005215                    INC     (R5)                    ;SET 'GO' BIT
2756
2757  010110  005765  000032    1$:     TST     TC(R5)                  ;BRANCH WHEN 'ACCL' RESETS
2758  010114  100002                    BPL     2$
2759  010116  000163  004340            JMP     TIMER(R3)               ;GO TO TIMER & RETURN VIA R2
2760
2761  010122  004737  005126    2$:     JSR     PC,WAITRDY
2762  010126  102403                    BVS     99$
2763  010130  004737  004376            JSR     PC,TIMOK
2764  010134  000401                    BR      100$
2765
2766  010136  104400    99$:    HLT
2767  010140  104000    100$:   SCOPE
2768
2769                    ;TEST 007 - READ SHUTDOWN
2770                    ;THIS TEST MEASURES TIME FROM 'FC REG'=FRAME COUNT TO 'SWDN'=1.
2771  010142  112737  000007  001122  TST007: MOVB    #7,@#TSTNUM             ;SET TEST #7
2772  010150  004737  005406            JSR     PC,WRT.BK               ;WRITE A RECORD & BACK SPACE
2773  010154  102430                    BVS     99$                     ;BRANCH IF ERROR
2774  010156  004737  005332            JSR     PC,READ
2775  010162  005215                    INC     (R5)                    ;SET 'GO' BIT
2776
2777  010164  022710  000400    1$:     CMP     #-FRMCNT,(R0)           ;WAIT FOR FRAME COUNT TO
2778  010170  001404                    BEQ     2$                      ;= # OF FRAMES WRITTEN
2779  010172  032711  040000            BIT     #ERR,(R1)               ;MONITOR ERROR BIT
2780  010176  001017                    BNE     99$
2781  010200  000771                    BR      1$
2782
2783  010202            2$:
2784  010202  004737  004250            JSR     PC,TIMON                ;TURN TIMER ON
2785  010206  010702                    MOV     PC,R2                   ;SET RETURN PC FROM TIMER
2786  010210  032711  000020            BIT     #SDWN,(R1)              ;BRANCH WHEN SDWN SETS
2787  010214  001002                    BNE     3$
```

```
2788  010216  000163  004340                  JMP     TIMER(R3)               ;GO TO TIMER & RETURN VIA R2
2789
2790  010222  004737  005126          3$:     JSR     PC,WAITRDY
2791  010226  102403                          BVS     99$
2792  010230  004737  004376                  JSR     PC,TIMOK
2793  010234  000401                          BR      100$
2794
2795  010236  104400                  99$:    HLT                             ;REPORT ERROR
2796  010240  104000                  100$:   SCOPE
2797
2798                                  ;TEST 010 - READ SETTLEDOWN
2799                                  ;THIS TEST MEASURES TIME FROM 'SWDN'=1 TO 'SWDN'=0.
2800  010242  112737  000010  001122 TST010: MOVB    #10,@#TSTNUM            ;SET TEST #10
2801  010250  012702  010326                  MOV     #4$,R2                  ;SET RETURN PC FROM TIMER
2802  010254  004737  005406                  JSR     PC,WRT.BK               ;WRITE A RECORD & BACK SPACE
2803  010260  102436                          BVS     99$
2804  010262  004737  005332                  JSR     PC,READ
2805  010266  005215                          INC     (R5)                    ;SET 'GO' BIT
2806
2807  010270  105711                  1$:     TSTB    (R1)                    ;WAIT FOR READY
2808  010272  100404                          BMI     2$                      ;BRANCH WHEN SET
2809  010274  032711  040000                  BIT     #ERR,(R1)               ;CHECK ERROR BIT
2810  010300  001026                          BNE     99$
2811  010302  000772                          BR      1$
2812
2813  010304  032711  000020          2$:     BIT     #SDWN,(R1)              ;WAIT FOR ASSERTION OF 'SDWN'
2814  010310  001004                          BNE     3$
2815  010312  032711  040000                  BIT     #ERR,(R1)               ;MONITOR ERROR BIT
2816  010316  001017                          BNE     99$
2817  010320  000771                          BR      2$
2818
2819  010322                          3$:
2820  010322  004737  004250                  JSR     PC,TIMON                ;TURN TIMER ON
2821  010326  032765  000020  000012  4$:     BIT     #SDWN,DS(R5)            ;WAIT FOR NEGATION OF SDWN
2822  010334  001402                          BEQ     5$
2823  010336  000163  004340                  JMP     TIMER(R3)               ;GO TO TIMER & RETURN VIA R2
2824
2825  010342  004737  005126          5$:     JSR     PC,WAITRDY
2826  010346  102403                          BVS     99$
2827  010350  004737  004376                  JSR     PC,TIMOK
2828  010354  000401                          BR      100$
2829
2830  010356  104400                  99$:    HLT
2831  010360  104000                  100$:   SCOPE
2832
2833
2834                                  ;TEST 011-READ REVERSE START
2835                                  ;THIS TEST MEASURES TIME FROM 'GO'=1 TO 'ACCL'=0.
2836  010362  112737  000011  001122 TST011: MOVB    #11,@#TSTNUM
2837  010370  012702  010426                  MOV     #1$,R2                  ;SET RETURN PC FROM TIMER
2838  010374  004737  005314                  JSR     PC,WRITE                ;WRITE A RECORD
2839  010400  005215                          INC     (R5)                    ;SET 'GO' BIT
2840  010402  004737  005126                  JSR     PC,WAITRDY
2841  010406  102422                          BVS     99$
2842  010410  004737  004634                  JSR     PC,DELAY                ;WAIT FOR TAPE MOTION TO STOP
2843  010414  004737  005350                  JSR     PC,REVRD
```

```
2844  010420  004737  004250            JSR     PC,TIMON                ;TURN TIMER ON
2845  010424  005215                    INC     (R5)                    ;SET 'GO' BIT
2846
2847  010426  005765  000032    1$:     TST     TC(R5)                  ;BRANCH WHEN 'ACCL' = 0
2848  010432  100002                    BPL     2$
2849  010434  000163  004340            JMP     TIMER(R3)               ;GO TO TIMER & RETURN VIA R2
2850
2851  010440  004737  005126    2$:     JSR     PC,WAITRDY
2852  010444  102403                    BVS     99$                     ;BRANCH IF ERROR
2853  010446  004737  004376            JSR     PC,TIMOK
2854  010452  000401                    BR      100$
2855
2856  010454  104400            99$:    HLT
2857  010456  104000            100$:   SCOPE
2858
2859                            ;TEST 012-READ REVERSE SHUTDOWN
2860                            ;THIS TEST MEASURES TIME FROM 'FC REG' = FRAME COUNT TO 'SDWN'=1.
2861  010460  112737  000012  001122 TST012: MOVB    #12,@#TSTNUM
2862  010466  012702  010536            MOV     #3$,R2                  ;SET RETURN PC FROM TIMER
2863  010472  004737  005314            JSR     PC,WRITE                ;WRITE A RECORD
2864  010476  005215                    INC     (R5)                    ;SET 'GO' BIT
2865  010500  004737  005126            JSR     PC,WAITRDY
2866  010504  102427                    BVS     99$
2867  010506  004737  005350            JSR     PC,REVRD
2868  010512  005215                    INC     (R5)                    ;SET 'GO' BIT
2869
2870  010514  022710  000400    1$:     CMP     #-FRMCNT,(R0)           ;BRANCH WHEN FRAME COUNT
2871  010520  001404                    BEQ     2$                      ;= # OF RECORD WRITTEN
2872  010522  032711  040000            BIT     #ERR,(R1)               ;MONITOR ERROR BIT IN 'DS' REG
2873  010526  001016                    BNE     99$
2874  010530  000771                    BR      1$
2875
2876  010532                    2$:
2877  010532  004737  004250            JSR     PC,TIMON                ;TURN TIMER ON
2878  010536  032711  000020    3$:     BIT     #SDWN,(R1)              ;BRANCH WHEN SDWN SETS
2879  010542  001002                    BNE     4$
2880  010544  000163  004340            JMP     TIMER(R3)               ;GO TO TIMER & RETURN VIA R2
2881
2882  010550  004737  005126    4$:     JSR     PC,WAITRDY              ;WAIT FOR READY
2883  010554  102403                    BVS     99$
2884  010556  004737  004376            JSR     PC,TIMOK
2885  010562  000401                    BR      100$
2886
2887  010564  104400            99$:    HLT
2888  010566  104000            100$:   SCOPE
2889
2890                            ;TEST 013-READ REVERSE SETTLEDOWN
2891                            ;THIS TEST MEASURES TIME FROM 'SDWN'=1 TO 'SDWN'=0.
2892  010570  112737  000013  001122 TST013: MOVB    #13,@#TSTNUM
2893  010576  012702  010662            MOV     #4$,R2                  ;SET RETURN PC FROM TIMER
2894  010602  004737  005314            JSR     PC,WRITE                ;WRITE A RECORD
2895  010606  005215                    INC     (R5)                    ;SET 'GO' BIT
2896  010610  004737  005126            JSR     PC,WAITRDY
2897  010614  102435                    BVS     99$
2898  010616  004737  005350            JSR     PC,REVRD
2899  010622  005215                    INC     (R5)                    ;SET 'GO' BIT
```

```
2900
2901  010624  105711                  1$:     TSTB    (R1)                    ;BRANCH WHEN
2902  010626  100404                          BMI     2$                      ;READY SETS
2903  010630  032711  040000                  BIT     #ERR,(R1)
2904  010634  001025                          BNE     99$
2905  010636  000772                          BR      1$
2906
2907  010640  032711  000020          2$:     BIT     #SDWN,(R1)
2908  010644  001004                          BNE     3$
2909  010646  032711  040000                  BIT     #ERR,(R1)
2910  010652  001016                          BNE     99$
2911  010654  000771                          BR      2$
2912
2913  010656                          3$:
2914  010656  004737  004250                  JSR     PC,TIMON                ;TURN TIMER ON
2915  010662  032711  000020          4$:     BIT     #SDWN,(R1)              ;BRANCH WHEN SWDN = 0
2916  010666  001402                          BEQ     5$
2917  010670  000163  004340                  JMP     TIMER(R3)               ;GO TO TIMER & RETURN VIA R2
2918
2919  010674  004737  005126          5$:     JSR     PC,WAITRDY              ;WAIT FOR READY
2920  010700  102403                          BVS     99$
2921  010702  004737  004376                  JSR     PC,TIMOK
2922  010706  000401                          BR      100$
2923
2924  010710  104400                  99$:    HLT
2925  010712  104000                  100$:   SCOPE
2926
2927                                  ;REWIND DRIVE
2928  010714                          A:
2929  010714  004737  005232                  JSR     PC,.REWIND              ;REWIND SLAVE
2930  010720  102401                          BVS     99$                     ;BRANCH IF ERROR ON REWIND
2931  010722  102002                          BVC     100$
2932  010724  104400                  99$:    HLT
2933  010726  000772                          BR      A
2934  010730                          100$:
2935
2936                                  ;TEST 014-TURN AROUND DELAY (FORWARD-REVERSE)
2937                                  ;THIS TEST MEASURES TIME FROM 'GO'=1 (READ REVERSE) TO 'ACCL'=0
2938  010730  112737  000014  001122  TST014: MOVB    #14,@#TSTNUM
2939  010736  012702  010770                  MOV     #2$,R2                  ;SET RETURN PC FROM TIMER
2940  010742  004737  005314                  JSR     PC,WRITE                ;WRITE A RECORD
2941  010746  005215                          INC     (R5)                    ;SET 'GO' BIT
2942  010750  004737  005126                  JSR     PC,WAITRDY
2943  010754  102420                          BVS     99$
2944
2945  010756  004737  005350          1$:     JSR     PC,REVRD                ;READ THE RECORD (REVERSE)
2946  010762  004737  00425C                  JSR     PC,TIMON                ;TURN TIMER ON
2947  010766  005215                          INC     (R5)                    ;SET 'GO' BIT
2948
2949  010770  005765  000032          2$:     TST     TC(R5)                  ;WAIT FOR 'ACCL' = 0
2950  010774  100002                          BPL     3$
2951  010776  000163  004340                  JMP     TIMER(R3)               ;GO TO TIMER & RETURN VIA R2
2952
2953  011002  004737  005126          3$:     JSR     PC,WAITRDY
2954  011006  102403                          BVS     99$
2955  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.
001122                          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)
001122                          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$
```

```
3012  011220  104400                  99$:    HLT
3013  011222  104000                  100$:   SCOPE
3014
3015                                  ;TEST 017-GAP SIZE (START HALF)
3016  011224  112737  000017  001122  TST017: MOVB    #17,@#TSTNUM
3017  011232  012702  011304                  MOV     #1$,R2          ;SET RETURN PC FROM TIMER
3018  011236  004737  005314                  JSR     PC,WRITE        ;WRITE A RECORD
3019  011242  005215                          INC     (R5)            ;SET 'GO' BIT
3020  011244  004737  005126                  JSR     PC,WAITRDY      ;WAIT FOR READY
3021  011250  102427                          BVS     99$
3022  011252  004737  005350                  JSR     PC,REVRD        ;READ REVERSE THE RECORD
3023  011256  005215                          INC     (R5)            ;SET 'GO' BIT
3024  011260  004737  005126                  JSR     PC,WAITRDY      ;WAIT FOR READY
3025  011264  102421                          BVS     99$             ;BRANCH ON ERROR
3026  011266  004737  004634                  JSR     PC,DELAY        ;WAIT FOR TAPE MOTION TO STOP
3027  011272  004737  005332                  JSR     PC,READ         ;READ RECORD
3028  011276  004737  004250                  JSR     PC,TIMON        ;TURN TIMER ON
3029  011302  005215                          INC     (R5)            ;SET 'GO' BIT
3030
3031  011304  005710                  1$:     TST     (R0)            ;WAIT FOR FRAME COUNT > 0
3032  011306  001002                          BNE     2$
3033  011310  000163  004340                  JMP     TIMER(R3)       ;GO TO TIMER & RETURN VIA R2
3034
3035  011314  004737  005126          2$:     JSR     PC,WAITRDY      ;WAIT FOR READY
3036  011320  102403                          BVS     99$
3037  011322  004737  004376                  JSR     PC,TIMOK        ;CHECK TIME
3038  011326  000401                          BR      100$
3039
3040  011330  104400                  99$:    HLT
3041  011332  104000                  100$:   SCOPE
3042
3043                                  ;TEST 020- GAP SIZE (INTERRECORD)
3044                                  ;THIS TEST MEASURES TIME FROM 'GO'=1 TO 'FC REG' >0.
3045  011334  112737  000020  001122  TST020: MOVB    #20,@#TSTNUM
3046  011342  012702  011424                  MOV     #1$,R2          ;SET RETURN PC FROM TIMER
3047  011346  004737  005314                  JSR     PC,WRITE        ;WRITE A RECORD
3048  011352  005215                          INC     (R5)            ;SET 'GO' BIT
3049  011354  004737  005126                  JSR     PC,WAITRDY      ;WAIT FOR READY
3050  011360  102433                          BVS     99$
3051  011362  004737  005314                  JSR     PC,WRITE        ;WRITE SECOND RECORD
3052  011366  005215                          INC     (R5)            ;SET 'GO' BIT
3053  011370  004737  005126                  JSR     PC,WAITRDY      ;WAIT FOR READY
3054  011374  102425                          BVS     99$
3055  011376  004737  005350                  JSR     PC,REVRD        ;READ REVERSE SECOND RECORD
3056  011402  005215                          INC     (R5)            ;SET 'GO' BIT
3057  011404  004737  005126                  JSR     PC,WAITRDY      ;WAIT FOR READY
3058  011410  102417                          BVS     99$
3059  011412  004737  005350                  JSR     PC,REVRD        ;READ REVERSE FIRST RECORD
3060  011416  004737  004250                  JSR     PC,TIMON        ;TURN TIMER ON
3061  011422  005215                          INC     (R5)            ;SET 'GO' BIT
3062
3063  011424  005710                  1$:     TST     (R0)            ;WAIT FOR FRAME COUNT > 0
3064  011426  001002                          BNE     2$
3065  011430  000163  004340                  JMP     TIMER(R3)       ;GO TO TIMER & RETURN VIA R2
3066
3067  011434  004737  005126          2$:     JSR     PC,WAITRDY      ;WAIT FOR READY
```

```
 3068  011440  102403                        BVS     99$
 3069  011442  004737  004376                JSR     PC,TIMOK
 3070  011446  000401                        BR      100$
 3071
 3072  011450  104400                99$:    HLT
 3073  011452  104000                100$:   SCOPE
 3074
 3075                                ;TEST 021- GAP CONSISTANCY
 3076                                ;THIS TEST MEASURES TIME FROM 'GO'=1 TO 'FC REG' > 0.
 3077                                ;THE TEST REWINDS THE TAPE,WRITES 17 RECORDS WITH A DELAY FROM 1-16 MS
 3078                                ;BETWEEN EACH WRITE COMMAND. AFTER THE 17. RECORDS ARE WRITTTEN THE
 3079                                ;PROGRAM READ REVERSES 16 RECORDS. AT THIS POINT THE TAPE IS STOPPED BE-
 3080                                ;TWEEN THE FIRST AND SECOND RECORD. A READ COMMAND IS EXECUTED TO READ
 3081                                ;THE 16 RECORDS WITH THE TIME BETEWEN GO=1 TO FC > 0 STORED IN 'GAPTBL'
 3082                                ;FOR EALH RECORD READ. AFTER 16 RECORDS HAVE BEEN READ THE TIME IS VER-
 3083                                ;IFIED FOR EACH READ. AFTER ALL RECORD TIMES ARE VERIFIED THEY ARE AVER-
 3084                                ;AGED AND PLACED IN THE 'ATIMTBL' (BY SCOPE). THE ABOVE PROCESS IS RE-
 3085                                ;PEATED FOR EACH ITERATION.
 3086
 3087  011454  112737  000021  001122  TST021: MOVB    #21,@#TSTNUM
 3088  011462  012702  011620                MOV     #4$,R2          ;SET RETURN PC FROM TIMER
 3089  011466  004737  005232                JSR     PC,.REWIND      ;REWIND SLAVE
 3090  011472  102530                        BVS     99$             ;BRANCH IF ERROR ON REWIND
 3091  011474  005037  001114                CLR     DELTIM          ;CLEAR VARIABLE DELAY TIME
 3092  011500  012700  000021                MOV     #17.,R0         ;SET # OF RECORDS TO WRITE
 3093  011504  004737  005314          1$:   JSR     PC,WRITE        ;WRITE 17. RECORDS
 3094  011510  005215                        INC     (R5)            ;SET 'GO' BIT
 3095  011512  004737  005126                JSR     PC,WAITRDY      ;WAIT FOR READY
 3096  011516  102516                        BVS     99$
 3097  011520  004737  004664                JSR     PC,DELAYV       ;DELAY BEFORE WRITING NEXT REC.
 3098  011524  062737  000022  001114        ADD     #18.,DELTIM     ;SET NEXT DELAY TIME
 3099  011532  005300                        DEC     R0              ;DECREMENT RECORDS WRITTEN COUNT
 3100  011534  001363                        BNE     1$
 3101
 3102  011536  012700  000021                MOV     #17.,R0         ;SET # OF RECS. TO REVERSE READ
 3103  011542  004737  005350          2$:   JSR     PC,REVRD        ;REVERSE READ 17. RECORDS
 3104  011546  005215                        INC     (R5)            ;SET 'GO' BIT
 3105  011550  004737  005126                JSR     PC,WAITRDY      ;WAIT FOR READY
 3106  011554  102477                        BVS     99$
 3107  011556  005300                        DEC     R0              ;DECREMENT RECORD COUNT
 3108  011560  001370                        BNE     2$
 3109
 3110  011562  012700  000020                MOV     #16.,R0         ;SET # OF RECORDS TO READ
 3111  011566  012701  001054                MOV     #GAPTBL,R1      ;SET PTR TO GAP TABLE FOR TEST
 3112  011572  004737  005332                JSR     PC,READ         ;READ A RECORD
 3113  011576  005215                        INC     (R5)            ;SET 'GO' BIT
 3114
 3115  011600  004737  005126          3$:   JSR     PC,WAITRDY      ;WAIT FOR READY
 3116  011604  102463                        BVS     99$
 3117  011606  004737  005332                JSR     PC,READ         ;READ NEXT RECORD
 3118  011612  004737  004250                JSR     PC,TIMON        ;TURN TIMER ON
 3119  011616  005215                        INC     (R5)            ;SET 'GO' BIT
 3120
 3121  011620  005765  000006          4$:   TST     FC(R5)          ;WAIT FOR FRAME COUNT > 0
 3122  011624  001002                        BNE     5$
 3123  011626  000163  004340                JMP     TIMER(R3)       ;GO TO TIMER & RETURN VIA R2
```

```
3124
3125  011632  004737  005126          5$:     JSR     PC,WAITRDY              ;WAIT FOR READY
3126  011636  102446                          BVS     99$
3127  011640  010421                          MOV     R4,(R1)+                ;STORE TIME IN GAPTBL
3128  011642  005300                          DEC     R0                      ;DECREMENT # OF RECORDS READ
3129  011644  001355                          BNE     3$
3130
3131  011646  105037  001120                  CLRB    @#GAP                   ;SET GAP # 0
3132  011652  012700  000020                  MOV     #16.,R0
3133  011656  012701  001054                  MOV     #GAPTBL,R1
3134
3135  011662  012104                  6$:     MOV     (R1)+,R4                ;GET GAP TICK COUNT
3136  011664  004737  004506                  JSR     PC,GAPOK                ;CHECK TIME
3137  011670  105237  001120                  INCB    @#GAP                   ;INCREMENT GAP #
3138  011674  122737  000020  001120          CMPB    #16.,@#GAP              ;BRANCH IF ALL GAPS NOT CHECKED
3139  011702  001367                          BNE     6$
3140
3141  011704  012700  000020                  MOV     #16.,R0                 ;SETUP TO AVERAGE GAP SIZES
3142  011710  012701  001054                  MOV     #GAPTBL,R1              ;SET PTR TO TABLE
3143  011714  005002                          CLR     R2                      ;CLEAR 'SUM' REGISTERS
3144  011716  005003                          CLR     R3
3145  011720  062102                  7$:     ADD     (R1)+,R2                ;ADD ALL GAP SIZES TOGETHER
3146  011722  005503                          ADC     R3
3147  011724  005300                          DEC     R0
3148  011726  001374                          BNE     7$
3149  011730  012700  000004                  MOV     #4,R0                   ;NOW DIVIDE BY 16.
3150  011734  006203                  8$:     ASR     R3                      ;BY SHIFTING 4 PLACES RIGHT
3151  011736  006002                          ROR     R2
3152  011740  005300                          DEC     R0
3153  011742  001374                          BNE     8$
3154  011744  010204                          MOV     R2,R4                   ;MOVE AVERAGED TIMES TO R4
3155  011746  004737  004376                  JSR     PC,TIMOK                ;CHECK AVERAGED TIMES
3156  011752  000401                          BR      100$
3157
3158  011754  104400                  99$:    HLT
3159  011756  104000                  100$:   SCOPE
3160
3161
3162
3163                                  ;TEST 022-DATA TIME (800BPI)
3164                                  ;THIS TEST MEASURES THE TIME FROM FC REG >-6400 TO 'RDY' = 1.
3165  011760  112737  000022  001122  TST022: MOVB    #022,@#TSTNUM
3166  011766  012702  012046                  MOV     #3$,R2                  ;SET RETURN PC FROM TIMER
3167  011772  004737  005232                  JSR     PC,.REWIND              ;REWIND SLAVE
3168  011776  102442                          BVS     99$                     ;BRANCH IF ERROR ON REWIND
3169  012000  052765  001700  000032          BIS     #NORM11,TC(R5)          ;SET 800 BPI
3170  012006  004337  00545C                  JSR     R3,TMCMD                ;WRITE 3200. WORD RECORD
3171  012012  015730                          .WORD   WTBUF
3172  012014  171600                          .WORD   -3200.
3173  012016  163400                          .WORD   -6400.
3174  012020  000060                          .WORD   WFWD
3175  012022  005215                          INC     (R5)                    ;SET 'GO' BIT
3176
3177  012024  022710  163400          1$:     CMP     #-6400.,(R0)            ;WAIT FOR WRITING TO START
3178  012030  001004                          BNE     2$
3179  012032  032711  040000                  BIT     #ERR,(R1)               ;MONITOR ERROR BIT
```

```
 3180  012036  001022                        BNE     99$
 3181  012040  000771                        BR      1$
 3182
 3183  012042                          2$:
 3184  012042  004737  004250                JSR     PC,TIMON                ;TURN TIMER ON
 3185  012046  105711                  3$:   TSTB    (R1)                    ;BRANCH WHEN READY SETS
 3186  012050  100402                        BMI     4$
 3187  012052  000163  004340                JMP     TIMER(R3)               ;GO TO TIMER & RETURN VIA R2
 3188
 3189  012056  012700  000003          4$:   MOV     #3,R0                   ;SET SHIFT COUNT
 3190  012062  006204                  5$:   ASR     R4
 3191  012064  005300                        DEC     R0
 3192  012066  001375                        BNE     5$
 3193  012070  004737  005126                JSR     PC,WAITRDY
 3194  012074  102403                        BVS     99$
 3195  012076  004737  004376                JSR     PC,TIMOK                ;CHECK TIME
 3196  012102  000401                        BR      100$
 3197
 3198  012104  104400                  99$:  HLT
 3199  012106  104000                  100$: SCOPE
 3200
 3201                                  ;TEST 023-DATA TIME (1600BPI)
 3202                                  ;THIS TEST MEASURES THE TIME FROM FC REG >-6400 TO 'RDY' = 1.
 3203  012110  112737  000023  001122  TST023: MOVB   #023,@#TSTNUM
 3204  012116  012702  012204                MOV     #3$,R2                  ;SET RETURN PC FROM TIMER
 3205  012122  004737  005232                JSR     PC,.REWIND              ;REWIND SLAVE
 3206  012126  102442                        BVS     99$                     ;BRANCH IF ERROR ON REWIND
 3207  012130  042765  003700  000032        BIC     #3700,TC(R5)            ;CLEAR CURRENT DENSITY
 3208  012136  052765  002300  000032        BIS     #PE1600,TC(R5)          ;SET 1600 BPI
 3209  012144  004337  005450                JSR     R3,TMCMD                ;WRITE 3200. WORD RECORD
 3210  012150  015730                        .WORD   WTBUF
 3211  012152  171600                        .WORD   -3200.
 3212  012154  163400                        .WORD   -6400.
 3213  012156  000060                        .WORD   WFWD
 3214  012160  005215                        INC     (R5)                    ;SET 'GO' BIT
 3215
 3216  012162  022710  163400          1$:   CMP     #-6400.,(R0)            ;BRANCH WHEN WRITING STARTS
 3217  012166  001004                        BNE     2$
 3218  012170  032711  040000                BIT     #ERR,(R1)               ;MONITOR ERROR BIT
 3219  012174  001017                        BNE     99$
 3220  012176  000771                        BR      1$
 3221
 3222  012200                          2$:
 3223  012200  004737  004250                JSR     PC,TIMON                ;TURN TIMER ON
 3224  012204  105711                  3$:   TSTB    (R1)                    ;BRANCH WHEN READY SETS
 3225  012206  100402                        BMI     4$
 3226  012210  000163  004340                JMP     TIMER(R3)               ;GO TO TIMER & RETURN VIA R2
 3227
 3228  012214  006204                  4$:   ASR     R4                      ;DIVIDE TIME BY 4
 3229  012216  006204                        ASR     R4
 3230  012220  004737  005126                JSR     PC,WAITRDY
 3231  012224  102403                        BVS     99$
 3232  012226  004737  004376                JSR     PC,TIMOK                ;CHECK TIME
 3233  012232  000401                        BR      100$
 3234
 3235  012234  104400                  99$:  HLT
```

```
 3236  012236  104000                  100$:   SCOPE
 3237
 3238                                  ;TEST 024-ERASE
 3239                                  ;THIS TEST MEASURES TIME FROM 'GO'=1 TO 'RDY'=1.
 3240  012240  112737  000024  001122  TST024: MOVB    #24,@#TSTNUM
 3241  012246  012702  012330                  MOV     #2$,R2                  ;SET RETURN PC FROM TIMER
 3242  012252  004737  005232                  JSR     PC,.REWIND              ;REWIND SLAVE
 3243  012256  102436                          BVS     99$                     ;BRANCH IF ERROR ON REWIND
 3244  012260  004737  005070                  JSR     PC,RHINIT               ;SET NRZ
 3245  012264  004737  005314                  JSR     PC,WRITE                ;WRITE A RECORD
 3246  012270  005215                          INC     (R5)                    ;SET 'GO' BIT
 3247  012272  004737  005126                  JSR     PC,WAITRDY
 3248  012276  102426                          BVS     99$
 3249  012300  012737  012306  001002          MOV     #1$,@#SCPADR
 3250  012306  004337  005450          1$:     JSR     R3,@#TMCMD
 3251  012312  000000                          .WORD   0
 3252  012314  000000                          .WORD   0
 3253  012316  000000                          .WORD   0
 3254  012320  000024                          .WORD   ERASE
 3255  012322  004737  004250                  JSR     PC,TIMON                ;TURN TIMER ON
 3256  012326  005215                          INC     (R5)                    ;SET 'GO' BIT
 3257
 3258  012330  105711                  2$:     TSTB    (R1)                    ;BRANCH WHEN READY SETS
 3259  012332  100402                          BMI     3$
 3260  012334  000163  004340                  JMP     TIMER(R3)               ;GO TO TIMER & RETURN VIA R2
 3261
 3262  012340  004737  005126          3$:     JSR     PC,WAITRDY
 3263  012344  102403                          BVS     99$
 3264  012346  004737  004376                  JSR     PC,TIMOK
 3265  012352  000401                          BR      100$
 3266
 3267  012354  104400                  99$:    HLT
 3268  012356  104000                  100$:   SCOPE
 3269
 3270                                  ;TEST 025 TAPE MARK
 3271                                  ;THIS TEST MEASURES TIME FROM 'GO'=1 TO 'RDY'=1.
 3272  012360  112737  000025  001122  TST025: MOVB    #25,@#TSTNUM
 3273  012366  012702  012430                  MOV     #1$,R2                  ;SET RETURN PC FROM TIMER
 3274  012372  004737  005314                  JSR     PC,WRITE                ;WRITE A RECORD
 3275  012376  005215                          INC     (R5)                    ;SET 'GO' BIT
 3276  012400  004737  005126                  JSR     PC,WAITRDY
 3277  012404  102423                          BVS     99$
 3278  012406  004337  005450                  JSR     R3,@#TMCMD
 3279  012412  000000                          .WORD   0
 3280  012414  000000                          .WORD   0
 3281  012416  000000                          .WORD   0
 3282  012420  000026                          .WORD   WFMK
 3283  012422  004737  004250                  JSR     PC,TIMON                ;TURN TIMER ON
 3284  012426  005215                          INC     (R5)                    ;SET 'GO' BIT
 3285
 3286  012430  105711                  1$:     TSTB    (R1)                    ;BRANCH WHEN READY SETS
 3287  012432  100402                          BMI     2$
 3288  012434  000163  004340                  JMP     TIMER(R3)               ;GO TO TIMER & RETURN VIA R2
 3289
 3290  012440  004737  005126          2$:     JSR     PC,WAITRDY
 3291  012444  102403                          BVS     99$
```

```
 3292  012446  004737  004376                  JSR     PC,TIMOK
 3293  012452  000401                          BR      100$
 3294
 3295  012454  104400                  99$:    HLT
 3296  012456                          100$:
 3297  012456  004737  005232                  JSR     PC,.REWIND              ;REWIND SLAVE
 3298  012462  102774                          BVS     99$                     ;BRANCH IF ERROR ON REWIND
 3299  012464  104000                          SCOPE
 3300
```

```
 3301  012466  032777  002000  166304  FINISH: BIT     #SW10,@SWR              ;DO NOT SPACE PAPER
 3302  012474  001011                          BNE     2$                      ;IF USER SELECTED NO OUTPUT
 3303  012476  005737  005716                  TST     CHNFLG                  ;OR IF IN CHAIN MODE
 3304  012502  001006                          BNE     2$
 3305  012504  012700  000012                  MOV     #10.,R0                 ;SET LINE FEED COUNT
 3306  012510  000004  001374          1$:     TYPE,CRLF
 3307  012514  005300                          DEC     R0
 3308  012516  001374                          BNE     1$
 3309
 3310
 3311  012520  105237  001005          2$:     INCB    @#SLVNUM                ;SET NEXT SLAVE #
 3312  012524  005237  001006                  INC     @#SLVPTR                ;AND ITS POINTER
 3313  012530  122737  000010  001005        CMPB    #8.,@#SLVNUM            ;BRANCH IF LAST SLAVE (7)
 3314  012536  001402                          BEQ     3$
 3315  012540  000137  006750                  JMP     @#BEGIN                 ;BEGIN TEST ON NEXT SLAVE
 3316  012544  105037  001005          3$:     CLRB    @#SLVNUM                ;SET SLAVE #0
 3317  012550  105237  001004                  INCB    @#DRVNUM                ;AND INCREMENT DRIVE #
 3318  012554  122737  000010  001004        CMPB    #8.,@#DRVNUM            ;AND CHECK IF LAST DRIVE
 3319  012562  001402                          BEQ     END
 3320  012564  000137  006750                  JMP     @#BEGIN
 3321
 3322  012570  105737  001125          END:    TSTB    @#UNTFND                ;BRANCH IF A UNIT WAS FOUND
 3323  012574  001004                          BNE     1$
 3324  012576  000004  014147                  TYPE,E.UNIT
 3325  012602  000137  005620                  JMP     @#INIT
 3326  012606  105237  001127          1$:     INCB    @#PSCNT         ;INCREMENT PASS COUNT
 3327  012612  000004  013406                  TYPE,M.EOP
 3328  012616  113702  001127                  MOVB    @#PSCNT,R2      ;GET PASSCOUNT
 3329  012622  004737  002426                  JSR     PC,TYPOCT       ;AND TYPE IT
 3330  012626  000004  001374                  TYPE,CRLF
 3331  012632  013700  000042                  MOV     @#42,R0         ;GET ACT11 RETURN ADDRESS
 3332  012636  001405                          BEQ     HERE            ;BRANCH IF NOT ACT11
 3333  012640  000005                          RESET
 3334  012642  004710                  $ENDAD: JSR     PC,(R0)
 3335  012644  000240                          NOP
 3336  012646  000240                          NOP
 3337  012650  000240                          NOP
 3338  012652  000240                  HERE:   NOP
 3339  012654  005737  005716                  TST     CHNFLG          ;BRANCH IF CHAIN MODE
 3340  012660  001004                          BNE     1$
 3341  012662  032777  000100  166110        BIT     #SW06,@SWR              ;BRANCH IF NOT CONTINOUS LOOP
 3342  012670  001402                          BEQ     2$
 3343  012672  000137  006726          1$:     JMP     @#RSTRT                 ;RESTART
 3344  012676  000000                  2$:     HALT
 3345  012700  000005                          RESET
 3346  012702  000137  005620                  JMP     @#INIT                  ;RESTART
```

```
 3347                                   ;SKEW TAPE TIMING TESTS
 3348                                   ;THE FOLLOWING TESTS REQUIRE A SPECIALLY WRITTEN 800 BPI SKEW TAPE
 3349  012706  012737  012714  001002   SKEWTST:MOV    #TST026,@#SCPADR        ;SET SCOPE POINTER
 3350
 3351                                   ;TEST 026- SKEW TAPE SPEED TEST-FORWARD
 3352                                   ;THIS TEST READS 32" OF TAPE (26400.-800. = 25600. FRAMES),THEN
 3353                                   ;DIVIDES TIME BY 32. TO GET TIME TO READ 1" (800. FRAMES) OF TAPE.
 3354  012714  112737  000026  001122   TST026: MOVB   #26,@#TSTNUM
 3355  012722  012702  013000                   MOV    #2$,R2                  ;SET RETURN PC FROM TIMER
 3356  012726  004737  005232                   JSR    PC,.REWIND              ;REWIND SLAVE
 3357  012732  102441                           BVS    99$                     ;BRANCH IF ERROR ON REWIND
 3358  012734  052765  001700  000032         BIS    #NORM11,TC(R5)  ;SET 800 BPI
 3359  012742  052765  000010  000010         BIS    #BAI,CS2(R5)            ;INHIBIT BUS ADDRESS INCREMENT
 3360  012750  004337  005450                   JSR    R3,@#TMCMD              ;READ 32" OF TAPE-FORWARD
 3361  012754  015730                           .WORD  RDBUF
 3362  012756  177777                           .WORD  -1.
 3363  012760  063440                 10$:      .WORD  26400.                  ;FRAME COUNT
 3364  012762  000070                           .WORD  RDFWD
 3365  012764  005215                           INC    (R5)                    ;SET 'GO' BIT
 3366
 3367  012766  022710  001440          1$:      CMP    #800.,(R0)              ;WAIT FOR FIRST 800 FRAMES
 3368  012772  101375                           BHI    1$                      ;TO BE READ
 3369
 3370  012774  004737  004250                   JSR    PC,TIMON                ;TURN TIMER ON
 3371  013000  023710  012760          2$:      CMP    @#10$,(R0)              ;WAIT FOR READING TO FINISH
 3372  013004  103402                           BLO    3$
 3373  013006  000163  004340                   JMP    TIMER(R3)               ;GO TO TIMER & RETURN VIA R2
 3374
 3375  013012  012700  000005          3$:      MOV    #5,R0                   ;DIVIDE TIME BY 32.
 3376  013016  006204                  4$:      ASR    R4
 3377  013020  005300                           DEC    R0
 3378  013022  001375                           BNE    4$
 3379  013024  004737  005070                   JSR    PC,RHINIT               ;INIT DRIVE
 3380  013030  004737  004376                   JSR    PC,TIMOK                ;CHECK TIME
 3381  013034  000401                           BR     100$
 3382
 3383  013036  104400                 99$:      HLT
 3384  013040  104000                 100$:     SCOPE
 3385
 3386                                   ;TEST 027-SKEW TAPE SPEED TEST-REVERSE
 3387                                   ;THIS TEST READS FORWARD 40" (32000. FRAMES) OF TAPE, THEN READS REVERSE
 3388                                   ;32" (26400.-800. = 25600. FRAMES) OF TAPE. THE TIME IS THEN DIVIDED BY
 3389                                   ;32. TO GET TIME TO READ 1" (800. FRAMES) OF TAPE.
 3390  013042  112737  000027  001122   TST027: MOVB   #27,@#TSTNUM
 3391  013050  012702  013176                   MOV    #3$,R2                  ;SET RETURN PC FROM TIMER
 3392  013054  004737  005232                   JSR    PC,.REWIND              ;REWIND SLAVE
 3393  013060  102465                           BVS    99$                     ;BRANCH IF ERROR ON REWIND
 3394  013062  052765  001700  000032         BIS    #NORM11,TC(R5)
 3395  013070  052765  000010  000010         BIS    #BAI,CS2(R5)
 3396  013076  004337  005450                   JSR    R3,@#TMCMD              ;READ FORWARD 32000. FRAMES
 3397  013102  015730                           .WORD  RDBUF
 3398  013104  177777                           .WORD  -1.                     ;WORD COUNT
 3399  013106  076400                 10$:      .WORD  32000.                  ;FRAME COUNT
 3400  013110  000070                           .WORD  RDFWD                   ;READ FORWARD
 3401  013112  005215                           INC    (R5)                    ;SET 'GO' BIT
 3402
```

```
3403  013114  023710  013106          1$:     CMP     @#10$,(R0)
3404  013120  101375                  BHI     1$
3405
3406  013122  004737  005070          JSR     PC,RHINIT               ;INIT DRIVE
3407  013126  004737  004634          JSR     PC,DELAY                ;WAIT FOR TAPE MOTION TO STOP
3408  013132  052765  001700  000032  BIS     #NORM11,TC(R5)          ;SET 800 BPI
3409  013140  052765  000010  000010  BIS     #BAI,CS2(R5)            ;INHIBIT BUS ADDRESS INCRENENT
3410  013146  004337  005450          JSR     R3,@#TMCMD              ;READ REVERSE 32'' OF TAPE
3411  013152  015730                  .WORD   RDBUF                   ;READ BUFFER
3412  013154  177777                  .WORD   -1.                     ;WORD COUNT
3413  013156  063440          11$:    .WORD   26400.                  ;FRAME COUNT
3414  013160  000076                  .WORD   RDREV                   ;READ REVERSE
3415  013162  005215                  INC     (R5)                    ;SET 'GO' BIT
3416
3417  013164  022710  001440          2$:     CMP     #800.,(R0)              ;WAIT FOR FIRST 800 FRAMES
3418  013170  101375                  BHI     2$                      ;TO BE READ
3419
3420  013172  004737  004250          JSR     PC,TIMON                ;TURN TIMER ON
3421  013176  023710  013156          3$:     CMP     @#11$,(R0)              ;WAIT FOR ALL FRAMES TO BE READ
3422  013202  103402                  BLO     4$
3423  013204  000163  004340          JMP     TIMER(R3)               ;GO TO TIMER & RETURN VIA R2
3424
3425  013210  012700  000005          4$:     MOV     #5,R0                   ;DIVIDE TIME BY 32.
3426  013214  006204                  5$:     ASR     R4
3427  013216  005300                  DEC     R0
3428  013220  001375                  BNE     5$
3429  013222  004737  005070          JSR     PC,RHINIT
3430  013226  004737  004376          JSR     PC,TIMOK
3431  013232  000401                  BR      100$
3432
3433  013234  104400          99$:    HLT
3434  013236                  100$:
3435  013236  004737  005232          JSR     PC,.REWIND              ;REWIND SLAVE
3436  013242  102774                  BVS     99$                     ;BRANCH IF ERROR ON REWIND
3437  013244  104000                  SCOPE
3438
3439  013246  000137  012466          JMP     @#FINISH
3440
3441
3442
```

```
                                                        .SBTTL    PROGRAM MESSAGES
                                                ;OPERATOR INSTRUCTIONS
013252  005015  046524  031460  M.NAM:  .ASCII  <CR><LF>'TM03/TU45 DRIVE FUNCTION TIMER (CZTUSA0)'
013260  052057  032125  020065
013266  051104  053111  020105
013274  052506  041516  044524
013302  047117  052040  046511
013310  051105  024040  055103
013316  052524  040523  024460
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 TU45 TO BE TESTED'
013462  053117  020105  046524
013470  050104  043040  047522
013476  020115  052524  032464
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  %0- 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
```

```
 3499  013722     000
 3500  013723     123  042520  042105  I.SPD:  .ASCIZ  'SPEED TESTS? (YES/NO): NO '
 3501  013730  052040  051505  051524
 3502  013736  020077  054450  051505
 3503  013744  047057  024517  020072
 3504  013752  047516  000040
 3505  013756  005015  047105  020104  M.EOT:  .ASCIZ  <CR><LF>'END OF TAPE'<CR><LF>
 3506  013764  043117  052040  050101
 3507  013772  006505  000012
 3508
 3509                                  ;ERROR MESSAGES
 3510  013776  005015  051124  050101  E.TRP4: .ASCIZ  <CR><LF>'TRAPPED TO 4'
 3511  014004  042520  020104  047524
 3512  014012  032040     000
 3513  014015     116  020117  047503  E.NCON: .ASCIZ  'NO CONTROLLER AT ADDRESS SPECIFIED'<CR><LF>
 3514  014022  052116  047522  046114
 3515  014030  051105  040440  020124
 3516  014036  042101  051104  051505
 3517  014044  020123  050123  041505
 3518  014052  043111  042511  006504
 3519  014060  000012
 3520  014062  046524  031460  042040  E.NDRV: .ASCIZ  'TM03 DRIVE '
 3521  014070  044522  042526  000040
 3522  014076  051104  053111  020105  E.NSLV: .ASCII  'DRIVE '
 3523  014104  020060  046123  053101  E.DRV:  .ASCII  '0 SLAVE '
 3524  014112  020105
 3525  014114  020060  047516  020124  E.NAVA: .ASCIZ  '0 NOT AVAILABLE FOR TEST'<CR><LF>
 3526  014122  053101  044501  040514
 3527  014130  046102  020105  047506
 3528  014136  020122  042524  052123
 3529  014144  005015     000
 3530  014147     116  020117  046524  E.UNIT: .ASCIZ  'NO TM03/TU45 UNIT FOUND TO TEST'<CR><LF>
 3531  014154  031460  052057  032125
 3532  014162  020065  047125  052111
 3533  014170  043040  052517  042116
 3534  014176  052040  020117  042524
 3535  014204  052123  005015     000
 3536  014211     123  043117  020124  E.SFT:  .ASCIZ  'SOFT ERROR (DATA)'<CR><LF>
 3537  014216  051105  047522  020122
 3538  014224  042050  052101  024501
 3539  014232  005015     000
 3540  014235     124  051505  020124  E.HDR:  .ASCIZ  'TEST # '
 3541  014242  020043     000
 3542  014245     040  042504  044526  E.HDR1: .ASCII  ' DEVICE ERROR'<CR><LF>
 3543  014252  042503  042440  051122
 3544  014260  051117  005015
 3545  014264  051503  004461  041527          .ASCIZ  'CS1'<HT>'WC'<HT>'BA'<HT>'FC'<HT>'CS2'<HT>'DS'<HT>'ER'<HT>'TC'<CR><LF>
 3546  014272  041011  004501  041506
 3547  014300  041411  031123  042011
 3548  014306  004523  051105  052011
 3549  014314  006503  000012
 3550  014320  047440  052125  047440  E.HDR2: .ASCIZ  ' OUT OF RANGE ERROR'<CR><LF>
 3551  014326  020106  040522  043516
 3552  014334  020105  051105  047522
 3553  014342  006522  000012
 3554  014346  005015  044524  042515  E.TIMOV:        .ASCIZ  <CR><LF>'TIMER OVERFLOWED'<CR><LF>
```

```
 3555  014354  020122  053117  051105
 3556  014362  046106  053517  042105
 3557  014370  005015     000
 3558  014373     015  052012  046511  E.TIMEX:        .ASCIZ  <CR><LF>'TIME EXPIRED WAITING FOR RDY'<CR><LF>
 3559  014400  020105  054105  044520
 3560  014406  042522  020104  040527
 3561  014414  052111  047111  020107
 3562  014422  047506  020122  042122
 3563  014430  006531  000012
 3564  014434  043440  050101  021440  E.GAP:  .ASCIZ  ' GAP # '
 3565  014442  000040
 3566
 3567                                  ;TIME DOCUMENT LINES
 3568  014444  025052  025052  025052  L.HDR1: .ASCIZ  '*********************************************************************************
 3569  014452  025052  025052  025052
 3570  014460  025052  025052  025052
 3571  014466  025052  025052  025052
 3572  014474  025052  025052  025052
 3573  014502  025052  025052  025052
 3574  014510  025052  025052  025052
 3575  014516  025052  025052  025052
 3576  014524  025052  025052  025052
 3577  014532  025052  025052  025052
 3578  014540  025052  025052  025052
 3579  014546  025052  025052  025052
 3580  014554  005015     000
 3581  014557     052  052040  030115  L.HDR2: .ASCII  '* TM03 DRIVE FUNCTION TIMES- DRIVE # '
 3582  014564  020063  051104  053111
 3583  014572  020105  052506  041516
 3584  014600  044524  047117  052040
 3585  014606  046511  051505  020055
 3586  014614  051104  053111  020105
 3587  014622  020043
 3588  014624  020060  046123  053101  L.DRV:  .ASCII  '0 SLAVE # '
 3589  014632  020105  020043
 3590  014636  020060     040          L.SLV:  .ASCII  '0  '
 3591  014641     071  041440  040510  L.CHAN: .ASCIZ  '9 CHAN. SER # '
 3592  014646  027116  051440  051105
 3593  014654  021440  000040
 3594  014660  006440  025012  005015  L.HDR3: .ASCII  ' '<CR><LF>'*'<CR><LF>
 3595  014666  020052  052506  041516          .ASCIZ  '* FUNCTION'<HT><HT>'TIME(SPECIFICATION)'<HT>'TIME(ACTUAL)'<CR><LF>
 3596  014674  044524  047117  004411
 3597  014702  044524  042515  051450
 3598  014710  042520  044503  044506
 3599  014716  040503  044524  047117
 3600  014724  004451  044524  042515
 3601  014732  040450  052103  040525
 3602  014740  024514  005015     000
 3603
 3604  014745     122  047101  042507  L.RNG:  .ASCIZ  'RANGE=<'
 3605  014752  036075     000
 3606  014755     101  052103  040525  L.ACT:  .ASCIZ  'ACTUAL='
 3607  014762  036514     000
 3608
 3609                                  ;TEST DESCRIPTOR HEADERS
 3610  014765     052  053440  044522  A.T001: .ASCIZ  '* WRITE FROM BOT'<HT>
```

```
 3611  014772  042524  043040  047522
 3612  015000  020115  047502  004524
 3613  015006     000
 3614  015007     052  053440  044522  A.T002: .ASCIZ  '* WRITE START'<HT><HT>
 3615  015014  042524  051440  040524
 3616  015022  052122  004411     000
 3617  015027     052  053440  044522  A.T003: .ASCIZ  '* WRITE SHUTDOWN'<HT>
 3618  015034  042524  051440  052510
 3619  015042  042124  053517  004516
 3620  015050     000
 3621  015051     052  053440  044522  A.T004: .ASCIZ  '* WRITE SETTLEDOWN'<HT>
 3622  015056  042524  051440  052105
 3623  015064  046124  042105  053517
 3624  015072  004516     000
 3625  015075     052  051040  040505  A.T005: .ASCIZ  '* READ FROM BOT'<HT><HT>
 3626  015102  020104  051106  046517
 3627  015110  041040  052117  004411
 3628  015116     000
 3629  015117     052  051040  040505  A.T006: .ASCIZ  '* READ START'<HT><HT>
 3630  015124  020104  052123  051101
 3631  015132  004524  000011
 3632  015136  020052  042522  042101  A.T007: .ASCIZ  '* READ SHUTDOWN'<HT><HT>
 3633  015144  051440  052510  042124
 3634  015152  053517  004516  000011
 3635  015160  020052  042522  042101  A.T010: .ASCIZ  '* READ SETTLEDOWN'<HT>
 3636  015166  051440  052105  046124
 3637  015174  042105  053517  004516
 3638  015202     000
 3639  015203     052  051040  040505  A.T011: .ASCIZ  '* READ REV START'<HT>
 3640  015210  020104  042522  020126
 3641  015216  052123  051101  004524
 3642  015224     000
 3643  015225     052  051040  040505  A.T012: .ASCIZ  '* READ REV SHUTDOWN'<HT>
 3644  015232  020104  042522  020126
 3645  015240  044123  052125  047504
 3646  015246  047127  000011
 3647  015252  020052  042522  042101  A.T013: .ASCIZ  '* READ REV SETTLEDOWN'<HT>
 3648  015260  051040  053105  051440
 3649  015266  052105  046124  042105
 3650  015274  053517  004516     000
 3651  015301     052  052040  051125  A.T014: .ASCIZ  '* TURN AROUND DELAY F-R'<HT>
 3652  015306  020116  051101  052517
 3653  015314  042116  042040  046105
 3654  015322  054501  043040  051055
 3655  015330  000011
 3656  015332  020052  052524  047122  A.T015: .ASCIZ  '* TURN AROUND DELAY R-F'<HT>
 3657  015340  040440  047522  047125
 3658  015346  020104  042504  040514
 3659  015354  020131  026522  004506
 3660  015362     000
 3661  015363     052  043440  050101  A.T016: .ASCIZ  '* GAP SIZE-STOP HALF'<HT>
 3662  015370  051440  055111  026505
 3663  015376  052123  050117  044040
 3664  015404  046101  004506     000
 3665  015411     052  043440  050101  A.T017: .ASCIZ  '* GAP SIZE-START HALF'<HT>
 3666  015416  051440  055111  026505
```

```
 3667  015424  052123  051101  020124
 3668  015432  040510  043114  000011
 3669  015440  020052  040507  020120  A.T020: .ASCIZ  '* GAP SIZE-INTERRECORD'<HT>
 3670  015446  044523  042532  044455
 3671  015454  052116  051105  042522
 3672  015462  047503  042122  000011
 3673  015470  020052  040507  020120  A.T021: .ASCIZ  '* GAP CONSISTANCY'<HT>
 3674  015476  047503  051516  051511
 3675  015504  040524  041516  004531
 3676  015512     000
 3677  015513     052  042040  052101  A.T022: .ASCIZ  '* DATA TIME-800BPI'<HT>
 3678  015520  020101  044524  042515
 3679  015526  034055  030060  050102
 3680  015534  004511     000
 3681  015537     052  042040  052101  A.T023: .ASCIZ  '* DATA TIME-1600BPI'<HT>
 3682  015544  020101  044524  042515
 3683  015552  030455  030066  041060
 3684  015560  044520  000011
 3685  015564  020052  051105  051501  A.T024: .ASCIZ  '* ERASE GAP TIME'<HT>
 3686  015572  020105  040507  020120
 3687  015600  044524  042515  000011
 3688  015606  020052  051127  052111  A.T025: .ASCIZ  '* WRITE FILE MARK'<HT>
 3689  015614  020105  044506  042514
 3690  015622  046440  051101  004513
 3691  015630     000
 3692  015631     052  052040  050101  A.T026: .ASCIZ  '* TAPE SPEED-FWD'<HT>
 3693  015636  020105  050123  042505
 3694  015644  026504  053506  004504
 3695  015652     000
 3696  015653     052  052040  050101  A.T027: .ASCIZ  '* TAPE SPEED-REV'<HT>
 3697  015660  020105  050123  042505
 3698  015666  026504  042522  004526
 3699  015674     000
 3700
 3701  015675     015  057012  000107  L.CNTG: .ASCIZ  <CR><LF>' G'
 3702  015702  005015  053523  036522  L.SWR:  .ASCIZ  <CR><LF>'SWR='
 3703  015710     000
 3704  015711     040  047040  053505  L.NEW:  .ASCIZ  '  NEW= '
 3705  015716  020075     000
 3706  015721     015  037412  005015  L.QUEST:        .ASCIZ  <CR><LF>'?'<CR><LF>
 3707  015726     000
 3708          015730                          .EVEN
 3709          015730                          RDBUF=.
 3710          015730                          WTBUF=.
 3711  015730  000200                          .BLKW   128.
 3712          000001                          .END
```

```
A       010714          CNTRLU= 000025          E.HDR2  014320          L.HDR3  014660          READ    005332
ACCL  = 100000          CNVDEC  002526          E.NAVA  014114          L.NEW   015711          RESVEC= 000010
ANGTAB  001412          CNVOCT  002420          E.NCON  014015          L.QUES  015721          REVRD   005350
AS    = 000016          CNVTAO  003056          E.NDRV  014062          L.RNG   014745          RHINIT  005070
ASFLG   001130          CNVTD   002540          E.NSLV  014076          L.SLV   014636          RMR   = 000004
ATA   = 100000          CNVTO   002432          E.SFT   014211          L.SWR   015702          RSTRT   006726
ATIME   001012          CR    = 000015          E.TIME  014373          MCPE  = 020000          RWD   = 000006
ATIMTB  001014          CRLF    001374          E.TIMO  014346          MDPE  = 000400          RWDOFF= 000002
A.T001  014765          CSITM = 002000          E.TRP4  013776          MMVEC = 000250          R10   =%000000
A.T002  015007          CS1   = 000000          E.UNIT  014147          MOL   = 010000          R11   =%000001
A.T003  015027          CS2   = 000010          FC    = 000006          MR    = 000024          R12   =%000002
A.T004  015051          DASH    001405          FCE   = 001000          MXF   = 001000          R13   =%000003
A.T005  015075          DB    = 000022          FINISH  012466          M.EOP   013406          R14   =%000004
A.T006  015117          DCONST  002634          FMT   = 000020          M.EOT   013756          R15   =%000005
A.T007  015136          DELAY   004634          FPEVEC= 000244          M.HSWR  013425          SC    = 100000
A.T010  015160          DELAYV  004664          FRMCNT= 177400          M.NAM   013252          SCOPE = 104000
A.T011  015203          DELTIM  001114          FWDSPC  005366          NAMPTR  001656          SCPADR  001002
A.T012  015225          DIGTAB  001132          GAP     001120          NED   = 010000          SDWN  = 000020
A.T013  015252          DIVIDE  004722          GAPOK   004506          NEF   = 004000          SKEWTS  012706
A.T014  015301          DLT   = 100000          GAPTBL  001054          NEM   = 004000          SLA   = 000001
A.T015  015332          DPR   = 000400          GO    = 000001          NOP   = 000000          SLAVES  006350
A.T016  015363          DRIVES  006104          GTIMTB  001556          NORM11= 001700          SLR   = 177774
A.T017  015411          DRVAVA  005012          GTSWR   002020          NSG   = 000400          SLVAVA  005040
A.T020  015440          DRVNUM  001004          HERE    012652          OCTAL0  001116          SLVNUM  001005
A.T021  015470          DRVTBL  001154          HLT   = 104400          ODIGIT  001144          SLVPTR  001006
A.T022  015513          DRY   = 000200          HT    = 000011          OPI   = 020000          SLVTBL  001164
A.T023  015537          DRYCLR= 000010          IDB   = 000010          OR    = 000200          SN    = 000030
A.T024  015564          DS    = 000012          IE    = 000100          OSC   = 000100          SNPT    005470
A.T025  015606          DT    = 000026          ILF   = 000001          OUTBUF= 005620          SPACE   001410
A.T026  015631          DTE   = 010000          ILR   = 000002          OUTGAP  002744          SPACE2  001407
A.T027  015653          DVA   = 004000          INBUF   001264          OUTSPC  002650          SPCFWD= 000030
A16   = 000400          DV0   = 000000          INCVAE= 000100          PARVEC= 000114          SPCREV= 000032
A17   = 001000          DV1   = 000001          INIT    005620          PAT   = 000020          SPR   = 002000
BA    = 000004          DV2   = 000002          IOTVEC= 000020          PEFLRC= 000200          SSC   = 000100
BAI   = 000010          DV3   = 000003          IR    = 000100          PES   = 000040          STIMTB  001416
BEGIN   006750          DV4   = 000004          ITCNT   001121          PE1600= 002300          STKPTR= 000600
BELL    001403          DV5   = 000005          I.DRV   013656          PFVEC = 000024          SWR     001000
BKSLSH  001377          DV6   = 000006          I.DRVS  013567          PGE   = 002000          SWREG   000176
BOT   = 000002          DV7   = 000007          I.REG   013522          PIP   = 020000          SW06  = 000100
BPI200= 000000          ECHO    001401          I.REM   013455          PIRQ  = 177772          SW07  = 000200
BPI556= 000400          EMTVEC= 000030          I.SLVS  013637          PIRVEC= 000240          SW08  = 000400
BPI800= 001000          END     012570          I.SPD   013723          PLKCSR= 172540          SW09  = 001000
BPTVEC= 000014          EOT   = 002000          LF    = 000012          PLKVEC= 000104          SW10  = 002000
CDM11 = 000320          ER    = 000014          LKS   = 177546          PRGFLG  001124          SW11  = 004000
CHKDRV  006250          ERASE = 000024          LKVEC = 000100          PSCNT   001127          SW13  = 020000
CHKSLV  006560          ERFLG   001123          LPB   = 177516          PSEL  = 002000          SW14  = 040000
CHNFLG  005716          ERR   = 040000          LPS   = 177514          PSW   = 177776          SW15  = 100000
CH7   = 010000          ERRTRP  003542          L.ACT   014755          PUBLIS  003144          TAP   = 040000
CLR   = 000040          ERRVEC= 000004          L.CHAN  014641          RDBUF = 015730          TBITVE= 000014
CNTRLA= 000001          E.DRV   014104          L.CNTG  015675          RDFWD = 000070          TC    = 000032
CNTRLC= 000003          E.GAP   014434          L.DRV   014624          RDREV = 000076          TCRLF   002156
CNTRLG= 000007          E.HDR   014235          L.HDR1  014444          RDSW    002126          TIB     002016
CNTRLO= 000017          E.HDR1  014245          L.HDR2  014557          RDY   = 000200          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

 ,CZTUSA.SEQ/SOL_CZTUSA.P11
 RUN-TIME: 31 47 2 SECONDS
 RUN-TIME RATIO: 139/81=1.7
 CORE USED:  14K  (28 PAGES)
```