TJU16

DRIVE FUNCTION TIMER

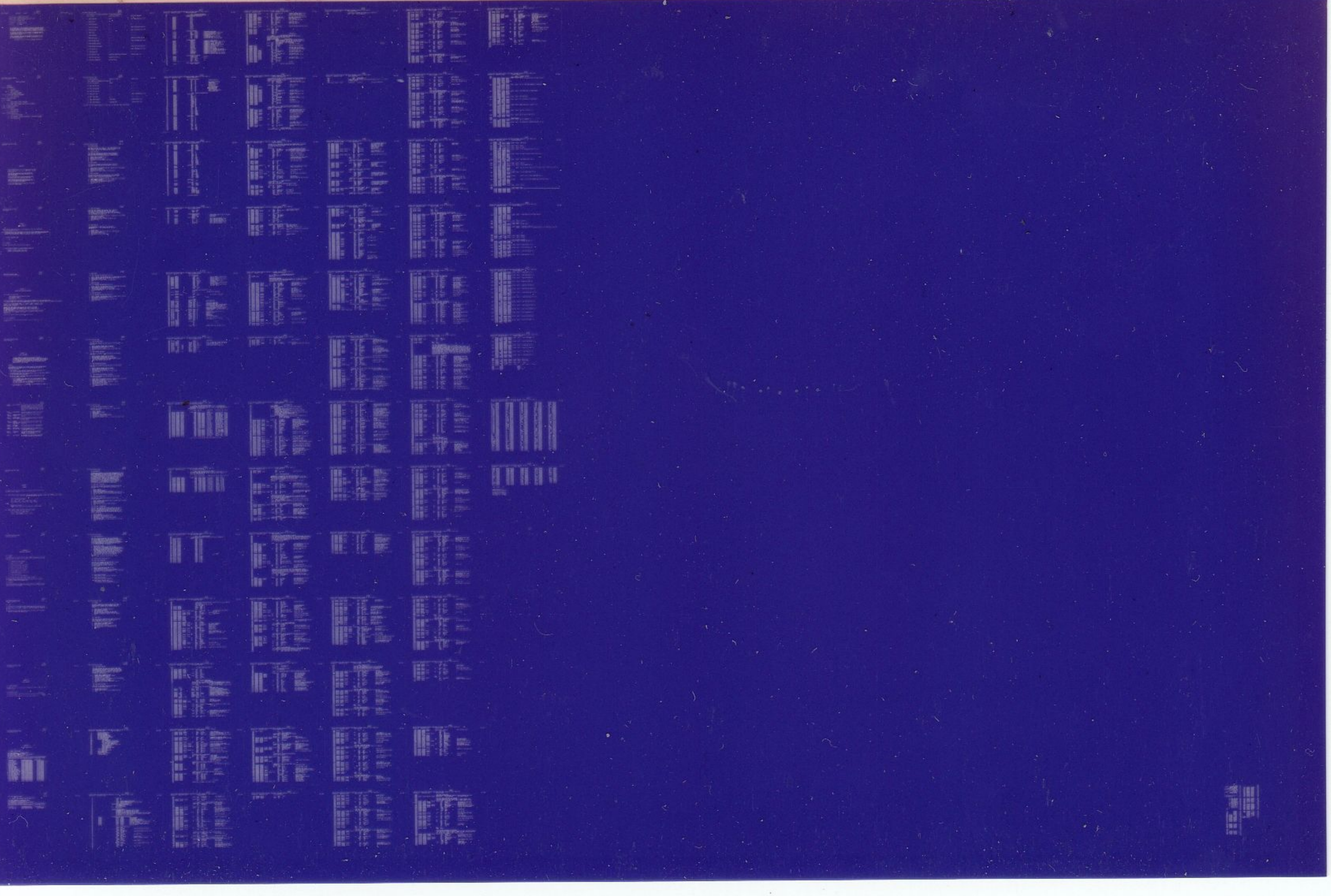
MD-11-DZTUG-A

EP DZTUG A DL A

FICHE 1 OF 1

MAR 1977

MADE IN USA



Product Code: MAINDEC-11-DZTUG-A-D

Product Name: TM02/TU16J DRIVE FUNCTION TIMER

Date Created: JANUARY 1977

Maintainer: Diagnostic Group

Author: R. BARNES

TABLE OF CONTENTS

ABSTRACT

Program DZTUG measures the time required and GAP sizes produced by the TM02/TU16J Magtape Drive/Slave.

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

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

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

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

CHAPTER 1
REQUIREMENTS

PDP-11 Family Central Processor with 4K memory with up to 64 TM02/TU16J controller/magtape stations.

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

1.1 OPTIONAL EQUIPMENT USED

1. none

1.2 STORAGE

Program loads and runs in the first 4K of memory.

1.3 PRELIMINARY PROGRAMS (TO ASSURE HARDWARE OPERATION)

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

CHAPTER 2

LOADING AND STARTING PROCEDURE

The procedure is as follows:

Load program using the Absolute Loader
Load address = 200
Set operating switches
Press start

***IF THE SOFTWARE SWITCH REGISTER IS USED THEN THE PROGRAM WILL TYPE SWR=XXXXXX NEW=
THIS WILL ALLOW LOC. 176 TO BE CHANGED BEFORE THE START OF THE TESTING.(REFER TO CHAPTER 3 FOR OPTIONS)

Program will request DRIVE (TM02) and SLAVE (TU16J) numbers to be tested. Type DRIVE/SLAVE numbers with a comma (,) between each DRIVE/SLAVE to be tested.

REQUESTS FOR TAPE SPEED TESTS AND NRZ ONLY MODE WILL BE MADE.
RESPONSE TO TAPE SPEED ONLY REQUEST WITH A ONE (1) WILL CAUSE
THE PROGRAM TO EXECUTE TEST 31 AND 32 ONLY. THIS IS THE ONLY WAY
TO TEST TAPE SPEED.
NRZ ONLY MODE WILL CAUSE THE PROGRAM TO SKIP THE 1600 BPI DATA TIME TEST.
Type Control U (↑U) to delete line typed or rubout to delete last character(s).

Program will publish times required and report errors.

2.1 ACT11 OPERATION

If the program is run in Quick Verify Mode, function tests are not iterated.

CHAPTER 3

SWITCH SETTINGS

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

CONTROL:

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

1) TYPE CONTROL G <↑G>; THIS WILL ALLOW THE TTY TO ENTER DATA INTO LOC. 176 AT SELECTED POINTS WITHIN THE PROGRAM.

2) THE MACHINE WILL THEN TYPE: SWR=XXXXXXNEW= (XXXXXX IS THE OCTAL CONTENTS OF THE SOFTWARE SWITCH REGISTER.)

3) AFTER THE ''NEW='' HAS BEEN TYPED THEN THE OPERATOR CAN DO ONE OF THE FOLLOWING AT THE TTY:

   A) TYPE A NUMBER TO BE LOADED INTO LOC. 176 FOLLOWED BY A <CR>. (ONLY NUMBERS BETWEEN 0-7 WILL BE ACCEPTED AND ONLY 6 NUMBERS WILL BE ALLOWED)
   IF A <CR> IS THE FIRST KEY DEPRESSED THE SOFTWARE SWITCH REGISTER CONTENTS WILL NOT BE CHANGED.

   B) IF A CONTROL U <↑U> IS DEPRESSED THEN THE PROGRAM WILL SEND YOU BACK TO STEP 2.

SW15 (100000)	HALT ON ERROR	This switch when set will halt the processor when an error is detected. The PC+2 and PSW at the time of the error is stored on the stack. Pressing continue will cause the error to be typed (if selected) and further testing resumed.
SW14 (040000)	LOOP SUBTEST	This switch when set loops the current subtest regardless of error condition.
SW13 (020000)	INHIBIT ERROR TYPEOUT	This switch when set inhibits error typeout.
SW11 (004000)	INHIBIT SUB-TEST ITERATION	This switch when set causes each subtest to be executed only once. (Initial Startup Only).
SW10 (002000)	INHIBIT FUNCTION TIME PUBLICATION	This switch when set will inhibit the printing of the function times. (See Chapter 8.)
SW09 (001000)	RING BELL ON ERROR	This switch when set will ring the bell on the TTY when an error is detected.
SW07 (000200)	HALT AFTER SELECTED TEST	This switch when set will cause the program to HALT after the test selected in SW05-SW00 is executed.
SW06 (000100)	CONTINUOUS CYCLE	THIS SWITCH WHEN SET WILL CAUSE THE PROGRAM TO RUN CONTINUOUSLY UNTIL STOPPED BY THE OPERATOR.
SW5-0	TEST SELECT	THE PROGRAM WILL HALT AFTER EXECUTION OF THE TEST SELECTED WHEN SW07 IS SET.

CHAPTER 4

ERRORS

Two types of errors are detected by this program, hardware errors and incorect function times.

4.1 ERROR TYPEOUT FORMAT (HARDWARE): DATA RELATED ERRORS (IE: PARITY ERROR) ARE PRINTED 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>
3. Provides continuous loop <SW14>
4. Moves function time into table
5. Outputs Line Item if selected
6. Provides HALT on test <SW07>
7. Delays 350MS before starting test
8. Init's Drive/Slave
9. Clears the error flag (ERFLG)

The routine monitors SW14, SW11, SW10, SW08, and SW07.

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

5.2 PUBLISH

The Publish Routine is called from the Scope Routine if SW10 is equal to 0 (Publish Time Document). The routine will print a "SINGLE LINE ITEM" each time it is called.

5.3 .HLT

The HLT Routine is called by the HLT (Trap) instruction when an error is detected. A HLT (TRAP) instruction formats the error information as shown in Sec 4.1. A HLT+1 (TRAP+1) formats tHE ERROR AS SHOWN IN SEC 4.2.

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

CHAPTER 6
MISCELLANEOUS

6.1 STACK POINTER

The Stack Pointer is initailly set to 500 and is reset to 500 by the SCOPEA Routine.

6.2 EXECUTION TIME

When SW11=1 (Inhibit Iterations) the time required is 2 min.

When SW11=0 (Iterate Subtests) the time required is 9 min.

CHAPTER 7

PROGRAM DESCRIPTION

7.1 SAMPLE TIME DOCUMENT

```
TYPE FIRST ADDRESS OF CONTROLER 172440
TYPE TM02 DRIVE #'S TO BE TESTED 0
FOR TM02 DRIVE 0- TYPE SLAVE #'S TO BE TESTED 7
TAPE SPEED TESTS ONLY? (YES/NO = 1/0) 0
NRZ ONLY? (YES/NO = 1/0) 0
*****************************************************************
* TM02 DRIVE FUNCTION TIMES- DRIVE # 0 SLAVE # 7 9 CHAN. SER. # 5009
*
* FUNCTION                TIME(SPECIFICATION)     TIME(ACTUAL)
* WRITE FROM BOT          RANGE=(176000-172000)       ACTUAL=174740
* WRITE START             RANGE=(009500-008700)       ACTUAL=009120
* WRITE SHUTDOWN          RANGE=(008900-008500)       ACTUAL=008840
* WRITE SETTLEDOWN        RANGE=(013500-008100)       ACTUAL=010970
* READ FROM BOT           RANGE=(045000-041000)       ACTUAL=043580
* READ START              RANGE=(003200-002600)       ACTUAL=002740
* READ SHUTDOWN           RANGE=(004650-004250)       ACTUAL=004360
* READ SETTLEDOWN         RANGE=(013500-008100)       ACTUAL=010970
* READ REV START          RANGE=(003200-002600)       ACTUAL=002740
* READ REV SHUTDOWN       RANGE=(003700-003300)       ACTUAL=003520
* READ REV SETTLEDOWN     RANGE=(013500-008100)       ACTUAL=010970
* TURN AROUND DELAY F-R   RANGE=(016700-010700)       ACTUAL=013600
* TURN AROUND DELAY R-F   RANGE=(016700-010700)       ACTUAL=013660
* GAP SIZE-STOP HALF      RANGE=(012900-009500)       ACTUAL=012200
* GAP SIZE-START HALF     RANGE=(011800-008500)       ACTUAL=010520
* GAP SIZE-INTERRECORD    RANGE=(014800-013700)       ACTUAL=014500
* GAP CONSISANCY          RANGE=(014000-012400)       ACTUAL=013040
* DATA TIME-200 BPI       RANGE=(024100-023100)       ACTUAL=023460
* DATA TIME-556 BPI       RANGE=(024000-023000)       ACTUAL=023350
* DATA TIME-800BPI        RANGE=(024000-023000)       ACTUAL=023400
* DATA TIME-1600BPI       RANGE=(025100-024100)       ACTUAL=024470
* ERASE GAP TIME          RANGE=(101000-099000)       ACTUAL=099510
* WRITE FILE MARK         RANGE=(105000-103000)       ACTUAL=103990
```

7.1.1 SAMPLE TIME DOCUMENT FOR TAPE SPEED TESTS

```
TYPE FIRST ADDRESS OF CONTROLLER 172440
TYPE TM02 DRIVE #'S TO BE TESTED 0
FOR TM02 DRIVE 0- TYPE SLAVE #'S TO BE TESTED 7
SPEED TESTS ONLY? (YES/NO = 1/0) 1
**********************************************************************
*TM02 DRIVE FUNCTION TIMES- DRIVE # 0 SLAVE # 7 9 CHAN. SER. # 5009
*
*FUNCTION                TIME(SPECIFICATION)     TIME(ACTUAL)
*TAPE SPEED FWD          RANGE=<022700-021700>       ACTUAL=022500
*TAPE SPEED REV          RANGE=<022700-021700>       ACTUAL=022500
```

7.2 TEST SEQUENCE WITH RELATED ADJUSTMENTS AND ASSOCIATED HARDWARE

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

TM02 DRIVE FUNCTION TIMER

SEQ 0015

21. GAP CONSISTENCY	*SAME AS IN TEST 16	*WRITE CLOCK
*TEST NUMBER 22 IS RESERVED FOR FUTURE USE		
23. DATA TIME 200 BPI	*NONE	* " "
24. DATA TIME 556 BPI	* "	* " "
25. DATA TIME 800 BPI	* "	* " "
26. DATA TIME 1600 BPI	* "	* " "
27. ERASE GAP TIME	* "	*M8911 ROM*M8903 ACCL CNTR
30. WRITE FILE MARK	* "	* " " * " " "
31. TAPE SPEED-FORWARD	*FWD SPEED	*CAPSTAN SERVO LOOP
32. TAPE SPEED-REVERSE	*REVERSE SPEED	*CAPSTAN SERVO LOOP

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

7.3 TEST DESCRIPTIONS:

THE FIRST THIRTEEN (13) TESTS (T1 - T15) ARE CHECKS OF THE ROM CIRCUITS IN THE TU16J (M9811), THE ACCL COUNTER IN THE TM02 (M8903), AND THE SETTLEDOWN ONE SHOT (M8910).

T1. WRITE FROM BOT:

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

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

T2. WRITE START:

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

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

T3. WRITE SHUTDOWN:

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

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

T4. WRITE SETTLEDOWN:

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

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

T5. READ FROM BOT

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

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

T6. READ START

THIS TEST MEASURES THE SAME DELAY AS IN T2.

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

T7. READ SHUTDOWN:

THIS TEST MEASURES THE SAME DELAY AS IN T3.

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

T10. READ SETTLEDOWN:

THIS TEST MEASURES THE SAME DELAY AS IN T4.

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

T11. READ REVERSE START:

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

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

T12. READ REVERSE SHUTDOWN

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

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

T13. READ REVERSE SETTLEDOWN:

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

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

T14. TURN AROUND DELAY-FORWARD TO REVERSE

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

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

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 TM02 OR TU16J 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 DZTUG LISTING FOR GAP TIMES)

T22. RESERVED FOR FUTURE USE*****

T23. DATA TIME AT 200 BPI:

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

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

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

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

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

T27. ERASE:

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

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

T30. TAPE MARK:

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

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

T31. TAPE SPEED FORWARD:

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

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

T32. TAPE SPEED REVERSE:

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

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

```
                                .NLIST  MC
                                .LIST   ME
                                .ABS
                                .MCALL  $CPVEC,$CPREG,$CATCH,$TYPE
                                .TITLE  DZTUG-A TM02/TU16J DRIVE FUNCTION TIMER
                                .SBTTL  STARTING INSTRUCTIONS
                        ;LOADING AND STARTING PROCEEDURE
                        ;       LOAD PROGRAM USING ABS LOADER
                        ;       LOAD ADDRESS 200
                        ;       SET SWITCH OPTIONS
                        ;       PRESS START

                        ;GENERAL REGISTER USAGE:
                        ;       R0=ADDRESS OF 'FC' REGISTER (SET BY SCOPE)
                        ;       R1=ADDRESS OF 'DS' REGISTER (SET BY SCOPE)
                        ;       R2=RETURN PC FROM TIMER (SET BY EACH TEST)
                        ;       R3=INDEX INDICATING PREVIOUS OSCILLATOR POLARITY (SET BY TIMER)
                        ;       R4=CONTAINS 'TICK' COUNT WHEN TIMER IS RUNNING (SET BY TIMER)
                        ;       R5=ADDRESS OF CS1 (SET BY SCOPE)

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

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

```
                        .SBTTL  REGISTER ASSIGNMENTS
                ;;DEFINITIONS AND REGISTER ASSIGNMENTS
                ;;GENERAL REGISTER ASSIGNMENTS
000000                  R0=%0
000001                  R1=%1
000002                  R2=%2
000003                  R3=%3
000004                  R4=%4
000005                  R5=%5
000006                  SP=%6
000007                  PC=%7
000000                  R10=%0
000001                  R11=%1
000002                  R12=%2
000003                  R13=%3
000004                  R14=%4
000005                  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

                ;RH11,TM02/TU16 REGISTERS
172440                  TMCS1=  172440
```

```
                        ;TM02/TU16 INDEX VALUES
000000                  CS1=    00              ;CONTROL STATUS #1
000002                  WC=     02
000004                  BA=     04              ;BUS ADDRESS REGISTER
000006                  FC=     06              ;FRAME COUNT
000010                  CS2=    10              ;CONTROL STATUS #2
000012                  DS=     12              ;DRIVE STATUS
000014                  ER=     14              ;ERROR REG #1
000016                  AS=     16              ;ATTENTION SUMMARY
000022                  DB=     22              ;DATA BUFFER REG
000024                  MR=     24              ;MAINTENANCE REG
000026                  DT=     26              ;DRIVE TYPE REG
000030                  SN=     30              ;SERIAL NUMBER REGISTER
000032                  TC=     32              ;TAPE CONTROL REG

                        .SBTTL  TM02/TU16 REGISTER BITS
                        ;RHCS1-CS1(R5)
000001                  GO=     1
000000                  NOP=    0
000002                  RWDOFF= 2
000006                  RWD=    6
000010                  DRYCLR= 10
000026                  WFMK=   26
000024                  ERASE=  24
000030                  SPCFWD= 30
000032                  SPCREV= 32
000050                  WCHKF=  50
000056                  WCHKR=  56
000060                  WFWD=   60
000070                  RDFWD=  70
000076                  RDREV=  76
000100                  IE=     100
000200                  RDY=    200
000400                  A16=    400
001000                  A17=    1000
002000                  PSEL=   2000
004000                  DVA=    4000
020000                  MCPE=   20000
040000                  TRE=    40000
100000                  SC=     100000
                        ;RHCS2-CS2(R5)
000000                  DV0=    0
000001                  DV1=    1
000002                  DV2=    2
000003                  DV3=    3
000004                  DV4=    4
000005                  DV5=    5
000006                  DV6=    6
000007                  DV7=    7
000010                  BAI=    10
000020                  PAT=    20
000040                  CLR=    40
000100                  IR=     100
000200                  OR=     200
000400                  MDPE=   400
001000                  MXF=    1000
```

```
                                        PGE=    2000
                                        NEM=    4000
                                        NED=    10000
                                        UPE=    20000
                                        WCE=    40000
                                        DLT=    100000
                                ;RHDS-DS(R5)
                                        SLA=    1
                                        BOT=    2
                                        TMK=    4
                                        IDB=    10
                                        SDWN=   20
                                        PES=    40
                                        SSC=    100
                                        DRY=    200
                                        DPR=    400
                                        EOT=    2000
                                        WRL=    4000
                                        MOL=    10000
                                        PIP=    20000
                                        ERR=    40000
                                        ATA=    100000
                                ;RHER-ER(R5)
                                        ILF=    1
                                        ILR=    2
                                        RMR=    4
                                        FMT=    20
                                        INCVAE= 100
                                        PEFLRC= 200
                                        NSG=    400
                                        FCE=    1000
                                        CSITM=  2000
                                        NEF=    4000
                                        DTE=    10000
                                        OPI=    20000
                                        UNS=    40000

                                ;RHMR-MR(R5)
                                        OSC=    100

                                ;RHDT-DT(R5)
                                        SPR=    2000
                                        CH7=    10000
                                        TAP=    40000

                                ;RHTC-TC(R5)
                                        NORM11= 300
                                        CDM11=  320
                                        BPI200= 0
                                        BPI556= 000400
                                        BPI800= 001000
                                        PE1600= 002000
                                        ACCL=   100000

                                ;INSTRUCTION EQUATES
```

```
                104400                  HLT=    TRAP
                104000                  SCOPE=  EMT
                000004                  TYPE=   IOT

                                ;MISCELLANEOUS EQUATES
                005724                  OUTBUF=INIT             ;OTPUT BUFFER START AT BEG OF PROGRAM
                177400                  FRMCNT= -256.           ;FRAME COUNT
                177600                  WRDCNT= -128.           ;WORD COUNT
                                ;ASCII EQUATES
                000003                  CNTRLC= 3               ;ASCII CODE FOR CONTROL C (↑C)
                000011                  HT=     11              ;ASCII CODE FOR HORIZONTAL TAB
                000012                  LF=     12              ;ASCII CODE FOR LINE FEED
                000015                  CR=     15              ;ASCII CODE FOR CARRIAGE RETURN
                000017                  CNTRLO= 17              ;ASCII CODE FOR CONTROL O (↑O)
                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      002332                          .WORD   .TYPE           ;SET IOT TRAP TO .TYPE ROUTINE
000022      000000                          .WORD   0               ;PRIOTITY LEVEL 0
000024      000026                          .WORD   PFVEC+2         ;POWER FAIL TRAP TO HALT
000026      000000                          .WORD   HALT            ;AT PFVEC+2
000030      004126                          .WORD   .SCOPE          ;SET EMT TRAP TO .SCOPE ROUTINE
000032      000340                          .WORD   340             ;PRIORITY LEVEL 7
000034      003652                          .WORD   .HLT            ;SET TRAP TRAP TO .HLT ROUTINE
000036      000340                          .WORD   340             ;PRIORITY LEVEL 7
            000060                          .=TKVEC
000060      003606                          .WORD   TKISR
000062      000340                          .WORD   340

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

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

            000500                          .=500
            000600              STKPTR= 600                 ;STACK

            001000                          .=1000
                                    ;PROGRAM TAGS
001000      177570                  SWR:    177570          ;SWITCH REGISTER
001002      000000                  SCPADR: .WORD   0
001004         000                  DRVNUM: .BYTE   0       ;TM02 DRIVE UNDER TEST
001005         000                  SLVNUM: .BYTE   0       ;TU16 SLAVE UNDER TEST
001006      000000                  SLVPTR: .WORD   0       ;POINTER TO SLAVE TABLE (SLVTBL) BELOW
001010      172440                  TMBASE: .WORD   TMCS1   ;BASE ADDRESS OF TM02/TU16 REGISTERS
001012      000000                  ATIME:  .WORD   0               ;CONTAINS 'TICK' COUNT
001014      000020                  ATIMTBL:.BLKW   16.     ;EACH ENTRY CONTAINS TIME FOR FUNCTION
                                                            ;ENTRIES ARE MADE BY 'SCOPE' ROUTINE
001054      000020                  GAPTBL: .BLKW   16.     ;TIMES RECORDED BY 'GAP CONSISTANCY' TEST
001114      000000                  DELTIM: .WORD.  0       ;VARIBLE DELAY
001116      000000                  OCTALO: .WORD   0
001120         000                  GAP:    .BYTE   0       ;CONTAINS GAP # (USED FOR TST 021)
001121         000                  ITCNT:  .BYTE   0       ;ITERATION COUNT
001122         000                  TSTNUM: .BYTE   0       ;TEST #
001123         000                  ERFLG:  .BYTE   0       ;ERROR FLAG
001124         000                  PRGFLG: .BYTE   0       ;PROGRAM FLAG
001125         000                  UNTFND: .BYTE   0       ;UNIT FOUND INDICATOR
001126         000                  TYPFLG: .BYTE   0
001127         000                  NRZFLG: .BYTE   0       ;INDICATES IF DRIVE IS NRZ ONLY.
001130         000                  ASFLG:  .BYTE   0       ;1/0 = YES/NO.
            001132                          .EVEN
001132      030460                  DIGTAB: "01
001134      031462                          "23
001136      032464                          "45
001140      033466                          "67
001142      034470                          "89
001144      000006                  ODIGITS:.BLKB   6               ;RESERVE SPACE FOR CONVERTED DIGITS
```

```
                                                .BYTE   0               ;TERMINATOR
                                                .EVEN
                                DRVTBL: .BLKB   8.              ;A 0/-1 = DRIVE NOT TO BE/TO BE TESTED
                                SLVTBL: .BLKB   64.             ;A 0/-1 = SLAVE NOT TO BE/TO BE TESTED
                                INBUF:  .BLKB   72.             ;TELETYPE INPUT BUFFER
                                CRLF:   .ASCIZ  <CR><LF>        ;MISCELLANEOUS ASCII CHARACTERS
                                BKSLSH: .ASCIZ  '\'
                                ECHO:   .ASCIZ  '0'
                                BELL:   .ASCIZ  <7>
                                DASH:   .ASCIZ  '-'
                                SPACE2: .ASCII  '  '
                                SPACE:  .ASCIZ  ' '
                                ANGTAB: .ASCIZ  '>'<HT>
                                        .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  042300  041460          .WORD   17600.,17200.   ;176.0-172.0   WRITE FROM BOT  TST001
001426  001666  001546          .WORD   00950.,00870.   ;9.5-8.7       WRITE START     TST002
001432  001572  001522          .WORD   00890.,00850.   ;8.9-8.5       WRITE SHUTDOWN  TST003
001436  002506  001452          .WORD   01350.,00810.   ;13.5-8.1       WRITE STLDOWN   TST004
001442  010624  010004          .WORD   04500.,04100.   ;45.0-41.0      READ FROM BOT   TST005
001446  000500  000404          .WORD   00320.,00260.   ;3.2-2.6        READ START      TST006
001452  000721  000651          .WORD   00465.,00425.   ;4.65-4.25      READ SHUTDOWN   TST007
001456  002506  001452          .WORD   01350.,00810.   ;13.5-8.1       READ SETTLEDOWN TST010
001462  000500  000404          .WORD   00320.,00260.   ;3.2-2.6        RD REV START    TSTC11
001466  000562  000512          .WORD   00370.,00330.   ;3.7-3.3        RD REV SHTDWN   TST012
001472  002506  001452          .WORD   01350.,00810.   ;13.5-8.1       RD REV STLDWN   TST013
001476  003206  002056          .WORD   01670.,01070.   ;16.7-10.7      TRN RND DLY F-R TST014
001502  003206  002056          .WORD   01670.,01070.   ;16.7-10.7      TRN RND DLY R-F TST015
001506  002412  001666          .WORD   01290.,00950.   ;12.9-9.5       GAP SIZE STOP   TST016
001512  002234  001522          .WORD   01180.,00850.   ;11.8-8.5       GAP SIZE STRT   TST017
001516  002710  002532          .WORD   01480.,01370.   ;14.8-13.7      GAP SIZE INTER  TST020
001522  002544  002330          .WORD   01380.,01240.   ;13.8-12.4      GAP CONSISANCY  TST021
001526  000000  000000          .WORD   0,0             ;0.0-0.0        DUMMY           TST022
001532  004552  004406          .WORD   02410.,02310.   ;24.1-23.1      DAT TIME 200BPI TST023
001536  004540  004374          .WORD   02400.,02300.   ;24.0-23.0      DAT TIME 556BPI TST024
001542  004540  004374          .WORD   02400.,02300.   ;24.0-23.0      DAT TIME 800BPI TST025
001546  004716  004552          .WORD   02510.,02410.   ;25.1-24.1      DAT TIME 1600PE TST026
001552  023564  023254          .WORD   10100.,09900.   ;101.0-99.0     ERASE           TST027
001556  024404  024074          .WORD   10500.,10300.   ;105.0-103.0    WRT FILE MARK   TST030
001562  004336  004172          .WORD   02270.,02170.   ;22.7-21.7      READ 1" TAPE    TST031
001566  004336  004172          .WORD   02270.,02170.   ;22.7-21.7      RD REV 1" TAPE  TST032

                ;NOTE: TEST 31 AND 32 REQUIRE PRERECORDED 800BPI SKEW TAPE.
```

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

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

001572  002602  002412 GTIMTBL:.WORD   01410.,01290.   ;14.1-12.9      GAP-0   0 MS
001576  002652  002506         .WORD   01450.,01350.   ;14.5-13.5      GAP-1   1.0 MS
001602  002734  002532         .WORD   01500.,01370.   ;15.0-13.7      GAP-2   2.0 MS
001606  002734  002532         .WORD   01500.,01370.   ;15.0-13.7      GAP-3   3.0 MS
001612  002734  002424         .WORD   01500.,01300.   ;15.0-13.0      GAP-4   4.0 MS
001616  002652  002260         .WORD   01450.,01200.   ;14.5-12.0      GAP-5   5.0 MS
001622  002652  002260         .WORD   01450.,01200.   ;14.5-12.0      GAP-6   6.0 MS
001626  002652  002260         .WORD   01450.,01200.   ;14.5-12.0      GAP-7   7.0 MS
001632  002532  002260         .WORD   01370.,01200.   ;13.7-12.0      GAP-10  8.0 MS
001636  002532  002260         .WORD   01370.,01200.   ;13.7-12.0      GAP-11  9.0 MS
001642  002532  002260         .WORD   01370.,01200.   ;13.7-12.0      GAP-12  10.0 MS
001646  002532  002260         .WORD   01370.,01200.   ;13.7-12.0      GAP-13  11.0 MS
001652  002532  002260         .WORD   01370.,01200.   ;13.7-12.0      GAP-14  12.0 MS
001656  002532  002260         .WORD   01370.,01200.   ;13.7-12.0      GAP-15  13.1 MS
001662  002532  002260         .WORD   01370.,01200.   ;13.7-12.0      GAP-16  14.1 MS
001666  002532  002260         .WORD   01370.,01200.   ;13.7-12.0      GAP-17  15.1 MS
```

```
                        .SBTTL  TEST HEADER POINTERS
                ;THE BELOW TABLE CONTAINS POINTERS TO EACH TEST'S DESCRIPTOR
001672  000000  NAMPTR: .WORD   0
001674  014667          .WORD   A.T001
001676  014711          .WORD   A.T002
001700  014731          .WORD   A.T003
001702  014753          .WORD   A.T004
001704  014777          .WORD   A.T005
001706  015021          .WORD   A.T006
001710  015040          .WORD   A.T007
001712  015062          .WORD   A.T010
001714  015105          .WORD   A.T011
001716  015127          .WORD   A.T012
001720  015154          .WORD   A.T013
001722  015203          .WORD   A.T014
001724  015234          .WORD   A.T015
001726  015265          .WORD   A.T016
001730  015313          .WORD   A.T017
001732  015342          .WORD   A.T020
001734  015372          .WORD   A.T021
001736  000000          .WORD   0                       ;DUMMY TEST
001740  015415          .WORD   A.T023
001742  015441          .WORD   A.T024
001744  015465          .WORD   A.T025
001746  015511          .WORD   A.T026
001750  015536          .WORD   A.T027
001752  015560          .WORD   A.T030
001754  015603          .WORD   A.T031
001756  015625          .WORD   A.T032
```

```
                                        ;CHECK SWITCH REGISTER ROUTINE. CHECKS FOR ↑G TO ALLOW CHANGING
                                        ;OF LOC.176.
                                        ;CALL IS BY WAY OF JSR  PC,CKSWR
                                        ;LOCATIONS USED:
001760  000000                          TIB:    .WORD   0
001762  000000                          TEMPST: .WORD   0
001764  000000                          COUNT:  .WORD   0
001766  000000                          RDSW:   .WORD   0

001770  022767  000176  177002  CKSWR:  CMP     #SWREG,SWR              ;SOFTWARE SWITCH REG PRESENT
001776  001120                          BNE     OUT                     ;NO, GET OUT
002000  016767  175556  177752          MOV     TKB,TIB         ;AND STRIP OFF
002006  042767  177600  177744          BIC     #177600,TIB             ;THE GARBAGE
002014  022767  000007  177736          CMP     #7,TIB          ;IS IT A <↑G>
002022  001106                          BNE     OUT
002024  000004  015647                  TYPE,L.CNTG
002030  000004  015654          CNTLU:  TYPE,L.SWR
002034  017702  176740                  MOV     @SWR,R2
002040  004767  000564                  JSR     PC,TYPOCT
002044  000004  015663                  TYPE,L.NEW

002050  005067  177706                  CLR     TEMPST
002054  012767  000007  177702          MOV     #7,COUNT
002062  004767  000154          1$:     JSR     PC,TTIN         ;GO  READ A CHARACTER
002066  042767  177600  177664          BIC     #177600,TIB     ;STRIP OFF GARBAGE
002074  122767  000025  177656          CMPB    #25,TIB         ;IS IT A ↑U?
002102  001001                          BNE     2$              ;BRANCH IF NOT
002104  000751                  3$:     BR CNTLU                ;START OVER
002106  122767  000015  177644  2$:     CMPB    #15,TIB         ;IS IT A <CR>?
002114  001012                          BNE     4$              ;BRANCH IF NOT
002116  012767  000200  177642          MOV     #200,RDSW
002124  004767  000230                  JSR     PC,TCRLF        ;ECHO IT WITH <LF>
002130  022767  000007  177626          CMP     #7,COUNT        ;WAS IT FIRST CHARACTER
002136  001034                          BNE     7$              ;CHANGE SWR IF NOT FIRST ONE
002140  000437                  8$:     BR      OUT             ;GET OUT
002142  122767  000060  177610  4$:     CMPB    #60,TIB
002150  003004                          BGT     5$
002152  122767  000067  177600          CMPB    #67,TIB
002160  002003                          BGE     6$
002162  000004  015673          5$:     TYPE,L.QUEST
002166  000746                          BR      3$              ;START OVER IF NOT LEGAL CHARACTER
002170  006367  177566          6$:     ASL     TEMPST
002174  006367  177562                  ASL     TEMPST
002200  006367  177556                  ASL     TEMPST
002204  142767  000060  177546          BICB    #60,TIB         ;GET NITTY-GRITTY
002212  156767  177542  177542          BISB    TIB,TEMPST
002220  005367  177540                  DEC     COUNT           ;ONLY WANT 6   DIGITS
002224  001756                          BEQ     5$
002226  000715                          BR      1$
002230  016777  177526  176542  7$:     MOV     TEMPST,@SWR     ;CHANGE SWITCH REGISTER CONTENTS
002236  000740                          BR      8$
002240  000207                  OUT:    RTS     PC
```

```
                                        ;TTY READ SUBROUTINE**********
002242  005067  175312          TTIN:   CLR     TKS
002246  005067  175310                  CLR     TKB
002252  005067  177502                  CLR     TIB
002256  005267  175276                  INC     TKS
002262  105767  175272          TTIN1:  TSTB    TKS
002266  100375                          BPL     TTIN1
002270  016767  175266  177462          MOV     TKB,TIB
002276  105767  175262          TTIN2:  TSTB    TPS
002302  100375                          BPL     TTIN2
002304  116767  177450  175254          MOVB    TIB,TPB
002312  000207                          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
002314     000          $NULL:  .BYTE   0                       ;;CONTAINS NULL CHARACTER
002315     002          $FILL:  .BYTE   2                       ;;CONTAINS # OF FILLER CHARACTERS
002316     000          $TPFLG: .BYTE   0                       ;;CONTAINS TELEPRINTER AVAILABLE FLAG
                                                                ;;0/377 = AVAIL/NOT AVAIL
002317     000          $TKFLG: .BYTE   0                       ;;CONTAINS KEYBOARD AVAILABLE FLAG
002320  177564          $TPS:   .WORD   177564                  ;;ADDRESS OF TLEPRINTER STATUS REGISTER
002322  177566          $TPB:   .WORD   177566                  ;;ADDRESS OF TELEPRINTER DATA BUFFER
002324     000          $CHARCNT:.BYTE  0                       ;;CONTAINS # OF CHARS TYPED
002325     000          $CNTRLO:.BYTE   0                       ;;CONTAINS CONTROL O CHAR (IF TYPED)
002326  005015  000      $CRLF:  .ASCIZ  <15><12>
002332                          .EVEN

002332  010046          .TYPE:  MOV     R0,-(SP)                ;;SAVE R0
002334  017600  000002          MOV     @2(SP),R0               ;;GET MESSAGE ADDRESS
002340  062766  000002  000002  ADD     #2,2(SP)                ;;ADJUST RETURN PC
002346  105067  177753          CLRB    $CNTRLO

002352  105767  177747  TYPE1:  TSTB    $CNTRLO                 ;;BRANCH IF CONTROL O(↑O) WASN'T TYPED
002356  001410                  BEQ     TYPE2
002360  000004  002326  TCRLF:  TYPE,$CRLF                      ;;TYPE <CR><LF>
002364  105767  177376          TSTB    RDSW
002370  100006                  BPL     TYPE3
002372  005067  177370          CLR     RDSW
002376  000207                  RTS     PC
002400  112046          TYPE2:  MOVB    (R0)+,-(SP)             ;;PUSH CHARACTER TO BE TYPED ONTO STACK
002402  001003                  BNE     TYPE4                   ;;BRANCH IF NOT THE TERMINATOR
002404  005726                  TST     (SP)+                   ;;POP TERMINATOR CHAR OFF THE STACK
002406  012600          TYPE3:  MOV     (SP)+,R0                ;;RESTORE R0
002410  000002                  RTI                             ;;RETURN TO CALLER

002412  122716  000011  TYPE4:  CMPB    #$HT,(SP)               ;;BRANCH IF HORIZONTAL TAB <HT>
002416  001445                  BEQ     9$
```

```
(1)  002420  004767  000026            3$:   JSR    PC,5$            ;;TYPE CHARACTER
(1)  002424  122726  000012                  CMPB   #12,(SP)+        ;;CHECK IF CHARACTER WAS A LINE FEED
(1)  002430  001350                          BNE    TYPE1            ;;BRANCH IF NOT LINE FEED
(1)  002432  016746  177656                  MOV    $NULL,-(SP)      ;;GET # OF FILLERS REQUIRED AND FILLER
(1)                                                                  ;;CHARACTER.
(1)
(1)  002436  105366  000001            4$:   DECB   1(SP)            ;;DECREMENT FILLERS REQ. COUNT
(1)  002442  002770                          BLT    3$               ;;BRANCH IF NO MORE FILLERS ARE REQUIRED
(1)  002444  004767  000002                  JSR    PC,5$            ;;TYPE FILLER CHARACTER
(1)  002450  000772                          BR     4$
(1)
(1)  002452  105777  177642            5$:   TSTB   @$TPS            ;;WAIT FOR OUTPUT DEVICE
(1)  002456  100375                          BPL    .-4
(1)  002460  122737  000017  002325          CMPB   #17,@#$CNTRLO     ;;CHECK IF CONTROL O WAS TYPED
(1)  002466  001403                          BEQ    6$               ;;STOP TYPING MESSAGE IF ↑O WAS TYPED
(1)  002470  116677  000002  177624          MOVB   2(SP),@$TPB       ;;OUTPUT CHARACTER
(1)  002476  122766  000015  000002    6$:   CMPB   #15,2(SP)         ;;BRANCH IF NOT <CR>
(1)  002504  001003                          BNE    7$
(1)  002506  105067  177612                  CLRB   $CHARCNT          ;;CLEAR CHARACTERS TYPED COUNT
(1)  002512  000406                          BR     8$
(1)  002514  122766  000012  000002    7$:   CMPB   #12,2(SP)         ;;BRANCH IF <LF> OR 'NULL'
(1)  002522  002002                          BGE    8$
(1)  002524  105267  177574                  INCB   $CHARCNT          ;;INCREMENT CHARACTER TYPED COUNT
(1)  002530  000207                    8$:   RTS    PC
(1)
(1)                                    ;;HORIZONTAL TAB <HT> PROCESSER
(1)  002532  112716  000040            9$:   MOVB   #40,(SP)         ;;LOAD 'SPACE'
(1)  002536  004767  177710            10$:  JSR    PC,5$            ;;TYPE 'SPACE'
(1)  002542  132767  000007  177554          BITB   #7,$CHARCNT       ;;TYPE SPACES UNTIL A MULTIPLE
(1)  002550  001372                          BNE    10$              ;;OF 8 CHARACTERS HAVE BEEN TYPED
(1)  002552  105726                          TSTB   (SP)+            ;;POP SPACE
(1)  002554  000676                          BR     TYPE1            ;;GET NEXT CHARACTER
435
436                                    ;SUBROUTINE TO SAVE GENERAL REGISTERS ON THE STACK
437                                    ;CALL:  SAVE
438  002556  010546                   .SAVE: MOV    R5,-(SP)                 ;SAVE REGISTERS ON THE STACK
439  002560  010446                          MOV    R4,-(SP)
440  002562  010346                          MOV    R3,-(SP)
441  002564  010246                          MOV    R2,-(SP)
442  002566  010146                          MOV    R1,-(SP)
443  002570  010046                          MOV    R0,-(SP)
444  002572  016646  000014                  MOV    14(SP),-(SP)             ;GET RETURN PC
445  002576  000207                          RTS    PC                       ;RETURN
446
447                                    ;SUBROUTINE TO RESTORE GENERAL REGISTERS FROM THE STACK
448                                    ;CALL:  RESTORE
449  002600  012666  000014           .RESTORE:MOV   (SP)+,14(SP)             ;MOVE RETURN PC
450  002604  012600                          MOV    (SP)+,R0                 ;RESTORE REGISTERS
451  002606  012601                          MOV    (SP)+,R1
452  002610  012602                          MOV    (SP)+,R2
453  002612  012603                          MOV    (SP)+,R3
454  002614  012604                          MOV    (SP)+,R4
455  002616  012605                          MOV    (SP)+,R5
456  002620  000207                          RTS    PC                       ;RETURN
457
458                                    ;SUBROUTINE TO CONVERT OCTAL DATA TO ASCII
```

```
                                ;CALL:  MOV     NUMBER,R2               ;MOVE NUMBER TO R2
                                ;       JSR     PC,CNVOCT
                                
002622  110667  176300          CNVOCT: MOVB    SP,TYPFLG               ;SET DO NOT TYPE FLAG
002626  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
                                
002630  105037  001126          TYPOCT: CLRB    @#TYPFLG                ;SET TYPE FLAG
002634                          CNVTO:
002634  004767  177716                  JSR     PC,.SAVE                ;SAVE REGISTERS ON THE STACK
002640  012704  001144                  MOV     #ODIGITS,R4             ;SET PTR TO OUTPUT
002644  005003                          CLR     R3                      ;R3 WILL CONTAIN OCTAL DIGIT
002646  010201                          MOV     R2,R1                   ;GET # TO BE TYPED
002650  006302                  1$:     ASL     R2                      ;SHIFT #
002652  006103                          ROL     R3
002654  012700  000006                  MOV     #6,R0                   ;SET DIGIT COUNTER
002660  000404                          BR      3$
                                
002662  006302                  2$:     ASL     R2                      ;SHIFT # 3 PLACES LEFT
002664  006103                          ROL     R3
002666  005301                          DEC     R1
002670  001374                          BNE     2$
002672  012701  000003          3$:     MOV     #3,R1                   ;SET SHIFT COUNTER
002676  116324  001132                  MOVB    DIGTAB(R3),(R4)+        ;MOVE ASCII EQUIV TO OUTPUT
002702  005003                          CLR     R3
002704  005300                          DEC     R0                      ;DECREMENT DIGIT COUNT
002706  001365                          BNE     2$                      ;GET NEXT DIGIT
002710  105737  001126                  TSTB    @#TYPFLG                ;BRANCH IF ASCII IS
002714  001002                          BNE     4$                      ;NOT TO BE TYPED
002716  000004  001144                  TYPE,ODIGITS
002722                          4$:
002722  004767  177652                  JSR     PC,.RESTORE             ;RESTORE REGISTERS FROM THE STACK
002726  000207                          RTS     PC
                                
                                ;SUBROUTINE TO CONVERT OCTAL DATA TO DECIMAL ASCII
                                ;CALL:  MOV     NUMBER,R2               ;MOVE NUMBER TO R2
                                ;       JSR     PC,CNVDEC
                                
002730  110637  001126          CNVDEC: MOVB    SP,@#TYPFLG             ;SET DO NOT TYPE FLAG
002734  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
                                
002736  105037  001126          TYPDEC: CLRB    @#TYPFLG                ;SET TYPE FLAG
002742                          CNVTD:
002742  004767  177610                  JSR     PC,.SAVE                ;SAVE REGISTERS ON THE STACK
002746  005000                          CLR     R0                      ;R0 IS INDEX TO DECIMAL CONSTANT
002750  012704  001144                  MOV     #ODIGITS,R4             ;SET OUTPUT PTR
002754  005003                  1$:     CLR     R3                      ;R3 CONTAINS DECIMAL DIGIT
```

```
                                2$:     SUB     DCONST(R0),R2           ;SUBTRACT DECIMAL CONSTANT UNTIL
                                        BLO     3$                      ;INPUT # GOES NEGATIVE
                                        INC     R3                      ;KEEPING TRACK OF SUBTRACTIONS
                                        BR      2$
                                3$:     ADD     DCONST(R0),R2           ;ADD BACK CONSTANT WHEN NEGATIVE
                                        MOVB    DIGTAB(R3),(R4)+        ;MOVE ASCII EQUIVILENT
                                        ADD     #2,R0                   ;NEXT CONSTANT
                                        TST     DCONST(R0)              ;UNTIL ALL CONSTANTS DONE
                                        BNE     1$
                                        MOVB    #'0,(R4)+               ;LAST DIGIT IS 0
                                        TSTB    @#TYPFLG                ;BRANCH IF ASCII IS
                                        BNE     4$                      ;NOT TO BE TYPED
                                        TYPE,ODIGITS
                                4$:
(1)                                     JSR     PC,.RESTORE             ;RESTORE REGISTERS FROM THE STACK
                                        RTS     PC

                                DCONST: .WORD   10000.
                                        .WORD   1000.
                                        .WORD   100.
                                        .WORD   10.
                                        .WORD   1.
                                        .WORD   0                       ;TERMINATOR


                                        .SBTTL          TYPE SPECIFIED TIMES ROUTINE
                                ;THIS SUBROUTINE OUTPUTS THE TIME SPECIFICATIONS FOR THE TEST
                                ;AND ALSO THE ACTUAL TIME RECORDED (ATIME)
                                ;FORMAT OF LINE TYPED
                                ;RANGE=<AAAAAA-BBBBBB>          ACTUAL=CCCCCC
                                ;WHERE:         AAAAAA IS MAXIMUM TIME FOR TEST (STIMTBL(TSTNUMX4)).
                                ;               BBBBBB IS MINIMUM TIME FOR TEST (STIMTBL(TSTNUMX4+2)).
                                ;               CCCCCC IS ACTUAL TIME RECORDED BY TEST (ATIME).
                                ;CALL:  MOVB    TEST NUMBER,R2  ;LOAD TEST NUMBER
                                ;       MOV     #TIME,@#ATIME           ;MOVE TIME TO ATIME
                                ;       JSR     PC,OUTSPC
                                OUTSPC: MOV     R2,-(SP)                ;SAVE R2 & R3 ON THE STACK
                                        MOV     R3,-(SP)
                                        ASL     R2                      ;MULTIPLY TEST # TIMES 4
                                        ASL     R2                      ;TO FORM INDEX INTO STIMTBL
                                        MOV     R2,R3                   ;R3 CONTAINS INDEX INTO TABLE
                                        TYPE,L.RNG
                                        MOV     STIMTBL(R3),R2          ;GET MAXIMUM SPEC TIME
                                        JSR     PC,TYPDEC               ;CONVERT TO DECIMAL & TYPE
                                        TYPE,DASH
                                        MOV     STIMTBL+2(R3),R2        ;GET MINIMUM TIME
                                        JSR     PC,TYPDEC               ;CONVERT TO DECIMAL & TYPE
                                        TYPE,ANGTAB
                                        TYPE,L.ACT
                                        MOV     @#ATIME,R2              ;GET ACTUAL TIME
                                        JSR     PC,TYPDEC               ;CONVERT TO DECIMAL & TYPE
                                        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
                                
003146  010246                  OUTGAP: MOV     R2,-(SP)                ;SAVE R2 AND R3
003150  010346                          MOV     R3,-(SP)
003152  116703  175742                  MOVB    GAP,R3                  ;GET GAP #
003156  006303                          ASL     R3
003160  006303                          ASL     R3
003162  000004  014647                  TYPE,L.RNG
003166  016302  001572                  MOV     GTIMTBL(R3),R2          ;GET MAX TIME
003172  004767  177540                  JSR     PC,TYPDEC               ;CONVERT TO DECIMAL & TYPE
003176  000004  001405                  TYPE,DASH
003202  016302  001574                  MOV     GTIMTBL+2(R3),R2        ;GET MIN TIME
003206  004767  177524                  JSR     PC,TYPDEC               ;CONVERT TO DECIMAL & TYPE
003212  000004  001412                  TYPE,ANGTAB                     ;TYPE <
003216  000004  014657                  TYPE,L.ACT
003222  013702  001012                  MOV     @#ATIME,R2              ;GET ACTUAL TIME
003226  004767  177504                  JSR     PC,TYPDEC               ;CONVERT TO DECIMAL & TYPE
003232  000004  014336                  TYPE,E.GAP
003236  113702  001120                  MOVB    @#GAP,R2                ;GET GAP #
003242  004767  177362                  JSR     PC,TYPOCT               ;TYPE GAP #
003246  000004  001374                  TYPE,CRLF
003252  012603                          MOV     (SP)+,R3                ;RESTORE R3 AND R2
003254  012602                          MOV     (SP)+,R2
003256  000207                          RTS     PC
                                
                                        .SBTTL          ASCII TO OCTAL CONVERT SUBROUTINE
                                ;SUBROUTINE TO CONVERT ASCII DATA TO OCTAL. CONVERTED OCTAL DATA
                                ;IS LEFT IN OCTALO <15-00>.
003260                          CNVTAO:
(1) 003260  004767  177272              JSR     PC,.SAVE                ;SAVE REGISTERS ON THE STACK
003264  012700  001264                  MOV     #INBUF,R0               ;SET PTR TO ASCII DATA
003270  012701  001116                  MOV     #OCTALO,R1              ;GET ADDRESS OF OCTAL DATA
003274  005011                          CLR     (R1)                    ;CLEAR OUT OLD OCTAL DATA
003276  005061  000002                  CLR     2(R1)
003302  122710  000015          1$:     CMPB    #CR,(R0)                ;<CR> TERMINATES INPUT
003306  001414                          BEQ     3$
003310  112002                          MOVB    (R0)+,R2                ;GET 'OCTAL' DATA
003312  042702  177770                  BIC     #177770,R2              ;STRIP UNUSED BITS
003316  012703  000003                  MOV     #3,R3                   ;SET SHIFT COUNT
003322  006311                  2$:     ASL     (R1)                    ;SHIFT LAST
003324  006161  000002                  ROL     2(R1)                   ;OCTAL DIGIT
003330  005303                          DEC     R3
003332  001373                          BNE     2$
003334  050211                          BIS     R2,(R1)                 ;AND INSERT THIS DIGIT
003336  000761                          BR      1$                      ;GO GET NEXT DIGIT
003340                          3$:
(1) 003340  004767  177234              JSR     PC,.RESTORE             ;RESTORE REGISTERS FROM THE STACK
003344  000207                          RTS     PC                      ;RETURN
                                
                                        .SBTTL          PUBLISH SUBROUTINE
                                ;THE PUBLISH SUBROUTINE AVERAGES THE RECORDED TIMES FOR EACH TEST IT-
```

```
                        ;ERATION (IF 16. ITERATIONS) AND PLACES THE AVERAGE RESULT IN 'ATIME'.
                        ;IT TYPES THE NAME OF THE FUNCTION THAT WAS TIMED,THE TIME SPEC-
                        ;IFICATION AND THE ACTUAL TIME .

003346                  PUBLISH:
(1) 003346 004767 177204        JSR     PC,.SAVE                ;SAVE REGISTERS ON THE STACK
003352 012700 001014            MOV     #ATIMTBL,R0     ;GET TABLE ADDRESS CONTAINING TIMES
003356 113701 001121            MOVB    @#ITCNT,R1      ;GET # OF ENTRIES (GIVEN BY ITERATION COUNT)
003362 122701 000001            CMPB    #1,R1                   ;BRANCH IF SINGLE ITERATION
003366 001423                   BEQ     4$
003370 005002                   CLR     R2                      ;CLEAR 'SUM' REGISTERS
003372 005003                   CLR     R3
003374 122701 000020            CMPB    #16.,R1                 ;BRANCH IF 16. ITERATIONS
003400 001402                   BEQ     1$
003402 000000                   HALT                            ;ITERATION COUNT MUST BE 1 OR 16.
003404 000777                   BR      .                       ;DO NOT CHANGE POSIT OF SW11
                                                                ;WHEN TEST IS RUNNING.

003406 062002           1$:     ADD     (R0)+,R2                ;SUM INDIVIDUAL TIMES
003410 005503                   ADC     R3
003412 005301                   DEC     R1
003414 001374                   BNE     1$

003416 012700 000004    2$:     MOV     #4,R0
003422 006203           3$:     ASR     R3                      ;SHIFT TIME IN R3 & R2 4 PLACES
003424 006002                   ROR     R2                      ;RIGHT = DIVIDE BY 16.
003426 005300                   DEC     R0
003430 001374                   BNE     3$
003432 010237 001012            MOV     R2,@#ATIME              ;MOVE AVERAGED TIMES

003436 113700 001122    4$:     MOVB    @#TSTNUM,R0             ;GET TEST #
003442 006300                   ASL     R0
003444 016067 001672 000002     MOV     NAMPTR(R0),5$           ;GET TEST NAME STRING ADDRESS
003452 000004                   TYPE
003454 000000           5$:     .WORD   0
003456 113702 001122            MOVB    @#TSTNUM,R2             ;GET TEST #
003462 004767 177364            JSR     PC,OUTSPC               ;OUTPUT TIMES
003466 004767 177106            JSR     PC,.RESTORE             ;RESTORE REGISTERS FROM THE STACK
003472 000207                   RTS     PC

                                .SBTTL          INPUT SUBROUTINE
                        ;SUBROUTINE TO GET TTY INPUT
                        ;CALL:  JSR     PC,.INPUT
                        ;INPUT DATA IS RETURNED IN BUFFER BEGINNING AT INBUF.

003474 010046           .INPUT: MOV     R0,-(SP)        ;SAVE R0 ON THE STACK
003476 012700 001264    1$:     MOV     #INBUF,R0
003502 105737 177560    2$:     TSTB    @#TKS
003506 100375                   BPL     2$

003510 113746 177562            MOVB    @#TKB,-(SP)     ;GET CHARACTER
003514 042716 000200            BIC     #200,(SP)
003520 122716 000177            CMPB    #177,(SP)       ;CHECK RUBOUT
003524 001004                   BNE     3$
003526 124026                   CMPB    -(R0),(SP)+     ;REMOVE CHARACTER FROM INPUT
003530 000004 001377            TYPE,BKSLSH
```

```
                                        BR      2$              ;WAIT FOR NEXT CHARACTER
                                3$:     CMPB    #CNTRLU,(SP)    ;CHECK CONTROL U (↑U)
                                        BNE     4$
                                        TST     (SP)+
                                        TYPE,CRLF
                                        BR      1$
                                4$:     MOVB    (SP),@#ECHO
                                        MOVB    (SP),(R0)+
                                        CMPB    #CR,(SP)+
                                        BEQ     5$
                                        TYPE,ECHO
                                        BR      2$
                                5$:     TYPE,CRLF
                                        MOV     (SP)+,R0
                                        RTS     PC

                                ;KEYBOARD INTERRUPT SERVICE ROUTINE
                                TKISR:  MOVB    @#TKB,-(SP)     ;GET TYPED CHARACTER
                                        BIC     #200,(SP)       ;STRIP PARITY BIT
                                        CMPB    #CNTRLO,(SP)    ;BRANCH IF NOT CONTROL O (↑O)
                                        BNE     1$
                                        MOVB    (SP)+,$CNTRLO   ;SET CONTROL O INDICATOR IN TYPE ROUTINE
                                        RTI                     ;EXIT

                                1$:     CMPB    #3,(SP)+        ;BRANCH IF NOT CONTROL C (↑C)
                                        BNE     2$
                                        RESET
                                        JMP     @#INIT          ;RESTART PROGRAM
                                2$:     RTI                     ;EXIT
```

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

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

003652  004767  176112  .HLT:   JSR     PC,CKSWR                ;CHECK FOR CNTL G
003656  004767  176674          JSR     PC,.SAVE                ;SAVE REGISTERS ON THE STACK
003662  110637  001123  1$:     MOVB    SP,@#ERFLG              ;SET ERROR FLAG
003666  032777  020000  175104  BIT     #SW13,@SWR              ;BRANCH IF NO TYPOUT
003674  001075          BNE     4$
003676  000004  014137          TYPE,E.HDR
003702  113702  001122          MOVB    @#TSTNUM,R2             ;GET TEST #
003706  004767  176716          JSR     PC,TYPOCT               ;AND TYPE IT
003712  016600  000016          MOV     16(SP),R0               ;GET RETURN PC
003716  162700  000002          SUB     #2,R0                   ;NOW PC OF HLT CALL
003722  111000                  MOVB    (R0),R0                 ;NOW HLT CALL ITSELF
003724  001417                  BEQ     2$                      ;BRANCH IF HLT
003726  000004  014222          TYPE,E.HDR2
003732  122700  000002          CMPB    #2,R0                   ;BRANCH IF NOT HLT+2
003736  001005                  BNE     10$
003740  004767  177202          JSR     PC,OUTGAP               ;TYPE GAP SPECIFIED TIMES
003744  000004  001374          TYPE,CRLF
003750  000447                  BR      4$
003752  004767  177074  10$:    JSR     PC,OUTSPC               ;TYPE SPECIFIED TIMES
003756  000004  001374          TYPE,CRLF
003762  000442                  BR      4$
003764  016500  000014  2$:     MOV     ER(R5),R0
003770  032765  002000  000032  BIT     #PE1600,TC(R5)
003776  001403                  BEQ     20$
004000  042700  102100          BIC     #102100,R0
004004  000402                  BR      21$
004006  042700  102300  20$:    BIC     #102300,R0
004012  005700  21$:    TST     R0
004014  001003                  BNE     22$
004016  000004  014113          TYPE,E.SFT              ;TYPE SOFT ERROR MESSAGE
004022  000434                  BR      6$

004024  000004  014147  22$:    TYPE,E.HDR1
004030  010500                  MOV     R5,R0                   ;GET FIRST ADDRESS OF REGS.
004032  012701  000007          MOV     #7.,R1                  ;TYPE FIRST 7 REGS.
004036  012002  3$:     MOV     (R0)+,R2                ;GET REG CONTENTS
004040  004767  176564          JSR     PC,TYPOCT               ;AND TYPE IT
004044  000004  001407          TYPE,SPACE2
004050  005301                  DEC     R1
004052  001371                  BNE     3$
004054  016502  000032          MOV     TC(R5),R2               ;GET CONTENTS OF TC REGISTER
004060  004767  176544          JSR     PC,TYPOCT
004064  000004  001374          TYPE,CRLF

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

```
   762  004104  005777  174670          5$:     TST     @SWR                    ;HALT ON ERROR?
   763  004110  100001                          BPL     6$
   764  004112  000000                          HALT
   765  004114  004767  175650          6$:     JSR     PC,CKSWR        ;CHECK FOR CNTL G
   766  004120  004767  176454                  JSR     PC,.RESTORE             ;RESTORE REGISTERS FROM THE STACK
   767  004124  000002                          RTI                             ;RETURN
   768
   769
```

```
                                        .SBTTL          SCOPE SUBROUTINE
                                ;SCOPE ROUTINE
                                ;THIS ROUTINE IS ENTERED UPON COMPLETION OF EACH SUBTEST
                                ;THE SCOPE ROUTINE:
                                ;       OUTPUTS TIME SPEC ON EACH ITERATION IF SW08 IS SET
                                ;       REPEATS TEST IF SW14 IS SET
                                ;       STORES ACTUAL TIME FOR FUNCTION IN TIME TABLE (ATIMTBL)
                                ;       PUBLISHES TIME IF SW10=0
                                ;       UPDATES ITERATION COUNT AND IF ITERATIONS COMPLETE CONTINUES
                                ;       TO NEXT TEST, OTHERWISE REPEATS TEST.
                                ;       DELAYS BEFORE CONTINUING OR REPEATING TEST.
                                ;       INITIALIZES DRIVE
                                ;RETURNS:       R5=BASE ADDRESS OF TM02 REGISTERS (ADDRESS OF CS1)
                                ;               R1='DS' REG ADDRESS
                                ;               R0='FC' REG ADDRESS

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

```
                                        HALT
                                        JSR     PC,CKSWR                ;CHECK FOR CNTL G
                                        BR      1$
                                        .SBTTL  TIMER SUBROUTINES
                                ;SUBROUTINE TO SYNCHRONIZE THE TIMER AND TURN IT ON.
                                ;REGISTER 4 IS CLEARED, AND THE OSCILLATOR POLARITY IS MONITORED
                                ;THE ROUTINE IS EXITED WHEN THE OSCILLATOR POLARITY CHANGES WITH R3
                                ;SET TO INDICATE THE POLARITY OF THE OSCILLATOR.
                                ;CALL:   JSR     PC,TIMON
                                ;RETURNS:       R3 SET TO INDICATE LAST POLARITY (+24/-24=0/1)
                                ;               R4 = 0
                                TIMON:  CLR     R4                      ;CLEAR TIME COUNT
                                        MOV     #24,R3                  ;SET POLARITY TO '0' STATE
                                        BIT     #OSC,MR(R5)             ;BRANCH IF POLARITY IS '0'
                                        BEQ     2$
                                1$:     BIT     #OSC,MR(R5)             ;WAIT FOR OSCILLATOR TO RETURN
                                        BNE     1$
                                        BR      4$
                                2$:     NEG     R3                      ;NEGATE PREV POLARITY INDICATOR
                                3$:     BIT     #OSC,MR(R5)             ;WAIT FOR OSCILLATOR TO RETURN
                                        BEQ     3$                      ;TO '1' STATE
                                4$:     RTS     PC
                                ;SUBROUTINE TO COUNT TIME
                                ;EACH TIME THE OSCILLATOR TOGGLES (BIT <06> IN MR REG) REGISTER
                                ;R4 IS INCREMENTED, AND THE REGISTER R3 IS NEGATED TO INDICATE
                                ;THE LAST STATE OF THE OSCILLATOR.
                                ;CALL    JMP     TIMER(R3)              ;R3 IS SET BY TIMON ROUTINE
                                ;       R2=RETURN ADDRESS TO CALLER
                                ;NOTE: THE TIME TO EXECUTE THIS ROUTINE IS VERY CRITICAL. IT MUST BE
                                ;LESS THAN 40 US.
                                ;ENTER HERE VIA JMP  TIMER(R3) WHEN R3=-24 (PREV STATE=1)
                                TIMER1: BIT     #OSC,MR(R5)             ;BRANCH IF CURRENT STATE IS '0'
                                        BEQ     TIMER                   ;GO INCREMENT TIME
                                        JMP     (R2)                    ;RETURN TO TEST
                                        .=TIMER1+24
                                TIMER:  NEG     R3                      ;NEGATE PREV STATE INDICATOR
                                        INC     R4                      ;INCREMENT 'TICK' COUNT
                                        BMI     TIMERR                  ;BRANCH ON OVERFLOW
                                        JMP     (R2)                    ;RETURN TO TEST
                                TIMERR: TYPE,E.TIMOV                    ;TYPE 'TIMER OVERFLOWED'
                                        HLT                             ;REPORT HARDWARE ERROR
                                        JMP     @SCPADR                 ;RETURN TO BEGINNING OF TEST
                                        .=TIMER+24
                                ;ENTER HERE VIA JMP  TIMER(R3) WHEN R3=+24 (PREV STATE=0)
                                TIMER0: BIT     #OSC,MR(R5)             ;BRANCH IF CURRENT STATE = '1'
                                        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

004502                          TIMOK:
(1) 004502  004767  176050              JSR     PC,.SAVE                ;SAVE REGISTERS ON THE STACK
004506  012700  000070                  MOV     #56.,R0                 ;GET TIME PER TICK
004512  010401                          MOV     R4,R1                   ;GET TICKS COUNT
004514  005002                          CLR     R2                      ;CLEAR SUMMING REGISTERS
004516  005003                          CLR     R3
004520  060002                  1$:     ADD     R0,R2                   ;MULTIPLY TIME PER TICK
004522  005503                          ADC     R3                      ;BY TICK COUNT
004524  005301                          DEC     R1
004526  001374                          BNE     1$
004530  010246                          MOV     R2,-(SP)                ;DIVIDE COUNT BY 10.

004532  010346                          MOV     R3,-(SP)
004534  012746  000012                  MOV     #10.,-(SP)
004540  004767  000262                  JSR     PC,DIVIDE
004544  005726                          TST     (SP)+                   ;DISCARD REMAINDER
004546  012637  001012                  MOV     (SP)+,@#ATIME           ;STORE QUOTIENT
004552  113700  001122                  MOVB    @#TSTNUM,R0             ;GET TEST #
004556  006300                          ASL     R0
004560  006300                          ASL     R0
004562  023760  001012  001416          CMP     @#ATIME,STIMTBL(R0)     ;CHECK THAT TIME IS WITHIN
004570  101004                          BHI     2$                      ;LIMITS SPECIFIED
004572  023760  001012  001420          CMP     @#ATIME,STIMTBL+2(R0)
004600  101001                          BHI     3$
004602  104401                  2$:     HLT+1                           ;CALL ERROR ROUTINE
004604                          3$:
(1) 004604  004767  175770              JSR     PC,.RESTORE             ;RESTORE REGISTERS FROM THE STACK
004610  000207                          RTS     PC                      ;RETURN

                                ;SUBROUTINE TO CHECK INDIVIDUAL GAP TIMES (PRODUCED BY TST021)
                                ;SUBROUTINE COMPUTES THE ACTUAL TIME (IN MICROSECONDS) AND CHECKS
                                ;THAT THE GAP TIME RECORDED BY THE SUBTEST (TST021) BY COMPARING THE
                                ;TIME WITH THE MAX LIMIT (GTIMTBL-GAPTBL(R1)) AND THE MIN LIMIT
                                ;(GTIMTBL+2-GAPTBL(R1)).
                                ;CALL:  MOV     #TICK COUNT,R4          ;R4 CONTAINS TICK COUNT
                                ;       MOVB    #GAP,@#GAP              ;LOCATION GAP CONTAINS GAP #
                                ;       JSR     PC,GAPOK

004612                          GAPOK:
(1) 004612  004767  175740              JSR     PC,.SAVE                ;SAVE REGISTERS ON THE STACK
004616  012700  000070                  MOV     #56.,R0                 ;GET TIME PER TICK
004622  010401                          MOV     R4,R1                   ;GET TICK COUNT
004624  005002                          CLR     R2                      ;CLEAR SUMMING REGISTERS
004626  005003                          CLR     R3
004630  060002                  1$:     ADD     R0,R2                   ;MULTIPLY TICK COUNT
004632  005503                          ADC     R3                      ;BY TIME PER TICK
004634  005301                          DEC     R1
004636  001374                          BNE     1$
```

```
                                                ;DIVIDE TIME BY 10.
004640  010246          MOV     R2,-(SP)
004642  010346          MOV     R3,-(SP)
004644  012746  000012  MOV     #10.,-(SP)
004650  004767  000152  JSR     PC,DIVIDE
004654  005726          TST     (SP)+           ;DISCARD REMAINDER
004656  012637  001012  MOV     (SP)+,@#ATIME   ;STORE QUOTIENT
004662  113703  001120  MOVB    @#GAP,R3        ;GET GAP #
004666  006303          ASL     R3              ;MULTPLY BY 4
004670  006303          ASL     R3              ;TO GET AT TABLE ENTRY
004672  023763  001012  001572  CMP     @#ATIME,GTIMTBL(R3)     ;CHECK TIME (MAX)
004700  101004          BHI     2$
004702  023763  001012  001574  CMP     @#ATIME,GTIMTBL+2(R3)   ;CHECK TIME (MIN)
004710  101002          BHI     3$
004712  104402  2$:     HLT+2                   ;REPORT OUT OF RANGE ERROR
004714  000406          BR      100$
004716  032777  000400  174054  3$:     BIT     #SW08,@SWR      ;BRANCH IF TIMES NOT WANTED
004724  001402          BEQ     100$
004726  004767  176214          JSR     PC,OUTGAP               ;TYPE GAP TIMES

004732  100$:
(1) 004732  004767  175642      JSR     PC,.RESTORE             ;RESTORE REGISTERS FROM THE STACK
004736  000207          RTS     PC                              ;RETURN TO TEST

                        .SBTTL          DELAY SUBROUTINES
                ;THIS SUBROUTINE CAUSES A DELAY OF 350 MS.
004740  004767  177410  DELAY:  JSR     PC,TIMON
004744  010246          MOV     R2,-(SP)                ;SAVE R2 ON THE STACK
004746  012702  004756  MOV     #2$,R2                  ;SET RETURN ADDRESS FOR TIMER
004752          1$:
(1) 004752  000163  004444  JMP     TIMER(R3)           ;GO TO TIMER & RETURN VIA R2
004756  032704  004000  2$:     BIT     #4000,R4
004762  001773          BEQ     1$
004764  012602          MOV     (SP)+,R2                ;RESTORE R2
004766  000207          RTS     PC

                ;THIS SUBROUTINE ALLOWS A CALLER SPECIFIED DELAY (UP TO 65MS.)
                ;CALL:  MOV     DELAY TIME,DELTIM       ;LOAD DELAY TIME (IN US)
                ;       JSR     PC,DELAYV
004770  005767  174120  DELAYV: TST     DELTIM          ;BRANCH IF 0 DELAY
004774  001413          BEQ     3$
004776  004767  177352  JSR     PC,TIMON                ;TURN TIMER ON
005002  010246          MOV     R2,-(SP)                ;SAVE R2 ON THE STACK
005004  012702  005014  MOV     #2$,R2                  ;SET RETURN ADDRESS FROM TIMER
005010          1$:
(1) 005010  000163  004444  JMP     TIMER(R3)           ;GO TO TIMER & RETURN VIA R2
005014  023704  001114  2$:     CMP     @#DELTIM,R4
005020  101373          BHI     1$
005022  012602          MOV     (SP)+,R2                ;RESTORE R2
005024  000207  3$:     RTS     PC

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

005026  005046                  DIVIDE: CLR     -(SP)                   ;SAVE LOC FOR SIGNS
005030  012746  000021                  MOV     #17.,-(SP)              ;SET ITERATION COUNT
005034  016601  000012                  MOV     12(SP),R1               ;GET LSH DIVIDEND
005040  016600  000010                  MOV     10(SP),R0               ;GET MSH DIVIDEND
005044  016602  000006                  MOV     6(SP),R2                ;GET DIVISOR
005050  005402                          NEG     R2                      ;NEGATE DIVISOR
005052  000241                          CLC                             ;CLEAR 'C' BIT IN PSW
005054  000405                          BR      2$
005056  006100                  1$:     ROL     R0                      ;ROTATE MSH DIVIDEND
005060  010003                          MOV     R0,R3                   ;SAVE IN R3
005062  060203                          ADD     R2,R3                   ;SUBTRACT DIVISOR FROM MSH DIVIDEND
005064  103001                          BCC     2$                      ;BRANCH IF DIVIDEND > DIVISOR
005066  010300                          MOV     R3,R0                   ;SAVE REMAINDER IN R0
005070  006101                  2$:     ROL     R1                      ;ROTATE LSH DIVIDEND
005072  005316                          DEC     (SP)                    ;DECREMENT ITERATION COUNT
005074  001370                          BNE     1$
005076  005726                          TST     (SP)+                   ;POP ITERATION COUNTER
005100  005726                          TST     (SP)+                   ;POP SIGN CORRECTION
005102  010166  000006                  MOV     R1,6(SP)                ;PUSH REMAINDER ON STACK
005106  010066  000004                  MOV     R0,4(SP)                ;PUSH QUOTIENT ONTO STACK
005112  012616                          MOV     (SP)+,(SP)
```

```
1019  005114  000207                          RTS     PC
1020
1021                                          .SBTTL  DRIVE SUBROUTINES
1022                                  ;SUBROUTINE TO CHECK IF DRIVE IS AVAILABLE
1023                                  ;CALL:  MOVB    DRIVE#,DRVNUM
1024                                  ;       JSR     PC,DRVAVA
1025                                  ;RETURN:        'C' BIT SET IF NOT AVAILABLE
1026  005116  113765  001004  000010  DRVAVA: MOVB    @#DRVNUM,CS2(R5)        ;LOAD DRIVE #
1027  005124  032765  040000  000026          BIT     #TAP,DT(R5)             ;CHECK IF TAPE UNIT
1028  005132  001003                          BNE     1$
1029  005134  004767  000034                  JSR     PC,RHINIT
1030  005140  000262                          SEV                             ;SET 'V' TO IND NOT AVAIL
1031  005142  000207                  1$:     RTS     PC                      ;RETURN
1032
1033                                  ;SUBROUTINE TO CHECK IF TU16 SLAVE IS AVAILABLE FOR TEST
1034                                  ;CALL:  MOVB    DRIVE #,@#DRVNUM        ;PASS DRIVE # VIA DRVNUM
1035                                  ;       MOVB    SLAVE #,@#SLVNUM        ;PASS SLAVE # VIA SLVNUM
1036                                  ;       JSR     PC,SLVAVA               ;CALL SUBROUTINE
1037  005144  113765  001004  000010  SLVAVA: MOVB    @#DRVNUM,CS2(R5)        ;LOAD DRIVE #
1038  005152  113765  001005  000032          MOVB    @#SLVNUM,TC(R5)         ;AND SLAVE #
1039  005160  032765  002000  000026          BIT     #SPR,DT(R5)             ;BRANCH IF SLAVE PRESENT
1040  005166  001001                          BNE     1$
1041  005170  000262                          SEV                             ;SET 'V' TO INDICATE NO SLAVE
1042  005172  000207                  1$:     RTS     PC
1043
1044                                  ;SUBROUTINE TO  INITIALIZE RH CONTROLLER
1045                                  ;CALL:  JSR PC,RHINIT
1046
1047  005174  012765  000040  000010  RHINIT: MOV     #40,CS2(R5)
1048  005202  113765  001004  000010          MOVB    @#DRVNUM,CS2(R5)
1049  005210  005046                          CLR     -(SP)
1050  005212  113716  001005                  MOVB    @#SLVNUM,(SP)
1051  005216  012665  000032                  MOV     (SP)+,TC(R5)            ;LOAD SLAVE # INTO TC REG
1052  005222  052765  000300  000032          BIS     #NORM11,TC(R5)
1053  005230  000207                          RTS     PC
1054
1055                                  ;SUBROUTINE TO WAIT FOR DRIVE READY (DRY)
1056  005232  005027                  WAITRDY:CLR     (PC)+                   ;CLEAR WAIT TIMER
1057  005234  000000                  WAITTIM:.WORD   0
1058  005236  105765  000012          1$:     TSTB    DS(R5)                  ;WAIT FOR READY TO SET
1059  005242  100406                          BMI     2$
1060  005244  005267  177764                  INC     WAITTIM                 ;INCREMENT WAIT TIMER
1061  005250  001372                          BNE     1$                      ;BRANCH IF TIME HAS NOT EXPIRED
1062  005252  000004  014275                  TYPE,E.TIMEXP                   ;TYPE 'TIME EXPIRED WAITING FOR RDY'
1063  005256  000425                          BR      99$                     ;TAKE ERROR EXIT
1064  005260  032765  002000  000012  2$:     BIT     #EOT,DS(R5)             ;CHECK FOR END OF TAPE
1065  005266  001415                          BEQ     3$                      ;BRANCH IF NO EOT
1066  005270  000004  013330                  TYPE,M.NAM
1067  005274  000004  013657                  TYPE,M.EOT                      ;TYPE 'END OF TAPE'
1068  005300  004767  000032                  JSR     PC,.REWIND              ;REWIND SLAVE
 (1)  005304  102412                          BVS     99$                     ;BRANCH IF ERROR ON REWIND
1069  005306  004767  000106                  JSR     PC,WRITE                ;WRITE A RECORD
1070  005312  005215                          INC     (R5)                    ;SET 'GO' BIT
1071  005314  004767  177712                  JSR     PC,WAITRDY              ;WAIT FOR READY
1072  005320  000404                          BR      99$                     ;TAKE ERROR EXIT
1073  005322  032765  040000  000012  3$:     BIT     #ERR,DS(R5)             ;CHECK ERROR EXIT
```

N04


```
1074  005330  001401                    BEQ     100$
1075  005332  000262            99$:    SEV
1076  005334  000207            100$:   RTS     PC
1077
```

```
1079                            ;SUBROUTINE TO REWIND A UNIT (DRIVE/SLAVE COMBINATION)
1080                            ;CALL   MOVB    DRIVE #,@#DRVNUM
```

```
1084                                    ;       MOVB    SLAVE #,@#SLVNUM
1085                                    ;       JSR     PC,.REWIND
1086                                    ;SUBROUTINE RETURNS TO CALLER WITH SELECTED SLAVE AT 'BOT', & 'V' SET IF
1087                                    ;AN ERROR OCCURS.
1088
1089  005336  004767  177632            .REWIND:JSR     PC,RHINIT               ;INITIALIZE CONTROLLER
```

```
1093  005342  004367  000206                JSR     R3,TMCMD            ;GO TO TM COMMAND SUBROUTINE
1094  005346  000000                        .WORD   0                   ;BUS ADDRESS (NOT USED)
1095  005350  000000                        .WORD   0                   ;WORD COUNT (NOT USED)
1096  005352  000000                        .WORD   0                   ;FRAME COUNT (NOT USED)
1097  005354  000006                        .WORD   RWD                 ;REWIND COMMAND
1098  005356  005215                        INC     (R5)                ;SET 'GO' BIT
1099  005360  032765  000002  000012  1$:   BIT     #BOT,DS(R5)         ;BRANCH IF 'BOT' SET
1100  005366  001005                        BNE     2$
1101  005370  032765  040000  000012        BIT     #ERR,DS(R5)         ;CHECK ERROR BIT
1102  005376  001006                        BNE     99$                 ;BRANCH IF ERROR BIT SET
1103  005400  000767                        BR      1$
1104
1105  005402  032765  020000  000012  2$:   BIT     #PIP,DS(R5)         ;WAIT FOR TAPE MOTION TO STOP
1106  005410  001374                        BNE     2$
1107  005412  000401                        BR      100$
1108  005414  000262                99$:    SEV
1109  005416  000207                100$:   RTS     PC
1110
1111                                  ;SUBROUTINE TO WRITE 256. WORD RECORD
1112                                  ;CALL:  JSR     PC,WRITE
1113
1114  005420  004367  000130          WRITE:  JSR     R3,TMCMD            ;GO TO TM COMMAND SUBROUTINE
1115  005424  015702                        .WORD   WTBUF               ;BUS ADDRESS
1116  005426  177600                        .WORD   WRDCNT              ;WORD COUNT
1117  005430  177400                        .WORD   FRMCNT              ;FRAME COUNT
1118  005432  000060                        .WORD   WFWD                ;WRITE FORWARD COMMAND
1119  005434  000207                        RTS     PC
1120
1121                                  ;SUBROUTINE TO READ A 256. WORD RECORD.
1122                                  ;CALL:  JSR     PC,READ
1123
1124  005436  004337  005554          READ:   JSR     R3,@#TMCMD
1125  005442  015702                        .WORD   RDBUF               ;ADDRESS OF READ BUFFER
1126  005444  177600                        .WORD   WRDCNT              ;2'S COMPLEMENT OF WORD COUNT
1127  005446  177400                        .WORD   FRMCNT              ;2'S COMPLEMENT OF FRAME COUNT
1128  005450  000070                        .WORD   RDFWD               ;READ FORWARD COMMAND
1129  005452  000207                        RTS     PC
1130
1131                                  ;SUBROUTINE TO INITIATE READ REVERSE COMMAND
1132                                  ;CALL:  JSR     PC,REVRD
1133                                  ;
1134  005454  004367  000074          REVRD:  JSR     R3,TMCMD
1135  005460  016302                        .WORD   RDBUF+256.          ;ADDRESS OF READ REVERSE BUFFER
1136  005462  177600                        .WORD   WRDCNT              ;2'S COMPLEMENT OF WORD COUNT
1137  005464  177400                        .WORD   FRMCNT              ;2'S COMPLEMENT OF FRAME COUNT
1138  005466  000076                        .WORD   RDREV               ;READ REVERSE COMMAND
1139  005470  000207                        RTS     PC
1140
1141                                  ;SUBROUTINE TO SPACE FORWARD 1 RECORD
1142  005472  012765  177777  000006  FWDSPC: MOV     #-1,FC(R5)          ;LOAD RECORD COUNT
1143  005500  012715  000031                MOV     #SPCFWD+1,(R5)      ;LOAD COMMAND
1144  005504  004767  177522                JSR     PC,WAITRDY          ;WAIT FOR READY
1145  005510  000207                        RTS     PC                  ;RETURN
1146
1147                                  ;SUBROUTINE TO WRITE A RECORD AND BACK SPACE OVER  THE RECORD.
1148  005512  004767  177702          WRT.BK: JSR     PC,WRITE            ;WRITE THE RECORD
```

```
1149  005516  005215                          INC     (R5)                    ;SET 'GO' BIT
1150  005520  004767  177506                  JSR     PC,WAITRDY
1151  005524  102412                          BVS     2$
1152  005526  012765  177777  000006          MOV     #-1,FC(R5)              ;LOAD RECORD COUNT
1153  005534  012715  000033                  MOV     #SPCREV+1,(R5)          ;LOAD COMMAND
1154  005540  004767  177466                  JSR     PC,WAITRDY
1155  005544  102402                          BVS     2$
1156  005546  004767  177166          1$:     JSR     PC,DELAY                ;WAIT FOR TAPE MOTION TO STOP
1157  005552  000207                  2$:     RTS     PC
1158
1159                                  ;SUBROUTINE TO LOAD A COMMAND
1160                                  ;CALL:  JSR     R3,TMCMD
1161                                  ;       .WORD   BUS ADDRESS
1162                                  ;       .WORD   WORD COUNT (2'S COMPLEMENT)
1163                                  ;       .WORD   FRAME COUNT (2'S COMPLEMENT)
1164                                  ;       .WORD   COMMAND
1165
1166  005554  012365  000004          TMCMD:  MOV     (R3)+,BA(R5)            ;LOAD BUS ADDRESS
1167  005560  012365  000002                  MOV     (R3)+,WC(R5)            ;LOAD WORD COUNT
1168  005564  012365  000006                  MOV     (R3)+,FC(R5)            ;LOAD FRAME COUNT
1169  005570  012315                          MOV     (R3)+,(R5)              ;LOAD COMMAND
1170  005572  000203                          RTS     R3                      ;RETURN
1171
1172                                          ;SUBROUTINE TO PRINT TU16 SERIAL NUMBER
1173                                          ;JSR    PC,SNPT
1174
1175  005574  016503  000030          SNPT:   MOV     SN(R5),R3
1176  005600  012701  001144                  MOV     #ODIGITS,R1
1177  005604  000303                          SWAB    R3
1178  005606  006003                          ROR     R3
1179  005610  006003                          ROR     R3
1180  005612  006003                          ROR     R3
1181  005614  006003                          ROR     R3                      ;GET FIRST DIGIT
1182  005616  042703  177760                  BIC     #177760,R3
1183  005622  052703  000260                  BIS     #260,R3
1184  005626  110321                          MOVB    R3,(R1)+                ;FILL FIRST DIGIT
1185  005630  016503  000030                  MOV     SN(R5),R3
1186  005634  000303                          SWAB    R3
1187  005636  042703  177760                  BIC     #177760,R3
1188  005642  052703  000260                  BIS     #260,R3
1189  005646  110321                          MOVB    R3,(R1)+                ;GET SECOND DIGIT
1190  005650  016503  000030                  MOV     SN(R5),R3
1191  005654  006003                          ROR     R3
1192  005656  006003                          ROR     R3
1193  005660  006003                          ROR     R3
1194  005662  006003                          ROR     R3
1195  005664  042703  177760                  BIC     #177760,R3
1196  005670  052703  000260                  BIS     #260,R3
1197  005674  110321                          MOVB    R3,(R1)+                ;GET THIRD DIGIT
1198  005676  016503  000030                  MOV     SN(R5),R3
1199  005702  042703  177760                  BIC     #177760,R3
1200  005706  052703  000260                  BIS     #260,R3
1201  005712  110321                          MOVB    R3,(R1)+                ;GET FOURTH DIGIT
1202  005714  105011                          CLRB    (R1)
1203  005716  000004  001144                  TYPE,ODIGITS                    ;TYPE SERIAL NUMBER
1204  005722  000207                          RTS     PC                      ;RETURN
```

```
1205
1206                                            .SBTTL  PROGRAM INITIALIZATION
1207  005724  012706  000600          INIT:     MOV     #STKPTR,SP              ;SET STACK PTR
1208
1209  005730  013746  000006          SUSWR:    MOV     @#6,-(SP)               ;SAVE VECTORS
1210  005734  013746  000004                    MOV     @#4,-(SP)
1211  005740  012737  005760  000004            MOV     #61$,@#4                        ;SET UP FOR TIMEOUT
1212  005746  022777  177777  173024            CMP     #-1,@SWR                ;REFERENCE HARDWARE SWITCH REGISTER
1213  005754  001402                           BEQ     60$
1214  005756  000404                           BR      62$
1215  005760  022626                  61$:     CMP     (SP)+,(SP)+             ;ADJUST STACK
1216  005762  012767  000176  173010  60$:     MOV     #SWREG,SWR              ;POINT TO SOFTWARE SWITCH REG
1217  005770  012637  000004          62$:     MOV     (SP)+,@#4               ;RESTORE VECTORS
1218  005774  012637  000006                    MOV     (SP)+,@#6
1219  006000  022737  000176  001000            CMP     #SWREG,@#SWR
1220  006006  001002                           BNE     64$
1221  006010  004767  174014                   JSR     PC,CNTLU
1222  006014  105037  001124          64$:     CLRB    @#PRGFLG                ;CLEAR PROGRAM FLAG
1223  006020  105037  001121                   CLRB    @#ITCNT                 ;CLEAR ITERATION COUNT
1224  006024  105037  001122                   CLRB    @#TSTNUM                ;SET TEST # 0
1225  006030  105037  001123                   CLRB    @#ERFLG                 ;CLEAR ERROR FLAG
1226  006034  105067  173070                   CLRB    ASFLG                   ;CLEAR ASK FLAG
1227  006040  012737  000006  000004            MOV     #ERRVEC+2,@#ERRVEC
1228  006046  012737  000002  000006            MOV     #RTI,@#ERRVEC+2         ;CHECK IF 'LP' IS AVAILABLE
1229  006054  005037  001264          2$:      CLR     @#INBUF
1230  006060  000004  001374                   TYPE,CRLF
1231  006064  000004  013330                   TYPE,M.NAM                      ;TYPE TITLE
1232  006070  000004  013377                   TYPE,I.REG                      ;ASK USER TO TYPE CONT BASE ADRS
1233  006074  004767  175374                   JSR     PC,.INPUT               ;GET USER INPUT
1234  006100  004767  175154          4$:      JSR     PC,CNVTAO               ;CONVERT ASCII TO OCTAL
1235  006104  013737  001116  001010            MOV     @#OCTALO,@#TMBASE       ;SET NEW ADDRESS
1236  006112  013705  001010          5$:      MOV     @#TMBASE,R5
1237
1238                                    ;ROUTINE TO CHECK IF CONTROLLER (RH11) IS AVAILAABLE
1239  006116  000261                           SEC                             ;SET 'C' IN PSW
1240  006120  005715                           TST     (R5)                    ;BRANCH IF CONTROLLER AVAIL
1241  006122  103003                           BCC     6$
1242  006124  000004  013716                   TYPE,E.NCON
1243  006130  000675                           BR      INIT
1244  006132  012737  003650  000004  6$:      MOV     #ERRTRP,@#ERRVEC        ;SET ERROR TRAP VECTOR
```

```
1246
1247  006140  105037  001123          ;ROUTINE TO GET TM02 DRIVES USER DESIRES TO TEST
1248  006144  012701  001154  DRIVES: CLRB    @#ERFLG                 ;CLEAR ERROR FLAG
1249  006150  012700  000004          MOV     #DRVTBL,R1              ;MARK ALL DRIVES AS NOT TO
1250  006154  005021                  MOV     #4,R0                   ;BE TESTED. A '0' INDICATES
1251  006156  005300          1$:     CLR     (R1)+                   ;THAT A DRIVE IS NOT TO BE
1252  006160  001375                  DEC     R0                      ;TESTED
1253  006162  000004  013444          BNE     1$
1254  006166  004767  175302          TYPE,I.DRVS
1255  006172  012700  001264          JSR     PC,.INPUT               ;GET USER INPUT
1256  006176  122710  000101          MOV     #INBUF,R0
1257  006202  001013                  CMPB    #'A,(R0)                ;AN 'A' SPECIFIES ALL
1258  006204  110667  172714          BNE     3$                      ;DRIVES TO BE TESTED
1259  006210  012701  001154          MOVB    SP,PRGFLG               ;SET FLAG TO IND ALL DRIVES
1260  006214  012700  000004          MOV     #DRVTBL,R1              ;MARK ALL DRIVES TO BE TESTED
1261  006220  012721  177777          MOV     #4,R0                   ;A '-1' INDICATES THAT A DRIVE
1262  006224  005300          2$:     MOV     #-1,(R1)+               ;IS TO BE TESTED
1263  006226  001374                  DEC     R0
1264  006230  000417                  BNE     2$
1265                                  BR      CHKDRV                  ;GO CHECK DRIVE AVAILABILITY
1266
1267  006232  122710  000015          ;GET USER SELECTED DRIVES AND MARK EACH DRIVE SELECTED TO BE TESTED
1268  006236  001414          3$:     CMPB    #CR,(R0)
1269  006240  121027  000054          BEQ     CHKDRV
1270  006244  001001                  CMPB    (R0),#',                ;CHECK IF 'COMMA'
1271  006246  105720                  BNE     4$
1272  006250  112001                  TSTB    (R0)+                   ;STEP PTR PAST 'COMMA'
1273  006252  042701  177770  4$:     MOVB    (R0)+,R1
1274  006256  112761  177777  001154  BIC     #177770,R1
1275  006264  000240                  MOVB    #-1,DRVTBL(R1)
1276  006266  000761                  NOP
1277                                  BR      3$
1278
1279  006270  005000          ;ASCERTAIN THAT DRIVES (TM02'S) SPECIFIED ARE AVAILABLE
1280  006272  105760  001154  CHKDRV: CLR     R0                      ;A 0/-1 INDICATES THAT THE
1281  006276  001005          1$:     TSTB    DRVTBL(R0)              ;DRIVE IS NOT/IS TO BE TESTED
1282  006300  005200                  BNE     3$
1283  006302  122700  000010  2$:     INC     R0
1284  006306  001371                  CMPB    #8.,R0
1285  006310  000421                  BNE     1$
1286  006312  110037  001004          BR      4$
1287  006316  004737  005116  3$:     MOVB    R0,@#DRVNUM
1288  006322  102366                  JSR     PC,@#DRVAVA             ;CHECK IF AVAILABLE
1289  006324  000004  013763          BVC     2$                      ;'V' BIT SET INDICATES NOT AVAIL
1290  006330  116037  001132  014015  TYPE,E.NDRV
1291  006336  000004  014015          MOVB    DIGTAB(R0),@#E.NAVA     ;SET DRIVE # IN MESSAGE
1292  006342  110637  001123          TYPE,E.NAVA
1293  006346  105060  001154          MOVB    SP,@#ERFLG              ;SET 'ERROR' FLAG
1294  006352  000752                  CLRB    DRVTBL(R0)              ;MARK DRIVE UNAVAILABLE
1295  006354  105737  001123          BR      2$                      ;CHECK NEXT DRIVE
1296  006360  001403          4$:     TSTB    @#ERFLG                 ;GO GET SLAVES IF NO ERROR
1297  006362  105737  001124          BEQ     SLAVES
1298  006366  001664                  TSTB    @#PRGFLG                ;ASK USER TO RETYPE DRIVES IF
1299                                  BEQ     DRIVES                  ;'ALL' NOT SPECIFIED
1300
1301  006370  105037  001123          ;ROUTINE TO GET SLAVES (TU16'S) USER DESIRES TO TEST
                                SLAVES: CLRB    @#ERFLG                 ;CLEAR 'ERROR' FLAG
```

```
1302  006374  012701  001164            MOV     #SLVTBL,R1          ;MARK ALL SLAVES (64.) AS NOT
1303  006400  012700  000040            MOV     #32.,R0             ;TO BE TESTED.A 0 INDICATES THAT
1304  006404  005021            1$:     CLR     (R1)+               ;A DRIVE'S SLAVE IS NOT TO BE
1305  006406  005300                    DEC     R0                  ;TESTED
1306  006410  001375                    BNE     1$
1307  006412  005000                    CLR     R0                  ;R0 = DRIVE # FOR SLAVES
1308  006414  012701  001164            MOV     #SLVTBL,R1          ;R1 POINTS TO DRIVE'S SLAVE
1309  006420  105760  001154    2$:     TSTB    DRVTBL(R0)          ;IF DRIVE IS TO BE TESTED
1310  006424  001007                    BNE     4$                  ;GO TO 4$ OTHERWISE
1311  006426  062701  000010    3$:     ADD     #8.,R1              ;STEP SLAVE PTR TO NEXT DRIVE'S
1312  006432  005200                    INC     R0                  ;SLAVES AND INCREMENT DRIVE #
1313  006434  122700  000010            CMPB    #8.,R0              ;CHECK ALL DRIVES
1314  006440  001367                    BNE     2$                  ;AND WHEN ALL DRIVES CHECKED
1315  006442  000454                    BR      CHKSLV              ;GO CHECK SLAVE AVAILABILITY
1316
1317  006444  105737  001124    4$:     TSTB    @#PRGFLG            ;BRANCH IF USER SELECTED ALL
1318  006450  001020                    BNE     5$                  ;DRIVES
1319  006452  110067  172326            MOVB    R0,DRVNUM           ;GET DRIVE #
1320  006456  116037  001132  013525    MOVB    DIGTAB(R0),@#I.DRV  ;PREPARE USER ACTION MESSAGE
1321  006464  000004  013506            TYPE,I.SLVS
1322  006470  004767  175000            JSR     PC,.INPUT           ;GET USER INPUT
1323  006474  012703  001264            MOV     #INBUF,R3           ;SET PTR TO USER INPUT
1324  006500  122710  000101            CMPB    #'A,(R0)            ;BRANCH IF USER DOES NOT WANT
1325  006504  001015                    BNE     7$                  ;'ALL' SLAVES
1326  006506  110637  001124            MOVB    SP,@#PRGFLG         ;SET 'ALL' INDICATOR
1327  006512  012701  001164    5$:     MOV     #SLVTBL,R1          ;MARK ALL SLAVES FOR ALL
1328  006516  012700  000040            MOV     #32.,R0             ;DRIVES AS TO BE TESTED
1329  006522  012721  177777    6$:     MOV     #-1,(R1)+
1330  006526  005300                    DEC     R0
1331  006530  001374                    BNE     6$
1332  006532  105737  001124            TSTB    @#PRGFLG            ;BRANCH IF ALL WAS SELECTED
1333  006536  001016                    BNE     CHKSLV
1334
1335  006540  122713  000015    7$:     CMPB    #CR,(R3)            ;GET USER SELECTED SLAVES FOR
1336  006544  001730                    BEQ     3$                  ;DRIVE
1337  006546  121327  000054            CMPB    (R3),#',            ;STEP PTR PAST 'COMMA'
1338  006552  001001                    BNE     8$
1339  006554  105723                    TSTB    (R3)+
1340  006556  112304    8$:     MOVB    (R3)+,R4            ;AND MARK SELECED SLAVE
1341  006560  042704  177770            BIC     #177770,R4          ;AS TO BE TESTED
1342  006564  060104                    ADD     R1,R4
1343  006566  112714  177777            MOVB    #-1,(R4)
1344  006572  000762                    BR      7$
1345
1346                                ;ASCERTAIN THAT SLAVES (TU16'S) SELECTED ARE AVAILABLE
1347  006574  005000            CHKSLV: CLR     R0                  ;R0 WILL CONTAIN THE DRIVE #
1348  006576  005001                    CLR     R1                  ;AND R1 THE SLAVE #
1349  006600  012702  001164            MOV     #SLVTBL,R2          ;SET PTR TO SLAVE TABLE
1350  006604  105760  001154    1$:     TSTB    DRVTBL(R0)          ;BRANCH IF DRIVE SELECTED
1351  006610  001007                    BNE     3$                  ;& AVAILABLE FOR TEST
1352  006612  005200    2$:     INC     R0                  ;INCREMENT DRIVE #
1353  006614  062702  000010            ADD     #8.,R2              ;STEP SLAVE PTR TO NEXT DRIVE'S
1354  006620  022700  000010            CMP     #8.,R0              ;SLAVES. BRANCH TO 1$ IF NOT ALL
1355  006624  001367                    BNE     1$                  ;DRIVES CHECKED OTHERWISE EXIT
1356  006626  000434                    BR      7$
1357
```

```
1358  006630  005001                  3$:     CLR     R1                      ;SET SLAVE # 0
1359  006632  105712                  4$:     TSTB    (R2)                    ;BRANCH IF DRIVE'S SLAVE IS SEL-
1360  006634  001006                          BNE     6$                      ;ECTED FOR TEST
1361  006636  005201                  5$:     INC     R1                      ;INCREMENT SLAVE #
1362  006640  005202                          INC     R2                      ;STEP PTR TO NEXT SLAVE
1363  006642  022701  000010                  CMP     #8.,R1                  ;GO TO 4$ IF ALL SLAVES NOT
1364  006646  001371                          BNE     4$                      ;CHECKED
1365  006650  000760                          BR      2$                      ;OTHERWISE GO TO 2$ ABOVE
1366
1367  006652  110037  001004          6$:     MOVB    R0,@#DRVNUM             ;PASS DRIVE & SLAVE #
1368  006656  110137  001005                  MOVB    R1,@#SLVNUM
1369  006662  004737  005144                  JSR     PC,@#SLVAVA             ;AND CHECK IF AVAILABLE
1370  006666  102363                          BVC     5$                      ;'V' BIT SET ON RETURN IND-
1371  006670  116037  001132  014005          MOVB    DIGTAB(R0),@#E.DRV      ;ICATES ERROR. PREPARE ERROR
1372  006676  116137  001132  014015          MOVB    DIGTAB(R1),@#E.NAVA     ;MESSAGE
1373  006704  000004  013777                  TYPE,E.NSLV
1374  006710  110637  001123                  MOVB    SP,@#ERFLG              ;SET ERROR INDICATOR
1375  006714  105012                          CLRB    (R2)                    ;CLEAR SLAVE TABLE ENTRY
1376  006716  000747                          BR      5$                      ;GET NEXT SLAVE
1377
1378  006720  105737  001123          7$:     TSTB    @#ERFLG                 ;BRANCH IF NO ERROR
1379  006724  001403                          BEQ     100$
1380  006726  105737  001124                  TSTB    @#PRGFLG                ;BRANCH IF NOT 'ALL'
1381  006732  001616                          BEQ     SLAVES                  ;ASK USER TO RETYPE SLAVES
1382  006734  012737  003650  000004  100$:   MOV     #ERRTRP,@#ERRVEC
1383
1384                                  ;SCAN DIVE AND SLAVE TABLE FOR DRIVE/SLAVE COMBINATION TO TEST
1385  006742  105037  001004                  CLRB    @#DRVNUM                ;SET DRIVE AND SLAVE # 0
1386  006746  105037  001005                  CLRB    @#SLVNUM
1387  006752  012737  001164  001006          MOV     #SLVTBL,@#SLVPTR        ;SET PTR TO SLAVE TABLE
1388  006760  105037  001125                  CLRB    @#UNTFND                ;CLEAR 'UNIT FOUND' IND.
1389
1390  006764  113700  001004          BEGIN:  MOVB    @#DRVNUM,R0             ;GET DRIVE #
1391  006770  113701  001005                  MOVB    @#SLVNUM,R1             ;AND SLAVE #
```

```
1394  006774  013702  001006          MOV     @#SLVPTR,R2             ;GET SLAVE PTR
1395  007000  105760  001154  1$:     TSTB    DRVTBL(R0)              ;BRANCH IF DRIVE AVAIL TO TEST
1396  007004  001011                  BNE     3$
1397  007006  005001                  CLR     R1                      ;CLEAR SLAVE #
1398  007010  062702  000010          ADD     #8.,R2                  ;AND STEP PTR TO NEXT DRIVE'S
1399  007014  005200          2$:     INC     R0                      ;SLAVES AND INCREMENT DRIVE #
1400  007016  022700  000010          CMP     #8.,R0                  ;EXIT TEST IF ALL DRIVES
1401  007022  001366                  BNE     1$                      ;CHECKED OTHERWISE CONTINUE
1402  007024  000137  012732          JMP     @#END                   ;SCAN FOR NEXT 'UNIT'
1403
1404  007030  105712          3$:     TSTB    (R2)                    ;BRANCH IF SLAVE ON DRIVE IS
1405  007032  001007                  BNE     4$                      ;AVAILABLE THERWISE STEP
1406  007034  005202                  INC     R2                      ;PTR TO NEXT SLAVE
1407  007036  005201                  INC     R1                      ;INCREMENT SLAVE #
1408  007040  122701  000010          CMPB    #8.,R1                  ;UNTIL ALL SLAVES CHECKED
1409  007044  001371                  BNE     3$                      ;WHEN ALL SLAVES CHECKED
1410  007046  005001                  CLR     R1                      ;SET SLAVE # 0
1411  007050  000761                  BR      2$                      ;AND CONTINUE SCAN
1412
1413  007052  110637  001125  4$:     MOVB    SP,@#UNTFND             ;INDICATE THAT A 'UNIT' IS FOUND
1414  007056  110037  001004          MOVB    R0,@#DRVNUM             ;SET DRIVE 3
```

```
1419  007062  110137  001005          MOVB    R1,@#SLVNUM             ;SET SLAVE #
1420  007066  010237  001006          MOV     R2,@#SLVPTR             ;SAVE SLAVE PTR
1421
1422  007072  105737  001130    5$:   TSTB    @#ASFLG
1423  007076  001044                  BNE     7$
1424  007100  112767  000001  172022  MOVB    #1,ASFLG
1425
1426  007106  105037  001124          CLRB    @#PRGFLG                ;CLEAR PROGRAM INDICATOR
1427  007112  000004  013565          TYPE,I.SKEW                     ;ASK USER IF HE WANTS TO RUN SKEW TESTS
1428  007116  004767  174352          JSR     PC,.INPUT                       ;GET USER INPUT
1429  007122  012703  001264          MOV     #INBUF,R3               ;GET REPLY
1430  007126  122713  000060          CMPB    #'0,(R3)                ;BRANCH IF 'NO' (0)
1431  007132  001406                  BEQ     6$
1432  007134  122713  000061          CMPB    #'1,(R3)                ;CHECK IF 'YES' (1)
1433  007140  001354                  BNE     5$                      ;NEITHER SO ASK AGAIN
1434  007142  111337  001124          MOVB    (R3),@#PRGFLG           ;SET INDICATOR
1435  007146  000420                  BR      7$
1436
1437  007150  105037  001127    6$:   CLRB    @#NRZFLG                ;CLEAR NRZ INDICATOR
1438  007154  000004  013626          TYPE,I.NRZ                      ;ASK USER IF DRIVE 'NRZ' ONLY
1439  007160  004767  174310          JSR     PC,.INPUT                       ;GET USER INPUT
1440  007164  012703  001264          MOV     #INBUF,R3               ;GET REPLY
1441  007170  122713  000060          CMPB    #'0,(R3)                ;BRANCH IF 'NO' (0)
1442  007174  001405                  BEQ     7$
1443  007176  122713  000061          CMPB    #'1,(R3)                ;CHECK IF 'YES' (1)
1444  007202  001362                  BNE     6$                      ;ASK AGAIN IF NEITHER
1445  007204  111337  001127          MOVB    (R3),@#NRZFLG           ;SET INDICATOR
1446  007210                    7$:
1447
1448  007210  052737  000100  177560 TYPHDR: BIS     #100,@#TKS              ;SET KEYBOARD IE BIT
1449  007216  000004  014346          TYPE,L.HDR1
1450  007222  116037  001132  014526        MOVB    DIGTAB(R0),@#L.DRV      ;SET DRIVE #
1451  007230  116137  001132  014540        MOVB    DIGTAB(R1),@#L.SLV      ;AND SLAVE #
1452  007236  112737  000071  014543        MOVB    #'9,@#L.CHAN            ;GET SLAVES CHANNEL TYPE
1453  007244  032765  010000  000026        BIT     #CH7,DT(R5)
1454  007252  001403                        BEQ     1$
1455  007254  112737  000067  014543        MOVB    #'7,@#L.CHAN            ;SET 7 CHANNEL
1456  007262  000004  014461          1$:   TYPE,L.HDR2
1457  007266  004767  176302          JSR     PC,SNPT         ;GO PRINT SERIAL NUMBER
1458  007272  000004  014562          TYPE,L.HDR3
1459  007276  012737  007336  001002        MOV     #TST001,@#SCPADR        ;SET 'SCOPE' ADDRESS FOR FIRST TEST
1460  007304  010500                        MOV     R5,R0
1461  007306  062700  000006                ADD     #FC,R0                  ;R0 CONTAINS ADDRESS OF FC REG
1462  007312  010501                        MOV     R5,R1
1463  007314  062701  000012                ADD     #DS,R1                  ;R1 CONTAINS ADDRESS OF DS REG
1464  007320  012703  004444                MOV     #TIMER,R3               ;SET JUMP ADDRESS TO TIMER
1465  007324  105737  001124                TSTB    @#PRGFLG                ;BRANCH IF NOT SKEW TESTS
1466  007330  001402                        BEQ     TST001
1467  007332  000137  012764                JMP     @#SKEWTST
```

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

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

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

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

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

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

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

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

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

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

```
1524  007526  000772                          BR      1$
1525
1526  007530                          2$:
 (1)  007530  004767  174620                  JSR     PC,TIMON                ;TURN TIMER ON
1527  007534  010702                          MOV     PC,R2                   ;LOAD RETURN PC FROM TIMER
1528  007536  032711  000020          3$:     BIT     #SDWN,(R1)              ;BRANCH WHEN DS <SDWN> SETS
1529  007542  001002                          BNE     4$
1530  007544  000163  004444                  JMP     TIMER(R3)               ;GO TO TIMER & RETURN VIA R2
1531
1532  007550  004767  175456          4$:     JSR     PC,WAITRDY              ;WAIT FOR READY
1533  007554  102403                          BVS     99$
1534  007556  004767  174720                  JSR     PC,TIMOK                ;GO CHECK TIME RECORDED
1535  007562  000401                          BR      100$
1536  007564  104400                  99$:    HLT                             ;REPORT ERROR
1537  007566  104000                  100$:   SCOPE
1538
1539                                  ;TEST 004 - WRITE SETTLEDOWN
1540                                  ;THIS TEST MEASURES TIME FROM 'SWDN'=1 TO 'SWDN'=0.
1541  007570  112737  000004  001122 TST004: MOVB    #4,@#TSTNUM
1542  007576  004767  175616                  JSR     PC,WRITE
1543  007602  005215                          INC     (R5)                    ;SET 'GO' BIT
1544
1545  007604  005710                  1$:     TST     (R0)                    ;BRANCH WHEN WRITING FINISHED
1546  007606  001404                          BEQ     2$
1547  007610  032711  040000                  BIT     #ERR,(R1)               ;CHECK ERROR BIT
1548  007614  001025                          BNE     99$
1549  007616  000772                          BR      1$
1550
1551  007620  032711  000020          2$:     BIT     #SDWN,(R1)              ;WAIT FOR ASSERTION OF 'SDWN'
1552  007624  001004                          BNE     3$
1553  007626  032711  040000                  BIT     #ERR,(R1)               ;MONITOR ERROR BIT
1554  007632  001017                          BNE     99$
1555  007634  000771                          BR      2$
1556
1557  007636                          3$:
 (1)  007636  004767  174512                  JSR     PC,TIMON                ;TURN TIMER ON
1558  007642  010702                          MOV     PC,R2                   ;SET RETURN PC FROM TIMER
1559  007644  032711  000020                  BIT     #SDWN,(R1)              ;BRANCH WHEN SWDN CLEARS
1560  007650  001402                          BEQ     5$
1561  007652  000163  004444                  JMP     TIMER(R3)               ;GO TO TIMER & RETURN VIA R2
1562
1563  007656  004767  175350          5$:     JSR     PC,WAITRDY              ;WAIT FOR READY
1564  007662  102403                          BVS     99$
1565  007664  004767  174612                  JSR     PC,TIMOK
1566  007670  000401                          BR      100$
1567
1568  007672  104400                  99$:    HLT
1569  007674  104000                  100$:   SCOPE
1570
1571                                  ;TEST 005 - READ FROM BOT
1572                                  ;THIS TEST MEASURES TIME FROM 'GO'=1 TO 'ACCL'=0.
1573  007676  112737  000005  001122 TST005: MOVB    #5,@#TSTNUM             ;SET TEST #5
1574  007704  004767  175426                  JSR     PC,.REWIND              ;REWIND SLAVE
 (1)  007710  102422                          BVS     99$                     ;BRANCH IF ERROR ON REWIND
1575  007712  004767  175520                  JSR     PC,READ
1576  007716  012702  007730                  MOV     #1$,R2                  ;SET RETURN PC FROM TIMER
```

```
1577  007722  004767  174426            JSR     PC,TIMON                ;TURN TIMER ON
1578  007726  005215                    INC     (R5)                    ;SET 'GO' BIT
1579
1580  007730  005765  000032    1$:     TST     TC(R5)                  ;BRANCH WHEN 'ACCL' RESETS
1581  007734  100002                    BPL     2$
1582  007736  000163  004444            JMP     TIMER(R3)               ;GO TO TIMER & RETURN VIA R2
1583
1584  007742  004767  175264    2$:     JSR     PC,WAITRDY              ;WAIT FOR READY
1585  007746  102403                    BVS     99$                     ;BRANCH IF ERROR
1586  007750  004767  174526            JSR     PC,TIMOK                ;CHECK RECORDED TIME
1587  007754  000401                    BR      100$
1588
1589  007756  104400            99$:    HLT
1590  007760  104000            100$:   SCOPE
1591
1592                            ;TEST 006 - READ START
1593                            ;THIS TEST MEASURES TIME FROM 'GO'=1 TO 'ACCL'=0.
1594  007762  112737  000006  001122  TST006: MOVB    #6,@#TSTNUM             ;SET TEST #6
1595  007770  004767  175516            JSR     PC,WRT.BK               ;WRITE A RECORD & BACK SPACE
1596  007774  102422                    BVS     99$
1597  007776  004767  175434            JSR     PC,READ
1598  010002  012702  010014            MOV     #1$,R2                  ;SET RETURN PC FROM TIMER
1599  010006  004767  174342            JSR     PC,TIMON                ;TURN TIMER ON
1600  010012  005215                    INC     (R5)                    ;SET 'GO' BIT
1601
1602  010014  005765  000032    1$:     TST     TC(R5)                  ;BRANCH WHEN 'ACCL' RESETS
1603  010020  100002                    BPL     2$
1604  010022  000163  004444            JMP     TIMER(R3)               ;GO TO TIMER & RETURN VIA R2
1605
1606  010026  004767  175200    2$:     JSR     PC,WAITRDY
1607  010032  102403                    BVS     99$
1608  010034  004767  174442            JSR     PC,TIMOK
1609  010040  000401                    BR      100$
1610
1611  010042  104400            99$:    HLT
1612  010044  104000            100$:   SCOPE
1613
1614                            ;TEST 007 - READ SHUTDOWN
1615                            ;THIS TEST MEASURES TIME FROM 'FC REG'=FRAME COUNT TO 'SWDN'=1.
1616  010046  112737  000007  001122  TST007: MOVB    #7,@#TSTNUM             ;SET TEST #7
1617  010054  004767  175432            JSR     PC,WRT.BK               ;WRITE A RECORD & BACK SPACE
1618  010060  102430                    BVS     99$                     ;BRANCH IF ERROR
1619  010062  004767  175350            JSR     PC,READ
1620  010066  005215                    INC     (R5)                    ;SET 'GO' BIT
1621
1622  010070  022710  000400    1$:     CMP     #-FRMCNT,(R0)           ;WAIT FOR FRAME COUNT TO
1623  010074  001404                    BEQ     2$                      ;= # OF FRAMES WRITTEN
1624  010076  032711  040000            BIT     #ERR,(R1)               ;MONITOR ERROR BIT
1625  010102  001017                    BNE     99$
1626  010104  000771                    BR      1$
1627
1628  010106                    2$:
 (1)  010106  004767  174242            JSR     PC,TIMON                ;TURN TIMER ON
1629  010112  010702                    MOV     PC,R2                   ;SET RETURN PC FROM TIMER
1630  010114  032711  000020            BIT     #SDWN,(R1)              ;BRANCH WHEN SDWN SETS
1631  010120  001002                    BNE     3$
```

```
1632  010122  000163  004444                    JMP     TIMER(R3)               ;GO TO TIMER & RETURN VIA R2
1633
1634  010126  004767  175100          3$:       JSR     PC,WAITRDY
1635  010132  102403                            BVS     99$
1636  010134  004767  174342                    JSR     PC,TIMOK
1637  010140  000401                            BR      100$
1638
1639  010142  104400                99$:        HLT                             ;REPORT ERROR
1640  010144  104000                100$:       SCOPE
1641
1642                                ;TEST 010 - READ SETTLEDOWN
1643                                ;THIS TEST MEASURES TIME FROM 'SWDN'=1 TO 'SWDN'=0.
1644  010146  112737  000010  001122 TST010:    MOVB    #10,@#TSTNUM            ;SET TEST #10
1645  010154  012702  010232                    MOV     #4$,R2                  ;SET RETURN PC FROM TIMER
1646  010160  004767  175326                    JSR     PC,WRT.BK               ;WRITE A RECORD & BACK SPACE
1647  010164  102436                            BVS     99$
1648  010166  004767  175244                    JSR     PC,READ
1649  010172  005215                            INC     (R5)                    ;SET 'GO' BIT
1650
1651  010174  105711                1$:         TSTB    (R1)                    ;WAIT FOR READY
1652  010176  100404                            BMI     2$                      ;BRANCH WHEN SET
1653  010200  032711  040000                    BIT     #ERR,(R1)               ;CHECK ERROR BIT
1654  010204  001026                            BNE     99$
1655  010206  000772                            BR      1$
1656
1657  010210  032711  000020          2$:       BIT     #SDWN,(R1)              ;WAIT FOR ASSERTION OF 'SDWN'
1658  010214  001004                            BNE     3$
1659  010216  032711  040000                    BIT     #ERR,(R1)               ;MONITOR ERROR BIT
1660  010222  001017                            BNE     99$
1661  010224  000771                            BR      2$
1662
1663  010226                            3$:
 (1)  010226  004767  174122                    JSR     PC,TIMON                ;TURN TIMER ON
1664  010232  032765  000020  000012  4$:       BIT     #SDWN,DS(R5)            ;WAIT FOR NEGATION OF SDWN
1665  010240  001402                            BEQ     5$
1666  010242  000163  004444                    JMP     TIMER(R3)               ;GO TO TIMER & RETURN VIA R2
1667
1668  010246  004767  174760          5$:       JSR     PC,WAITRDY
1669  010252  102403                            BVS     99$
1670  010254  004767  174222                    JSR     PC,TIMOK
1671  010260  000401                            BR      100$
1672
1673  010262  104400                99$:        HLT
1674  010264  104000                100$:       SCOPE
1675
1676
1677                                ;TEST 011-READ REVERSE START
1678                                ;THIS TEST MEASURES TIME FROM 'GO'=1 TO 'ACCL'=0.
1679  010266  112737  000011  001122 TST011:    MOVB    #11,@#TSTNUM
1680  010274  012702  010332                    MOV     #1$,R2                  ;SET RETURN PC FROM TIMER
1681  010300  004767  175114                    JSR     PC,WRITE                ;WRITE A RECORD
1682  010304  005215                            INC     (R5)                    ;SET 'GO' BIT
1683  010306  004767  174720                    JSR     PC,WAITRDY
1684  010312  102422                            BVS     99$
1685  010314  004767  174420                    JSR     PC,DELAY                ;WAIT FOR TAPE MOTION TO STOP
1686  010320  004767  175130                    JSR     PC,REVRD
```

```
1687  010324  004767  174024          JSR     PC,TIMON                ;TURN TIMER ON
1688  010330  005215                  INC     (R5)                    ;SET 'GO' BIT
1689
1690  010332  005765  000032  1$:     TST     TC(R5)                  ;BRANCH WHEN 'ACCL' = 0
1691  010336  100002                  BPL     2$
1692  010340  000163  004444          JMP     TIMER(R3)               ;GO TO TIMER & RETURN VIA R2
1693
1694  010344  004767  174662  2$:     JSR     PC,WAITRDY
1695  010350  102403                  BVS     99$                     ;BRANCH IF ERROR
1696  010352  004767  174124          JSR     PC,TIMOK
1697  010356  000401                  BR      100$
1698
1699  010360  104400          99$:    HLT
1700  010362  104000          100$:   SCOPE
1701
1702                          ;TEST 012-READ REVERSE SHUTDOWN
1703                          ;THIS TEST MEASURES TIME FROM 'FC REG' = FRAME COUNT TO 'SDWN'=1.
1704  010364  112737  000012  001122  TST012: MOVB    #12,@#TSTNUM
1705  010372  012702  010442          MOV     #3$,R2                  ;SET RETURN PC FROM TIMER
1706  010376  004767  175016          JSR     PC,WRITE                ;WRITE A RECORD
1707  010402  005215                  INC     (R5)                    ;SET 'GO' BIT
1708  010404  004767  174622          JSR     PC,WAITRDY
1709  010410  102427                  BVS     99$
1710  010412  004767  175036          JSR     PC,REVRD
1711  010416  005215                  INC     (R5)                    ;SET 'GO' BIT
1712
1713  010420  022710  000400  1$:     CMP     #-FRMCNT,(R0)           ;BRANCH WHEN FRAME COUNT
1714  010424  001404                  BEQ     2$                      ;= # OF RECORD WRITTEN
1715  010426  032711  040000          BIT     #ERR,(R1)               ;MONITOR ERROR BIT IN 'DS' REG
1716  010432  001016                  BNE     99$
1717  010434  000771                  BR      1$
1718
1719  010436          2$:
 (1)  010436  004767  173712          JSR     PC,TIMON                ;TURN TIMER ON
1720  010442  032711  000020  3$:     BIT     #SDWN,(R1)              ;BRANCH WHEN SDWN SETS
1721  010446  001002                  BNE     4$
1722  010450  000163  004444          JMP     TIMER(R3)               ;GO TO TIMER & RETURN VIA R2
1723
1724  010454  004767  174552  4$:     JSR     PC,WAITRDY              ;WAIT FOR READY
1725  010460  102403                  BVS     99$
1726  010462  004767  174014          JSR     PC,TIMOK
1727  010466  000401                  BR      100$
1728
1729  010470  104400          99$:    HLT
1730  010472  104000          100$:   SCOPE
1731
1732                          ;TEST 013-READ REVERSE SETTLEDOWN
1733                          ;THIS TEST MEASURES TIME FROM 'SDWN'=1 TO 'SDWN'=0.
1734  010474  112737  000013  001122  TST013: MOVB    #13,@#TSTNUM
1735  010502  012702  010566          MOV     #4$,R2                  ;SET RETURN PC FROM TIMER
1736  010506  004767  174706          JSR     PC,WRITE                ;WRITE A RECORD
1737  010512  005215                  INC     (R5)                    ;SET 'GO' BIT
1738  010514  004767  174512          JSR     PC,WAITRDY
1739  010520  102435                  BVS     99$
1740  010522  004767  174726          JSR     PC,REVRD
1741  010526  005215                  INC     (R5)                    ;SET 'GO' BIT
```

```
1742
1743  010530  105711              1$:     TSTB    (R1)                ;BRANCH WHEN
1744  010532  100404                      BMI     2$                  ;READY SETS
1745  010534  032711  040000              BIT     #ERR,(R1)
1746  010540  001025                      BNE     99$
1747  010542  000772                      BR      1$
1748
1749  010544  032711  000020      2$:     BIT     #SDWN,(R1)
1750  010550  001004                      BNE     3$
1751  010552  032711  040000              BIT     #ERR,(R1)
1752  010556  001016                      BNE     99$
1753  010560  000771                      BR      2$
1754
1755  010562                      3$:
 (1)  010562  004767  173566              JSR     PC,TIMON            ;TURN TIMER ON
1756  010566  032711  000020      4$:     BIT     #SDWN,(R1)          ;BRANCH WHEN SWDN = 0
1757  010572  001402                      BEQ     5$
1758  010574  000163  004444              JMP     TIMER(R3)           ;GO TO TIMER & RETURN VIA R2
1759
1760  010600  004767  174426      5$:     JSR     PC,WAITRDY          ;WAIT FOR READY
1761  010604  102403                      BVS     99$
1762  010606  004767  173670              JSR     PC,TIMOK
1763  010612  000401                      BR      100$
1764
1765  010614  104400              99$:    HLT
1766  010616  104000              100$:   SCOPE
1767
1768                              ;REWIND DRIVE
1769  010620                      A:
 (1)  010620  004767  174512              JSR     PC,.REWIND          ;REWIND SLAVE
 (1)  010624  102401                      BVS     99$                 ;BRANCH IF ERROR ON REWIND
1770  010626  102002                      BVC     100$
1771  010630  104400              99$:    HLT
1772  010632  000772                      BR      A
1773  010634              100$:
1774
1775                              ;TEST 014-TURN AROUND DELAY (FORWARD-REVERSE)
1776                              ;THIS TEST MEASURES TIME FROM 'GO'=1 (READ REVERSE) TO 'ACCL'=0
1777  010634  112737  000014  001122  TST014: MOVB    #14,@#TSTNUM
1778  010642  012702  010674              MOV     #2$,R2              ;SET RETURN PC FROM TIMER
1779  010646  004767  174546              JSR     PC,WRITE            ;WRITE A RECORD
1780  010652  005215                      INC     (R5)                ;SET 'GO' BIT
1781  010654  004767  174352              JSR     PC,WAITRDY
1782  010660  102420                      BVS     99$
1783
1784  010662  004767  174566      1$:     JSR     PC,REVRD            ;READ THE RECORD (REVERSE)
1785  010666  004767  173462              JSR     PC,TIMON            ;TURN TIMER ON
1786  010672  005215                      INC     (R5)                ;SET 'GO' BIT
1787
1788  010674  005765  000032      2$:     TST     TC(R5)              ;WAIT FOR 'ACCL' = 0
1789  010700  100002                      BPL     3$
1790  010702  000163  004444              JMP     TIMER(R3)           ;GO TO TIMER & RETURN VIA R2
1791
1792  010706  004767  174320      3$:     JSR     PC,WAITRDY
1793  010712  102403                      BVS     99$
1794  010714  004767  173562              JSR     PC,TIMOK
```

```
1795  010720  000401                          BR      100$
1796
1797  010722  104400                  99$:    HLT
1798  010724  104000                  100$:   SCOPE
1799
1800                                  ;TEST 015- TURN AROUND DELAY (REVERSE-FORWARD)
1801                                  ;THIS TEST MEASURES TIME FROM 'GO'=1 (READ) TO 'ACCL'=0.
1802  010726  112737  000015  001122  TST015: MOVB    #15,@#TSTNUM
1803  010734  012702  011002                  MOV     #2$,R2                  ;SET RETURN PC FROM TIMER
1804  010740  004767  174454                  JSR     PC,WRITE                ;WRITE A RECORD
1805  010744  005215                          INC     (R5)                    ;SET 'GO' BIT
1806  010746  004767  174260                  JSR     PC,WAITRDY              ;WAIT FOR READY
1807  010752  102426                          BVS     99$
1808  010754  004767  174474                  JSR     PC,REVRD                ;READ A RECORD IN THE
1809  010760  005215                          INC     (R5)                    ;SET 'GO' BIT
1810
1811  010762  004767  174244                  JSR     PC,WAITRDY
1812  010766  102420                          BVS     99$
1813
1814  010770  004767  174442          1$:     JSR     PC,READ                 ;READ RECORD FORWARD
1815  010774  004767  173354                  JSR     PC,TIMON                ;TURN TIMER ON
1816  011000  005215                          INC     (R5)                    ;SET 'GO' BIT
1817
1818  011002  005765  000032          2$:     TST     TC(R5)                  ;WAIT FOR 'ACCL' = 0
1819  011006  100002                          BPL     3$
1820  011010  000163  004444                  JMP     TIMER(R3)               ;GO TO TIMER & RETURN VIA R2
1821
1822  011014  004767  174212          3$:     JSR     PC,WAITRDY
1823  011020  102403                          BVS     99$
1824  011022  004767  173454                  JSR     PC,TIMOK
1825  011026  000401                          BR      100$
1826
1827  011030  104400                  99$:    HLT
1828  011032  104000                  100$:   SCOPE
1829
1830                                  ;TEST 016-GAP SIZE (STOP HALF)
1831  011034  112737  000016  001122  TST016: MOVB    #16,@#TSTNUM
1832  011042  012702  011100                  MOV     #1$,R2                  ;SET RETURN PC FROM TIMER
1833  011046  004767  174346                  JSR     PC,WRITE                ;WRITE A RECORD
1834  011052  005215                          INC     (R5)                    ;SET 'GO' BIT
1835  011054  004767  174152                  JSR     PC,WAITRDY
1836  011060  102421                          BVS     99$
1837  011062  004767  173652                  JSR     PC,DELAY                ;DELAY 350 MS
1838  011066  004767  174362                  JSR     PC,REVRD                ;READ REVERSE RECORD
1839  011072  004767  173256                  JSR     PC,TIMON                ;TURN TIMER ON
1840  011076  005215                          INC     (R5)                    ;SET 'GO' BIT
1841
1842  011100  005710                  1$:     TST     (R0)                    ;WAIT FOR FRAME COUNT > 0
1843  011102  001002                          BNE     2$
1844  011104  000163  004444                  JMP     TIMER(R3)               ;GO TO TIMER & RETURN VIA R2
1845
1846  011110  004767  174116          2$:     JSR     PC,WAITRDY              ;WAIT FOR READY BIT TO SET
1847  011114  102403                          BVS     99$
1848  011116  004767  173360                  JSR     PC,TIMOK                ;CHECK TIME
1849  011122  000401                          BR      100$
1850
```

```
1851  011124  104400                  99$:    HLT
1852  011126  104000                  100$:   SCOPE
1853
1854                                  ;TEST 017-GAP SIZE (START HALF)
1855  011130  112737  000017  001122  TST017: MOVB    #17,@#TSTNUM
1856  011136  012702  011210                  MOV     #1$,R2                  ;SET RETURN PC FROM TIMER
1857  011142  004767  174252                  JSR     PC,WRITE                ;WRITE A RECORD
1858  011146  005215                          INC     (R5)                    ;SET 'GO' BIT
1859  011150  004767  174056                  JSR     PC,WAITRDY              ;WAIT FOR READY
1860  011154  102427                          BVS     99$
1861  011156  004767  174272                  JSR     PC,REVRD                ;READ REVERSE THE RECORD
1862  011162  005215                          INC     (R5)                    ;SET 'GO' BIT
1863  011164  004767  174042                  JSR     PC,WAITRDY              ;WAIT FOR READY
1864  011170  102421                          BVS     99$                     ;BRANCH ON ERROR
1865  011172  004767  173542                  JSR     PC,DELAY                ;WAIT FOR TAPE MOTION TO STOP
1866  011176  004767  174234                  JSR     PC,READ                 ;READ RECORD
1867  011202  004767  173146                  JSR     PC,TIMON                ;TURN TIMER ON
1868  011206  005215                          INC     (R5)                    ;SET 'GO' BIT
1869
1870  011210  005710                  1$:     TST     (R0)                    ;WAIT FOR FRAME COUNT > 0
1871  011212  001002                          BNE     2$
1872  011214  000163  004444                  JMP     TIMER(R3)               ;GO TO TIMER & RETURN VIA R2
1873
1874  011220  004767  174006          2$:     JSR     PC,WAITRDY              ;WAIT FOR READY
1875  011224  102403                          BVS     99$
1876  011226  004767  173250                  JSR     PC,TIMOK                ;CHECK TIME
1877  011232  000401                          BR      100$
1878
1879  011234  104400                  99$:    HLT
1880  011236  104000                  100$:   SCOPE
1881
1882                                  ;TEST 020- GAP SIZE (INTERRECORD)
1883                                  ;THIS TEST MEASURES TIME FROM 'GO'=1 TO 'FC REG' >0.
1884  011240  112737  000020  001122  TST020: MOVB    #20,@#TSTNUM
1885  011246  012702  011330                  MOV     #1$,R2                  ;SET RETURN PC FROM TIMER
1886  011252  004767  174142                  JSR     PC,WRITE                ;WRITE A RECORD
1887  011256  005215                          INC     (R5)                    ;SET 'GO' BIT
1888  011260  004767  173746                  JSR     PC,WAITRDY              ;WAIT FOR READY
1889  011264  102433                          BVS     99$
1890  011266  004767  174126                  JSR     PC,WRITE                ;WRITE SECOND RECORD
1891  011272  005215                          INC     (R5)                    ;SET 'GO' BIT
1892  011274  004767  173732                  JSR     PC,WAITRDY              ;WAIT FOR READY
1893  011300  102425                          BVS     99$
1894  011302  004767  174146                  JSR     PC,REVRD                ;READ REVERSE SECOND RECORD
1895  011306  005215                          INC     (R5)                    ;SET 'GO' BIT
1896  011310  004767  173716                  JSR     PC,WAITRDY              ;WAIT FOR READY
1897  011314  102417                          BVS     99$
1898  011316  004767  174132                  JSR     PC,REVRD                ;READ REVERSE FIRST RECORD
1899  011322  004767  173026                  JSR     PC,TIMON                ;TURN TIMER ON
1900  011326  005215                          INC     (R5)                    ;SET 'GO' BIT
1901
1902  011330  005710                  1$:     TST     (R0)                    ;WAIT FOR FRAME COUNT > 0
1903  011332  001002                          BNE     2$
1904  011334  000163  004444                  JMP     TIMER(R3)               ;GO TO TIMER & RETURN VIA R2
1905
1906  011340  004767  173666          2$:     JSR     PC,WAITRDY              ;WAIT FOR READY
```

```
1907  011344  102403                          BVS     99$
1908  011346  004767  173130                  JSR     PC,TIMOK
1909  011352  000401                          BR      100$
1910
1911  011354  104400                  99$:    HLT
1912  011356  104000                  100$:   SCOPE
1913
1914                                  ;TEST 021- GAP CONSISTANCY
1915                                  ;THIS TEST MEASURES TIME FROM 'GO'=1 TO 'FC REG' > 0.
1916                                  ;THE TEST REWINDS THE TAPE,WRITES 17 RECORDS WITH A DELAY FROM 1-16 MS
1917                                  ;BETWEEN EACH WRITE COMMAND. AFTER THE 17. RECORDS ARE WRITTTEN THE
1918                                  ;PROGRAM READ REVERSES 16 RECORDS. AT THIS POINT THE TAPE IS STOPPED BE-
1919                                  ;TWEEN THE FIRST AND SECOND RECORD. A READ COMMAND IS EXECUTED TO READ
1920                                  ;THE 16 RECORDS WITH THE TIME BETEWEN GO=1 TO FC > 0 STORED IN 'GAPTBL'
1921                                  ;FOR EACH RECORD READ. AFTER 16 RECORDS HAVE BEEN READ THE TIME IS VER-
1922                                  ;IFIED FOR EACH READ. AFTER ALL RECORD TIMES ARE VERIFIED THEY ARE AVER-
1923                                  ;AGED AND PLACED IN THE 'ATIMTBL' (BY SCOPE). THE ABOVE PROCESS IS RE-
1924                                  ;PEATED FOR EACH ITERATION.
1925
1926  011360  112737  000021  001122  TST021: MOVB    #21,@#TSTNUM
1927  011366  012702  011524                  MOV     #4$,R2                  ;SET RETURN PC FROM TIMER
1928  011372  004767  173740                  JSR     PC,.REWIND              ;REWIND SLAVE
 (1)  011376  102530                          BVS     99$                     ;BRANCH IF ERROR ON REWIND
1929  011400  005067  167510                  CLR     DELTIM                  ;CLEAR VARIABLE DELAY TIME
1930  011404  012700  000021                  MOV     #17.,R0                 ;SET # OF RECORDS TO WRITE
1931  011410  004767  174004          1$:     JSR     PC,WRITE                ;WRITE 17. RECORDS
1932  011414  005215                          INC     (R5)                    ;SET 'GO' BIT
1933  011416  004767  173610                  JSR     PC,WAITRDY              ;WAIT FOR READY
1934  011422  102516                          BVS     99$
1935  011424  004767  173340                  JSR     PC,DELAYV               ;DELAY BEFORE WRITING NEXT REC.
1936  011430  062767  000022  167456          ADD     #18.,DELTIM             ;SET NEXT DELAY TIME
1937  011436  005300                          DEC     R0                      ;DECREMENT RECORDS WRITTEN COUNT
1938  011440  001363                          BNE     1$
1939
1940  011442  012700  000021                  MOV     #17.,R0                 ;SET # OF RECS. TO REVERSE READ
1941  011446  004767  174002          2$:     JSR     PC,REVRD                ;REVERSE READ 17. RECORDS
1942  011452  005215                          INC     (R5)                    ;SET 'GO' BIT
1943  011454  004767  173552                  JSR     PC,WAITRDY              ;WAIT FOR READY
1944  011460  102477                          BVS     99$
1945  011462  005300                          DEC     R0                      ;DECREMENT RECORD COUNT
1946  011464  001370                          BNE     2$
1947
1948  011466  012700  000020                  MOV     #16.,R0                 ;SET # OF RECORDS TO READ
1949  011472  012701  001054                  MOV     #GAPTBL,R1              ;SET PTR TO GAP TABLE FOR TEST
1950  011476  004767  173734                  JSR     PC,READ                 ;READ A RECORD
1951  011502  005215                          INC     (R5)                    ;SET 'GO' BIT
1952
1953  011504  004767  173522          3$:     JSR     PC,WAITRDY              ;WAIT FOR READY
1954  011510  102463                          BVS     99$
1955  011512  004767  173720                  JSR     PC,READ                 ;READ NEXT RECORD
1956  011516  004767  172632                  JSR     PC,TIMON                ;TURN TIMER ON
1957  011522  005215                          INC     (R5)                    ;SET 'GO' BIT
1958
1959  011524  005765  000006          4$:     TST     FC(R5)                  ;WAIT FOR FRAME COUNT > 0
1960  011530  001002                          BNE     5$
1961  011532  000163  004444                  JMP     TIMER(R3)               ;GO TO TIMER & RETURN VIA R2
```

```
                                                ;WAIT FOR READY
011536  004767  173470          5$:   JSR     PC,WAITRDY
011542  102446                        BVS     99$
011544  010421                        MOV     R4,(R1)+        ;STORE TIME IN GAPTBL
011546  005300                        DEC     R0              ;DECREMENT # OF RECORDS READ
011550  001355                        BNE     3$

011552  105037  001120                CLRB    @#GAP           ;SET GAP # 0
011556  012700  000020                MOV     #16.,R0
011562  012701  001054                MOV     #GAPTBL,R1

011566  012104                6$:   MOV     (R1)+,R4        ;GET GAP TICK COUNT
011570  004767  173016                JSR     PC,GAPOK        ;CHECK TIME
011574  105237  001120                INCB    @#GAP           ;INCREMENT GAP #
011600  122737  000020  001120        CMPB    #16.,@#GAP      ;BRANCH IF ALL GAPS NOT CHECKED
011606  001367                        BNE     6$

011610  012700  000020                MOV     #16.,R0         ;SETUP TO AVERAGE GAP SIZES
011614  012701  001054                MOV     #GAPTBL,R1      ;SET PTR TO TABLE
011620  005002                        CLR     R2              ;CLEAR 'SUM' REGISTERS
011622  005003                        CLR     R3
011624  062102                7$:   ADD     (R1)+,R2        ;ADD ALL GAP SIZES TOGETHER
011626  005503                        ADC     R3
011630  005300                        DEC     R0
011632  001374                        BNE     7$
011634  012700  000004                MOV     #4,R0           ;NOW DIVIDE BY 16.
011640  006203                8$:   ASR     R3              ;BY SHIFTING 4 PLACES RIGHT
011642  006002                        ROR     R2
011644  005300                        DEC     R0
011646  001374                        BNE     8$
011650  010204                        MOV     R2,R4           ;MOVE AVERAGED TIMES TO R4
011652  004767  172624                JSR     PC,TIMOK        ;CHECK AVERAGED TIMES
011656  000401                        BR      100$

011660  104400                99$:   HLT
011662  104000                100$:  SCOPE

                              ;TEST 022-DUMMY TEST
                              ;THIS TEST MEASURES NOTHING
011664  112737  000022  001122 TST022: MOVB    #22,@#TSTNUM

                              ;TEST 023-DATA TIME (200BPI)
                              ;THIS TEST MEASURES TIME FROM 'FC REG' CHANGES TO 'RDY'=1.
011672  112737  000023  001122 TST023: MOVB    #23,@#TSTNUM
011700  012702  011752                MOV     #3$,R2          ;SET RETURN PC FROM TIMER
011704  004767  173426                JSR     PC,.REWIND      ;REWIND SLAVE
  (1)   011710  102437                BVS     99$             ;BRANCH IF ERROR ON REWIND
011712  004367  173636                JSR     R3,TMCMD        ;WRITE 800 WORD RECORD
011716  015702                        .WORD   WTBUF           ;SET WRITE BUFFER ADDRESS
011720  176340                        .WORD   -800.           ;WORD COUNT
011722  174700                        .WORD   -1600.          ;FRAME COUNT
011724  000060                        .WORD   WFWD            ;WRITE COMMAND
011726  005215                        INC     (R5)            ;SET 'GO' BIT

011730  022710  174700          1$:   CMP     #-1600.,(R0)    ;WAIT FOR FRAME COUNT TO CHANGE
011734  001004                        BNE     2$
```

```
2017  011736  032711  040000            BIT     #ERR,(R1)               ;MONITOR ERROR BIT
2018  011742  001022                    BNE     99$
2019  011744  000771                    BR      1$
2020
2021  011746                    2$:
 (1)  011746  004767  172402            JSR     PC,TIMON                ;TURN TIMER ON
2022  011752  105711              3$:   TSTB    (R1)                    ;WAIT FOR READY TO SET
2023  011754  100402                    BMI     4$
2024  011756  000163  004444            JMP     TIMER(R3)               ;GO TO TIMER & RETURN VIA R2
2025  011762  012700  000003        4$: MOV     #3,R0                   ;SET TO DIVIDE BY 8
2026  011766  006204                5$: ASR     R4                      ;BY SHIFTING RIGHT 3 PLACES
2027  011770  005300                    DEC     R0
2028  011772  001375                    BNE     5$
2029  011774  004767  173232            JSR     PC,WAITRDY
2030  012000  102403                    BVS     99$
2031  012002  004767  172474            JSR     PC,TIMOK                ;CHECK TIME
2032  012006  000401                    BR      100$
2033
2034  012010  104400              99$:  HLT
2035  012012  104000              100$: SCOPE
2036
2037                              ;TEST 024-DATA TIME (556BPI)
2038  012014  112737  000024  001122  TST024: MOVB   #24,@#TSTNUM
2039  012022  012702  012102            MOV     #3$,R2                  ;SET RETURN PC FROM TIMER
2040  012026  004767  173304            JSR     PC,.REWIND              ;REWIND SLAVE
 (1)  012032  102442                    BVS     99$                     ;BRANCH IF ERROR ON REWIND
2041  012034  052765  000700  000032    BIS     #BPI556+NORM11,TC(R5)   ;LOAD TAPE CONTROL REGISTER
2042  012042  004367  173506            JSR     R3,TMCMD                ;WRITE 2224. WORD RECORD
2043  012046  015702                    .WORD   WTBUF
2044  012050  173520                    .WORD   -2224.
2045  012052  167240                    .WORD   -4448.
2046  012054  000060                    .WORD   WFWD
2047  012056  005215                    INC     (R5)                    ;SET 'GO' BIT
2048
2049  012060  022710  167240      1$:   CMP     #-4448.,(R0)            ;BRANCH WHEN WRITING BEGINS
2050  012064  001004                    BNE     2$
2051  012066  032711  040000            BIT     #ERR,(R1)               ;MONITOR ERROR BIT
2052  012072  001022                    BNE     99$
2053  012074  000771                    BR      1$
2054
2055  012076                    2$:
 (1)  012076  004767  172252            JSR     PC,TIMON                ;TURN TIMER ON
2056  012102  105711              3$:   TSTB    (R1)                    ;BRANCH WHEN READY SETS
2057  012104  100402                    BMI     4$
2058  012106  000163  004444            JMP     TIMER(R3)               ;GO TO TIMER & RETURN VIA R2
2059
2060  012112  012700  000003        4$: MOV     #3,R0                   ;SET SHIFT COUNT
2061  012116  006204                5$: ASR     R4
2062  012120  005300                    DEC     R0
2063  012122  001375                    BNE     5$
2064  012124  004767  173102            JSR     PC,WAITRDY
2065  012130  102403                    BVS     99$
2066  012132  004767  172344            JSR     PC,TIMOK                ;CHECK TIME
2067  012136  000401                    BR      100$
2068
2069  012140  104400              99$:  HLT
```

```
                                100$:   SCOPE

                                ;TEST 025-DATA TIME (800BPI)
                        001122  TST025: MOVB    #025,@#TSTNUM
                                        MOV     #3$,R2                  ;SET RETURN PC FROM TIMER
                                        JSR     PC,.REWIND              ;REWIND SLAVE
                                        BVS     99$                     ;BRANCH IF ERROR ON REWIND
                        000032          BIS     #BPI800+NORM11,TC(R5)   ;SET 800 BPI
                                        JSR     R3,TMCMD                ;WRITE 3200. WORD RECORD
                                        .WORD   WTBUF
                                        .WORD   -3200.
                                        .WORD   -6400.
                                        .WORD   WFWD
                                        INC     (R5)                    ;SET 'GO' BIT

                                1$:     CMP     #-6400.,(R0)            ;WAIT FOR WRITING TO START
                                        BNE     2$
                                        BIT     #ERR,(R1)               ;MONITOR ERROR BIT
                                        BNE     99$
                                        BR      1$

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

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

                                99$:    HLT
                                100$:   SCOPE

                                ;TEST 026-DATA TIME (1600BPI)
                        001122  TST026: MOVB    #026,@#TSTNUM
                                        TSTB    @#NRZFLG                ;BRANCH IF DRIVE 'NRZ ONLY'
                                        BNE     TST027
                                        MOV     #3$,R2                  ;SET RETURN PC FROM TIMER
                                        JSR     PC,.REWIND              ;REWIND SLAVE
                                        BVS     99$                     ;BRANCH IF ERROR ON REWIND
                        000032          BIS     #PE1600+NORM11,TC(R5)   ;SET 1600 BPI
                                        JSR     R3,TMCMD                ;WRITE 3200. WORD RECORD
                                        .WORD   WTBUF
                                        .WORD   -3200.
                                        .WORD   -6400.
                                        .WORD   WFWD
                                        INC     (R5)                    ;SET 'GO' BIT

                                1$:     CMP     #-6400.,(R0)            ;BRANCH WHEN WRITING STARTS
                                        BNE     2$
```

```
2123  012354  032711  040000              BIT     #ERR,(R1)               ;MONITOR ERROR BIT
2124  012360  001017                      BNE     99$
2125  012362  000771                      BR      1$
2126
2127  012364                      2$:
 (1)  012364  004767  171764              JSR     PC,TIMON                ;TURN TIMER ON
2128  012370  105711              3$:     TSTB    (R1)                    ;BRANCH WHEN READY SETS
2129  012372  100402                      BMI     4$
2130  012374  000163  004444              JMP     TIMER(R3)               ;GO TO TIMER & RETURN VIA R2
2131
2132  012400  006204              4$:     ASR     R4                      ;DIVIDE TIME BY 4
2133  012402  006204                      ASR     R4
2134  012404  004767  172622              JSR     PC,WAITRDY
2135  012410  102403                      BVS     99$
2136  012412  004767  172064              JSR     PC,TIMOK                ;CHECK TIME
2137  012416  000401                      BR      100$
2138
2139  012420  104400              99$:    HLT
2140  012422  104000              100$:   SCOPE
2141
2142                                      ;TEST 027-ERASE
2143                                      ;THIS TST MEASURES TIME FROM 'GO'=1 TO 'RDY'=1.
2144  012424  112737  000027  001122  TST027: MOVB   #27,@#TSTNUM
2145  012432  012702  012460              MOV     #1$,R2                  ;SET RETURN PC FROM TIMER
2146  012436  004337  005554              JSR     R3,@#TMCMD
2147  012442  000000                      .WORD   0
2148  012444  000000                      .WORD   0
2149  012446  000000                      .WORD   0
2150  012450  000024                      .WORD   ERASE
2151  012452  004767  171676              JSR     PC,TIMON                ;TURN TIMER ON
2152  012456  005215                      INC     (R5)                    ;SET 'GO' BIT
2153
2154  012460  105711              1$:     TSTB    (R1)                    ;BRANCH WHEN READY SETS
2155  012462  100402                      BMI     2$
2156  012464  000163  004444              JMP     TIMER(R3)               ;GO TO TIMER & RETURN VIA R2
2157
2158  012470  004767  172536          2$: JSR     PC,WAITRDY
2159  012474  102403                      BVS     99$
2160  012476  004767  172000              JSR     PC,TIMOK
2161  012502  000401                      BR      100$
2162
2163  012504  104400              99$:    HLT
2164  012506  104000              100$:   SCOPE
2165
2166                                      ;TEST-030 TAPE MARK
2167                                      ;THIS TEST MEASURES TIME FROM 'GO'=1 TO 'RDY'=1.
2168  012510  112737  000030  001122  TST030: MOVB   #30,@#TSTNUM
2169  012516  012702  012560              MOV     #1$,R2                  ;SET RETURN PC FROM TIMER
2170  012522  004767  172672              JSR     PC,WRITE                ;WRITE A RECORD
2171  012526  005215                      INC     (R5)                    ;SET 'GO' BIT
2172  012530  004767  172476              JSR     PC,WAITRDY
2173  012534  102423                      BVS     99$
2174  012536  004337  005554              JSR     R3,@#TMCMD
2175  012542  000000                      .WORD   0
2176  012544  000000                      .WORD   0
2177  012546  000000                      .WORD   0
```

```
2178  012550  000026                        .WORD   WFMK
2179  012552  004767  171576                JSR     PC,TIMON        ;TURN TIMER ON
2180  012556  005215                        INC     (R5)            ;SET 'GO' BIT
2181
2182  012560  105711                1$:     TSTB    (R1)            ;BRANCH WHEN READY SETS
2183  012562  100402                        BMI     2$
2184  012564  000163  004444                JMP     TIMER(R3)       ;GO TO TIMER & RETURN VIA R2
2185
2186  012570  004767  172436        2$:     JSR     PC,WAITRDY
2187  012574  102403                        BVS     99$
2188  012576  004767  171700                JSR     PC,TIMOK
2189  012602  000401                        BR      100$
2190
2191  012604  104400                99$:    HLT
2192  012606                        100$:
  (1) 012606  004767  172524                JSR     PC,.REWIND      ;REWIND SLAVE
  (1) 012612  102774                        BVS     99$             ;BRANCH IF ERROR ON REWIND
2193  012614  104000                        SCOPE
2194
```

```
2196  012616  012700  000012          FINISH: MOV     #10.,R0                 ;SET LINE FEED COUNT
2197  012622  000004  001374          1$:     TYPE,CRLF
2198  012626  005300                          DEC     R0
2199  012630  001374                          BNE     1$
2200  012632  032777  000100  166140          BIT     #SW06,@SWR
2201  012640  001410                          BEQ     2$
2202  012642  113700  001004                  MOVB    @#DRVNUM,R0
2203  012646  113701  001005                  MOVB    @#SLVNUM,R1
2204  012652  113702  001006                  MOVB    @#SLVPTR,R2
2205  012656  000137  007210                  JMP     @#TYPHDR
2206  012662  105237  001005          2$:     INCB    @#SLVNUM                ;SET NEXT SLAVE #
2207  012666  005237  001006                  INC     @#SLVPTR                ;AND ITS POINTER
2208  012672  122737  000010  001005          CMPB    #8.,@#SLVNUM            ;BRANCH IF LAST SLAVE (7)
2209  012700  001402                          BEQ     3$
2210  012702  000137  006764                  JMP     @#BEGIN                 ;BEGIN TEST ON NEXT SLAVE
2211  012706  105037  001005          3$:     CLRB    @#SLVNUM                ;SET SLAVE #0
2212  012712  105237  001004                  INCB    @#DRVNUM                ;AND INCREMENT DRIVE #
2213  012716  122737  000010  001004          CMPB    #8.,@#DRVNUM            ;AND CHECK IF LAST DRIVE
2214  012724  001402                          BEQ     END
2215  012726  000137  006764                  JMP     @#BEGIN
2216
2217  012732  105737  001125          END:    TSTB    @#UNTFND                ;BRANCH IF A UNIT WAS FOUND
2218  012736  001004                          BNE     1$
2219  012740  000004  014050                  TYPE,E.UNIT
2220  012744  000137  005724                  JMP     @#INIT
2221  012750  000000          1$:     HALT
2222  012752  004767  167012                  JSR     PC,CKSWR                ;CHECK FOR CNTL G
2223  012756  000005                          RESET
2224  012760  000137  005724                  JMP     @#INIT                  ;RESTART
```

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

                                        ;TEST 031- SKEW TAPE SPEED TEST-FORWARD
                                        ;THIS TEST READS 32" OF TAPE (26400.-800. = 25600. FRAMES) THEN
                                        ;DIVIDES TIME BY 32. TO GET TIME TO READ 1" (800. FRAMES) OF TAPE.
012772 112737 000031 001122 TST031: MOVB    #31,@#TSTNUM
013000 012702 013056                MOV     #2$,R2                  ;SET RETURN PC FROM TIMER
013004 004767 172326                JSR     PC,.REWIND              ;REWIND SLAVE
 (1)   013010 102441                BVS     99$                     ;BRANCH IF ERROR ON REWIND
013012 052765 001300 000032         BIS     #BPI800+NORM11,TC(R5)   ;SET 800 BPI
013020 052765 000010 000010         BIS     #BAI,CS2(R5)            ;INHIBIT BUS ADDRESS INCREMENT
013026 004337 005554                JSR     R3,@#TMCMD              ;READ 32" OF TAPE-FORWARD
013032 015702                       .WORD   RDBUF
013034 177777                       .WORD   -1.
013036 063440               10$:    .WORD   26400.                  ;FRAME COUNT
013040 000070                       .WORD   RDFWD
013042 005215                       INC     (R5)                    ;SET 'GO' BIT

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

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

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

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

                                        ;TEST 032-SKEW TAPE SPEED TEST-REVERSE
                                        ;THIS TEST READS FORWARD 40" (32000. FRAMES) OF TAPE, THEN READS REVERSE
                                        ;32" (26400.-800. = 25600. FRAMES) OF TAPE. THE TIME IS THEN DIVIDED BY
                                        ;32. TO GET TIME TO READ 1" (800. FRAMES) OF TAPE.
013120 112737 000032 001122 TST032: MOVB    #32,@#TSTNUM
013126 012702 013254                MOV     #3$,R2                  ;SET RETURN PC FROM TIMER
013132 004767 172200                JSR     PC,.REWIND              ;REWIND SLAVE
 (1)   013136 102465                BVS     99$                     ;BRANCH IF ERROR ON REWIND
013140 052765 001300 000032         BIS     #BPI800+NORM11,TC(R5)
013146 052765 000010 000010         BIS     #BAI,CS2(R5)
013154 004337 005554                JSR     R3,@#TMCMD              ;READ FORWARD 32000. FRAMES
013160 015702                       .WORD   RDBUF
013162 177777                       .WORD   -1.                     ;WORD COUNT
013164 076400               10$:    .WORD   32000.                  ;FRAME COUNT
013166 000070                       .WORD   RDFWD                   ;READ FORWARD
013170 005215                       INC     (R5)                    ;SET 'GO' BIT

```

```
2280  013172  023710  013164          1$:     CMP     @#10$,(R0)
2281  013176  101375                          BHI     1$
2282
2283  013200  004767  171770                  JSR     PC,RHINIT               ;INIT DRIVE
2284  013204  004767  171530                  JSR     PC,DELAY                ;WAIT FOR TAPE MOTION TO STOP
2285  013210  052765  001300  000032          BIS     #BPI800+NORM11,TC(R5)   ;SET 800 BPI
2286  013216  052765  000010  000010          BIS     #BAI,CS2(R5)            ;INHIBIT BUS ADDRESS INCRENENT
2287  013224  004337  005554                  JSR     R3,@#TMCMD              ;READ REVERSE 32" OF TAPE
2288  013230  015702                          .WORD   RDBUF                   ;READ BUFFER
2289  013232  177777                          .WORD   -1.                     ;WORD COUNT
2290  013234  063440                  11$:    .WORD   26400.                  ;FRAME COUNT
2291  013236  000076                          .WORD   RDREV                   ;READ REVERSE
2292  013240  005215                          INC     (R5)                    ;SET 'GO' BIT
2293
2294  013242  022710  001440          2$:     CMP     #800.,(R0)              ;WAIT FOR FIRST 800 FRAMES
2295  013246  101375                          BHI     2$                      ;TO BE READ
2296
2297  013250  004767  171100                  JSR     PC,TIMON                ;TURN TIMER ON
2298  013254  023710  013234          3$:     CMP     @#11$,(R0)              ;WAIT FOR ALL FRAMES TO BE READ
2299  013260  103402                          BLO     4$
2300  013262  000163  004444                  JMP     TIMER(R3)               ;GO TO TIMER & RETURN VIA R2
2301
2302  013266  012700  000005          4$:     MOV     #5,R0                   ;DIVIDE TIME BY 32.
2303  013272  006204                  5$:     ASR     R4
2304  013274  005300                          DEC     R0
2305  013276  001375                          BNE     5$
2306  013300  004767  171670                  JSR     PC,RHINIT
2307  013304  004767  171172                  JSR     PC,TIMOK
2308  013310  000401                          BR      100$
2309
2310  013312  104400                  99$:    HLT
2311  013314                          100$:
 (1)  013314  004767  172016                  JSR     PC,.REWIND              ;REWIND SLAVE
 (1)  013320  102774                          BVS     99$                     ;BRANCH IF ERROR ON REWIND
2312  013322  104000                          SCOPE
2313
2314  013324  000137  012616                  JMP     @#FINISH
2315
2316
2317
```

```
                                        .SBTTL    PROGRAM MESSAGES
                                    ;OPERATOR INSTRUCTIONS
013330  005015  052524  033061    M.NAM:   .ASCIZ  <CR><LF>'TU16J DRIVE FUNCTION TIMER (DZTUG-A)'
013336  020112  051104  053111
013344  020105  052506  041516
013352  044524  047117  052040
013360  046511  051105  024040
013366  055104  052524  026507
013374  024501     000
013377     015  052012  050131    I.REG:   .ASCIZ  <CR><LF>'TYPE FIRST ADDRESS OF CONTROLLER  '
013404  020105  044506  051522
013412  020124  042101  051104
013420  051505  020123  043117
013426  041440  047117  051124
013434  046117  042514  020122
013442  000040
013444  054524  042520  052040    I.DRVS: .ASCIZ  %TYPE TM02 DRIVE #'S TO BE TESTED %
013452  030115  020062  051104
013460  053111  020105  023443
013466  020123  047524  041040
013474  020105  042524  052123
013502  042105  000040
013506  047506  020122  046524    I.SLVS: .ASCII  'FOR TM02 DRIVE '
013514  031060  042040  044522
013522  042526     040
013525     060  020055  054524    I.DRV:  .ASCIZ  %0- TYPE SLAVE #'S TO BE TESTED %
013532  042520  051440  040514
013540  042526  021440  051447
013546  052040  020117  042502
013554  052040  051505  042524
013562  020104     000
013565     123  042520  042105    I.SKEW: .ASCIZ  'SPEED TESTS ONLY? (YES/NO = 1/0)'
013572  052040  051505  051524
013600  047440  046116  037531
013606  024040  042531  027523
013614  047516  036440  030440
013622  030057  000051
013626  051116  020132  047117    I.NRZ:  .ASCIZ  'NRZ ONLY? (YES/NO = 1/0)'
013634  054514  020077  054450
013642  051505  047057  020117
013650  020075  027461  024460
013656     000
013657     015  042412  042116    M.EOT:  .ASCIZ  <CR><LF>'END OF TAPE'<CR><LF>
013664  047440  020106  040524
013672  042520  005015     000

                                    ;ERROR MESSAGES
013677     015  052012  040522    E.TRP4: .ASCIZ  <CR><LF>'TRAPPED TO 4'
013704  050120  042105  052040
013712  020117  000064
013716  047516  041440  047117    E.NCON: .ASCIZ  'NO CONTROLLER AT ADDRESS SPECIFIED'<CR><LF>
013724  051124  046117  042514
013732  020122  052101  040440
013740  042104  042522  051523
013746  051440  042520  044503
013754  044506  042105  005015
```

```
                013762   000
2333            013763   124  030115  020062  E.NDRV: .ASCIZ  'TM02 DRIVE '
                013770 051104  053111  020105
                013776   000
2334            013777   104  044522  042526  E.NSLV: .ASCII  'DRIVE '
                014004   040
2335            014005   060  051440  040514  E.DRV:  .ASCII  '0 SLAVE '
                014012 042526    040
2336            014015   060  047040  052117  E.NAVA: .ASCIZ  '0 NOT AVAILABLE FOR TEST'<CR><LF>
                014022 040440  040526  046111
                014030 041101  042514  043040
                014036 051117  052040  051505
                014044 006524  000012
2337            014050 047516  052040  030115  E.UNIT: .ASCIZ  'NO TM02/TU16J UNIT FOUND TO TEST'<CR><LF>
                014056 027462  052524  033061
                014064 020112  047125  052111
                014072 043040  052517  042116
                014100 052040  020117  042524
                014106 052123  005015    000
2338            014113   123  043117  020124  E.SFT:  .ASCIZ  'SOFT ERROR (DATA)'<CR><LF>
                014120 051105  047522  020122
                014126 042050  052101  024501
                014134 005015    000
2339            014137   124  051505  020124  E.HDR:  .ASCIZ  'TEST # '
                014144 020043    000
2340            014147   040  042504  044526  E.HDR1: .ASCII  ' DEVICE ERROR'<CR><LF>
                014154 042503  042440  051122
                014162 051117  005015
2341            014166 051503  004461  041527          .ASCIZ  'CS1'<HT>'WC'<HT>'BA'<HT>'FC'<HT>'CS2'<HT>'DS'<HT>'ER'<HT>'TC'<CR><LF>
                014174 041011  004501  041506
                014202 041411  031123  042011
                014210 004523  051105  052011
                014216 006503  000012
2342            014222 047440  052125  047440  E.HDR2: .ASCIZ  ' OUT OF RANGE ERROR'<CR><LF>
                014230 020106  040522  043516
                014236 020105  051105  047522
                014244 006522  000012
2343            014250 005015  044524  042515  E.TIMOV:        .ASCIZ  <CR><LF>'TIMER OVERFLOWED'<CR><LF>
                014256 020122  053117  051105
                014264 046106  053517  042105
                014272 005015    000
2344            014275    015  052012  046511  E.TIMEX:        .ASCIZ  <CR><LF>'TIME EXPIRED WAITING FOR RDY'<CR><LF>
                014302 020105  054105  044520
                014310 042522  020104  040527
                014316 052111  047111  020107
                014324 047506  020122  042122
                014332 006531  000012
2345            014336 043440  050101  021440  E.GAP:  .ASCIZ  ' GAP # '
                014344 000040
2346
2347                                            ;TIME DOCUMENT LINES
2348            014346 025052  025052  025052  L.HDR1: .ASCIZ  '*******************************************************************************
                014354 025052  025052  025052
                014362 025052  025052  025052
                014370 025052  025052  025052
                014376 025052  025052  025052
```

```
                014404  025052  025052  025052
                014412  025052  025052  025052
                014420  025052  025052  025052
                014426  025052  025052  025052
                014434  025052  025052  025052
                014442  025052  025052  025052
                014450  025052  025052  025052
                014456  005015      000
 2349           014461      052  052040  030115  L.HDR2: .ASCII  '* TM02 DRIVE FUNCTION TIMES- DRIVE # '
                014466  020062  051104  053111
                014474  020105  052506  041516
                014502  044524  047117  052040
                014510  046511  051505  020055
                014516  051104  053111  020105
                014524  020043
 2350           014526  020060  046123  053101  L.DRV:  .ASCII  'D SLAVE # '
                014534  020105  020043
 2351           014540  020060      040          L.SLV:  .ASCII  'D  '
 2352           014543      071  041440  040510  L.CHAN: .ASCIZ  '9 CHAN. SER # '
                014550  027116  051440  051105
                014556  021440  000040
 2353           014562  006440  025012  005015  L.HDR3: .ASCII  ' '<CR><LF>'*'<CR><LF>
 2354           014570  020052  052506  041516          .ASCIZ  '* FUNCTION'<HT><HT>'TIME(SPECIFICATION)'<HT>'TIME(ACTUAL)'<CR><LF>
                014576  044524  047117  004411
                014604  044524  042515  051450
                014612  042520  044503  044506
                014620  040503  044524  047117
                014626  004451  044524  042515
                014634  040450  052103  040525
                014642  024514  005015      000
 2355
 2356           014647      122  047101  042507  L.RNG:  .ASCIZ  'RANGE=<'
                014654  036075      000
 2357           014657      101  052103  040525  L.ACT:  .ASCIZ  'ACTUAL='
                014664  036514      000
 2358
 2359                                            ;TEST DESCRIPTOR HEADERS
 2360           014667      052  053440  044522  A.TOO1: .ASCIZ  '* WRITE FROM BOT'<HT>
                014674  042524  043040  047522
                014702  020115  047502  004524
                014710      000
 2361           014711      052  053440  044522  A.TOO2: .ASCIZ  '* WRITE START'<HT><HT>
                014716  042524  051440  040524
                014724  052122  004411      000
 2362           014731      052  053440  044522  A.TOO3: .ASCIZ  '* WRITE SHUTDOWN'<HT>
                014736  042524  051440  052510
                014744  042124  053517  004516
                014752      000
 2363           014753      052  053440  044522  A.TOO4: .ASCIZ  '* WRITE SETTLEDOWN'<HT>
                014760  042524  051440  052105
                014766  046124  042105  053517
                014774  004516      000
 2364           014777      052  051040  040505  A.TOO5: .ASCIZ  '* READ FROM BOT'<HT><HT>
                015004  020104  051106  046517
                015012  041040  052117  004411
                015020      000
```

```
2365  015021  052     051040  040505  A.T006: .ASCIZ  '* READ START'<HT><HT>
      015026  020104  052123  051101
      015034  004524  000011
2366  015040  020052  042522  042101  A.T007: .ASCIZ  '* READ SHUTDOWN'<HT><HT>
      015046  051440  052510  042124
      015054  053517  004516  000011
2367  015062  020052  042522  042101  A.T010: .ASCIZ  '* READ SETTLEDOWN'<HT>
      015070  051440  052105  046124
      015076  042105  053517  004516
      015104  000
2368  015105  052     051040  040505  A.T011: .ASCIZ  '* READ REV START'<HT>
      015112  020104  042522  020126
      015120  052123  051101  004524
      015126  000
2369  015127  052     051040  040505  A.T012: .ASCIZ  '* READ REV SHUTDOWN'<HT>
      015134  020104  042522  020126
      015142  044123  052125  047504
      015150  047127  000011
2370  015154  020052  042522  042101  A.T013: .ASCIZ  '* READ REV SETTLEDOWN'<HT>
      015162  051040  053105  051440
      015170  052105  046124  042105
      015176  053517  004516  000
2371  015203  052     052040  051125  A.T014: .ASCIZ  '* TURN AROUND DELAY F-R'<HT>
      015210  020116  051101  052517
      015216  042116  042040  046105
      015224  054501  043040  051055
      015232  000011
2372  015234  020052  052524  047122  A.T015: .ASCIZ  '* TURN AROUND DELAY R-F'<HT>
      015242  040440  047522  047125
      015250  020104  042504  040514
      015256  020131  026522  004506
      015264  000
2373  015265  052     043440  050101  A.T016: .ASCIZ  '* GAP SIZE-STOP HALF'<HT>
      015272  051440  055111  026505
      015300  052123  050117  044040
      015306  046101  004506  000
2374  015313  052     043440  050101  A.T017: .ASCIZ  '* GAP SIZE-START HALF'<HT>
      015320  051440  055111  026505
      015326  052123  051101  020124
      015334  040510  043114  000011
2375  015342  020052  040507  020120  A.T020: .ASCIZ  '* GAP SIZE-INTERRECORD'<HT>
      015350  044523  042532  044455
      015356  052116  051105  042522
      015364  047503  042122  000011
2376  015372  020052  040507  020120  A.T021: .ASCIZ  '* GAP CONSISTANCY'<HT>
      015400  047503  051516  051511
      015406  040524  041516  004531
      015414  000
2377  015415  052     042040  052101  A.T023: .ASCIZ  '* DATA TIME-200BPI'<HT>
      015422  020101  044524  042515
      015430  031055  030060  050102
      015436  004511  000
2378  015441  052     042040  052101  A.T024: .ASCIZ  '* DATA TIME-556BPI'<HT>
      015446  020101  044524  042515
      015454  032455  033065  050102
      015462  004511  000
```

```
2379  015465     052  042040  052101  A.T025: .ASCIZ  '* DATA TIME-800BPI'<HT>
      015472  020101  044524  042515
      015500  034055  030060  050102
      015506  004511     000
2380  015511     052  042040  052101  A.T026: .ASCIZ  '* DATA TIME-1600BPI'<HT>
      015516  020101  044524  042515
      015524  030455  030066  041060
      015532  044520  000011
2381  015536  020052  051105  051501  A.T027: .ASCIZ  '* ERASE GAP TIME'<HT>
      015544  020105  040507  020120
      015552  044524  042515  000011
2382  015560  020052  051127  052111  A.T030: .ASCIZ  '* WRITE FILE MARK'<HT>
      015566  020105  044506  042514
      015574  046440  051101  004513
      015602     000
2383  015603     052  052040  050101  A.T031: .ASCIZ  '* TAPE SPEED-FWD'<HT>
      015610  020105  050123  042505
      015616  026504  053506  004504
      015624     000
2384  015625     052  052040  050101  A.T032: .ASCIZ  '* TAPE SPEED-REV'<HT>
      015632  020105  050123  042505
      015640  026504  042522  004526
      015646     000
2385
2386  015647     015  057012  000107  L.CNTG: .ASCIZ  <CR><LF>'↑G'
2387  015654  005015  053523  036522  L.SWR:  .ASCIZ  <CR><LF>'SWR='
      015662     000
2388  015663     040  047040  053505  L.NEW:  .ASCIZ  '  NEW= '
      015670  020075     000
2389  015673     015  037412  005015  L.QUEST:        .ASCIZ  <CR><LF>'?'<CR><LF>
      015700     000
2390          015702                          .EVEN
2391          015702                  RDBUF=.
2392          015702                  WTBUF=.
2393  015702  000200                  .BLKW   128.
2394          000001                  .END
```

```
A       010620
ACCL  = 100000
ANGTAB  001412
AS    = 000016
ASFLG   001130
ATA   = 100000
ATIME   001012
ATIMTB  001014
A.TOO1  014667
A.TOO2  014711
A.TOO3  014731
A.TOO4  014753
A.TOO5  014777
A.TOO6  015021
A.TOO7  015040
A.TO10  015062
A.TO11  015105
A.TO12  015127
A.TO13  015154
A.TO14  015203
A.TO15  015234
A.TO16  015265
A.TO17  015313
A.TO20  015342
A.TO21  015372
A.TO23  015415
A.TO24  015441
A.TO25  015465
A.TO26  015511
A.TO27  015536
A.TO30  015560
A.TO31  015603
A.TO32  015625
A16   = 000400
A17   = 001000
BA    = 000004
BAI   = 000010
BEGIN   006764
BELL    001403
BKSLSH  001377
BOT   = 000002
BPI200= 000000
BPI556= 000400
BPI800= 001000
BPTVEC= 000014
CDM11 = 000320
CHKDRV  006270
CHKSLV  006574
CH7   = 010000
CKSWR   001770
CLR   = 000040
CNTLU   002030
CNTRLC= 000003
CNTRLO= 000017
CNTRLU= 000025
CNVDEC  002730
CNVOCT  002622
CNVTAO  003260
CNVTD   002742
CNVTO   002634
COUNT   001764
CR    = 000015
CRLF    001374
CSITM = 002000
CS1   = 000000
CS2   = 000010
DASH    001405
DB    = 000022
DCONST  003036
DELAY   004740
DELAYV  004770
DELTIM  001114
DIGTAB  001132
DIVIDE  005026
DLT   = 100000
DPR   = 000400
DRIVES  006140
DRVAVA  005116
DRVNUM  001004
DRVTBL  001154
DRY   = 000200
DRYCLR= 000010
DS    = 000012
DT    = 000026
DTE   = 010000
DVA   = 004000
DV0   = 000000
DV1   = 000001
DV2   = 000002
DV3   = 000003
DV4   = 000004
DV5   = 000005
DV6   = 000006
DV7   = 000007
ECHO    001401
EMTVEC= 000030
END     012732
EOT   = 002000
ER    = 000014
ERASE = 000024
ERFLG   001123
ERR   = 040000
ERRTRP  003650
ERRVEC= 000004
E.DRV   014005
E.GAP   014336
E.HDR   014137
E.HDR1  014147
E.HDR2  014222
E.NAVA  014015
E.NCON  013716
E.NDRV  013763
E.NSLV  013777
E.SFT   014113
E.TIME  014275
E.TIMO  014250
E.TRP4  013677
E.UNIT  014050
FC    = 000006
FCE   = 001000
FINISH  012616
FMT   = 000020
FPEVEC= 000244
FRMCNT= 177400
FWDSPC  005472
GAP     001120
GAPOK   004612
GAPTBL  001054
GO    = 000001
GTIMTB  001572
HLT   = 104400
HT    = 000011
IDB   = 000010
IE    = 000100
ILF   = 000001
ILR   = 000002
INBUF   001264
INCVAE= 000100
INIT    005724
IOTVEC= 000020
IR    = 000100
ITCNT   001121
I.DRV   013525
I.DRVS  013444
I.NRZ   013626
I.REG   013377
I.SKEW  013565
I.SLVS  013506
LF    = 000012
LKS   = 177546
LKVEC = 000100
LPB   = 177516
LPS   = 177514
L.ACT   014657
L.CHAN  014543
L.CNTG  015647
L.DRV   014526
L.HDR1  014346
L.HDR2  014461
L.HDR3  014562
L.NEW   015663
L.QUES  015673
L.RNG   014647
L.SLV   014540
L.SWR   015654
MCPE  = 020000
MDPE  = 000400
MMVEC = 000250
MOL   = 010000
MR    = 000024
MXF   = 001000
M.EOT   013657
M.NAM   013330
NAMPTR  001672
NED   = 010000
NEF   = 004000
NEM   = 004000
NOP   = 000000
NORM11= 000300
NRZFLG  001127
NSG   = 000400
OCTALO  001116
ODIGIT  001144
OPI   = 020000
OR    = 000200
OSC   = 000100
OUT     002240
OUTBUF= 005724
OUTGAP  003146
OUTSPC  003052
PARVEC= 000114
PAT   = 000020
PEFLRC= 000200
PES   = 000040
PE1600= 002000
PFVEC = 000024
PGE   = 002000
PIP   = 020000
PIRQ  = 177772
PIRVEC= 000240
PLKCSR= 172540
PLKVEC= 000104
PRGFLG  001124
PSEL  = 002000
PSW   = 177776
PUBLIS  003346
RDBUF = 015702
RDFWD = 000070
RDREV = 000076
RDSW    001766
RDY   = 000200
READ    005436
RESVEC= 000010
REVRD   005454
RHINIT  005174
RMR   = 000004
RWD   = 000006
RWDOFF= 000002
R10   =%000000
R11   =%000001
R12   =%000002
R13   =%000003
R14   =%000004
R15   =%000005
SC    = 100000
SCOPE = 104000
SCPADR  001002
SDWN  = 000020
SKEWTS  012764
SLA   = 000001
SLAVES  006370
SLR   = 177774
SLVAVA  005144
SLVNUM  001005
SLVPTR  001006
SLVTBL  001164
SN    = 000030
SNPT    005574
SPACE   001410
SPACE2  001407
SPCFWD= 000030
SPCREV= 000032
SPR   = 002000
SSC   = 000100
STIMTB  001416
STKPTR= 000600
SUSWR   005730
SWR     001000
SWREG   000176
SW06  = 000100
SW07  = 000200
SW08  = 000400
SW09  = 001000
SW10  = 002000
SW11  = 004000
SW13  = 020000
SW14  = 040000
SW15  = 100000
TAP   = 040000
TBITVE= 000014
TC    = 000032
TCRLF   002360
TEMPST  001762
TIB     001760
TIMER   004444
```

```
TIMERR  004454          TSTNUM  001122          TST023  011672          TYPOCT  002630          $CNTRL  002325
TIMERO  004470          TST001  007336          TST024  012014          UBREAK= 177770          $CRLF   002326
TIMER1  004420          TST002  007422          TST025  012144          UNS     = 040000          $FILL   002315
TIMOK   004502          TST003  007500          TST026  012274          UNTFND  001125          $HT     = 000011
TIMON   004354          TST004  007570          TST027  012424          UPE     = 020000          $NULL   002314
TKB     = 177562        TST005  007676          TST030  012510          WAITRD  005232          $TKFLG  002317
TKISR   003606          TST006  007762          TST031  012772          WAITTI  005234          $TPB    002322
TKS     = 177560        TST007  010046          TST032  013120          WC      = 000002          $TPFLG  002316
TKVEC   = 000060        TST010  010146          TTIN    002242          WCE     = 040000          $TPS    002320
TMBASE  001010          TST011  010266          TTIN1   002262          WCHKF   = 000050          .       = 016302
TMCMD   005554          TST012  010364          TTIN2   002276          WCHKR   = 000056          .HLT    003652
TMCS1   = 172440        TST013  010474          TYPDEC  002736          WFMK    = 000026          .INPUT  003474
TMK     = 000004        TST014  010634          TYPE    = 000004        WFWD    = 000060          .RESTO  002600
TPB     = 177566        TST015  010726          TYPE1   002352          WRDCNT= 177600          .REWIN  005336
TPS     = 177564        TST016  011034          TYPE2   002400          WRITE   005420          .SAVE   002556
TPVEC   = 000064        TST017  011130          TYPE3   002406          WRL     = 004000          .SCOPE  004126
TRAPVE= 000034          TST020  011240          TYPE4   002412          WRT.BK  005512          .TYPE   002332
TRE     = 040000        TST021  011360          TYPFLG  001126          WTBUF   = 015702
TRTVEC= 000014          TST022  011664          TYPHDR  007210          $CHARC  002324

. ABS.  016302      000

 ERRORS DETECTED:  0
 DEFAULT GLOBALS GENERATED:  0

 DZTUGA,DZTUGA=DZTUGA.P11
 RUN-TIME:  11 17  1 SECONDS
 RUN-TIME RATIO: 64/30=2.1
 CORE USED:  7K  (13 PAGES)
```

J07