TM02

DRIVE FUNCTION TIMER

MD-11-DZTUD-C

EP-DZTUD-C-DL

FICHE 1 OF 1

JAN 1978

digital

MADE IN USA

PRODUCT CODE: MAINDEC-11-DZTUD-C-D

PRODUCT NAME: TM02 DRIVE FUNCTION TIMER

DATE CREATED: 21 APRIL 76

MAINTAINER: DIAGNOSTIC GROUP

AUTHOR: JOHN ADAMS/R, BARNES/S, CARPENTER

TABLE OF CONTENTS

ABSTRACT

PROGRAM DITUD MEASURES THE TIME REQUIRED AND GAP SIZES PRODUCED BY THE TM02/TU16 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 TM11/TM02 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 (TU16) 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 C <^C> 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=<180000-184000>       ACTUAL=184740
* WRITE START             RANGE=<089500-088700>       ACTUAL=089120
* WRITE SHUTDOWN          RANGE=<088900-088500>       ACTUAL=088840
* WRITE SETTLEDOWN        RANGE=<013500-008100>       ACTUAL=010970
* READ FROM BOT           RANGE=<152000-149000>       ACTUAL=150580
* 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=<011000-008500>       ACTUAL=010520
* GAP SIZE-INTERRECORD    RANGE=<014000-013700>       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*M8983 ACCL CNTR
2. WRITE START	* "	* " * "
3. WRITE SHUTDOWN	* "	* " * "
4. WRITE SETTLEDOWN	* "	*M8910 SETTLEDOWN ONE SHOT
5. READ FROM BOT	* "	*M8911 ROM*M8983 ACCL CNTR
6. READ START	* "	* " * "
7. READ SHUTDOWN	* "	* " * "
10. READ SETTLEDOWN	* "	*M8910 SETTLEDOWN ONE SHOT
11. READ REVERSE START	* "	*M8911 ROM*M8983 ACCL CNTR
12. READ REVERSE SHUTDOWN	* "	* " * "
13. READ REVERSE SETTLEDOWN	* "	*M8910 SETTLEDOWN ONE SHOT
14. TURN AROUND F-R	* "	*M8911 ROM*M8983 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	* " " "

A2

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*M8983 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 TU16 (M9011), 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 INTERRECORD GAP.

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

T4. WRITE SETTLEDOWN:

SEQ 0017

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

SEQ 0018

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 TU16 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

H2

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 DITUD 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(16)
4. MONITOR FC FOR FC = 8800(16)
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.

K2

DZTUDC,P11 TABLE OF CONTENTS

DZTUDC,P11

```
                                        .NLIST  ME
                                        .LIST   ME
                                        .ABS
                                        .MCALL  $CPREG,$CPVEC,$CPREG,$CATCH,$TYPE
                                        .TITLE  DZTUD-C TM02 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 'PC' REGISTER (SET BY SCOPE)
                                ;       R1=ADDRESS OF 'DS' REGISTER (SET BY SCOPE)
                                ;       R2=RETURN PC FROM TIMER (SET BY EACH TEST)
                                ;       R3=INDEX INDICATING PREVIOUS OSCILLATOR POLARITY (SET BY TIMER)
                                ;       R4=CONTAINS 'TICK' COUNT WHEN TIMER IS RUNNING (SET BY TIMER)
                                ;       R5=ADDRESS OF CSI (SET BY SCOPE)

                                ;SWITCH REGISTER SWITCH ASSIGNMENTS
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
```

m2

```
                                        .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
```

A3

```
                        ;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                          MDPT=   400
```

```
DZTUDC,P11      TM02/TU16 REGISTER BITS

   169          001000                      MXF=    1000
   170          002000                      PGE=    2000
   171          004000                      NEM=    4000
   172          010000                      NED=    10000
   173          020000                      UPE=    20000
   174          040000                      WCE=    40000
   175          100000                      DLT=    100000
   176                                  ;RHDS-DS(R5)
   177          000001                      SLA=    1
   178          000002                      BOT=    2
   179          000004                      TMK=    4
   180          000010                      IDB=    10
   181          000020                      SDWN=   20
   182          000040                      PES=    40
   183          000100                      SSC=    100
   184          000200                      DRY=    200
   185          000400                      DPR=    400
   186          002000                      EOT=    2000
   187          004000                      WRL=    4000
   188          010000                      MOL=    10000
   189          020000                      PIP=    20000
   190          040000                      ERR=    40000
   191          100000                      ATA=    100000
   192                                  ;RHER-ER(R5)
   193          000001                      ILF=    1
   194          000002                      ILR=    2
   195          000004                      RMR=    4
   196
   197          000020                      FMT=    20
   198          000100                      INCVAE= 100
   199          000200                      PEFLRC= 200
   200          000400                      NSG=    400
   201          001000                      FCE=    1000
   202          002000                      CSITM=  2000
   203          004000                      NEF=    4000
   204          010000                      DTE=    10000
   205          020000                      OPI=    20000
   206          040000                      UNS=    40000
   207
   208                                  ;RHMR-MR(R5)
   209          000100                      OSC=    100
   210
   211                                  ;RHDT-DT(R5)
   212          002000                      SPR=    2000
   213          010000                      CH7=    10000
   214          040000                      TAP=    40000
   215
   216                                  ;RHTC-TC(R5)
   217          000300                      NORM11= 300
   218          000320                      CDM11=  320
   219          000000                      BPI200= 0
   220          000400                      BPI556= 000400
   221          001000                      BPI800= 001000
   222          002000                      PE1600= 002000
   223          100000                      ACCL=   100000
   224
```

```
                        ;INSTRUCTION EQUATES
000400                          HLT=    TRAP
000000                          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      OCTAL0: .WORD   0
001120     000      GAP:    .BYTE   0           ;CONTAINS GAP 0 (USED FOR TST 021)
001121     000      ITCNT:  .BYTE   0           ;ITERATION COUNT
001122     000      TSTNUM: .BYTE   0           ;TEST 0
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  000000      ODIGIT: .WORD   0           ;OCTAL DIGITS
```

E3

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

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

001416  000000  000000  STIMTBL:.WORD   0,0              ;SPARE
001422  044560  043740          .WORD   18800.,18400.   ;188.0-184.0   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  035540  035064          .WORD   15200.,14900.   ;152.0-149.0   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    TST011
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 NS(10) GAP #   DELAY IN NS(10)
001572  002602  002412         GTIMTBL:.WORD   01410.,01290.   ;14.1-12.9      GAP=0   0 NS
001576  002652  002506                 .WORD   01450.,01350.   ;14.5-13.0      GAP=1   1.0 NS
001602  002734  002532                 .WORD   01500.,01370.   ;15.0-13.7      GAP=2   2.0 NS
001606  002734  002532                 .WORD   01500.,01370.   ;15.0-13.7      GAP=3   3.0 NS
001612  002734  002424                 .WORD   01500.,01300.   ;15.0-13.0      GAP=4   4.0 NS
001616  002652  002260                 .WORD   01450.,01200.   ;14.5-12.0      GAP=5   5.0 NS
001622  002652  002260                 .WORD   01450.,01200.   ;14.5-12.0      GAP=6   6.0 NS
001626  002652  002260                 .WORD   01450.,01200.   ;14.5-12.0      GAP=7   7.0 NS
001632  002532  002260                 .WORD   01370.,01200.   ;13.7-12.0      GAP=10  8.0 NS
001636  002532  002260                 .WORD   01370.,01200.   ;13.7-12.0      GAP=11  9.0 NS
001642  002532  002260                 .WORD   01370.,01200.   ;13.7-12.0      GAP=12  10.0 NS
001646  002532  002260                 .WORD   01370.,01200.   ;13.7-12.0      GAP=13  11.0 NS
001652  002532  002260                 .WORD   01370.,01200.   ;13.7-12.0      GAP=14  12.0 NS
001656  002532  002260                 .WORD   01370.,01200.   ;13.7-12.0      GAP=15  13.1 NS
001662  002532  002260                 .WORD   01370.,01200.   ;13.7-12.0      GAP=16  14.1 NS
001666  002532  002260                 .WORD   01370.,01200.   ;13.7-12.0      GAP=17  15.1 NS
```

```
                        .SBTTL  TEST HEADER POINTERS
                    ;THE BELOW TABLE CONTAINS POINTERS TO EACH TEST'S DESCRIPTOR
                    NAMPTR: .WORD   0
                            .WORD   A.T001
                            .WORD   A.T002
                            .WORD   A.T003
                            .WORD   A.T004
                            .WORD   A.T005
                            .WORD   A.T006
                            .WORD   A.T007
                            .WORD   A.T010
                            .WORD   A.T011
                            .WORD   A.T012
                            .WORD   A.T013
                            .WORD   A.T014
                            .WORD   A.T015
                            .WORD   A.T016
                            .WORD   A.T017
                            .WORD   A.T020
                            .WORD   A.T021
                            .WORD   0                       ;DUMMY TEST
                            .WORD   A.T023
                            .WORD   A.T024
                            .WORD   A.T025
                            .WORD   A.T026
                            .WORD   A.T027
                            .WORD   A.T030
                            .WORD   A.T031
                            .WORD   A.T032

  370
  371
  372  001672  000000
  373  001674  014665
  374  001676  014707
  375  001700  014727
  376  001702  014751
  377  001704  014775
  378  001706  015017
  379  001710  015036
  380  001712  015060
  381  001714  015103
  382  001716  015125
  383  001720  015152
  384  001722  015201
  385  001724  015232
  386  001726  015263
  387  001730  015311
  388  001732  015340
  389  001734  015370
  390  001736  000000
  391  001740  015413
  392  001742  015437
  393  001744  015463
  394  001746  015507
  395  001750  015534
  396  001752  015556
  397  001754  015601
  398  001756  015623
```

```
                                        ;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  015645                  TYPE,L,CNTG
002030  000004  015652          CNTLU:  TYPE,L,SWR
002034  017702  176740                  MOV     @SWR,R2
002040  004767  000564                  JSR     PC,TYPOCT
002044  000004  015661                  TYPE,L,NEW

002050  005067  177706                  CLR     TEMPST
002054  012767  000007  177702          MOV     #7,COUNT
002062  004767  000154          10:     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     20              ;BRANCH IF NOT
002104  000751                  30:     BR CNTLU                ;START OVER
002106  122767  000015  177644  20:     CMPB    #15,TIB         ;IS IT A <CR>?
002114  001012                          BNE     40              ;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     70              ;CHANGE SWR IF NOT FIRST ONE
002140  000437                  60:     BR      OUT             ;GET OUT
002142  122767  000060  177610  40:     CMPB    #60,TIB
002150  003004                          BGT     50
002152  122767  000067  177600          CMPB    #67,TIB
002160  002003                          BGE     60
002162  000004  015671          50:     TYPE,L,QUEST
002166  000746                          BR      30              ;START OVER IF NOT LEGAL CHARACTER
002170  006367  177566          60:     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     50
002226  000715                          BR      10
002230  016777  177526  176542  70:     MOV     TEMPST,@SWR     ;CHANGE SWITCH REGISTER CONTENTS
002236  000740                          BR      60
002240  000207                  OUT:    RTS     PC
```

J3

```
                                        ;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 ASCII 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 TELEPRINTER 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    #$HT,(SP)       ;;BRANCH IF HORIZONTAL TAB <HT>
002416  001446                  BEQ     8$
```

K3

```
 509  002420  004767  000026            JSR     PC,5$           ;;TYPE CHARACTER
 510  002424  122726  000012      3$:   CMPB    #12,(SP)+       ;;CHECK IF CHARACTER WAS A LINE FEED
 511  002430  001350                    BNE     TYPE1           ;;BRANCH IF NOT LINE FEED
 512  002432  016746  177656            MOV     $NULL,-(SP)     ;;GET # OF FILLERS REQUIRED AND FILLER
 513                                                            ;;CHARACTER.
 514
 515  002436  105366  000001      4$:   DECB    1(SP)           ;;DECREMENT FILLERS REQ. COUNT
 516  002442  002770                    BLT     3$              ;;BRANCH IF NO MORE FILLERS ARE REQUIRED
 517  002444  004767  000002            JSR     PC,5$           ;;TYPE FILLER CHARACTER
 518  002450  000772                    BR      4$
 519
 520  002452  105777  177642      5$:   TSTB    @#TPS           ;;WAIT FOR OUTPUT DEVICE
 521  002456  100375                    BPL     .-4
 522  002460  122737  000017  002325    CMPB    #17,@#CNTRLO    ;;CHECK IF CONTROL O WAS TYPED
 523  002466  001403                    BEQ     6$              ;;STOP TYPING MESSAGE IF ^O WAS TYPED
 524  002470  116677  000002  177624    MOVB    2(SP),@#TPB     ;;OUTPUT CHARACTER
 525  002476  122766  000015  000002  6$:   CMPB    #15,2(SP)       ;;BRANCH IF NOT <CR>
 526  002504  001003                    BNE     7$
 527  002506  105067  177612            CLRB    $CHARCNT        ;;CLEAR CHARACTERS TYPED COUNT
 528  002512  000406                    BR      8$
 529  002514  122766  000012  000002  7$:   CMPB    #12,2(SP)       ;;BRANCH IF <LF> OR 'NULL'
 530  002522  002002                    BGE     8$
 531  002524  105267  177574            INCB    $CHARCNT        ;;INCREMENT CHARACTER TYPED COUNT
 532  002530  000207                8$:   RTS     PC
 533
 534                                    ;;HORIZONTAL TAB <HT> PROCESSER
 535  002532  112716  000040        9$:   MOVB    #40,(SP)        ;;LOAD 'SPACE'
 536  002536  004767  177710       10$:   JSR     PC,5$           ;;TYPE 'SPACE'
 537  002542  132767  000007  177554      BITB    #7,$CHARCNT     ;;TYPE SPACES UNTIL A MULTIPLE
 538  002550  001372                    BNE     10$             ;;OF 8 CHARACTERS HAVE BEEN TYPED
 539  002552  105726                    TSTB    (SP)+           ;;POP SPACE
 540  002554  000676                    BR      TYPE1           ;;GET NEXT CHARACTER
 541
 542                                    ;SUBROUTINE TO SAVE GENERAL REGISTERS ON THE STACK
 543                                    ;CALL:  SAVE
 544  002556  010546            .SAVE:  MOV     R5,-(SP)                ;SAVE REGISTERS ON THE STACK
 545  002560  010446                    MOV     R4,-(SP)
 546  002562  010346                    MOV     R3,-(SP)
 547  002564  010246                    MOV     R2,-(SP)
 548  002566  010146                    MOV     R1,-(SP)
 549  002570  010046                    MOV     R0,-(SP)
 550  002572  016646  000014            MOV     14(SP),-(SP)            ;GET RETURN PC
 551  002576  000207                    RTS     PC                      ;RETURN
 552
 553                                    ;SUBROUTINE TO RESTORE GENERAL REGISTERS FROM THE STACK
 554                                    ;CALL:  RESTORE
 555  002600  012666  000014    .RESTORE:MOV    (SP)+,14(SP)            ;MOVE RETURN PC
 556  002604  012600                    MOV     (SP)+,R0                ;RESTORE REGISTERS
 557  002606  012601                    MOV     (SP)+,R1
 558  002610  012602                    MOV     (SP)+,R2
 559  002612  012603                    MOV     (SP)+,R3
 560  002614  012604                    MOV     (SP)+,R4
 561  002616  012605                    MOV     (SP)+,R5
 562  002620  000207                    RTS     PC                      ;RETURN
 563
 564                                    ;SUBROUTINE TO CONVERT OCTAL DATA TO ASCII
```

```
                                        ;CALL:  MOV     NUMBER,R2               ;MOVE NUMBER TO R2
                                        ;       JSR     PC,CNVOCT
                                        CNVOCT: MOVB    SP,TYPFLG               ;SET DO NOT TYPE FLAG
                                                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
                                        TYPOCT: CLRB    @#TYPFLG                ;SET TYPE FLAG
                                        CNVTO:
                                                JSR     PC,,SAVE                ;SAVE REGISTERS ON THE STACK
                                                MOV     #ODIGITS,R4             ;SET PTR TO OUTPUT
                                                CLR     R3                      ;R3 WILL CONTAIN OCTAL DIGIT
                                                MOV     R2,R1                   ;GET # TO BE TYPED
                                        10:     ASL     R2                      ;SHIFT #
                                                ROL     R3
                                                MOV     #6,R0                   ;SET DIGIT COUNTER
                                                BR      30
                                        20:     ASL     R2                      ;SHIFT # 3 PLACES LEFT
                                                ROL     R3
                                                DEC     R1
                                                BNE     20
                                        30:     MOV     #3,R1                   ;SET SHIFT COUNTER
                                                MOVB    DIGTAB(R3),(R4)+        ;MOVE ASCII EQUIV TO OUTPUT
                                                CLR     R3
                                                DEC     R0                      ;DECREMENT DIGIT COUNT
                                                BNE     20                      ;GET NEXT DIGIT
                                                TSTB    @#TYPFLG                ;BRANCH IF ASCII IS
                                                BNE     40                      ;NOT TO BE TYPED
                                                TYPE,@#ODIGITS
                                        40:
                                                JSR     PC,,RESTORE             ;RESTORE REGISTERS FROM THE STACK
                                                RTS     PC


                                        ;SUBROUTINE TO CONVERT OCTAL DATA TO DECIMAL ASCII
                                        ;CALL:  MOV     NUMBER,R2               ;MOVE NUMBER TO R2
                                        ;       JSR     PC,CNVDEC

                                        CNVDEC: MOVB    SP,@#TYPFLG             ;SET DO NOT TYPE FLAG
                                                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

                                        TYPDEC: CLRB    @#TYPFLG                ;SET TYPE FLAG
                                        CNVTD:
                                                JSR     PC,,SAVE                ;SAVE REGISTERS ON THE STACK
                                                CLR     R0                      ;R0 IS INDEX TO DECIMAL CONSTANT
                                                MOV     #ODIGITS,R4             ;SET OUTPUT PTR
                                        10:     CLR
```

m3

```
002756  166002  003036     2$:     SUB     DCONST(R0),R2           ;SUBTRACT DECIMAL CONSTANT UNTIL
002762  103402                     BLO     3$                      ;INPUT # GOES NEGATIVE
002764  005203                     INC     R3                      ;KEEPING TRACK OF SUBTRACTIONS
002766  000773                     BR      2$
002770  066002  003036     3$:     ADD     DCONST(R0),R2           ;ADD BACK CONSTANT WHEN NEGATIVE
002774  116324  001132             MOVB    DIGTAB(R3),(R4)+        ;MOVE ASCII EQUIVILENT
003000  062700  000002             ADD     #2,R0                   ;NEXT CONSTANT
003004  005760  003036             TST     DCONST(R0)              ;UNTIL ALL CONSTANTS DONE
003010  001361                     BNE     1$
003012  112724  000060             MOVB    #'0,(R4)+               ;LAST DIGIT IS 0
003016  105737  001126             TSTB    @#TYPFLG                ;BRANCH IF ASCII IS
003022  001002                     BNE     4$                      ;NOT TO BE TYPED
003024  000004  001144             TYPE,ODIGITS
003030                     4$:
003030  004767  177544             JSR     PC,.RESTORE             ;RESTORE REGISTERS FROM THE STACK
003034  000207                     RTS     PC

003036  023420             DCONST: .WORD   10000.
003040  001750                     .WORD   1000.
003042  000144                     .WORD   100.
003044  000012                     .WORD   10.
003046  000001                     .WORD   1.
003050  000000                     .WORD   0                       ;TERMINATOR

                                   .SBTTL          TYPE SPECIFIED TIMES ROUTINE
                           ;THIS SUBROUTINE OUTPUTS THE TIME SPECIFICATIONS FOR THE TEST
                           ;AND ALSO THE ACTUAL TIME RECORDED (ATIME)
                           ;FORMAT OF LINE TYPED
                           ;RANGE=<AAAAAA-BBBBBB>          ACTUAL=CCCCCC
                           ;WHERE:          AAAAAA IS MAXIMUM TIME FOR TEST (STIMTBL(TSTNUMX4)),
                           ;                BBBBBB IS MINIMUM TIME FOR TEST (STIMTBL(TSTNUMX4+2)),
                           ;                CCCCCC IS ACTUAL TIME RECORDED BY TEST (ATIME),
                           ;CALL:   MOVB    TEST NUMBER,R2  ;LOAD TEST NUMBER
                           ;        MOV     $TIME,@#ATIME           ;MOVE TIME TO ATIME
                           ;        JSR     PC,OUTSPC
003052  010246             OUTSPC: MOV     R2,-(SP)                ;SAVE R2 & R3 ON THE STACK
003054  010346                     MOV     R3,-(SP)
003056  006302                     ASL     R2                      ;MULTIPLY TEST # TIMES 4
003060  006302                     ASL     R2                      ;TO FORM INDEX INTO STIMTBL
003062  010203                     MOV     R2,R3                   ;R3 CONTAINS INDEX INTO TABLE
003064  000004  014645             TYPE,L.RNG
003070  016302  001416             MOV     STIMTBL(R3),R2          ;GET MAXIMUM SPEC TIME
003074  004767  177636             JSR     PC,TYPDEC               ;CONVERT TO DECIMAL & TYPE
003100  000004  001405             TYPE,DASH
003104  016302  001420             MOV     STIMTBL+2(R3),R2        ;GET MINIMUM TIME
003110  004767  177622             JSR     PC,TYPDEC               ;CONVERT TO DECIMAL & TYPE
003114  000004  001412             TYPE,ANGTAB
003120  000004  014655             TYPE,L.ACT
003124  013702  001012             MOV     @#ATIME,R2              ;GET ACTUAL TIME
003130  004767  177602             JSR     PC,TYPDEC               ;CONVERT TO DECIMAL & TYPE
003134  000004  001374             TYPE,CRLF
003140  012603                     MOV     (SP)+,R3
003142  012602                     MOV     (SP)+,R2
003144  000207                     RTS     PC                      ;RETURN

                                   .SBTTL          TYPE C...
```

```
                                        ;THIS SUBROUTINE IS USED TO TYPE THE SPECIFIED GAP SIZES (RECORDED IN
                                        ;TST021), IT IS CALLED BY THE GAPOK ROUTINE IF THE GAP SIZE IS OUT OF
                                        ;RANGE VIA THE HLT ROUTINE (HLT+2).
                                        ;CALL:  MOVB    #GAP,GAP            ;LOAD GAP # INTO GAP
                                        ;       MOV     #TIME,ATIME         ;LOAD ACTUAL TIME INTO ATIME
                                        ;       JSR     PC,OUTGAP
                                        OUTGAP: MOV     R2,-(SP)            ;SAVE R2 AND R3
                                                MOV     R3,-(SP)
003152  116703  175742                          MOVB    GAP,R3              ;GET GAP #
003156  006303                                  ASL     R3
003160  006303                                  ASL     R3
003162  000004  014645                          TYPE,L,RNG
003166  016302  001572                          MOV     GTIMTBL(R3),R2      ;GET MAX TIME
003172  004767  177540                          JSR     PC,TYPDEC           ;CONVERT TO DECIMAL & TYPE
003176  000004  001405                          TYPE,DASH
003202  016302  001574                          MOV     GTIMTBL+2(R3),R2    ;GET MIN TIME
003206  004767  177524                          JSR     PC,TYPDEC           ;CONVERT TO DECIMAL & TYPE
003212  000004  001412                          TYPE,ANGTAB                 ;TYPE <
003216  000004  014655                          TYPE,L,ACT
003222  013702  001012                          MOV     @#ATIME,R2          ;GET ACTUAL TIME
003226  004767  177504                          JSR     PC,TYPDEC           ;CONVERT TO DECIMAL & TYPE
003232  000004  014334                          TYPE,E,GAP
003236  113702  001120                          MOVB    @#GAP,R2            ;GET GAP #
003242  004767  177362                          JSR     PC,TYPOCT           ;TYPE GAP #
003246  000004  001374                          TYPE,CRLF
003252  012603                                  MOV     (SP)+,R3            ;RESTORE R3 AND R2
003254  012602                                  MOV     (SP)+,R2
003256  000207                                  RTS     PC

                                                .SBTTL          ASCII TO OCTAL CONVERT SUBROUTINE
                                        ;SUBROUTINE TO CONVERT ASCII DATA TO OCTAL, CONVERTED OCTAL DATA
                                        ;IS LEFT IN OCTAL0 <15-00>.
003260                                  CNVTAO:
003260  004767  177272                          JSR     PC,,SAVE            ;SAVE REGISTERS ON THE STACK
003264  012700  001264                          MOV     #INBUF,R0           ;SET PTR TO ASCII DATA
003270  012701  001116                          MOV     #OCTAL0,R1          ;GET ADDRESS OF OCTAL DATA
003274  005011                                  CLR     (R1)                ;CLEAR OUT OLD OCTAL DATA
003276  005061  000002                          CLR     2(R1)
003302  122710  000015                  1$:     CMPB    #CR,(R0)            ;<CR> TERMINATES INPUT
003306  001414                                  BEQ     3$
003310  112002                                  MOVB    (R0)+,R2            ;GET 'OCTAL' DATA
003312  042702  177770                          BIC     #177770,R2          ;STRIP UNUSED BITS
003316  012703  000003                          MOV     #3,R3               ;SET SHIFT COUNT
003322  006311                          2$:     ASL     (R1)                ;SHIFT LAST
003324  006161  000002                          ROL     2(R1)               ;OCTAL DIGIT
003330  005303                                  DEC     R3
003332  001373                                  BNE     2$
003334  050211                                  BIS     R2,(R1)             ;AND INSERT THIS DIGIT
003336  000761                                  BR      1$                  ;GO GET NEXT DIGIT
003340                                  3$:
003340  004767  177234                          JSR     PC,,RESTORE         ;RESTORE REGISTERS FROM THE STACK
003344  000207                                  RTS     PC                  ;RETURN

                                                .SBTTL          PUBLISH SUBROUTINE
                                        ;THE PUBLISH SUBROUTINE AVERAGES THE RECORDED RUNS FOR EACH ERROR
```

B4

```
                                ;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:
003346  004767  177284                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 RETUNED 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    BS,BELL
```

C4

```
 789  003534  000762                        BR      28              ;WAIT FOR NEXT CHARACTER
 790  003536  122716  000025        38:     CMPB    @CNTRLU,(SP)    ;CHECK CONTROL U (^U)
 791  003542  001004                        BNE     48
 792  003544  005726                        TST     (SP)+
 793  003546  000004  001374                TYPE,CRLF
 794  003552  000751                        BR      18
 795  003554  111637  001401        48:     MOVB    (SP),@#ECHO
 796  003560  111620                        MOVB    (SP),(R0)+
 797  003562  122726  000015                CMPB    @CR,(SP)+
 798  003566  001403                        BEQ     58
 799  003570  000004  001401                TYPE,ECHO
 800  003574  000742                        BR      28
 801  003576  000004  001374        58:     TYPE,CRLF
 802  003602  012600                        MOV     (SP)+,R0
 803  003604  000207                        RTS     PC
 804
 805                                  ;KEYBOARD INTERRUPT SERVICE ROUTINE
 806  003606  113746  177562        TKISR:  MOVB    @#TKB,-(SP)     ;GET TYPED CHARACTER
 807  003612  042716  000200                BIC     #200,(SP)       ;STRIP PARITY BIT
 808  003616  122716  000017                CMPB    @CNTRLO,(SP)    ;BRANCH IF NOT CONTROL O (^O)
 809  003622  001003                        BNE     18
 810  003624  112667  176475                MOVB    (SP)+,@CNTRLO   ;SET CONTROL O INDICATOR IN TYPE ROUTINE
 811  003630  000002                        RTI                     ;EXIT
 812
 813  003632  122726  000003        18:     CMPB    #3,(SP)+        ;BRANCH IF NOT CONTROL C (^C)
 814  003636  001003                        BNE     28
 815  003640  000005                        RESET
 816  003642  000137  005724                JMP     @#INIT          ;RESTART PROGRAM
 817  003646  000002                28:     RTI                     ;EXIT
```

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

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

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

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

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

E4

```
   874  004104  005777  174670          58:     TST     @SWR                    ;HALT ON ERROR?
   875  004110  100001                          BPL     68
   876  004112  000000                          HALT
   877  004114  004767  175650          68:     JSR     PC,CKSWR        ;CHECK FOR CNTL C
   878  004120  004767  176454                  JSR     PC,,RESTORE             ;RESTORE REGISTERS FROM THE STACK
   879  004124  000002                          RTI                             ;RETURN
   880
   881
```

```
                                        .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 (ATINTBL)
                                ;       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,RMINIT               ;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,ATINTBL(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
004344  000000          JSR     PC,CKSWR                ;CHECK FOR CNTL G
004346  004767  175416  BR      18
004352  000765
                        .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

004354  005004          TIMON:  CLR     R4                      ;CLEAR TIME COUNT
004356  012703  000024          MOV     #24,R3                  ;SET POLARITY TO '0' STATE
004362  032765  000100  000024  BIT     #OSC,MR(R5)             ;BRANCH IF POLARITY IS '0'
004370  001405                  BEQ     2$
004372  032765  000100  000024  1$:     BIT     #OSC,MR(R5)             ;WAIT FOR OSCILLATOR TO RETURN
004400  001374                  BNE     1$
004402  000405                  BR      4$

004404  005403          2$:     NEG     R3                      ;NEGATE PREV POLARITY INDICATOR
004406  032765  000100  000024  3$:     BIT     #OSC,MR(R5)             ;WAIT FOR OSCILLATOR TO RETURN
004414  001774                  BEQ     3$                      ;TO '1' STATE
004416  000207          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 48 US.

                        ;ENTER HERE VIA JMP  TIMER(R3) WHEN R3=-24 (PREV STATE=1)
004420  032765  000100  000024  TIMER1: BIT     #OSC,MR(R5)             ;BRANCH IF CURRENT STATE IS '0'
004426  001406                  BEQ     TIMER                   ;GO INCREMENT TIME
004430  000112                  JMP     (R2)                    ;RETURN TO TEST

        004444                  .=TIMER1+24
004444  005403          TIMER:  NEG     R3                      ;NEGATE PREV STATE INDICATOR
004446  005204                  INC     R4                      ;INCREMENT 'TICK' COUNT
004450  100401                  BMI     TIMERR                  ;BRANCH ON OVERFLOW
004452  000112                  JMP     (R2)                    ;RETURN TO TEST
004454  000004  014246  TIMERR: TYPE,E,TIMOV                   ;TYPE 'TIMER OVERFLOWED'
004460  104400                  HLT                             ;REPORT HARDWARE ERROR
004462  000177  174314          JMP     @#OSCPADR               ;RETURN TO BEGINNING OF TEST

        004470                  .=TIMER+24
                        ;ENTER HERE VIA JMP  TIMER(R3) WHEN R3=+24 (PREV STATE=0)
004470  032765  000100  000024  TIMER0: BIT     #OSC,MR(R5)             ;BRANCH IF CURRENT STATE = '1'
004476  001362                  BNE     TIMER
004500  000112                  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 (@TIMTBL(R0)) AND THE LOW LIMIT (@TIMTBL+2(R0)).
                                        ;IF THE TIME IS OUT OF RANGE AN OUT OF RANGE ERROR TYPEOUT RESULTS.
                                        ;THE SUBROUTINE IS ENTERED WITH:
                                        ;       R4=TICK COUNT

004502                                  TIMOK:
004502  004767  176958                          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                          10:     ADD     R0,R2                   ;MULTIPLY TIME PER TICK
004522  005503                                  ADC     R3                      ;BY TICK COUNT
004524  005301                                  DEC     R1
004526  001374                                  BNE     10
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,$TIMTBL(R0)     ;CHECK THAT TIME IS WITHIN
004570  101004                                  BHI     20                      ;LIMITS SPECIFIED
004572  023760  001012  001420                  CMP     @#ATIME,$TIMTBL+2(R0)
004600  101001                                  BHI     30
004602  104401                          20:     MLT+1                           ;CALL ERROR ROUTINE
004604                                  30:
004604  004767  175770                          JSR     PC,,RESTORE             ;RESTORE REGISTERS FROM THE STACK
004610  000207                                  RTS     PC                      ;RETURN

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

004612                                  GAPOK:
004612  004767  175740                          JSR     PC,,SAVE                ;SAVE REGISTERS ON THE STACK
004616  012700  000070                          MOV     #56.,,R0                ;GET TIME PER TICK
004622  010401                                  MOV     R4,R1                   ;GET TICK COUNT
004624  005002                                  CLR     R2                      ;CLEAR SUMMING REGISTERS
004626  005003                                  CLR     R3
004630  060002                          10:     ADD     R0,R2                   ;MULTIPLY TICK COUNT
004632  005503                                  ADC     R3                      ;BY TIME PER TICK
004634  005301                                  DEC     R1
004636  001374                                  BNE     10
```

```
1050
1051  004640  010246                        MOV     R2,-(SP)                ;DIVIDE TIME BY 10.
1052  004642  010346                        MOV     R3,-(SP)
1053  004644  012746  000012                MOV     #10.,-(SP)
1054  004650  004767  000152                JSR     PC,DIVIDE
1055  004654  005726                        TST     (SP)+                   ;DISCARD REMAINDER
1056  004656  012637  001012                MOV     (SP)+,@#ATIME           ;STORE QUOTIENT
1057  004662  113703  001120                MOVB    @#GAP,R3                ;GET GAP #
1058  004666  006303                        ASL     R3                      ;MULTPLY BY 4
1059  004670  006303                        ASL     R3                      ;TO GET AT TABLE ENTRY
1060  004672  023763  001012  001572        CMP     @#ATIME,GTIMTBL(R3)     ;CHECK TIME (MAX)
1061  004700  101004                        BHI     2$
1062  004702  023763  001012  001574        CMP     @#ATIME,GTIMTBL+2(R3)   ;CHECK TIME (MIN)
1063  004710  101002                        BHI     3$
1064  004712  104402                2$:     HLT+2                           ;REPORT OUT OF RANGE ERROR
1065  004714  000406                        BR      1000$
1066  004716  032777  000400  174054  3$:   BIT     @#SWR0,@#SWR            ;BRANCH IF TIMES NOT WANTED
1067  004724  001402                        BEQ     1000$
1068  004726  004767  176214                JSR     PC,OUTGAP               ;TYPE GAP TIMES
1069
1070  004732                        1000$:
1071  004732  004767  175642                JSR     PC,,RESTORE             ;RESTORE REGISTERS FROM THE STACK
1072  004736  000207                        RTS     PC                              ;RETURN TO TEST
1073
1074                                        .SBTTL          DELAY SUBROUTINES
1075                                ;THIS SUBROUTINE CAUSES A DELAY OF 350 MS.
1076  004740  004767  177410        DELAY:  JSR     PC,TIMON
1077  004744  010246                        MOV     R2,-(SP)                ;SAVE R2 ON THE STACK
1078  004746  012702  004756                MOV     #2$,R2                  ;SET RETURN ADDRESS FOR TIMER
1079  004752                        1$:
1080  004752  000163  004444                JMP     TIMER(R3)               ;GO TO TIMER & RETURN VIA R2
1081  004756  032704  004000        2$:     BIT     #4000,R4
1082  004762  001773                        BEQ     1$
1083  004764  012602                        MOV     (SP)+,R2                ;RESTORE R2
1084  004766  000207                        RTS     PC
1085
1086                                ;THIS SUBROUTINE ALLOWS A CALLER SPECIFIED DELAY (UP TO 65MS.)
1087                                ;CALL:  MOV     DELAY TIME,DELTIM       ;LOAD DELAY TIME (IN US)
1088                                ;       JSR     PC,DELAYV
1089  004770  005767  174120        DELAYV: TST     DELTIM                  ;BRANCH IF 0 DELAY
1090  004774  001413                        BEQ     3$
1091  004776  004767  177352                JSR     PC,TIMON                ;TURN TIMER ON
1092  005002  010246                        MOV     R2,-(SP)                ;SAVE R2 ON THE STACK
1093  005004  012702  005014                MOV     #2$,R2                  ;SET RETURN ADDRESS FROM TIMER
1094  005010                        1$:
1095  005010  000163  004444                JMP     TIMER(R3)               ;GO TO TIMER & RETURN VIA R2
1096  005014  023704  001114        2$:     CMP     @#DELTIM,R4
1097  005020  101373                        BHI     1$
1098  005022  012602                        MOV     (SP)+,R2                ;RESTORE R2
1099  005024  000207                3$:     RTS     PC
1100
1101                                        .SBTTL  DIVIDE SUBROUTINE
1102                                ;THIS SUBROUTINE DIVIDES A DOUBLE PRECISION # AND RETURNS THE RESULT
1103                                ;TO THE CALLER ON THE STACK. BOTH DIVIDEND & DIVISOR MUST BE POSITIVE.
1104                                ;CALL:  MOV     LEAST SIGNIFICANT HALF DIVIDEND,-(SP)
1105                                ;       MOV     ;MOST SIGNIFICANT HALF DI
```

J4

```
1106                                    ;       MOV     #DIVISOR,-(SP)
1107                                    ;       JSR     PC,DIVIDE
1108                                    ;RETURN
1109                                    ;       (SP)=REMAINDER ON STACK
1110                                    ;       2(SP)=QUOTIENT
1111
1112                                    ;NOTE:  THIS SUBROUTINE DESTROYS PREVIOUS CONTENTS OF R0,R1,R2 & R3.
1113
1114  005026  005046                    DIVIDE: CLR     -(SP)                   ;SAVE LOC FOR SIGNS
1115  005030  012746  000021                    MOV     #17.,-(SP)              ;SET ITERATION COUNT
1116  005034  016601  000012                    MOV     12(SP),R1               ;GET LSW DIVIDEND
1117  005040  016600  000010                    MOV     10(SP),R0               ;GET MSW DIVIDEND
1118  005044  016602  000006                    MOV     6(SP),R2                ;GET DIVISOR
1119  005050  005402                    NEG     R2                      ;NEGATE DIVISOR
1120  005052  000241                    CLC                             ;CLEAR 'C' BIT IN PSW
1121  005054  000405                    BR      2$
1122  005056  006100                    1$:     ROL     R0                      ;ROTATE MSW DIVIDEND
1123  005060  010003                    MOV     R0,R3                   ;SAVE IN R3
1124  005062  060203                    ADD     R2,R3                   ;SUBTRACT DIVISOR FROM MSW DIVIDEND
1125  005064  103001                    BCC     2$                      ;BRANCH IF DIVIDEND > DIVISOR
1126  005066  010300                    MOV     R3,R0                   ;SAVE REMAINDER IN R0
1127  005070  006101                    2$:     ROL     R1                      ;ROTATE LSW DIVIDEND
1128  005072  005316                    DEC     (SP)                    ;DECREMENT ITERATION COUNT
1129  005074  001370                    BNE     1$
1130  005076  005726                    TST     (SP)+                   ;POP ITERATION COUNTER
1131  005100  005726                    TST     (SP)+                   ;POP SIGN CORRECTION
1132  005102  010166  000006                    MOV     R1,6(SP)                ;PUSH REMAINDER ON STACK
1133  005106  010066  000004                    MOV     R0,4(SP)                ;PUSH QUOTIENT ONTO STACK
1134  005112  012616                    MOV     (SP)+,(SP)
```

K4

```
1135  005114  000207                  RTS     PC
1136
1137                                  .SBTTL  DRIVE SUBROUTINES
1138                                  ;SUBROUTINE TO CHECK IF DRIVE IS AVAILABLE
1139                                  ;CALL:  MOVB    DRIVE#,DRVNUM
1140                                  ;       JSR     PC,DRVAVA
1141                                  ;RETURN:        'C' BIT SET IF NOT AVAILABLE
1142  005116  113765  001004  000010 DRVAVA: MOVB    @#DRVNUM,CS2(R5)        ;LOAD DRIVE #
1143  005124  032765  040000  000026         BIT     #TAP,DT(R5)             ;CHECK IF TAPE UNIT
1144  005132  001003                        BNE     1$
1145  005134  004767  000034                JSR     PC,RHINIT
1146  005140  000262                        SEV                             ;SET 'V' TO IND NOT AVAIL
1147  005142  000207                  1$:   RTS     PC                      ;RETURN
1148
1149                                  ;SUBROUTINE TO CHECK IF TU16 SLAVE IS AVAILABLE FOR TEST
1150                                  ;CALL:  MOVB    DRIVE #,@#DRVNUM        ;PASS DRIVE # VIA DRVNUM
1151                                  ;       MOVB    SLAVE #,@#SLVNUM        ;PASS SLAVE # VIA SLVNUM
1152                                  ;       JSR     PC,SLVAVA               ;CALL SUBROUTINE
1153  005144  113765  001004  000010 SLVAVA: MOVB    @#DRVNUM,CS2(R5)        ;LOAD DRIVE #
1154  005152  113765  001005  000032         MOVB    @#SLVNUM,TC(R5)         ;AND SLAVE #
1155  005160  032765  002000  000026         BIT     #SPR,DT(R5)             ;BRANCH IF SLAVE PRESENT
1156  005166  001001                        BNE     1$
1157  005170  000262                        SEV                             ;SET 'V' TO INDICATE NO SLAVE
1158  005172  000207                  1$:   RTS     PC
1159
1160                                  ;SUBROUTINE TO  INITIALIZE RH CONTROLLER
1161                                  ;CALL:  JSR PC,RHINIT
1162
1163  005174  012765  000040  000010 RHINIT: MOV     #40,CS2(R5)
1164  005202  113765  001004  000010         MOVB    @#DRVNUM,CS2(R5)
1165  005210  005046                        CLR     -(SP)
1166  005212  113716  001005                MOVB    @#SLVNUM,(SP)
1167  005216  012665  000032                MOV     (SP)+,TC(R5)            ;LOAD SLAVE # INTO TC REG
1168  005222  052765  000300  000032         BIS     #NORM11,TC(R5)
1169  005230  000207                        RTS     PC
1170
1171                                  ;SUBROUTINE TO WAIT FOR DRIVE READY (DRY)
1172  005232  005027                  WAITRDY:CLR     (PC)+                   ;CLEAR WAIT TIMER
1173  005234  000000                  WAITTIM:.WORD   0
1174  005236  105765  000012          1$:   TSTB    DS(R5)                  ;WAIT FOR READY TO SET
1175  005242  100406                        BMI     2$
1176  005244  005267  177764                INC     WAITTIM                 ;INCREMENT WAIT TIMER
1177  005250  001372                        BNE     1$                      ;BRANCH IF TIME HAS NOT EXPIRED
1178  005252  000004  014273                TYPE,E,TIMEXP                   ;TYPE 'TIME EXPIRED WAITING FOR RDY'
1179  005256  000425                        BR      99$                     ;TAKE ERROR EXIT
1180  005260  032765  002000  000012  2$:   BIT     #EOT,DS(R5)             ;CHECK FOR END OF TAPE
1181  005266  001415                        BEQ     3$                      ;BRANCH IF NO EOT
1182  005270  000004  013330                TYPE,M,NAM
1183  005274  000004  013656                TYPE,M,EOT                      ;TYPE 'END OF TAPE'
1184  005300  004767  000032                JSR     PC,,REWIND              ;REWIND SLAVE
1185  005304  102412                        BVS     99$                     ;BRANCH IF ERROR ON REWIND
1186  005306  004767  000106                JSR     PC,WRITE                ;WRITE A RECORD
1187  005312  005215                        INC     (R5)                    ;SET 'GO' BIT
1188  005314  004767  177712                JSR     PC,WAITRDY              ;WAIT FOR READY
1189  005320  000404                        BR      99$                     ;TAKE ERROR EXIT
1190  005322  032765  040000  000012  3$:   BIT     #ATA,DS(R5)
```

L4

```
1191  005330  001401                    BEQ     1000
1192  005332  000262            9981:   SEV
1193  005334  000207            10001:  RTS     PC
1194
```

m4

```
1195                            ;SUBROUTINE TO REWIND A UNIT (DRIVE/SLAVE COMBINATION)
1196                            ;CALL   MOVB    DRIVE 0,@#DRVNUM
```

```
                                ;       MOVB    SLAVE 0,@#SLVNUM
                                ;       JSR     PC,.REWIND
                                ;SUBROUTINE RETURNS TO CALLER WITH SELECTED SLAVE AT 'BOT', & 'V' SET IF
                                ;AN ERROR OCCURS.

1202  005336  004767  177632  .REWIND:JSR     PC,RHINIT               ;INITIALIZE CONTROLLER
```

```
1203  005342  004367  000206          JSR     R3,TMCMD                ;GO TO TM COMMAND SUBROUTINE
1204  005346  000000                  .WORD   0                       ;BUS ADDRESS (NOT USED)
1205  005350  000000                  .WORD   0                       ;WORD COUNT (NOT USED)
1206  005352  000000                  .WORD   0                       ;FRAME COUNT (NOT USED)
1207  005354  000006                  .WORD   RWD                     ;REWIND COMMAND
1208  005356  005215                  INC     (R5)                    ;SET 'GO' BIT
1209  005360  032765  000002  000012  10$:    BIT     #BOT,DS(R5)             ;BRANCH IF 'BOT' SET
1210  005366  001005                  BNE     20$
1211  005370  032765  040000  000012          BIT     #ERR,DS(R5)             ;CHECK ERROR BIT
1212  005376  001006                  BNE     99$                     ;BRANCH IF ERROR BIT SET
1213  005400  000767                  BR      10$
1214
1215  005402  032765  020000  000012  20$:    BIT     #PIP,DS(R5)             ;WAIT FOR TAPE MOTION TO STOP
1216  005410  001374                  BNE     20$
1217  005412  000401                  BR      100$
1218  005414  000262          99$:    SEV
1219  005416  000207          100$:   RTS     PC
1220
1221                          ;SUBROUTINE TO WRITE 256. WORD RECORD
1222                          ;CALL:  JSR     PC,WRITE
1223
1224  005420  004367  000130  WRITE:  JSR     R3,TMCMD                ;GO TO TM COMMAND SUBROUTINE
1225  005424  015700                  .WORD   WTBUF                   ;BUS ADDRESS
1226  005426  177680                  .WORD   WRDCNT                  ;WORD COUNT
1227  005430  177480                  .WORD   FRMCNT                  ;FRAME COUNT
1228  005432  000060                  .WORD   WFWD                    ;WRITE FORWARD COMMAND
1229  005434  000207                  RTS     PC
1230
1231                          ;SUBROUTINE TO READ A 256. WORD RECORD.
1232                          ;CALL:  JSR     PC,READ
1233
1234  005436  004337  005554  READ:   JSR     R3,@#TMCMD
1235  005442  015700                  .WORD   RDBUF                   ;ADDRESS OF READ BUFFER
1236  005444  177680                  .WORD   WRDCNT                  ;2'S COMPLEMENT OF WORD COUNT
1237  005446  177480                  .WORD   FRMCNT                  ;2'S COMPLEMENT OF FRAME COUNT
1238  005450  000070                  .WORD   RDFWD                   ;READ FORWARD COMMAND
1239  005452  000207                  RTS     PC
1240
1241                          ;SUBROUTINE TO INITIATE READ REVERSE COMMAND
1242                          ;CALL:  JSR     PC,REVRD
1243                          ;
1244  005454  004367  000074  REVRD:  JSR     R3,TMCMD
1245  005460  016300                  .WORD   RDBUF+256.              ;ADDRESS OF READ REVERSE BUFFER
1246  005462  177680                  .WORD   WRDCNT                  ;2'S COMPLEMENT OF WORD COUNT
1247  005464  177480                  .WORD   FRMCNT                  ;2'S COMPLEMENT OF FRAME COUNT
1248  005466  000076                  .WORD   RDREV                   ;READ REVERSE COMMAND
1249  005470  000207                  RTS     PC
1250
1251                          ;SUBROUTINE TO SPACE FORWARD 1 RECORD
1252  005472  012765  177777  000006  FWDSPC: MOV     #-1,FC(R5)              ;LOAD RECORD COUNT
1253  005500  012715  000031          MOV     #SPCFWD+1,(R5)          ;LOAD COMMAND
1254  005504  004767  177522          JSR     PC,WAITRDY              ;WAIT FOR READY
1255  005510  000207                  RTS     PC                      ;RETURN
1256
1257                          ;SUBROUTINE TO WRITE A RECORD AND BACK SPACE OVER  THE RECORD.
1258  005512  004767  177782  WRT.BK:  JSR     PC,WRITE                ;WRITE THE RECORD
```

C5

```
1259  005516  005215                          INC     (R5)                    ;SET 'GO' BIT
1260  005520  004767  177586                  JSR     PC,WAITRDY
1261  005524  102412                          BVS     2$
1262  005526  012765  177777  000006          MOV     #-1,FC(R5)              ;LOAD RECORD COUNT
1263  005534  012715  000033                  MOV     #$PCREV+1,(R5)          ;LOAD COMMAND
1264  005540  004767  177466                  JSR     PC,WAITRDY
1265  005544  102402                          BVS     2$
1266  005546  004767  177166          1$:     JSR     PC,DELAY                ;WAIT FOR TAPE MOTION TO STOP
1267  005552  000207                  2$:     RTS     PC
1268
1269                                  ;SUBROUTINE TO LOAD A COMMAND
1270                                  ;CALL: JSR     R3,TMCMD
1271                                  ;       .WORD   BUS ADDRESS
1272                                  ;       .WORD   WORD COUNT (2'S COMPLEMENT)
1273                                  ;       .WORD   FRAME COUNT (2'S COMPLEMENT)
1274                                  ;       .WORD   COMMAND
1275
1276  005554  012365  000004          TMCMD:  MOV     (R3)+,BA(R5)            ;LOAD BUS ADDRESS
1277  005560  012365  000002                  MOV     (R3)+,WC(R5)            ;LOAD WORD COUNT
1278  005564  012365  000006                  MOV     (R3)+,FC(R5)            ;LOAD FRAME COUNT
1279  005570  012315                          MOV     (R3)+,(R5)              ;LOAD COMMAND
1280  005572  000203                          RTS     R3                      ;RETURN
1281
1282                                  ;SUBROUTINE TO PRINT TU16 SERIAL NUMBER
1283                                  ;JSR     PC,SNPT
1284
1285  005574  016503  000030          SNPT:   MOV     SN(R5),R3
1286  005600  012701  001144                  MOV     #ODIGITS,R1
1287  005604  000303                          SWAB    R3
1288  005606  006003                          ROR     R3
1289  005610  006003                          ROR     R3
1290  005612  006003                          ROR     R3
1291  005614  006003                          ROR     R3                      ;GET FIRST DIGIT
1292  005616  042703  177760                  BIC     #177760,R3
1293  005622  052703  000260                  BIS     #260,R3
1294  005626  110321                          MOVB    R3,(R1)+                ;FILL FIRST DIGIT
1295  005630  016503  000030                  MOV     SN(R5),R3
1296  005634  000303                          SWAB    R3
1297  005636  042703  177760                  BIC     #177760,R3
1298  005642  052703  000260                  BIS     #260,R3
1299  005646  110321                          MOVB    R3,(R1)+                ;GET SECOND DIGIT
1300  005650  016503  000030                  MOV     SN(R5),R3
1301  005654  006003                          ROR     R3
1302  005656  006003                          ROR     R3
1303  005660  006003                          ROR     R3
1304  005662  006003                          ROR     R3
1305  005664  042703  177760                  BIC     #177760,R3
1306  005670  052703  000260                  BIS     #260,R3
1307  005674  110321                          MOVB    R3,(R1)+                ;GET THIRD DIGIT
1308  005676  016503  000030                  MOV     SN(R5),R3
1309  005702  042703  177760                  BIC     #177760,R3
1310  005706  052703  000260                  BIS     #260,R3
1311  005712  110321                          MOVB    R3,(R1)+                ;GET FOURTH DIGIT
1312  005714  105011                          CLRB    (R1)
1313  005716  000004  001144                  TYPE,ODIGITS                    ;TYPE SERIAL NUMBER
1314  005722  000207                          RTS     PC                      ;RETURN
```

```
1315
1316                                        .SBTTL  PROGRAM INITIALIZATION
1317  005724  012706  000600        INIT:   MOV     #STKPTR,SP              ;SET STACK PTR
1318
1319  005730  013746  000006        SUSWR:  MOV     @6,-(SP)                ;SAVE VECTORS
1320  005734  013746  000004                MOV     @4,-(SP)
1321  005740  012737  005760  000004        MOV     #618,@4                         ;SET UP FOR TIMEOUT
1322  005746  022777  177777  173024        CMP     @-1,@SWR                ;REFERENCE HARDWARE SWITCH REGISTER
1323  005754  001402                       BEQ     608
1324  005756  000404                       BR      628
1325  005760  022626                6181:   CMP     (SP)+,(SP)+             ;ADJUST STACK
1326  005762  012767  000176  173010  6081:   MOV     #SWREG,SWR              ;POINT TO SOFTWARE SWITCH REG
1327  005770  012637  000004        6281:   MOV     (SP)+,@4                ;RESTORE VECTORS
1328  005774  012637  000006                MOV     (SP)+,@6
1329  006000  022737  000176  001000        CMP     #SWREG,@#SWR
1330  006006  001002                       BNE     648
1331  006010  004767  174014                JSR     PC,CNTLU
1332  006014  105037  001124        6481:   CLRB    @#PRGFLG                ;CLEAR PROGRAM FLAG
1333  006020  105037  001121                CLRB    @#ITCNT                 ;CLEAR ITERATION COUNT
1334  006024  105037  001122                CLRB    @#TSTNUM                ;SET TEST # 0
1335  006030  105037  001123                CLRB    @#ERFLG                 ;CLEAR ERROR FLAG
1336  006034  105067  173070                CLRB    ASFLG                   ;CLEAR ASK FLAG
1337  006040  012737  000006  000004        MOV     #ERRVEC+2,@#ERRVEC
1338  006046  012737  000002  000006        MOV     #RTI,@#ERRVEC+2         ;CHECK IF 'LP' IS AVAILABLE
1339  006054  005037  001264          28:   CLR     @#INBUF
1340  006060  000004  001374                TYPE,CRLF
1341  006064  000004  013330                TYPE,M,NAM                      ;TYPE TITLE
1342  006070  000004  013376                TYPE,I,REG                      ;ASK USER TO TYPE CONT BASE ADRS
1343  006074  004767  175374                JSR     PC,,INPUT               ;GET USER INPUT
1344  006100  004767  175154          48:   JSR     PC,CNVTAO               ;CONVERT ASCII TO OCTAL
1345  006104  013737  001116  001010        MOV     @#OCTALO,@#TMBASE       ;SET NEW ADDRESS
1346  006112  013705  001010          58:   MOV     @#TMBASE,R5
1347
1348                                ;ROUTINE TO CHECK IF CONTROLLER (RH11) IS AVAILAABLE
1349  006116  000261                       SEC                             ;SET 'C' IN PSW
1350  006120  005715                       TST     (R5)                    ;BRANCH IF CONTROLLER AVAIL
1351  006122  103003                       BCC     68
1352  006124  000004  013715               TYPE,E,NCON
1353  006130  000675                       BR      INIT
1354  006132  012737  003650  000004  68:   MOV     #ERRTRP,@#ERRVEC        ;SET ERROR TRAP VECTOR
```

```
                                        ;ROUTINE TO GET TM02 DRIVES USER DESIRES TO TEST
006140  105037  001123  DRIVES: CLRB    @#ERFLG                 ;CLEAR ERROR FLAG
006144  012701  001154         MOV     #DRVTBL,R1              ;MARK ALL DRIVES AS NOT TO
006150  012700  000004         MOV     #4,R0                   ;BE TESTED, A '0' INDICATES
006154  005021          1$:    CLR     (R1)+                   ;THAT A DRIVE IS NOT TO BE
006156  005300                 DEC     R0                      ;TESTED
006160  001375                 BNE     1$
006162  000004  013443         TYPE,I,DRV$
006166  004767  175302         JSR     PC,,INPUT               ;GET USER INPUT
006172  012700  001264         MOV     #INBUF,R0
006176  122710  000101         CMPB    #'A,(R0)                ;AN 'A' SPECIFIES ALL
006202  001013                 BNE     3$                      ;DRIVES TO BE TESTED
006204  110667  172714         MOVB    $P,PRGFLG               ;SET FLAG TO IND ALL DRIVES
006210  012701  001154         MOV     #DRVTBL,R1              ;MARK ALL DRIVES TO BE TESTED
006214  012700  000004         MOV     #4,R0                   ;A '-1' INDICATES THAT A DRIVE
006220  012721  177777   2$:   MOV     #-1,(R1)+               ;IS TO BE TESTED
006224  005300                 DEC     R0
006226  001374                 BNE     2$
006230  000417                 BR      CHKDRV                  ;GO CHECK DRIVE AVAILABILITY

                                        ;GET USER SELECTED DRIVES AND MARK EACH DRIVE SELECTED TO BE TESTED
006232  122710  000015   3$:   CMPB    #CR,(R0)
006236  001414                 BEQ     CHKDRV
006240  121027  000054         CMPB    (R0),#',                ;CHECK IF 'COMMA'
006244  001001                 BNE     4$
006246  105720                 TSTB    (R0)+                   ;STEP PTR PAST 'COMMA'
006250  112001          4$:    MOVB    (R0)+,R1
006252  042701  177770         BIC     #177770,R1
006256  112761  177777  001154 MOVB    #-1,DRVTBL(R1)
006264  000240                 NOP
006266  000761                 BR      3$

                                        ;ASCERTAIN THAT DRIVES (TM02'S) SPECIFIED ARE AVAILABLE
006270  005000   CHKDRV: CLR    R0                      ;A 0/-1 INDICATES THAT THE
006272  105760  001154   1$:   TSTB    DRVTBL(R0)              ;DRIVE IS NOT/IS TO BE TESTED
006276  001005                 BNE     3$
006300  005200          2$:    INC     R0
006302  122700  000010         CMPB    #8.,R0
006306  001371                 BNE     1$
006310  000421                 BR      4$
006312  110037  001004   3$:   MOVB    R0,@#DRVNUM
006316  004737  005116         JSR     PC,@#DRVAVA             ;CHECK IF AVAILABLE
006322  102366                 BVC     2$                      ;'V' BIT SET INDICATES NOT AVAIL
006324  000004  013762         TYPE,E,NDRV
006330  116037  001132  014014 MOVB    DIGTAB(R0),@#E,NAVA     ;SET DRIVE # IN MESSAGE
006336  000004  014014         TYPE,E,NAVA
006342  110637  001123         MOVB    $P,@#ERFLG              ;SET 'ERROR' FLAG
006346  105060  001154         CLRB    DRVTBL(R0)              ;MARK DRIVE UNAVAILABLE
006352  000752                 BR      2$                      ;CHECK NEXT DRIVE
006354  105737  001123   4$:   TSTB    @#ERFLG                 ;GO GET SLAVES IF NO ERROR
006360  001403                 BEQ     SLAVES
006362  105737  001124         TSTB    @#PRGFLG                ;ASK USER TO RETYPE DRIVES IF
006366  001664                 BEQ     DRIVES                  ;'ALL' NOT SPECIFIED

                                        ;ROUTINE TO GET SLAVES (TU16'S) USER DESIRES TO TEST
006370  105037  001123 SLAVES: CLRB    @#ERFLG                 ;CLEAR ERROR FLAG
```

```
1411  006374  012701  001164          MOV     #SLVTBL,R1          ;MARK ALL SLAVES (64.) AS NOT
1412  006400  012700  000040          MOV     #32.,R0             ;TO BE TESTED,A 0 INDICATES THAT
1413  006404  005021              1$: CLR     (R1)+               ;A DRIVE'S SLAVE IS NOT TO BE
1414  006406  005300                  DEC     R0                  ;TESTED
1415  006410  001375                  BNE     1$
1416  006412  005000                  CLR     R0                  ;R0 = DRIVE 0 FOR SLAVES
1417  006414  012701  001164          MOV     #SLVTBL,R1          ;R1 POINTS TO DRIVE'S SLAVE
1418  006420  105760  001154      2$: TSTB    DRVTBL(R0)          ;IF DRIVE IS TO BE TESTED
1419  006424  001007                  BNE     4$                  ;GO TO 4$ OTHERWISE
1420  006426  062701  000010      3$: ADD     #8.,R1              ;STEP SLAVE PTR TO NEXT DRIVE'S
1421  006432  005200                  INC     R0                  ;SLAVES AND INCREMENT DRIVE #
1422  006434  122700  000010          CMPB    #8.,R0              ;CHECK ALL DRIVES
1423  006440  001367                  BNE     2$                  ;AND WHEN ALL DRIVES CHECKED
1424  006442  000454                  BR      CHKSLV              ;GO CHECK SLAVE AVAILABILITY
1425
1426  006444  105737  001124      4$: TSTB    @#PRGFLG            ;BRANCH IF USER SELECTED ALL
1427  006450  001020                  BNE     5$                  ;DRIVES
1428  006452  110067  172326          MOVB    R0,DRVNUM           ;GET DRIVE #
1429  006456  116037  001132  013524  MOVB    DIGTAB(R0),@#I,DRV  ;PREPARE USER ACTION MESSAGE
1430  006464  000004  013505          TYPE,I,SLVS
1431  006470  004767  175000          JSR     PC,.INPUT           ;GET USER INPUT
1432  006474  012703  001264          MOV     #INBUF,R3           ;SET PTR TO USER INPUT
1433  006500  122710  000101          CMPB    #'A,(R0)            ;BRANCH IF USER DOES NOT WANT
1434  006504  001015                  BNE     7$                  ;'ALL' SLAVES
1435  006506  110637  001124          MOVB    SP,@#PRGFLG         ;SET 'ALL' INDICATOR
1436  006512  012701  001164      5$: MOV     #SLVTBL,R1          ;MARK ALL SLAVES FOR ALL
1437  006516  012700  000040          MOV     #32.,R0             ;DRIVES AS TO BE TESTED
1438  006522  012721  177777      6$: MOV     #-1,(R1)+
1439  006526  005300                  DEC     R0
1440  006530  001374                  BNE     6$
1441  006532  105737  001124          TSTB    @#PRGFLG            ;BRANCH IF ALL WAS SELECTED
1442  006536  001016                  BNE     CHKSLV
1443
1444  006540  122713  000015      7$: CMPB    #CR,(R3)            ;GET USER SELECTED SLAVES FOR
1445  006544  001730                  BEQ     3$                  ;DRIVE
1446  006546  121327  000054          CMPB    (R3),#',            ;STEP PTR PAST 'COMMA'
1447  006552  001001                  BNE     8$
1448  006554  105723                  TSTB    (R3)+
1449  006556  112304          8$: MOVB    (R3)+,R4            ;AND MARK SELECED SLAVE
1450  006560  042704  177770          BIC     #177770,R4          ;AS TO BE TESTED
1451  006564  060104                  ADD     R1,R4
1452  006566  112714  177777          MOVB    #-1,(R4)
1453  006572  000762                  BR      7$
1454
1455
1456  006574  005000          ;ASCERTAIN THAT SLAVES (TU16'S) SELECTED ARE AVAILABLE
1457  006576  005001  CHKSLV: CLR     R0                  ;R0 WILL CONTAIN THE DRIVE #
1458  006600  012702  001164      CLR     R1                  ;AND R1 THE SLAVE #
1459  006604  105760  001154      MOV     #SLVTBL,R2          ;SET PTR TO SLAVE TABLE
1460  006610  001007              1$: TSTB    DRVTBL(R0)          ;BRANCH IF DRIVE SELECTED
1461  006612  005200                  BNE     3$                  ;& AVAILABLE FOR TEST
1462  006614  062702  000010      2$: INC     R0                  ;INCREMENT DRIVE #
1463  006620  022700  000010          ADD     #8.,R2              ;STEP SLAVE PTR TO NEXT DRIVE'S
1464  006624  001367                  CMP     #8.,R0              ;SLAVES. BRANCH TO 1$ IF NOT ALL
1465  006626  000434                  BNE     1$                  ;DRIVES CHECKED OTHERWISE EXIT
1466                                  BR      7$
```

```
1467  006630  005001                  38:   CLR     R1                      ;SET SLAVE # 0
1468  006632  105712                  48:   TSTB    (R2)                    ;BRANCH IF DRIVE'S SLAVE IS SEL-
1469  006634  001006                        BNE     68                      ;ECTED FOR TEST
1470  006636  005201                  58:   INC     R1                      ;INCREMENT SLAVE #
1471  006640  005202                        INC     R2                      ;STEP PTR TO NEXT SLAVE
1472  006642  022701  000010                CMP     #10.,R1                 ;GO TO 48 IF ALL SLAVES NOT
1473  006646  001371                        BNE     48                      ;CHECKED
1474  006650  000760                        BR      28                      ;OTHERWISE GO TO 28 ABOVE
1475
1476  006652  110037  001004            68:   MOVB    R0,@#DRVNUM             ;PASS DRIVE & SLAVE #
1477  006656  110137  001005                  MOVB    R1,@#SLVNUM
1478  006662  004737  005144                  JSR     PC,@#SLVAVA             ;AND CHECK IF AVAILABLE
1479  006666  102363                        BVC     58                      ;'V' BIT SET ON RETURN IND-
1480  006670  116037  001132  014004        MOVB    DIGTAB(R0),@#E.DRV      ;ICATES ERROR, PREPARE ERROR
1481  006676  116137  001132  014014        MOVB    DIGTAB(R1),@#E.NAVA     ;MESSAGE
1482  006704  000004  013776                  TYPE,E.NSLV
1483  006710  110637  001123                  MOVB    SP,@#ERFLG              ;SET ERROR INDICATOR
1484  006714  105012                        CLRB    (R2)                    ;CLEAR SLAVE TABLE ENTRY
1485  006716  000747                        BR      58                      ;GET NEXT SLAVE
1486
1487  006720  105737  001123            78:   TSTB    @#ERFLG                 ;BRANCH IF NO ERROR
1488  006724  001403                        BEQ     1008
1489  006726  105737  001124                  TSTB    @#PRGFLG                ;BRANCH IF NOT 'ALL'
1490  006732  001616                        BEQ     SLAVES                  ;ASK USER TO RETYPE SLAVES
1491  006734  012737  003650  000004  1008:   MOV     #ERRTRP,@#ERRVEC
1492
1493                                    ;SCAN DIVE AND SLAVE TABLE FOR DRIVE/SLAVE COMBINATION TO TEST
1494  006742  105037  001004                  CLRB    @#DRVNUM                ;SET DRIVE AND SLAVE # #
1495  006746  105037  001005                  CLRB    @#SLVNUM
1496  006752  012737  001164  001006          MOV     #SLVTBL,@#SLVPTR        ;SET PTR TO SLAVE TABLE
1497  006760  105037  001125                  CLRB    @#UNITFND               ;CLEAR 'UNIT FOUND' IND.
1498
1499  006764  113700  001004            BEGIN:MOVB    @#DRVNUM,R0             ;GET DRIVE #
1500  006770  113701  001005                  MOVB    @#SLVNUM,R1             ;AND SLAVE #
```

```
 1501  006774  013702  001006          MOV     @#SLVPTR,R2             ;GET SLAVE PTR
 1502  007000  105760  001154  10$:    TSTB    DRVTBL(R0)              ;BRANCH IF DRIVE AVAIL TO TEST
 1503  007004  001011                  BNE     3$
 1504  007006  005001                  CLR     R1                      ;CLEAR SLAVE #
 1505  007010  062702  000010          ADD     #8.,R2                  ;AND STEP PTR TO NEXT DRIVE'S
 1506  007014  005200                  INC     R0                      ;SLAVES AND INCREMENT DRIVE #
 1507  007016  022700  000010  2$:     CMP     #8.,R0                  ;EXIT TEST IF ALL DRIVES
 1508  007022  001366                  BNE     1$                      ;CHECKED OTHERWISE CONTINUE
 1509  007024  000137  012732          JMP     @#END                   ;SCAN FOR NEXT 'UNIT'
 1510
 1511  007030  105712                  TSTB    (R2)            3$:     ;BRANCH IF SLAVE ON DRIVE IS
 1512  007032  001007                  BNE     4$                      ;AVAILABLE THERWISE STEP
 1513  007034  005202                  INC     R2                      ;PTR TO NEXT SLAVE
 1514  007036  005201                  INC     R1                      ;INCREMENT SLAVE #
 1515  007040  122701  000010          CMPB    #8.,R1                  ;UNTIL ALL SLAVES CHECKED
 1516  007044  001371                  BNE     3$                      ;WHEN ALL SLAVES CHECKED
 1517  007046  005001                  CLR     R1                      ;SET SLAVE # 0
 1518  007050  000761                  BR      2$                      ;AND CONTINUE SCAN
 1519
 1520  007052  110637  001125  4$:     MOVB    SP,@#UNTFND             ;INDICATE THAT A 'UNIT' IS FOUND
 1521  007056  110037  001004          MOVB    R0,@#DRVNUM             ;SET DRIVE #
```

```
 1522  007062  116137  001085                MOVB    R1,@#SLVNUM             ;SET SLAVE #
 1523  007066  016237  001086                MOV     R2,@#SLVPTR             ;SAVE SLAVE PTR
 1524
 1525  007072  105737  001130          58:   TSTB    @#ASFLG
 1526  007076  001044                       BNE     7$
 1527  007100  112767  000001  172022        MOVB    #1,ASFLG
 1528
 1529  007106  105037  001124                CLRB    @#PRGFLG        ;CLEAR PROGRAM INDICATOR
 1530  007112  000004  013564                TYPE,I,SKEW             ;ASK USER IF HE WANTS TO RUN SKEW TESTS
 1531  007116  004767  174352                JSR     PC,,INPUT               ;GET USER INPUT
 1532  007122  012703  001264                MOV     #INBUF,R3       ;GET REPLY
 1533  007126  122713  000060                CMPB    #'0,(R3)        ;BRANCH IF 'NO' (0)
 1534  007132  001406                       BEQ     6$
 1535  007134  122713  000061                CMPB    #'1,(R3)        ;CHECK IF 'YES' (1)
 1536  007140  001354                       BNE     5$              ;NEITHER SO ASK AGAIN
 1537  007142  111337  001124                MOVB    (R3),@#PRGFLG   ;SET INDICATOR
 1538  007146  000420                       BR      7$
 1539
 1540  007150  105037  001127          6$:   CLRB    @#NRZFLG        ;CLEAR NRZ INDICATOR
 1541  007154  000004  013625                TYPE,I,NRZ              ;ASK USER IF DRIVE 'NRZ' ONLY
 1542  007160  004767  174310                JSR     PC,,INPUT               ;GET USER INPUT
 1543  007164  012703  001264                MOV     #INBUF,R3       ;GET REPLY
 1544  007170  122713  000060                CMPB    #'0,(R3)        ;BRANCH IF 'NO' (0)
 1545  007174  001405                       BEQ     7$
 1546  007176  122713  000061                CMPB    #'1,(R3)        ;CHECK IF 'YES' (1)
 1547  007202  001362                       BNE     6$              ;ASK AGAIN IF NEITHER
 1548  007204  111337  001127                MOVB    (R3),@#NRZFLG   ;SET INDICATOR
 1549  007210                           7$:
 1550
 1551  007210  052737  000100  177560  TYPHDR: BIS     #100,@#TKS              ;SET KEYBOARD IE BIT
 1552  007216  000004  014344                TYPE,L,HDR1
 1553  007222  116037  001132  014524        MOVB    DIGTAB(R0),@#L,DRV      ;SET DRIVE #
 1554  007230  116137  001132  014536        MOVB    DIGTAB(R1),@#L,SLV      ;AND SLAVE #
 1555  007236  112737  000071  014541        MOVB    #'9,@#L,CHAN            ;GET SLAVES CHANNEL TYPE
 1556  007244  032765  010000  000026        BIT     #CH7,DT(R5)
 1557  007252  001403                       BEQ     1$
 1558  007254  112737  000067  014541        MOVB    #'7,@#L,CHAN            ;SET 7 CHANNEL
 1559  007262  000004  014457          1$:   TYPE,L,HDR2
 1560  007266  004767  176302                JSR     PC,$NPT         ;GO PRINT SERIAL NUMBER
 1561  007272  000004  014560                TYPE,L,HDR3
 1562  007276  012737  007336  001002        MOV     #TST001,@#SCPADR        ;SET 'SCOPE' ADDRESS FOR FIRST TEST
 1563  007304  010500                       MOV     R5,R0
 1564  007306  062700  000006                ADD     #FC,R0                  ;R0 CONTAINS ADDRESS OF FC REG
 1565  007312  010501                       MOV     R5,R1
 1566  007314  062701  000012                ADD     #DS,R1                  ;R1 CONTAINS ADDRESS OF DS REG
 1567  007320  012703  004444                MOV     #TIMER,R3               ;SET JUMP ADDRESS TO TIMER
 1568  007324  105737  001124                TSTB    @#PRGFLG                ;BRANCH IF NOT SKEW TESTS
 1569  007330  001402                       BEQ     TST001
 1570  007332  000137  012764                JMP     @#SKEWTST
```

```
 1571                                          .SBTTL  START OF TESTS
 1572                                          ;TEST 001 - WRITE FROM BOT
 1573                                          ;THIS TEST WILL MEASURE ACCELERATION DELAY REQUIRED TO
 1574                                          ;MOVE THE TAPE APPROXIMATELY SEVEN (7) INCHES FORWARD
 1575                                          ;FROM DEAD STOP BEFORE STARTING TO TRANSFER DATA.
 1576
 1577                                          ;THIS TEST MEASURES TIME FROM 'GO'=1 TO 'ACCL'=0.
 1578  007336  112737  000001  001122  TST001: MOVB    #1,@#TSTNUM             ;SET TEST #
 1579  007344  012702  007370                 MOV     #18,R2                  ;SET RETURN PC FROM TIMER
 1580  007350  004767  175762                 JSR     PC,,REWIND              ;REWIND SLAVE
 1581  007354  102420                         BVS     998                     ;BRANCH IF ERROR ON REWIND
 1582  007356  004767  176036                 JSR     PC,WRITE                ;GO SETUP WRITE COMMAND
 1583  007362  004767  174766                 JSR     PC,TIMON                ;TURN TIMER ON
 1584  007366  005215                         INC     (R5)                    ;SET 'GO' BIT
 1585
 1586  007370  005765  000032          18:    TST     TC(R5)                  ;BRANCH WHEN 'ACCL'=0
 1587  007374  100002                         BPL     28
 1588  007376  000163  004444                 JMP     TIMER(R3)               ;GO TO TIMER & RETURN VIA R2
 1589
 1590  007402  004767  175624          28:    JSR     PC,WAITRDY              ;WAIT FOR COMMAND TO FINISH
 1591  007406  102403                         BVS     998                     ;BRANCH IF ERROR
 1592  007410  004767  175066                 JSR     PC,TIMOK                ;GO CHECK TIME
 1593  007414  000401                         BR      1008
 1594  007416  104400                 998:    HLT
 1595  007420  104000                 1008:   $COPE
 1596
 1597                                          ;TEST 002 - WRITE START
 1598                                          ;THIS TST MEASURES TIME FROM 'GO'=1 TO 'ACCL'=0.
 1599  007422  112737  000002  001122  TST002: MOVB    #2,@#TSTNUM             ;SET TEST # 2
 1600  007430  004767  175764                 JSR     PC,WRITE                ;INITIATE WRITE COMMAND
 1601  007434  012702  007446                 MOV     #18,R2                  ;SET RETURN PC FROM TIMER
 1602  007440  004767  174710                 JSR     PC,TIMON
 1603  007444  005215                         INC     (R5)                    ;SET 'GO' BIT
 1604
 1605  007446  005765  000032          18:    TST     TC(R5)                  ;BRANCH WHEN 'ACCL'=0
 1606  007452  100002                         BPL     28
 1607  007454  000163  004444                 JMP     TIMER(R3)               ;GO TO TIMER & RETURN VIA R2
 1608
 1609  007460  004767  175546          28:    JSR     PC,WAITRDY              ;WAIT FOR READY
 1610  007464  102403                         BVS     998                     ;BRANCH IF ERROR
 1611  007466  004767  175010                 JSR     PC,TIMOK                ;GO CHECK TIME RECORDED
 1612  007472  000401                         BR      1008                    ;EXIT VIA $COPE
 1613
 1614  007474  104400                 998:    HLT                             ;REPORT ERROR
 1615  007476  104000                 1008:   $COPE
 1616
 1617                                          ;TEST 003- WRITE SHUTDOWN
 1618                                          ;THIS TEST MEASURES TIME FROM 'FC REG'=0 TO 'SHDN'=1.
 1619  007500  112737  000003  001122  TST003: MOVB    #3,@#TSTNUM             ;SET TEST#3
 1620  007506  004767  175706                 JSR     PC,WRITE                ;INITIATE WRITE COMMAND
 1621  007512  005215                         INC     (R5)                    ;SET 'GO' BIT
 1622
 1623  007514  005710                 18:     TST     (R0)                    ;BRANCH WHEN WRITING FINISHED
 1624  007516  001404                         BEQ     28
 1625  007520  032711  040000                 BIT     #ERR,(R1)               ;MONITOR ERROR BIT
 1626  007524  001017                         BNE     998
```

K5

```
 1627  007526  000772                        BR      1$
 1628
 1629  007530                          2$:
 1630  007530  004767  174620                JSR     PC,TIMON                ;TURN TIMER ON
 1631  007534  010702                        MOV     PC,R2                   ;LOAD RETURN PC FROM TIMER
 1632  007536  032711  000020          3$:   BIT     #SDWN,(R1)              ;BRANCH WHEN DS <SDWN> SETS
 1633  007542  001002                        BNE     4$
 1634  007544  000163  004444                JMP     TIMER(R3)               ;GO TO TIMER & RETURN VIA R2
 1635
 1636  007550  004767  175456          4$:   JSR     PC,WAITRDY              ;WAIT FOR READY
 1637  007554  102403                        BVS     99$
 1638  007556  004767  174720                JSR     PC,TIMOK                ;GO CHECK TIME RECORDED
 1639  007562  000401                        BR      100$
 1640  007564  104400                99$:    HLT                             ;REPORT ERROR
 1641  007566  104000                100$:   SCOPE
 1642
 1643                                  ;TEST 004 - WRITE SETTLEDOWN
 1644                                  ;THIS TEST MEASURES TIME FROM 'SWDN'=1 TO 'SWDN'=0.
 1645  007570  112737  000004  001122  TST004: MOVB   #4,@#TSTNUM
 1646  007576  004767  175616                JSR     PC,WRITE
 1647  007602  005210                        INC     (R0)                    ;SET 'GO' BIT
 1648
 1649  007604  005710                1$:     TST     (R0)                    ;BRANCH WHEN WRITING FINISHED
 1650  007606  001404                        BEQ     2$
 1651  007610  032711  040000                BIT     #ERR,(R1)               ;CHECK ERROR BIT
 1652  007614  001026                        BNE     99$
 1653  007616  000772                        BR      1$
 1654
 1655  007620  032711  000020          2$:   BIT     #SDWN,(R1)              ;WAIT FOR ASSERTION OF 'SDWN'
 1656  007624  001004                        BNE     3$
 1657  007626  032711  040000                BIT     #ERR,(R1)               ;MONITOR ERROR BIT
 1658  007632  001017                        BNE     99$
 1659  007634  000771                        BR      2$
 1660
 1661  007636                          3$:
 1662  007636  004767  174512                JSR     PC,TIMON                ;TURN TIMER ON
 1663  007642  010702                        MOV     PC,R2                   ;SET RETURN PC FROM TIMER
 1664  007644  032711  000020                BIT     #SDWN,(R1)              ;BRANCH WHEN SWDN CLEARS
 1665  007650  001402                        BEQ     5$
 1666  007652  000163  004444                JMP     TIMER(R3)               ;GO TO TIMER & RETURN VIA R2
 1667
 1668  007656  004767  175350          5$:   JSR     PC,WAITRDY              ;WAIT FOR READY
 1669  007662  102403                        BVS     99$
 1670  007664  004767  174612                JSR     PC,TIMOK
 1671  007670  000401                        BR      100$
 1672
 1673  007672  104400                99$:    HLT
 1674  007674  104000                100$:   SCOPE
 1675
 1676                                  ;TEST 005 - READ FROM BOT
 1677                                  ;THIS TEST MEASURES TIME FROM 'GO'=1 TO 'ACCL'=0.
 1678  007676  112737  000005  001122  TST005: MOVB   #5,@#TSTNUM             ;SET TEST #5
 1679  007704  004767  175426                JSR     PC,.REWIND              ;REWIND SLAVE
 1680  007710  102422                        BVS     99$                     ;BRANCH IF ERROR ON REWIND
 1681  007712  004767  175520                JSR     PC,READ
 1682  007716  012702  007730                MOV     #1$,R2                  ;SET RETURN PC FROM TIMER
```

```
 1683  007722  004767  174426            JSR     PC,TIMON                ;TURN TIMER ON
 1684  007726  005215                    INC     (R5)                    ;SET 'GO' BIT
 1685
 1686  007730  005765  000032    18:     TST     TC(R5)                  ;BRANCH WHEN 'ACCL' RESETS
 1687  007734  100002                    BPL     28
 1688  007736  000163  004444            JMP     TIMER(R3)               ;GO TO TIMER & RETURN VIA R2
 1689
 1690  007742  004767  175264    28:     JSR     PC,WAITRDY              ;WAIT FOR READY
 1691  007746  102403                    BVS     998                     ;BRANCH IF ERROR
 1692  007750  004767  174526            JSR     PC,TIMOK                ;CHECK RECORDED TIME
 1693  007754  000401                    BR      1008
 1694
 1695  007756  104400            998:    HLT
 1696  007760  104000            1008:   SCOPE
 1697
 1698                            ;TEST 006 - READ START
 1699                            ;THIS TEST MEASURES TIME FROM 'GO'=1 TO 'ACCL'=0.
 1700  007762  112737  000006  001122  TST006: MOVB    #6,@#TSTNUM             ;SET TEST #6
 1701  007770  004767  175516            JSR     PC,WRT,BK               ;WRITE A RECORD & BACK SPACE
 1702  007774  102422                    BVS     998
 1703  007776  004767  175434            JSR     PC,READ
 1704  010002  012702  010014            MOV     #18,R2                  ;SET RETURN PC FROM TIMER
 1705  010006  004767  174342            JSR     PC,TIMON                ;TURN TIMER ON
 1706  010012  005215                    INC     (R5)                    ;SET 'GO' BIT
 1707
 1708  010014  005765  000032    18:     TST     TC(R5)                  ;BRANCH WHEN 'ACCL' RESETS
 1709  010020  100002                    BPL     28
 1710  010022  000163  004444            JMP     TIMER(R3)               ;GO TO TIMER & RETURN VIA R2
 1711
 1712  010026  004767  175200    28:     JSR     PC,WAITRDY
 1713  010032  102403                    BVS     998
 1714  010034  004767  174442            JSR     PC,TIMOK
 1715  010040  000401                    BR      1008
 1716
 1717  010042  104400            998:    HLT
 1718  010044  104000            1008:   SCOPE
 1719
 1720                            ;TEST 007 - READ SHUTDOWN
 1721                            ;THIS TEST MEASURES TIME FROM 'FC REG'=FRAME COUNT TO 'SWDN'=1.
 1722  010046  112737  000007  001122  TST007: MOVB    #7,@#TSTNUM             ;SET TEST #7
 1723  010054  004767  175432            JSR     PC,WRT,BK               ;WRITE A RECORD & BACK SPACE
 1724  010060  102430                    BVS     998                     ;BRANCH IF ERROR
 1725  010062  004767  175350            JSR     PC,READ
 1726  010066  005215                    INC     (R5)                    ;SET 'GO' BIT
 1727
 1728  010070  022710  000400    18:     CMP     #-FRMCNT,(R0)           ;WAIT FOR FRAME COUNT TO
 1729  010074  001404                    BEQ     28                      ;= # OF FRAMES WRITTEN
 1730  010076  032711  040000            BIT     #ERR,(R1)               ;MONITOR ERROR BIT
 1731  010102  001017                    BNE     998
 1732  010104  000771                    BR      18
 1733
 1734  010106                    28:
 1735  010106  004767  174242            JSR     PC,TIMON                ;TURN TIMER ON
 1736  010112  010702                    MOV     PC,R2                   ;SET RETURN PC FROM TIMER
 1737  010114  032711  000020            BIT     #SDWN,(R1)              ;BRANCH WHEN SDWN SETS
 1738  010120  001[illegible]
```

m5

```
DZTUDC,P11      START OF TESTS

  1739  010122  000163  004444                    JMP     TIMER(R3)               ;GO TO TIMER & RETURN VIA R2
  1740
  1741  010126  004767  175100          38:       JSR     PC,WAITRDY
  1742  010132  102403                            BVS     998
  1743  010134  004767  174342                    JSR     PC,TIMOK
  1744  010140  000401                            BR      1008
  1745
  1746  010142  104400                998:       HLT                             ;REPORT ERROR
  1747  010144  104000                1008:      SCOPE
  1748
  1749                                          ;TEST 010 - READ SETTLEDOWN
  1750                                          ;THIS TEST MEASURES TIME FROM 'SWDN'=1 TO 'SWDN'=0.
  1751  010146  112737  000010  001122  TST010: MOVB    #10,@#TSTNUM            ;SET TEST #10
  1752  010154  012702  010232                    MOV     #48,R2                  ;SET RETURN PC FROM TIMER
  1753  010160  004767  175326                    JSR     PC,WRT,BK               ;WRITE A RECORD & BACK SPACE
  1754  010164  102436                            BVS     998
  1755  010166  004767  175244                    JSR     PC,READ
  1756  010172  005215                            INC     (R5)                    ;SET 'GO' BIT
  1757
  1758  010174  105711                18:        TSTB    (R1)                    ;WAIT FOR READY
  1759  010176  100404                            BMI     28                      ;BRANCH WHEN SET
  1760  010200  032711  040000                    BIT     #ERR,(R1)               ;CHECK ERROR BIT
  1761  010204  001026                            BNE     998
  1762  010206  000772                            BR      18
  1763
  1764  010210  032711  000020          28:       BIT     #SDWN,(R1)              ;WAIT FOR ASSERTION OF 'SDWN'
  1765  010214  001004                            BNE     38
  1766  010216  032711  040000                    BIT     #ERR,(R1)               ;MONITOR ERROR BIT
  1767  010222  001017                            BNE     998
  1768  010224  000771                            BR      28
  1769
  1770  010226                        38:
  1771  010226  004767  174122                    JSR     PC,TIMON                ;TURN TIMER ON
  1772  010232  032765  000020  000012  48:       BIT     #SDWN,DS(R5)            ;WAIT FOR NEGATION OF SDWN
  1773  010240  001402                            BEQ     58
  1774  010242  000163  004444                    JMP     TIMER(R3)               ;GO TO TIMER & RETURN VIA R2
  1775
  1776  010246  004767  174760          58:       JSR     PC,WAITRDY
  1777  010252  102403                            BVS     998
  1778  010254  004767  174222                    JSR     PC,TIMOK
  1779  010260  000401                            BR      1008
  1780
  1781  010262  104400                998:       HLT
  1782  010264  104000                1008:      SCOPE
  1783
  1784
  1785                                          ;TEST 011-READ REVERSE START
  1786                                          ;THIS TEST MEASURES TIME FROM 'GO'=1 TO 'ACCL'=0.
  1787  010266  112737  000011  001122  TST011: MOVB    #11,@#TSTNUM
  1788  010274  012702  010332                    MOV     #10,R2                  ;SET RETURN PC FROM TIMER
  1789  010300  004767  175114                    JSR     PC,WRITE                ;WRITE A RECORD
  1790  010304  005215                            INC     (R5)                    ;SET 'GO' BIT
  1791  010306  004767  174720                    JSR     PC,WAITRDY
  1792  010312  102422                            BVS     998
  1793  010314  004767  174420                    JSR     PC,DELAY                ;WAIT FOR TAPE MOTION TO STOP
  1794  010320  004767  175130                    JSR     PC,REWND
```

```
DZTUDC,P11      START OF TEST8

  1795  010324  004767  174024            JSR     PC,TIMON                ;TURN TIMER ON
  1796  010330  005215                      INC     (R5)                    ;SET 'GO' BIT
  1797
  1798  010332  005765  000032      18:     TST     TC(R5)                  ;BRANCH WHEN 'ACCL' = 0
  1799  010336  100002                      BPL     28
  1800  010340  000163  004444              JMP     TIMER(R3)               ;GO TO TIMER & RETURN VIA R2
  1801
  1802  010344  004767  174662      28:     JSR     PC,WAITRDY
  1803  010350  102403                      BVS     998                     ;BRANCH IF ERROR
  1804  010352  004767  174124              JSR     PC,TIMOK
  1805  010356  000401                      BR      1008
  1806
  1807  010360  104400              998:    HLT
  1808  010362  104000              1008:   $COPE
  1809
  1810                                      ;TEST 012-READ REVERSE SHUTDOWN
  1811                                      ;THIS TEST MEASURES TIME FROM 'FC REG' = FRAME COUNT TO 'SDWN'=1.
  1812  010364  112737  000012  001122  TST012: MOVB    #12,@#TSTNUM
  1813  010372  012702  010442              MOV     #38,R2                  ;SET RETURN PC FROM TIMER
  1814  010376  004767  175016              JSR     PC,WRITE                ;WRITE A RECORD
  1815  010402  005215                      INC     (R5)                    ;SET 'GO' BIT
  1816  010404  004767  174622              JSR     PC,WAITRDY
  1817  010410  102427                      BVS     998
  1818  010412  004767  175036              JSR     PC,REVRD
  1819  010416  005215                      INC     (R5)                    ;SET 'GO' BIT
  1820
  1821  010420  022710  000400      18:     CMP     @-FRMCNT,(R0)           ;BRANCH WHEN FRAME COUNT
  1822  010424  001404                      BEQ     28                      ;= 0 OF RECORD WRITTEN
  1823  010426  032711  040000              BIT     $ERR,(R1)               ;MONITOR ERROR BIT IN 'DS' REG
  1824  010432  001016                      BNE     998
  1825  010434  000771                      BR      18
  1826
  1827  010436                      28:
  1828  010436  004767  173712              JSR     PC,TIMON                ;TURN TIMER ON
  1829  010442  032711  000020      38:     BIT     $SDWN,(R1)              ;BRANCH WHEN SDWN SETS
  1830  010446  001002                      BNE     48
  1831  010450  000163  004444              JMP     TIMER(R3)               ;GO TO TIMER & RETURN VIA R2
  1832
  1833  010454  004767  174552      48:     JSR     PC,WAITRDY              ;WAIT FOR READY
  1834  010460  102403                      BVS     998
  1835  010462  004767  174014              JSR     PC,TIMOK
  1836  010466  000401                      BR      1008
  1837
  1838  010470  104400              998:    HLT
  1839  010472  104000              1008:   $COPE
  1840
  1841                                      ;TEST 013-READ REVERSE SETTLEDOWN
  1842                                      ;THIS TEST MEASURES TIME FROM 'SDWN'=1 TO 'SDWN'=0.
  1843  010474  112737  000013  001122  TST013: MOVB    #13,@#TSTNUM
  1844  010502  012702  010566              MOV     #48,R2                  ;SET RETURN PC FROM TIMER
  1845  010506  004767  174706              JSR     PC,WRITE                ;WRITE A RECORD
  1846  010512  005215                      INC     (R5)                    ;SET 'GO' BIT
  1847  010514  004767  174512              JSR     PC,WAITRDY
  1848  010520  102435                      BVS     998
  1849  010522  004767  174726              JSR     PC,REVRD
  1850  010526  005215                      INC     (R5)
```

DZTUDC,P11 START OF TESTS

```
1851
1852  010530  105711                  10$:    TSTB    (R1)                    ;BRANCH WHEN
1853  010532  100404                          BMI     20$                     ;READY SETS
1854  010534  032711  040000                  BIT     #ERR,(R1)
1855  010540  001025                          BNE     99$
1856  010542  000772                          BR      10$
1857
1858  010544  032711  000020          20$:    BIT     #SDWN,(R1)
1859  010550  001004                          BNE     30$
1860  010552  032711  040000                  BIT     #ERR,(R1)
1861  010556  001016                          BNE     99$
1862  010560  000771                          BR      20$
1863
1864  010562                          30$:
1865  010562  004767  173566                  JSR     PC,TIMON                ;TURN TIMER ON
1866  010566  032711  000020          40$:    BIT     #SDWN,(R1)              ;BRANCH WHEN SWDN = 0
1867  010572  001402                          BEQ     50$
1868  010574  000163  004444                  JMP     TIMER(R3)               ;GO TO TIMER & RETURN VIA R2
1869
1870  010600  004767  174426          50$:    JSR     PC,WAITRDY              ;WAIT FOR READY
1871  010604  102403                          BVS     99$
1872  010606  004767  173670                  JSR     PC,TIMOK
1873  010612  000401                          BR      100$
1874
1875  010614  104400                  99$:    HLT
1876  010616  104000                  100$:   $COPE
1877
1878                                  ;REWIND DRIVE
1879  010620                          A:
1880  010620  004767  174512                  JSR     PC,,REWIND              ;REWIND SLAVE
1881  010624  102401                          BVS     99$                     ;BRANCH IF ERROR ON REWIND
1882  010626  102002                          BVC     100$
1883  010630  104400                  99$:    HLT
1884  010632  000772                          BR      A
1885  010634                          100$:
1886
1887                                  ;TEST 014-TURN AROUND DELAY (FORWARD-REVERSE)
1888                                  ;THIS TEST MEASURES TIME FROM 'GO'=1 (READ REVERSE) TO 'ACCL'=0
1889  010634  112737  000014  001122  TST014: MOVB    #14,@#TSTNUM
1890  010642  012702  010674                  MOV     #20$,R2                 ;SET RETURN PC FROM TIMER
1891  010646  004767  174546                  JSR     PC,WRITE                ;WRITE A RECORD
1892  010652  005215                          INC     (R5)                    ;SET 'GO' BIT
1893  010654  004767  174352                  JSR     PC,WAITRDY
1894  010660  102420                          BVS     99$
1895
1896  010662  004767  174566          10$:    JSR     PC,REVRD                ;READ THE RECORD (REVERSE)
1897  010666  004767  173462                  JSR     PC,TIMON                ;TURN TIMER ON
1898  010672  005215                          INC     (R5)                    ;SET 'GO' BIT
1899
1900  010674  005765  000032          20$:    TST     TC(R5)                  ;WAIT FOR 'ACCL' = 0
1901  010700  100002                          BPL     30$
1902  010702  000163  004444                  JMP     TIMER(R3)               ;GO TO TIMER & RETURN VIA R2
1903
1904  010706  004767  174320          30$:    JSR     PC,WAITRDY
1905  010712  102403                          BVS     99$
1906  010714  004767  173562                  JSR     PC,TIMOK
```

C6

DZTUDC,P11 START OF TESTS

```
1907  010720  000401                        BR      1000
1908
1909  010722  104400                  999:  HLT
1910  010724  104000                  1000: SCOPE
1911
1912                                  ;TEST 015- TURN AROUND DELAY (REVERSE-FORWARD)
1913                                  ;THIS TEST MEASURES TIME FROM 'GO'=1 (READ) TO 'ACCL'=0.
1914  010726  112737  000015  001122  TST015: MOVB    #15,@#TSTNUM
1915  010734  012702  011002          MOV     #20$,R2                 ;SET RETURN PC FROM TIMER
1916  010740  004767  174454          JSR     PC,WRITE                ;WRITE A RECORD
1917  010744  005215                  INC     (R5)                    ;SET 'GO' BIT
1918  010746  004767  174260          JSR     PC,WAITRDY              ;WAIT FOR READY
1919  010752  102426                  BVS     99$
1920  010754  004767  174474          JSR     PC,REVRD                ;READ A RECORD IN THE
1921  010760  005215                  INC     (R5)                    ;SET 'GO' BIT
1922
1923  010762  004767  174244          JSR     PC,WAITRDY
1924  010766  102420                  BVS     99$
1925
1926  010770  004767  174442          1$:     JSR     PC,READ                 ;READ RECORD FORWARD
1927  010774  004767  173354          JSR     PC,TIMON                ;TURN TIMER ON
1928  011000  005215                  INC     (R5)                    ;SET 'GO' BIT
1929
1930  011002  005765  000032          2$:     TST     TC(R5)                  ;WAIT FOR 'ACCL' = 0
1931  011006  100002                  BPL     3$
1932  011010  000163  004444          JMP     TIMER(R3)               ;GO TO TIMER & RETURN VIA R2
1933
1934  011014  004767  174212          3$:     JSR     PC,WAITRDY
1935  011020  102403                  BVS     99$
1936  011022  004767  173454          JSR     PC,TIMOK
1937  011026  000401                  BR      1000
1938
1939  011030  104400                  999:  HLT
1940  011032  104000                  1000: SCOPE
1941
1942                                  ;TEST 016-GAP SIZE (STOP HALF)
1943  011034  112737  000016  001122  TST016: MOVB    #16,@#TSTNUM
1944  011042  012702  011100          MOV     #11$,R2                 ;SET RETURN PC FROM TIMER
1945  011046  004767  174346          JSR     PC,WRITE                ;WRITE A RECORD
1946  011052  005215                  INC     (R5)                    ;SET 'GO' BIT
1947  011054  004767  174152          JSR     PC,WAITRDY
1948  011060  102421                  BVS     99$
1949  011062  004767  173652          JSR     PC,DELAY                ;DELAY 350 MS
1950  011066  004767  174362          JSR     PC,REVRD                ;READ REVERSE RECORD
1951  011072  004767  173256          JSR     PC,TIMON                ;TURN TIMER ON
1952  011076  005215                  INC     (R5)                    ;SET 'GO' BIT
1953
1954  011100  005710                  1$:     TST     (R0)                    ;WAIT FOR FRAME COUNT > 0
1955  011102  001002                  BNE     2$
1956  011104  000163  004444          JMP     TIMER(R3)               ;GO TO TIMER & RETURN VIA R2
1957
1958  011110  004767  174116          2$:     JSR     PC,WAITRDY              ;WAIT FOR READY BIT TO SET
1959  011114  102403                  BVS     99$
1960  011116  004767  173360          JSR     PC,TIMOK                ;CHECK TIME
1961  011122  000401                  BR      1000
1962
```

D6

```
 1963  011124  104400                  9981:   HLT
 1964  011126  104000                  1000:   SCOPE
 1965
 1966                                  ;TEST 017-GAP SIZE (START HALF)
 1967  011130  112737  000017  001122  TST017: MOVB    #17,@#TSTNUM
 1968  011136  012702  011210                  MOV     #10,R2                  ;SET RETURN PC FROM TIMER
 1969  011142  004767  174252                  JSR     PC,WRITE                ;WRITE A RECORD
 1970  011146  005215                          INC     (R5)                    ;SET 'GO' BIT
 1971  011150  004767  174056                  JSR     PC,WAITRDY              ;WAIT FOR READY
 1972  011154  102427                          BVS     998
 1973  011156  004767  174272                  JSR     PC,REVRD                ;READ REVERSE THE RECORD
 1974  011162  005215                          INC     (R5)                    ;SET 'GO' BIT
 1975  011164  004767  174042                  JSR     PC,WAITRDY              ;WAIT FOR READY
 1976  011170  102421                          BVS     998                     ;BRANCH ON ERROR
 1977  011172  004767  173542                  JSR     PC,DELAY                ;WAIT FOR TAPE MOTION TO STOP
 1978  011176  004767  174234                  JSR     PC,READ                 ;READ RECORD
 1979  011202  004767  173146                  JSR     PC,TIMON                ;TURN TIMER ON
 1980  011206  005215                          INC     (R5)                    ;SET 'GO' BIT
 1981
 1982  011210  005710                  10:     TST     (R0)                    ;WAIT FOR FRAME COUNT > 0
 1983  011212  001002                          BNE     20
 1984  011214  000163  004444                  JMP     TIMER(R3)               ;GO TO TIMER & RETURN VIA R2
 1985
 1986  011220  004767  174006          20:     JSR     PC,WAITRDY              ;WAIT FOR READY
 1987  011224  102403                          BVS     998
 1988  011226  004767  173250                  JSR     PC,TIMOK                ;CHECK TIME
 1989  011232  000401                          BR      1000
 1990
 1991  011234  104400                  9981:   HLT
 1992  011236  104000                  1000:   SCOPE
 1993
 1994                                  ;TEST 020- GAP SIZE (INTERRECORD)
 1995                                  ;THIS TEST MEASURES TIME FROM 'GO'=1 TO 'FC REG' >0.
 1996  011240  112737  000020  001122  TST020: MOVB    #20,@#TSTNUM
 1997  011246  012702  011330                  MOV     #10,R2                  ;SET RETURN PC FROM TIMER
 1998  011252  004767  174142                  JSR     PC,WRITE                ;WRITE A RECORD
 1999  011256  005215                          INC     (R5)                    ;SET 'GO' BIT
 2000  011260  004767  173746                  JSR     PC,WAITRDY              ;WAIT FOR READY
 2001  011264  102433                          BVS     998
 2002  011266  004767  174126                  JSR     PC,WRITE                ;WRITE SECOND RECORD
 2003  011272  005215                          INC     (R5)                    ;SET 'GO' BIT
 2004  011274  004767  173732                  JSR     PC,WAITRDY              ;WAIT FOR READY
 2005  011300  102425                          BVS     998
 2006  011302  004767  174146                  JSR     PC,REVRD                ;READ REVERSE SECOND RECORD
 2007  011306  005215                          INC     (R5)                    ;SET 'GO' BIT
 2008  011310  004767  173716                  JSR     PC,WAITRDY              ;WAIT FOR READY
 2009  011314  102417                          BVS     998
 2010  011316  004767  174132                  JSR     PC,REVRD                ;READ REVERSE FIRST RECORD
 2011  011322  004767  173026                  JSR     PC,TIMON                ;TURN TIMER ON
 2012  011326  005215                          INC     (R5)                    ;SET 'GO' BIT
 2013
 2014  011330  005710                  10:     TST     (R0)                    ;WAIT FOR FRAME COUNT > 0
 2015  011332  001002                          BNE     20
 2016  011334  000163  004444                  JMP     TIMER(R3)               ;GO TO TIMER & RETURN VIA R2
 2017
 2018  011340  004767  173666          20:     JSR     PC,WAITRDY              ;WAIT FOR READY
```

E6

DZTUDC,P11 START OF TESTS

```
 2019  011344  102403                        BVS     99$
 2020  011346  004767  173130                JSR     PC,TIMOK
 2021  011352  000401                        BR      100$
 2022
 2023  011354  104400                99$:    HLT
 2024  011356  104000                100$:   $SCOPE
 2025
 2026                                ;TEST 021- GAP CONSISTANCY
 2027                                ;THIS TEST MEASURES TIME FROM 'GO'=1 TO 'FC REG' > 0,
 2028                                ;THE TEST REWINDS THE TAPE,WRITES 17 RECORDS WITH A DELAY FROM 1-16 MS
 2029                                ;BETWEEN EACH WRITE COMMAND, AFTER THE 17. RECORDS ARE WRITTEN THE
 2030                                ;PROGRAM READ REVERSES 16 RECORDS, AT THIS POINT THE TAPE IS STOPPED BE-
 2031                                ;TWEEN THE FIRST AND SECOND RECORD, A READ COMMAND IS EXECUTED TO READ
 2032                                ;THE 16 RECORDS WITH THE TIME BETWEEN GO=1 TO FC > 0 STORED IN 'GAPTBL'
 2033                                ;FOR EACH RECORD READ, AFTER 16 RECORDS HAVE BEEN READ THE TIME IS VER-
 2034                                ;IFIED FOR EACH READ, AFTER ALL RECORD TIMES ARE VERIFIED THEY ARE AVER-
 2035                                ;AGED AND PLACED IN THE 'ATIMTBL' (BY SCOPE), THE ABOVE PROCESS IS RE-
 2036                                ;PEATED FOR EACH ITERATION.
 2037
 2038  011360  112737  000021  001122 TST021: MOVB    #21,@#TSTNUM
 2039  011366  012702  011524                MOV     #$$,R2                  ;SET RETURN PC FROM TIMER
 2040  011372  004767  173740                JSR     PC,,REWIND              ;REWIND SLAVE
 2041  011376  102530                        BVS     99$                     ;BRANCH IF ERROR ON REWIND
 2042  011400  005067  167510                CLR     DELTIM                  ;CLEAR VARIABLE DELAY TIME
 2043  011404  012700  000021                MOV     #17.,,R0                ;SET # OF RECORDS TO WRITE
 2044  011410  004767  174004          10$:  JSR     PC,WRITE                ;WRITE 17. RECORDS
 2045  011414  005215                        INC     (R5)                    ;SET 'GO' BIT
 2046  011416  004767  173610                JSR     PC,WAITRDY              ;WAIT FOR READY
 2047  011422  102516                        BVS     99$
 2048  011424  004767  173340                JSR     PC,DELAYV               ;DELAY BEFORE WRITING NEXT REC,
 2049  011430  062767  000022  167456        ADD     #18.,,DELTIM            ;SET NEXT DELAY TIME
 2050  011436  005300                        DEC     R0                      ;DECREMENT RECORDS WRITTEN COUNT
 2051  011440  001363                        BNE     10$
 2052
 2053  011442  012700  000021                MOV     #17.,,R0                ;SET # OF RECS, TO REVERSE READ
 2054  011446  004767  174002          20$:  JSR     PC,REVRD                ;REVERSE READ 17. RECORDS
 2055  011452  005215                        INC     (R5)                    ;SET 'GO' BIT
 2056  011454  004767  173552                JSR     PC,WAITRDY              ;WAIT FOR READY
 2057  011460  102477                        BVS     99$
 2058  011462  005300                        DEC     R0                      ;DECREMENT RECORD COUNT
 2059  011464  001370                        BNE     20$
 2060
 2061  011466  012700  000020                MOV     #16.,,R0                ;SET # OF RECORDS TO READ
 2062  011472  012701  001054                MOV     #GAPTBL,R1              ;SET PTR TO GAP TABLE FOR TEST
 2063  011476  004767  173734                JSR     PC,READ                 ;READ A RECORD
 2064  011502  005215                        INC     (R5)                    ;SET 'GO' BIT
 2065
 2066  011504  004767  173522          30$:  JSR     PC,WAITRDY              ;WAIT FOR READY
 2067  011510  102463                        BVS     99$
 2068  011512  004767  173720                JSR     PC,READ                 ;READ NEXT RECORD
 2069  011516  004767  172632                JSR     PC,TIMON                ;TURN TIMER ON
 2070  011522  005215                        INC     (R5)                    ;SET 'GO' BIT
 2071
 2072  011524  005765  000006          40$:  TST     FC(R5)                  ;WAIT FOR FRAME COUNT > 0
 2073  011530  001002                        BNE     50$
 2074  011532  000161  004444                [illegible]                     [illegible]
```

```

011536  004767  173470          5$:     JSR     PC,WAITRDY              ;WAIT FOR READY
011542  102446                          BVS     99$
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      1000$

011660  104400                  99$:    HLT
011662  104000                  1000$:  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$,R2                  ;SET RETURN PC FROM TIMER
011704  004767  173426                  JSR     PC,.REWIND              ;REWIND SLAVE
011710  102437                          BVS     99$                     ;BRANCH IF ERROR ON REWIND
011712  004367  173636                  JSR     R3,TMCMD                ;WRITE 800 WORD RECORD
011716  015700                          .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
```

DZTUDC,P11 START OF TESTS

```
 2131  011736  032711  040000          BIT     #ERR,(R1)               ;MONITOR ERROR BIT
 2132  011742  001022                  BNE     99$
 2133  011744  000771                  BR      1$
 2134
 2135  011746                  20$:
 2136  011746  004767  172402          JSR     PC,TIMON                ;TURN TIMER ON
 2137  011752  105711          30$:    TSTB    (R1)                    ;WAIT FOR READY TO SET
 2138  011754  100402                  BMI     40$
 2139  011756  000163  004444          JMP     TIMER(R3)               ;GO TO TIMER & RETURN VIA R2
 2140  011762  012700  000003  40$:    MOV     #3,R0                   ;SET TO DIVIDE BY 8
 2141  011766  006204          50$:    ASR     R4                      ;BY SHIFTING RIGHT 3 PLACES
 2142  011770  005300                  DEC     R0
 2143  011772  001375                  BNE     50$
 2144  011774  004767  173232          JSR     PC,WAITRDY
 2145  012000  102403                  BVS     99$
 2146  012002  004767  172474          JSR     PC,TIMOK                ;CHECK TIME
 2147  012006  000401                  BR      100$
 2148
 2149  012010  104400          99$:    HLT
 2150  012012  104000          100$:   SCOPE
 2151
 2152                          ;TEST 024-DATA TIME (556BPI)
 2153  012014  112737  000024  001122  TST024: MOVB    #24,@#TSTNUM
 2154  012022  012702  012102          MOV     #38,R2                  ;SET RETURN PC FROM TIMER
 2155  012026  004767  173304          JSR     PC,,REWIND              ;REWIND SLAVE
 2156  012032  102442                  BVS     99$                     ;BRANCH IF ERROR ON REWIND
 2157  012034  052765  000700  000032  BIS     #BPI556+NORM11,TC(R5)   ;LOAD TAPE CONTROL REGISTER
 2158  012042  004367  173586          JSR     R3,TMCMD                ;WRITE 2224. WORD RECORD
 2159  012046  015700                  .WORD   WTBUF
 2160  012050  173520                  .WORD   -2224.
 2161  012052  167240                  .WORD   -4448.
 2162  012054  000060                  .WORD   WFWD
 2163  012056  005215                  INC     (R5)                    ;SET 'GO' BIT
 2164
 2165  012060  022710  167240  10$:    CMP     @-4448.,,(R0)           ;BRANCH WHEN WRITING BEGINS
 2166  012064  001004                  BNE     20$
 2167  012066  032711  040000          BIT     #ERR,(R1)               ;MONITOR ERROR BIT
 2168  012072  001022                  BNE     99$
 2169  012074  000771                  BR      1$
 2170
 2171  012076                  20$:
 2172  012076  004767  172252          JSR     PC,TIMON                ;TURN TIMER ON
 2173  012102  105711          30$:    TSTB    (R1)                    ;BRANCH WHEN READY SETS
 2174  012104  100402                  BMI     40$
 2175  012106  000163  004444          JMP     TIMER(R3)               ;GO TO TIMER & RETURN VIA R2
 2176
 2177  012112  012700  000003  40$:    MOV     #3,R0                   ;SET SHIFT COUNT
 2178  012116  006204          50$:    ASR     R4
 2179  012120  005300                  DEC     R0
 2180  012122  001375                  BNE     50$
 2181  012124  004767  173102          JSR     PC,WAITRDY
 2182  012130  102403                  BVS     99$
 2183  012132  004767  172344          JSR     PC,TIMOK                ;CHECK TIME
 2184  012136  000401                  BR      100$
 2185
 2186  012140  104400          99$:    HLT
```

H6

```
 2187  012142  104000                  1000$:   SCOPE
 2188
 2189                                  ;TEST 025-DATA TIME (800BPI)
 2190  012144  112737  000025  001122  TST025$: MOVB    #025,@#TSTNUM
 2191  012152  012702  012232                   MOV     #38,R2                  ;SET RETURN PC FROM TIMER
 2192  012156  004767  173154                   JSR     PC,,REWIND              ;REWIND SLAVE
 2193  012162  102442                           BVS     998                     ;BRANCH IF ERROR ON REWIND
 2194  012164  052765  001300  000032           BIS     #BPI800+NORM11,TC(R5)   ;SET 800 BPI
 2195  012172  004367  173356                   JSR     R3,TMCMD                ;WRITE 3200. WORD RECORD
 2196  012176  015700                           .WORD   WTBUF
 2197  012200  171600                           .WORD   -3200.
 2198  012202  163400                           .WORD   -6400.
 2199  012204  000060                           .WORD   WFWD
 2200  012206  005215                           INC     (R5)                    ;SET 'GO' BIT
 2201
 2202  012210  022710  163400          10$:     CMP     @-6400.,,(R0)           ;WAIT FOR WRITING TO START
 2203  012214  001004                           BNE     20$
 2204  012216  032711  040000                   BIT     #ERR,(R1)               ;MONITOR ERROR BIT
 2205  012222  001022                           BNE     998
 2206  012224  000771                           BR      10$
 2207
 2208  012226                          20$:
 2209  012226  004767  172122                   JSR     PC,TIMON                ;TURN TIMER ON
 2210  012232  105711                  30$:     TSTB    (R1)                    ;BRANCH WHEN READY SETS
 2211  012234  100402                           BMI     40$
 2212  012236  000163  004444                   JMP     TIMER(R3)               ;GO TO TIMER & RETURN VIA R2
 2213
 2214  012242  012700  000003          40$:     MOV     #3,R0                   ;SET SHIFT COUNT
 2215  012246  006204                  50$:     ASR     R4
 2216  012250  005300                           DEC     R0
 2217  012252  001375                           BNE     50$
 2218  012254  004767  172752                   JSR     PC,WAITRDY
 2219  012260  102403                           BVS     998
 2220  012262  004767  172214                   JSR     PC,TIMOK                ;CHECK TIME
 2221  012266  000401                           BR      1000$
 2222
 2223  012270  104400                  998$:    HLT
 2224  012272  104000                  1000$:   SCOPE
 2225
 2226                                  ;TEST 026-DATA TIME (1600BPI)
 2227  012274  112737  000026  001122  TST026: MOVB    #026,@#TSTNUM
 2228  012302  105737  001127                   TSTB    @#NRZFLG                ;BRANCH IF DRIVE 'NRZ ONLY'
 2229  012306  001046                           BNE     TST027
 2230  012310  012702  012370                   MOV     #38,R2                  ;SET RETURN PC FROM TIMER
 2231  012314  004767  173016                   JSR     PC,,REWIND              ;REWIND SLAVE
 2232  012320  102437                           BVS     998                     ;BRANCH IF ERROR ON REWIND
 2233  012322  052765  002300  000032           BIS     #PE1600+NORM11,TC(R5)   ;SET 1600 BPI
 2234  012330  004367  173220                   JSR     R3,TMCMD                ;WRITE 3200. WORD RECORD
 2235  012334  015700                           .WORD   WTBUF
 2236  012336  171600                           .WORD   -3200.
 2237  012340  163400                           .WORD   -6400.
 2238  012342  000060                           .WORD   WFWD
 2239  012344  005215                           INC     (R5)                    ;SET 'GO' BIT
 2240
 2241  012346  022710  163400          10$:     CMP     @-6400.,,(R0)           ;BRANCH WHEN WRITING STARTS
 2242  012352  001004                           BNE     20$
```

```
 2243  012354  032711  040000          BIT     #ERR,(R1)               ;MONITOR ERROR BIT
. 44   012360  001017                  BNE     998
 2245  012362  000771                  BR      18
 2246
 2247  012364                      28:
 2248  012364  004767  171764          JSR     PC,TIMON                ;TURN TIMER ON
 2249  012370  105711              38: TSTB    (R1)                    ;BRANCH WHEN READY SETS
 2250  012372  100402                  BMI     48
 2251  012374  000163  004444          JMP     TIMER(R3)               ;GO TO TIMER & RETURN VIA R2
 2252
 2253  012400  006204              48: ASR     R4                      ;DIVIDE TIME BY 4
 2254  012402  006204                  ASR     R4
 2255  012404  004767  172622          JSR     PC,WAITRDY
 2256  012410  102403                  BVS     998
 2257  012412  004767  172664          JSR     PC,TIMOK                ;CHECK TIME
 2258  012416  000401                  BR      1008
 2259
 2260  012420  104400              998:    HLT
 2261  012422  104000              1008:   $COPE
 2262
 2263                              ;TEST 027-ERASE
 2264                              ;THIS TST MEASURES TIME FROM 'GO'=1 TO 'RDY'=1.
 2265  012424  112737  000027  001122 TST027: MOVB    #27,@#TSTNUM
 2266  012432  012702  012460          MOV     #18,R2                  ;SET RETURN PC FROM TIMER
 2267  012436  004337  005554          JSR     R3,@#TMCMD
 2268  012442  000000                  .WORD   0
 2269  012444  000000                  .WORD   0
 2270  012446  000000                  .WORD   0
 2271  012450  000024                  .WORD   ERASE
 2272  012452  004767  171676          JSR     PC,TIMON                ;TURN TIMER ON
 2273  012456  005215                  INC     (R5)                    ;SET 'GO' BIT
 2274
 2275  012460  105711              18: TSTB    (R1)                    ;BRANCH WHEN READY SETS
 2276  012462  100402                  BMI     28
 2277  012464  000163  004444          JMP     TIMER(R3)               ;GO TO TIMER & RETURN VIA R2
 2278
 2279  012470  004767  172536      28: JSR     PC,WAITRDY
 2280  012474  102403                  BVS     998
 2281  012476  004767  172000          JSR     PC,TIMOK
 2282  012502  000401                  BR      1008
 2283
 2284  012504  104400              998:    HLT
 2285  012506  104000              1008:   $COPE
 2286
 2287                              ;TEST-030 TAPE MARK
 2288                              ;THIS TEST MEASURES TIME FROM 'GO'=1 TO 'RDY'=1.
 2289  012510  112737  000030  001122 TST030: MOVB    #30,@#TSTNUM
 2290  012516  012702  012560          MOV     #18,R2                  ;SET RETURN PC FROM TIMER
 2291  012522  004767  172672          JSR     PC,WRITE                ;WRITE A RECORD
 2292  012526  005215                  INC     (R5)                    ;SET 'GO' BIT
 2293  012530  004767  172476          JSR     PC,WAITRDY
 2294  012534  102423                  BVS     998
 2295  012536  004337  005554          JSR     R3,@#TMCMD
 2296  012542  000000                  .WORD   0
 2297  012544  000000                  .WORD   0
 2298  012546  000000                  .WORD   0
```

J6

DZTUDC,P11 START OF TESTS

```
2299   012550  000026
2300   012552  004767  171576          .WORD   WFMK
2301   012556  005215                  JSR     PC,TIMON                ;TURN TIMER ON
2302                                   INC     (R5)                    ;SET 'GO' BIT
2303   012560  105711          10$:    TSTB    (R1)                    ;BRANCH WHEN READY SETS
2304   012562  100402                  BMI     20$
2305   012564  000163  004444          JMP     TIMER(R3)               ;GO TO TIMER & RETURN VIA R2
2306
2307   012570  004767  172436  20$:    JSR     PC,WAITRDY
2308   012574  102403                  BVS     99$
2309   012576  004767  171700          JSR     PC,TIMOK
2310   012602  000401                  BR      100$
2311
2312   012604  104400          99$:    HLT
2313   012606                  100$:
2314   012606  004767  172524          JSR     PC,,REWIND              ;REWIND SLAVE
2315   012612  102774                  BVS     99$                     ;BRANCH IF ERROR ON REWIND
2316   012614  104000                  $COPE
2317
```

```
2318  012616  012700  000012          FINISH: MOV     #10.,R0                 ;SET LINE FEED COUNT
2319  012622  000004  001374          10:     TYPE,CRLF
2320  012626  005300                          DEC     R0
2321  012630  001374                          BNE     10
2322  012632  032777  000100  166140          BIT     #SW06,@#SWR
2323  012640  001410                          BEQ     20
2324  012642  113700  001004                  MOVB    @#DRVNUM,R0
2325  012646  113701  001005                  MOVB    @#SLVNUM,R1
2326  012652  113702  001006                  MOVB    @#SLVPTR,R2
2327  012656  000137  007210                  JMP     @#TYPMDR
2328  012662  105237  001005          20:     INCB    @#SLVNUM                ;SET NEXT SLAVE #
2329  012666  005237  001006                  INC     @#SLVPTR                ;AND ITS POINTER
2330  012672  122737  000010  001005          CMPB    #8.,@#SLVNUM            ;BRANCH IF LAST SLAVE (7)
2331  012700  001402                          BEQ     30
2332  012702  000137  006764                  JMP     @#BEGIN                 ;BEGIN TEST ON NEXT SLAVE
2333  012706  105037  001005          30:     CLRB    @#SLVNUM                ;SET SLAVE #0
2334  012712  105237  001004                  INCB    @#DRVNUM                ;AND INCREMENT DRIVE #
2335  012716  122737  000010  001004          CMPB    #8.,@#DRVNUM            ;AND CHECK IF LAST DRIVE
2336  012724  001402                          BEQ     END
2337  012726  000137  006764                  JMP     @#BEGIN
2338
2339  012732  105737  001125          END:    TSTB    @#UNITFND               ;BRANCH IF A UNIT WAS FOUND
2340  012736  001004                          BNE     16
2341  012740  000004  014047                  TYPE,E,UNIT
2342  012744  000137  005724                  JMP     @#INIT
2343  012750  000000                  16:     HALT
2344  012752  004767  167012                  JSR     PC,CKSWR                ;CHECK FOR CNTL G
2345  012756  000005                          RESET
2346  012760  000137  005724                  JMP     @#INIT                  ;RESTART
```

```
 2347                                          ;SKEW TAPE TIMING TESTS
 2348                                          ;THE FOLLOWING TESTS REQUIRE A SPECIALLY WRITTEN 800 BPI SKEW TAPE
 2349  012764  012737  012772  001002  SKEWTST:MOV     #TST031,@#SCPADR        ;SET SCOPE POINTER
 2350
 2351                                          ;TEST 031- SKEW TAPE SPEED TEST-FORWARD
 2352                                          ;THIS TEST READS 32" OF TAPE (26400.-800. = 25600. FRAMES),THEN
 2353                                          ;DIVIDES TIME BY 32. TO GET TIME TO READ 1" (800. FRAMES) OF TAPE.
 2354  012772  112737  000031  001122  TST031: MOVB    #31,@#TSTNUM
 2355  013000  012702  013056                 MOV     #26,R2                  ;SET RETURN PC FROM TIMER
 2356  013004  004767  172326                 JSR     PC,.REWIND              ;REWIND SLAVE
 2357  013010  102441                         BVS     99$                     ;BRANCH IF ERROR ON REWIND
 2358  013012  052765  001300  000032         BIS     #BPI800+NORM11,TC(R5)   ;SET 800 BPI
 2359  013020  052765  000010  000010         BIS     #BAI,CS2(R5)            ;INHIBIT BUS ADDRESS INCREMENT
 2360  013026  004337  005554                 JSR     R3,@#TMCMD              ;READ 32" OF TAPE-FORWARD
 2361  013032  015700                         .WORD   RDBUF
 2362  013034  177777                         .WORD   -1.
 2363  013036  063440                 10$:    .WORD   26400.                  ;FRAME COUNT
 2364  013040  000070                         .WORD   RDFWD
 2365  013042  005215                         INC     (R5)                    ;SET 'GO' BIT
 2366
 2367  013044  022710  001440          1$:    CMP     #800.,(R0)              ;WAIT FOR FIRST 800 FRAMES
 2368  013050  101375                         BHI     1$                      ;TO BE READ
 2369
 2370  013052  004767  171276                 JSR     PC,TIMON                ;TURN TIMER ON
 2371  013056  023710  013036          2$:    CMP     @#10$,(R5)              ;WAIT FOR READING TO FINISH
 2372  013062  103402                         BLO     3$
 2373  013064  000163  004444                 JMP     TIMER(R3)               ;GO TO TIMER & RETURN VIA R2
 2374
 2375  013070  012700  000005          3$:    MOV     #5,R0                   ;DIVIDE TIME BY 32.
 2376  013074  006204                  4$:    ASR     R4
 2377  013076  005300                         DEC     R0
 2378  013100  001375                         BNE     4$
 2379  013102  004767  172066                 JSR     PC,RHINIT               ;INIT DRIVE
 2380  013106  004767  171378                 JSR     PC,TIMOK                ;CHECK TIME
 2381  013112  000401                         BR      100$
 2382
 2383  013114  104400                  99$:   MLT
 2384  013116  104000                  100$:  SCOPE
 2385
 2386                                          ;TEST 032-SKEW TAPE SPEED TEST-REVERSE
 2387                                          ;THIS TEST READS FORWARD 40" (32000. FRAMES) OF TAPE, THEN READS REVERSE
 2388                                          ;32" (26400.-800. = 25600. FRAMES) OF TAPE, THE TIME IS THEN DIVIDED BY
 2389                                          ;32. TO GET TIME TO READ 1" (800. FRAMES) OF TAPE.
 2390  013120  112737  000032  001122  TST032: MOVB    #32,@#TSTNUM
 2391  013126  012702  013254                 MOV     #38,R2                  ;SET RETURN PC FROM TIMER
 2392  013132  004767  172200                 JSR     PC,.REWIND              ;REWIND SLAVE
 2393  013136  102465                         BVS     99$                     ;BRANCH IF ERROR ON REWIND
 2394  013140  052765  001300  000032         BIS     #BPI800+NORM11,TC(R5)
 2395  013146  052765  000010  000010         BIS     #BAI,CS2(R5)
 2396  013154  004337  005554                 JSR     R3,@#TMCMD              ;READ FORWARD 32000. FRAMES
 2397  013160  015700                         .WORD   RDBUF
 2398  013162  177777                         .WORD   -1.                     ;WORD COUNT
 2399  013164  076400                 10$:    .WORD   32000.                  ;FRAME COUNT
 2400  013166  000070                         .WORD   RDFWD                   ;READ FORWARD
 2401  013170  005215                         INC     (R5)                    ;SET 'GO' BIT
 2402
```

M6

DZTUDC,P11 START OF TESTS

```
2403  013172  023710  013164          18$:    CMP     @#10$,(R0)
2404  013176  101375                          BHI     18
2405
2406  013200  004767  171770                  JSR     PC,RHINIT               ;INIT DRIVE
2407  013204  004767  171530                  JSR     PC,DELAY                ;WAIT FOR TAPE MOTION TO STOP
2408  013210  052765  001300  000032          BIS     #BPI800+NORM11,TC(R5)   ;SET 800 BPI
2409  013216  052765  000010  000010          BIS     #BAI,CS2(R5)            ;INHIBIT BUS ADDRESS INCREMENT
2410  013224  004337  005554                  JSR     R3,@#TMCMD              ;READ REVERSE 32" OF TAPE
2411  013230  015700                          .WORD   RDBUF                   ;READ BUFFER
2412  013232  177777                          .WORD   -1.                     ;WORD COUNT
2413  013234  063440                  11$:    .WORD   26400.                  ;FRAME COUNT
2414  013236  000076                          .WORD   RDREV                   ;READ REVERSE
2415  013240  005215                          INC     (R5)                    ;SET 'GO' BIT
2416
2417  013242  022710  001440          28:     CMP     #800.,,(R0)             ;WAIT FOR FIRST 800 FRAMES
2418  013246  101375                          BMI     28                      ;TO BE READ
2419
2420  013250  004767  171100                  JSR     PC,TIMON                ;TURN TIMER ON
2421  013254  023710  013234          38:     CMP     @#11$,(R0)              ;WAIT FOR ALL FRAMES TO BE READ
2422  013260  103402                          BLO     48
2423  013262  000163  004444                  JMP     TIMER(R3)               ;GO TO TIMER & RETURN VIA R2
2424
2425  013266  012700  000005          48:     MOV     #5,R0                   ;DIVIDE TIME BY 32.
2426  013272  006204                  58:     ASR     R4
2427  013274  005300                          DEC     R0
2428  013276  001375                          BNE     5$
2429  013300  004767  171670                  JSR     PC,RHINIT
2430  013304  004767  171172                  JSR     PC,TIMOK
2431  013310  000401                          BR      100$
2432
2433  013312  104400                  99$:    HLT
2434  013314                          100$:
2435  013314  004767  172016                  JSR     PC,,REWIND              ;REWIND SLAVE
2436  013320  102774                          BVS     99$                     ;BRANCH IF ERROR ON REWIND
2437  013322  104000                          SCOPE
2438
2439  013324  000137  012616                  JMP     @#FINISH
2440
2441
2442
```

```
                                                        .SBTTL    PROGRAM MESSAGES
                                                    ;OPERATOR INSTRUCTIONS
013330  005015  052524  033061  M.NAM:  .ASCIZ  <CR><LF>'TU16 DRIVE FUNCTION TIMER (DZTUD-C)'
013336  042040  044522  042526
013344  043040  047125  052103
013352  047511  020116  044524
013360  042515  020122  042050
013366  052132  042125  041455
013374  000051
013376  005015  054524  042520  I.REG:  .ASCIZ  <CR><LF>'TYPE FIRST ADDRESS OF CONTROLLER  '
013404  043040  051111  052123
013412  040440  042104  042522
013420  051523  047440  020106
013426  047503  052116  047522
013434  046114  051105  020040
013442     000
013443     124  050131  020105  I.DRV8: .ASCIZ  %TYPE TM02 DRIVE #'S TO BE TESTED %
013450  046524  031060  042040
013456  044522  042526  021440
013464  051447  052040  020117
013472  042502  052040  051505
013500  042524  020104     000
013505     106  051117  052040  I.SLV8: .ASCII  'FOR TM02 DRIVE '
013512  030115  020062  051104
013520  053111  020105
013524  026460  052040  050131  I.DRV:  .ASCIZ  %0= TYPE SLAVE #'S TO BE TESTED %
013532  020105  046123  053101
013540  020105  023443  020123
013546  047524  041040  020105
013554  042524  052123  042105
013562  000040
013564  050123  042505  020104  I.SKEW: .ASCIZ  'SPEED TESTS ONLY? (YES/NO = 1/0)'
013572  042524  052123  020123
013600  047117  054514  020077
013606  054450  051505  047057
013614  020117  020075  027461
013622  024460     000
013625     116  055122  047440  I.NRZ:  .ASCIZ  'NRZ ONLY? (YES/NO = 1/0)'
013632  046116  037531  024040
013640  042531  027523  047516
013646  036440  030440  030057
013654  000051
013656  005015  047105  020104  M.EOT:  .ASCIZ  <CR><LF>'END OF TAPE'<CR><LF>
013664  043117  052040  050101
013672  006505  000012
                                                    ;ERROR MESSAGES
013676  005015  051124  050101  E.TRP4: .ASCIZ  <CR><LF>'TRAPPED TO 4'
013704  042520  020104  047524
013712  032040     000
013715     116  020117  047503  E.NCON: .ASCIZ  'NO CONTROLLER AT ADDRESS SPECIFIED'<CR><LF>
013722  052116  047522  046114
013730  051105  040440  020124
013736  042101  051104  051505
013744  020123  050123  041505
013752  043111  042511  006504
```

```
2499  013760  000012
2500  013762  046524  031060  042040  E.NDRV: .ASCIZ  'TM02 DRIVE '
2501  013770  044522  042526  000040
2502  013776  051104  053111  020105  E.NSLV: .ASCII  'DRIVE '
2503  014004  020060  046123  053101  E.DRV:  .ASCII  '0 SLAVE '
2504  014012  020105
2505  014014  020060  047516  020124  E.NAVA: .ASCIZ  '0 NOT AVAILABLE FOR TEST'<CR><LF>
2506  014022  053101  044501  040514
2507  014030  046102  020105  047506
2508  014036  020122  042524  052123
2509  014044  005015      000
2510  014047     116  020117  046524  E.UNIT: .ASCIZ  'NO TM02/TU16 UNIT FOUND TO TEST'<CR><LF>
2511  014054  031060  052057  030525
2512  014062  020066  047125  052111
2513  014070  043040  052517  042116
2514  014076  052040  020117  042524
2515  014104  052123  005015      000
2516  014111     123  043117  020124  E.SFT:  .ASCIZ  'SOFT ERROR (DATA)'<CR><LF>
2517  014116  051105  047522  020122
2518  014124  042050  052101  024501
2519  014132  005015      000
2520  014135     124  051505  020124  E.HDR:  .ASCIZ  'TEST 0 '
2521  014142  020043      000
2522  014145     040  042504  044526  E.HDR1: .ASCII  ' DEVICE ERROR'<CR><LF>
2523  014152  042503  042440  051122
2524  014160  051117  005015
2525  014164  051503  004461  041527          .ASCIZ  'CS1'<HT>'WC'<HT>'BA'<HT>'FC'<HT>'CS2'<HT>'DB'<HT>'ER'<HT>'IC'<CR><LF>
2526  014172  041011  004501  041506
2527  014200  041411  031123  042011
2528  014206  004523  051105  052011
2529  014214  006503  000012
2530  014220  047440  052125  047440  E.HDR2: .ASCIZ  ' OUT OF RANGE ERROR'<CR><LF>
2531  014226  020106  040522  043516
2532  014234  020105  051105  047522
2533  014242  006522  000012
2534  014246  005015  044524  042515  E.TIMOV:        .ASCIZ  <CR><LF>'TIMER OVERFLOWED'<CR><LF>
2535  014254  020122  053117  051105
2536  014262  046106  053517  042105
2537  014270  005015      000
2538  014273     015  052012  046511  E.TIMEX:        .ASCIZ  <CR><LF>'TIME EXPIRED WAITING FOR RDY'<CR><LF>
2539  014300  020105  054105  044520
2540  014306  042522  020104  040527
2541  014314  052111  047111  020107
2542  014322  047506  020122  042122
2543  014330  006531  000012
2544  014334  043440  050101  021440  E.GAP:  .ASCIZ  ' GAP 0 '
2545  014342  000040
2546
2547                                  ;TIME DOCUMENT LINES
2548  014344  025052  025052  025052  L.HDR1: .ASCIZ  '**********************************************************************'
2549  014352  025052  025052  025052
2550  014360  025052  025052  025052
2551  014366  025052  025052  025052
2552  014374  025052  025052  025052
2553  014402  025052  025052  025052
2554  014410  025052  025052  025052
```

C7

```
2555  014416  025052  025052  025052
2556  014424  025052  025052  025052
2557  014432  025052  025052  025052
2558  014440  025052  025052  025052
2559  014446  025052  025052  025052
2560  014454  005015     000
2561  014457     052  052040  030115  L.HDR2: .ASCII  '* TM02 DRIVE FUNCTION TIMES- DRIVE # '
2562  014464  020062  051104  053111
2563  014472  020105  052506  041516
2564  014500  044524  047117  052040
2565  014506  046511  051505  020055
2566  014514  051104  053111  020105
2567  014522  020043
2568  014524  020060  046123  053101  L.DRV:  .ASCII  '# SLAVE # '
2569  014532  020105  020043
2570  014536  020060     040           L.SLV:  .ASCII  '#  '
2571  014541     071  041440  040510  L.CHAN: .ASCII  '9 CHAN, SER # '
2572  014546  027116  051440  051105
2573  014554  021440  000040
2574  014560  006440  025012  005015  L.HDR3: .ASCII  ' '<CR><LF>'*'<CR><LF>
2575  014566  020052  052506  041516          .ASCII  '* FUNCTION'<HT><HT>'TIME(SPECIFICATION)'<HT>'TIME(ACTUAL)'<CR><LF>
2576  014574  044524  047117  004411
2577  014602  044524  042515  051460
2578  014610  042520  044503  044506
2579  014616  040503  044524  047117
2580  014624  004451  044524  042515
2581  014632  040450  052103  040525
2582  014640  024514  005015     000
2583
2584  014645     122  047101  042507  L.RNG:  .ASCII  'RANGE=<'
2585  014652  036075     000
2586  014655     101  052103  040525  L.ACT:  .ASCII  'ACTUAL='
2587  014662  036514     000
2588
2589                                  ;TEST DESCRIPTOR HEADERS
2590  014665     052  053440  044522  A.T001: .ASCII  '* WRITE FROM BOT'<HT>
2591  014672  042524  043040  047522
2592  014700  020115  047502  004524
2593  014706     000
2594  014707     052  053440  044522  A.T002: .ASCII  '* WRITE START'<HT><HT>
2595  014714  042524  051440  040524
2596  014722  052122  004411     000
2597  014727     052  053440  044522  A.T003: .ASCII  '* WRITE SHUTDOWN'<HT>
2598  014734  042524  051440  052510
2599  014742  042124  053517  004516
2600  014750     000
2601  014751     052  053440  044522  A.T004: .ASCII  '* WRITE SETTLEDOWN'<HT>
2602  014756  042524  051440  052105
2603  014764  046124  042105  053517
2604  014772  004516     000
2605  014775     052  051040  040505  A.T005: .ASCII  '* READ FROM BOT'<HT><HT>
2606  015002  020104  051106  046517
2607  015010  041040  052117  004411
2608  015016     000
2609  015017     052  051040  040505  A.T006: .ASCII  '* READ START'<HT><HT>
2610  015024  020104  052123  051101
```

D7

```
 2611  015032  004524  000011
 2612  015036  020052  042522  042101  A.T007: .ASCIZ  '* READ SHUTDOWN'<HT><HT>
 2613  015044  051440  052516  042124
 2614  015052  053517  004516  000011
 2615  015060  020052  042522  042101  A.T010: .ASCIZ  '* READ SETTLEDOWN'<HT>
 2616  015066  051440  052105  046124
 2617  015074  042105  053517  004516
 2618  015102     000
 2619  015103     052  051040  040505  A.T011: .ASCIZ  '* READ REV START'<HT>
 2620  015110  020104  042522  020126
 2621  015116  052123  051101  004524
 2622  015124     000
 2623  015125     052  051040  040505  A.T012: .ASCIZ  '* READ REV SHUTDOWN'<HT>
 2624  015132  020104  042522  020126
 2625  015140  044123  052125  047504
 2626  015146  047127  000011
 2627  015152  020052  042522  042101  A.T013: .ASCIZ  '* READ REV SETTLEDOWN'<HT>
 2628  015160  051040  053105  051440
 2629  015166  052105  046124  042105
 2630  015174  053517  004516     000
 2631  015201     052  052040  051125  A.T014: .ASCIZ  '* TURN AROUND DELAY F-R'<HT>
 2632  015206  020116  051101  052517
 2633  015214  042116  042040  046105
 2634  015222  054501  043040  051055
 2635  015230  000011
 2636  015232  020052  052524  047122  A.T015: .ASCIZ  '* TURN AROUND DELAY R-F'<HT>
 2637  015240  040440  047522  047125
 2638  015246  020104  042504  040514
 2639  015254  020131  026522  004506
 2640  015262     000
 2641  015263     052  043440  050101  A.T016: .ASCIZ  '* GAP SIZE-STOP HALF'<HT>
 2642  015270  051440  055111  026505
 2643  015276  052123  050117  044040
 2644  015304  046101  004506     000
 2645  015311     052  043440  050101  A.T017: .ASCIZ  '* GAP SIZE-START HALF'<HT>
 2646  015316  051440  055111  026505
 2647  015324  052123  051101  020124
 2648  015332  040510  043114  000011
 2649  015340  020052  040507  020120  A.T020: .ASCIZ  '* GAP SIZE-INTERRECORD'<HT>
 2650  015346  044523  042532  044455
 2651  015354  052116  051105  042522
 2652  015362  047503  042122  000011
 2653  015370  020052  040507  020120  A.T021: .ASCIZ  '* GAP CONSISTANCY'<HT>
 2654  015376  047503  051516  051511
 2655  015404  040524  041516  004531
 2656  015412     000
 2657  015413     052  042040  052101  A.T023: .ASCIZ  '* DATA TIME-200BPI'<HT>
 2658  015420  020101  044524  042515
 2659  015426  031055  030060  050102
 2660  015434  004511     000
 2661  015437     052  042040  052101  A.T024: .ASCIZ  '* DATA TIME-556BPI'<HT>
 2662  015444  020101  044524  042515
 2663  015452  032455  033065  050102
 2664  015460  004511     000
 2665  015463     052  042040  052101  A.T025: .ASCIZ  '* DATA TIME-800BPI'<HT>
 2666  015470  020101  044524  042515
```

```
 2667  015476  034055  030060  050102
 2668  015504  004511     000
 2669  015507     052  042040  052101  A.TO26: .ASCIZ  '* DATA TIME=1600DPI'<HT>
 2670  015514  020101  044524  042515
 2671  015522  030455  030066  041060
 2672  015530  044520  000011
 2673  015534  020052  051105  051501  A.TO27: .ASCIZ  '* ERASE GAP TIME'<HT>
 2674  015542  020105  040507  020120
 2675  015550  044524  042515  000011
 2676  015556  020052  051127  052111  A.TO30: .ASCIZ  '* WRITE FILE MARK'<HT>
 2677  015564  020105  044506  042514
 2678  015572  046440  051101  004513
 2679  015600     000
 2680  015601     052  052040  050101  A.TO31: .ASCIZ  '* TAPE SPEED=FWD'<HT>
 2681  015606  020105  050123  042505
 2682  015614  026504  053506  004504
 2683  015622     000
 2684  015623     052  052040  050101  A.TO32: .ASCIZ  '* TAPE SPEED=REV'<HT>
 2685  015630  020105  050123  042505
 2686  015636  026504  042522  004526
 2687  015644     000
 2688
 2689  015645     015  057012  000107  L.CNTG: .ASCIZ  <CR><LF>'?G'
 2690  015652  005015  053523  036522  L.SWR:  .ASCIZ  <CR><LF>'SWR='
 2691  015660     000
 2692  015661     040  047040  053505  L.NEW:  .ASCIZ  '  NEW= '
 2693  015666  020075     000
 2694  015671     015  037412  005015  L.QUES: .ASCIZ  <CR><LF>'?'<CR><LF>
 2695  015676     000
 2696          015700                  .EVEN
 2697          015700                  RDBUF=.
 2698          015700                  WTBUF=.
 2699  015700 000200                  .BLKW   128.
 2700          000001                  .END
```

DZTUDC.P11 SYMBOL TABLE

```
A       010620          CNTRLO= 000017          E.HDR   014135          L.HDR3  014560          READ    005436
ACCL  = 100000          CNTRLU= 000025          E.HDR1  014145          L.NEW   015661          RESVEC= 000010
ANGTAB  001412          CNVDEC  002730          E.HDR2  014220          L.QUES  015671          REVRD   005454
AS    = 000016          CNVOCT  002622          E.NAVA  014014          L.RNG   014645          RHINIT  005174
ASFLG   001130          CNVTAO  003260          E.NCON  013718          L.SLV   014536          RMR   = 000004
ATA   = 100000          CNVTD   002742          E.NDRV  013762          L.SWR   015652          RWD   = 000006
ATIME   001012          CNVTO   002634          E.NSLV  013776          MCPE  = 020000          RWDOFF= 000002
ATIMTB  001014          COUNT   001764          E.SFT   014111          MDPE  = 000400          R0    =%000000
A.T001  014665          CR    = 000015          E.TIME  014273          MNVEC = 000250          R1    =%000001
A.T002  014707          CRLF    001374          E.TIMO  014246          NOL   = 010000          R10   =%000000
A.T003  014727          CSITM = 002000          E.TRP4  013676          NR    = 000024          R11   =%000001
A.T004  014751          CS1   = 000000          E.UNIT  014047          NXF   = 001000          R12   =%000002
A.T005  014775          CS2   = 000010          FC    = 000006          N.EOT   013656          R13   =%000003
A.T006  015017          DASH    001405          FCE   = 001000          N.NAM   013330          R14   =%000004
A.T007  015036          DB    = 000022          FINISH  012616          NAMPTR  001672          R15   =%000005
A.T010  015060          DCONST  003036          FMT   = 000020          NED   = 010000          R2    =%000002
A.T011  015103          DELAY   004740          FPEVEC= 000244          NEF   = 004000          R3    =%000003
A.T012  015125          DELAYV  004770          FRNCNT= 177400          NEM   = 004000          R4    =%000004
A.T013  015152          DELTIM  001114          FWDSPC  005472          NOP   = 000000          R5    =%000005
A.T014  015201          DIGTAB  001132          GAP     001120          NORM11= 000300          SC    = 100000
A.T015  015232          DIVIDE  005026          GAPOK   004612          NREFLG  001127          SCOPE = 104000
A.T016  015263          DLT   = 100000          GAPTBL  001054          NSG   = 000400          SCPADR  001002
A.T017  015311          DPR   = 000400          GO    = 000001          OCTALS  001116          SDWN  = 000020
A.T020  015340          DRIVES  006140          GTIMTB  001572          ODIGIT  001144          SKEWTB  012764
A.T021  015370          DRVAVA  005116          HLT   = 104400          OPI   = 020000          SLA   = 000001
A.T023  015413          DRVNUM  001004          HT    = 000011          OR    = 000200          SLAVES  006370
A.T024  015437          DRVTBL  001154          IDB   = 000010          OSC   = 000100          SLR   = 177774
A.T025  015463          DRY   = 000200          IE    = 000100          OUT     002240          SLVAVA  005144
A.T026  015507          DRYCLR= 000010          ILF   = 000001          OUTBUF= 005724          SLVNUM  001005
A.T027  015534          DS    = 000012          ILR   = 000002          OUTGAP  001146          SLVPTR  001006
A.T030  015556          DT    = 000026          INBUF   001264          OUTSPC  003052          SLVTBL  001164
A.T031  015601          DTE   = 010000          INCVAL= 000100          PARVEC= 000114          SN    = 000030
A.T032  015623          DVA   = 004000          INIT    005724          PAT   = 000020          SNPT    005574
A16   = 000400          DV0   = 000000          IOTVEC= 000020          PC    =%000007          SP    =%000006
A17   = 001000          DV1   = 000001          IR    = 000100          PEFLRC= 000200          SPACE   001410
BA    = 000004          DV2   = 000002          ITCNT   001121          PES   = 000040          SPACE2  001407
BAI   = 000010          DV3   = 000003          I.DRV   013524          PE1600= 002000          SPCFWD= 000030
BEGIN   006764          DV4   = 000004          I.DRVS  013443          PFVEC = 000024          SPCREV= 000032
BELL    001403          DV5   = 000005          I.NRI   013625          PGE   = 002000          SPR   = 002000
BKSLSH  001377          DV6   = 000006          I.REG   013376          PIP   = 020000          SSC   = 000100
BOT   = 000002          DV7   = 000007          I.SKEW  013564          PIRQ  = 177772          STINTB  001416
BPI200= 000000          ECHO    001401          I.SLVS  013505          PIRVEC= 000240          STKPTR= 000460
BPI556= 000400          EMTVEC= 000030          LF    = 000012          PLKCSR= 172540          SUBWR   005730
BPI800= 001000          END     012732          LKS   = 177546          PLKVEC= 000104          SWR     001000
BPIVEC= 000014          EOT   = 002000          LKVEC = 000100          PRGFLG  001124          SWREG   000176
CDH11 = 000320          ER    = 000014          LPB   = 177516          PSEL  = 002000          SW06  = 000100
CHKDRV  006270          ERASE = 000024          LPS   = 177514          PSW   = 177776          SW07  = 000200
CHKSLV  006574          ERFLG   001123          L.ACT   014655          PUBLIS  003346          SW08  = 000400
CH7   = 010000          ERR   = 040000          L.CHAN  014541          RDBUF = 015700          SW09  = 001000
CKSWR   001770          ERRTRP  003650          L.CNTG  015645          RDFWD = 000070          SW10  = 002000
CLR   = 000040          ERRVEC= 000004          L.DRV   014524          RDREV = 000076          SW11  = 004000
CNTLU   002030          E.DRV   014004          L.HDR1  014344          RDSW    001766          SW13  = 020000
CNTRLC= 000003          E.GAP   014334          L.HDR2  014457          RDY   = 000200          SW14  = 040000
```

DZTUDC,P11 SYMBOL TABLE

```
SW15  = 100000          TMK   = 000004          TST015  010726          TYPE2   002400          WRL   = 004000
TAP   = 040000          TPB   = 177566          TST016  011034          TYPE3   002406          WRT,BK  000512
TBITVE= 000014          TPS   = 177564          TST017  011130          TYPE4   002412          WTBUF = 010700
TC    = 000032          TPVEC = 000064          TST020  011240          TYPFLG  001126          $CHARC  002324
TCRLF   002360          TRAPVE= 000034          TST021  011360          TYPHDR  007210          $CNTRL  002325
TEMPST  001762          TRE   = 040000          TST022  011664          TYPOCT  002630          $CRLF   002326
TIB     001760          TRTVEC= 000014          TST023  011672          UBREAK= 177770          $FILL   002315
TIMER   004444          TSTNUM  001122          TST024  012014          UNB   = 040000          $NT   = 000011
TIMERR  004454          TST001  007336          TST025  012144          UNTFND  001125          $NULL   002314
TIMER0  004470          TST002  007422          TST026  012274          UPE   = 020000          $TKFLG  002317
TIMER1  004420          TST003  007500          TST027  012424          WAITRD  005232          $TPB    002322
TIMOK   004502          TST004  007570          TST030  012510          WAITTI  005234          $TPFLG  002316
TIMON   004364          TST005  007676          TST031  012772          WC    = 000002          $TPS    002320
TKB   = 177562          TST006  007762          TST032  013120          WCE   = 040000          .HLT    003652
TKISR   003606          TST007  010046          TTIN    002242          WCHKF = 000050          .INPUT  003474
TKS   = 177560          TST010  010146          TTIN1   002262          WCHKR = 000056          .RESTO  002600
TKVEC = 000060          TST011  010266          TTIN2   002276          WPNK  = 000026          .RENIN  005336
TMBASE  001010          TST012  010364          TYPDEC  002736          WPWD  = 000060          .SAVE   002556
TMCMD   005554          TST013  010474          TYPE  = 000004          WRDCNT= 177600          .SCOPE  004126
TMC81 = 172440          TST014  010634          TYPE1   002352          WRITE   005420          .TYPE   002332
.     = 016300
```

```
ERRORS DETECTED:  0
DEFAULT GLOBALS GENERATED:  0

*DSKM:DZTUDC,DZTUD/SOL=DSKM:DZTUDC,P11
RUN-TIME:  10 16 1 SECONDS
RUN-TIME RATIO: 62/28=2.2
CORE USED:  7K  (13 PAGES)
```