TM02/TE16

DRIVE FUNCTION TIMER
CZTUGC0

AH-9473C-MC

FICHE 1 OF 1

JAN 1978
digital
MADE IN USA

```
                                .REM %
```

IDENTIFICATION

PRODUCT CODE: AC-9471C-MC

PRODUCT NAME: CZTUGC0 TM02/TE16 DR FCNT TMR

DATE CREATED: AUG 77

MAINTAINER: DIAGNOSTIC GROUP

AUTHOR: R. BARNES

REVISED: 11-NOV-77 BY CLEM WALSH

TM02 DRIVE FUNCTION TIMER PAGE 2
TABLE OF CONTENTS

TABLE OF CONTENTS

```
                        TM02 DRIVE FUNCTION TIMER                       PAGE 3
                        ABSTRACT


                                        ABSTRACT

                PROGRAM DZTUG MEASURES THE TIME REQUIRED AND GAP
                SIZES PRODUCED BY THE TM02/TE16 MAGTAPE
                DRIVE/SLAVE.

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

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

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

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

```
                                TM02 DRIVE FUNCTION TIMER                                PAGE 4
                                REQUIREMENTS
```

CHAPTER 1

REQUIREMENTS

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

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

1.1 OPTIONAL EQUIPMENT USED

1. NONE

1.2 STORAGE

PROGRAM LOADS AND RUNS IN THE FIRST 4K OF MEMORY.

1.3 PRELIMINARY PROGRAMS (TO ASSURE HARDWARE OPERATION)

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

CHAPTER 2

LOADING AND STARTING PROCEDURE

THE PROCEDURE IS AS FOLLOWS:

LOAD PROGRAM USING THE ABSOLUTE LOADER
LOAD ADDRESS = 200
SET OPERATING SWITCHES
PRESS START

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

PROGRAM WILL REQUEST DRIVE (TM02) AND SLAVE (TE16) NUMBERS TO BE TESTED. TYPE DRIVE/SLAVE NUMBERS WITH A COMMA (,) BETWEEN EACH DRIVE/SLAVE TO BE TESTED.

REQUESTS FOR TAPE SPEED TESTS AND NRZ ONLY MODE WILL BE MADE.
RESPONSE TO TAPE SPEED ONLY REQUEST WITH A ONE (1) WILL CAUSE THE PROGRAM TO EXECUTE TEST 31 AND 32 ONLY. THIS IS THE ONLY WAY TO TEST TAPE SPEED.
NRZ ONLY MODE WILL CAUSE THE PROGRAM TO SKIP THE 1600 BPI DATA TIME TEST.
TYPE CONTROL U (U) TO DELETE LINE TYPED OR RUBOUT TO DELETE LAST CHARACTER(S).

PROGRAM WILL PUBLISH TIMES REQUIRED AND REPORT ERRORS.

2.1 ACT11 OPERATION

IF THE PROGRAM IS RUN IN QUICK VERIFY MODE, FUNCTION TESTS ARE NOT ITERATED.

TM02 DRIVE FUNCTION TIMER PAGE 6
SWITCH SETTINGS

CHAPTER 3

SWITCH SETTINGS

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

CONTROL:

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

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

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

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

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

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

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

CHAPTER 4

ERRORS

TWO TYPES OF ERRORS ARE DETECTED BY THIS PROGRAM, HARDWARE ERRORS AND INCORECT FUNCTION TIMES.

4.1 ERROR TYPEOUT FORMAT (HARDWARE): DATA RELATED ERRORS (IE: PARITY ERROR) ARE PRINTED AND HAVE NO EFFECT ON TIME.

```
TEST # XXXXXX DEVICE ERROR

CS1    WE     BA     FC     CS2    DS     ER1
AAAAAA BBBBBB CCCCCC DDDDDD EEEEEE FFFFFF GGGGGG
```

WHERE:

XXXXXX = TEST NUMBER
AAAAAA-IIIIII = CONTENTS OF TAPE REGISTER 172440-172454

4.2 ERROR TYPEOUT FORMAT (FUNCTION TIME OUT OF RANGE)

```
TEST # XXXXXX OUT OF RANGE ERROR

RANGE = <AAAAAA-BBBBBB> ACTUAL = CCCCCC
```

CHAPTER 5

SUBROUTINE ABSTRACTS

5.1 .SCOPE

THE SCOPE ROUTINE IS CALLED BY THE SCOPE (EMT) INSTRUCTION AT THE START OF EACH SUBTEST. THE .SCOPE ROUTINE PERFORMS THE FOLLOWING FUNCTIONS:

1. LOADS R5 WITH BASE ADDRESS
2. TYPES TIME LINE <SW08>
3. PROVIDES CONTINUOUS LOOP <SW14>
4. MOVES FUNCTION TIME INTO TABLE
5. OUTPUTS LINE ITEM IF SELECTED
6. PROVIDES HALT ON TEST <SW07>
7. DELAYS 350MS BEFORE STARTING TEST
8. INIT'S DRIVE/SLAVE
9. CLEARS THE ERROR FLAG (ERFLG)

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

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

5.2 PUBLISH

THE PUBLISH ROUTINE IS CALLED FROM THE SCOPE ROUTINE IF SW10 IS EQUAL TO 0 (PUBLISH TIME DOCUMENT). THE ROUTINE WILL PRINT A "SINGLE LINE ITEM" EACH TIME IT IS CALLED.

```
                        TM02 DRIVE FUNCTION TIMER                                PAGE 9
                        SUBROUTINE ABSTRACTS


                        5.3  .HLT

                        THE HLT ROUTINE IS CALLED BY THE HLT (TRAP) INSTRUCTION WHEN AN  ERROR
                        IS  DETECTED.    A HLT (TRAP) INSTRUCTION FORMATS THE ERROR INFORMATION
                        AS SHOWN IN SEC 4.1.  A HLT+1  (TRAP+1) FORMATS THE  ERROR AS  SHOWN IN
                        SEC 4.2.

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

CHAPTER 6

MISCELLANEOUS

6.1 STACK POINTER

THE STACK POINTER IS INITAILLY SET TO 500 AND IS RESET TO 500 BY THE SCOPEA ROUTINE.

6.2 EXECUTION TIME

WHEN SW11=1 (INHIBIT ITERATIONS) THE TIME REQUIRED IS 2 MIN.

WHEN SW11=0 (ITERATE SUBTESTS) THE TIME REQUIRED IS 9 MIN.

CHAPTER 7

PROGRAM DESCRIPTION

7.1 SAMPLE TIME DOCUMENT

```
TYPE FIRST ADDRESS OF CONTROLER 172440
TYPE TM02 DRIVE #'S TO BE TESTED 0
FOR TM02 DRIVE 0- TYPE SLAVE #'S TO BE TESTED 7
TAPE SPEED TESTS ONLY? (YES/NO = 1/0) 0
NRZ ONLY? (YES/NO = 1/0) 0
*********************************************************************
* TM02 DRIVE FUNCTION TIMES- DRIVE # 0 SLAVE # 7 9 CHAN. SER. # 5009
*
* FUNCTION                TIME(SPECIFICATION)     TIME(ACTUAL)
* WRITE FROM BOT          RANGE=<154000-150000>       ACTUAL=152740
* WRITE START             RANGE=<009500-008700>       ACTUAL=009120
* WRITE SHUTDOWN          RANGE=<008900-008500>       ACTUAL=008840
* WRITE SETTLEDOWN        RANGE=<013500-008100>       ACTUAL=010970
* READ FROM BOT           RANGE=<037000-033000>       ACTUAL=035580
* READ START              RANGE=<003200-002600>       ACTUAL=002740
* READ SHUTDOWN           RANGE=<004650-004250>       ACTUAL=004360
* READ SETTLEDOWN         RANGE=<013500-008100>       ACTUAL=010970
* READ REV START          RANGE=<003200-002600>       ACTUAL=002740
* READ REV SHUTDOWN       RANGE=<003700-003300>       ACTUAL=003520
* READ REV SETTLEDOWN     RANGE=<013500-008100>       ACTUAL=010970
* TURN AROUND DELAY F-R   RANGE=<016700-010700>       ACTUAL=013600
* TURN AROUND DELAY R-F   RANGE=<016700-010700>       ACTUAL=013660
* GAP SIZE-STOP HALF      RANGE=<012900-009500>       ACTUAL=012200
* GAP SIZE-START HALF     RANGE=<011800-008500>       ACTUAL=010520
* GAP SIZE-INTERRECORD    RANGE=<014800-013700>       ACTUAL=014500
* GAP CONSISANCY          RANGE=<014000-012400>       ACTUAL=013040
* DATA TIME-200 BPI       RANGE=<024100-023100>       ACTUAL=023460
* DATA TIME-556 BPI       RANGE=<024000-023000>       ACTUAL=023350
* DATA TIME-800BPI        RANGE=<024000-023000>       ACTUAL=023400
* DATA TIME-1600BPI       RANGE=<025100-024100>       ACTUAL=024470
* ERASE GAP TIME          RANGE=<101000-099000>       ACTUAL=099510
* WRITE FILE MARK         RANGE=<105000-103000>       ACTUAL=103990
```

```
                                                                N  1

   532
```

```
533
534                                 TM02 DRIVE FUNCTION TIMER                                   PAGE 12
535
536                                 7.1.1   SAMPLE TIME DOCUMENT FOR TAPE SPEED TESTS
537
538                                 TYPE FIRST ADDRESS OF CONTROLLER 172440
539                                 TYPE TM02 DRIVE #'S TO BE TESTED 0
540                                 FOR TM02 DRIVE 0- TYPE SLAVE #'S TO BE TESTED 7
541                                 SPEED TESTS ONLY? (YES/NO = 1/0) 1
542                                 ************************************************************
543                                 *TM02 DRIVE FUNCTION TIMES- DRIVE # 0 SLAVE # 7 9 CHAN. SER. # 5009
544                                 *
545                                 *FUNCTION               TIME(SPECIFICATION)     TIME(ACTUAL)
546                                 *TAPE SPEED FWD         RANGE=<022700-021700>       ACTUAL=022500
547                                 *TAPE SPEED REV         RANGE=<022700-021700>       ACTUAL=022500
548
```

7.2 TEST SEQUENCE WITH RELATED ADJUSTMENTS AND ASSOCIATED HARDWARE

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

```
                                TM02 DRIVE FUNCTION TIMER                                PAGE 14

                        21.  GAP CONSISTENCY                    *SAME AS IN TEST 16

                        *TEST NUMBER 22 IS RESERVED FOR FUTURE USE

                        23.  DATA TIME 200 BPI                  *NONE

                        24.  DATA TIME 556 BPI                  *  "

                        25.  DATA TIME 800 BPI                  *  "

                        26.  DATA TIME 1600 BPI                 *  "

                        27.  ERASE GAP TIME                     *  "

                        30.  WRITE FILE MARK                    *  "

                        31.  TAPE SPEED-FORWARD                 *FWD SPEED

                        32.  TAPE SPEED-REVERSE                 *REVERSE SPEED

                *****NOTE:  IF TIME PROBLEMS APPEAR IN T1 THRU T3?, RUN TAPE SPEED TESTS FIRST*******
```

```
                                                    E 2
635
636                             TM02 DRIVE FUNCTION TIMER                                   PAGE 15
637
638                             7.3     TEST DESCRIPTIONS:
639
640                                     THE FIRST THIRTEEN (13) TESTS (T1 - T15) ARE CHECKS OF THE
641                                     ROM CIRCUITS IN THE TE16 (M9811), THE ACCL COUNTER IN THE
642                                     TM02 (M8903), AND THE SETTLEDOWN ONE SHOT (M8910).
643
644                                     T1.  WRITE FROM BOT:
645
646                                     THIS TEST WILL MEASURE ACCELLERATION DELAY REQUIRED TO
647                                     MOVE THE TAPE APPROXIMATELY SEVEN (7) INCHES FORWARD
648                                     FROM DEAD STOP AT BOT BEFORE STARTING TO TRANSFER DATA.
649
650                                     1.   ASSURE TAPE IS STOPPED AT BOT.
651                                     2.   ISSUE A WRITE COMMAND
652                                     3.   MONITOR BIT 15 OF TC (ACCL)
653                                     4.   TIME FROM GO TO ACCL RESET IS BOT DELAY
654                                     5.   STOP
655
656                                     T2.  WRITE START:
657
658                                     THIS TEST WILL MEASURE ACCELLERATION DELAY JUST AS IN
659                                     T1.  HOWEVER THE TIME WILL BE LESS WHEN NOT STARTING
660                                     FROM BOT.
661
662                                     1.   LEAVE TAPE AT ITS PRESENT POSITION.  ASSURE THAT IT IS STOPPED
663                                     2.   ISSUE A WRITE COMMAND
664                                     3.   MONITOR BIT 15 OF TC (ACCL)
665                                     4.   TIME FROM GO TO RESET OF ACCL IS START DELAY
666                                     5.   STOP
667
668                                     T3.  WRITE SHUTDOWN:
669
670                                     THIS TEST WILL MEASURE THE TIME FROM EOR (LAST CHARACTER
671                                     WRITTEN ON TAPE) TO THE START OF SETTLEDOWN TIME.  THIS
672                                     ASSURES, IN PART, A PROPER INTERRECORD GAP.
673
674                                     1.   LEAVE TAPE AT ITS PRESENT POSITION.  ASSURE THAT IT IS STOPPED
675                                     2.   ISSUE A WRITE COMMAND.
676                                     3.   MONITOR FRAME COUNTER AND BIT 4 OF DS (SDWN)
677                                     4.   TIME FROM FC=0 TO ASSERTION OF SDWN IS
678                                          THE SHUTDOWN TIME.
679                                     5.   STOP
680
```

```
                                        T4.  WRITE SETTLEDOWN:

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

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



                                        T5.  READ FROM BOT

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

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

```
                        T6.  READ START

                        THIS TEST MEASURES THE SAME DELAY AS IN T2.

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

                        T7.  READ SHUTDOWN:

                        THIS TEST MEASURES THE SAME DELAY AS IN T3.

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

                        T10.  READ SETTLEDOWN:

                        THIS TEST MEASURES THE SAME DELAY AS IN T4.

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

T11. READ REVERSE START:

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

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

T12. READ REVERSE SHUTDOWN

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

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

T13. READ REVERSE SETTLEDOWN:

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

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

T14. TURN AROUND DELAY-FORWARD TO REVERSE

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

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

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

GAP MEASUREMENTS:

THE PREVIOUS THIRTEEN (13) TESTS WERE MEASUREMENTS OF LOGIC DELAYS PERFORMED BY THE TM02 OR TE16 IN ORDER TO ALLOW FOR PROPER ACCELERATION AND DECELLERATION OF TAPE ACCORDING TO THE DESIRED INTEROCORD GAP (.6 INCHES). THIS TEST, HOWEVER, WILL MEASURE THE PHYSICAL SIZE OF THE INTERCORD GAP THAT EXISTS ON TAPE AS A RESULT OF THE START/STOP TIMES OF THE CAPSTAN ITSELF. BECAUSE THE INTERRCORD GAP IS CREATED BY TWO ACTIONS, THE START OF MOTION AND THE STOP OF MOTION IT IS NECESSARY TO MAKE TWO SEPERATE MEASUREMENTS. A THIRD MEASUREMENT, MADE ON THE FLY, OF THE ENTIRE LENGTH OF THE GAP WILL ALSO BE MADE.

T16. GAP SIZE (STOP HALF)

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

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

T17. GAP SIZE (START HALF)

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

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

T20. GAP SIZE (INTERRECORD)

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

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

```
                                                                K  2
   871                                  7.   STOP
```

```
                                TM02 DRIVE FUNCTION TIMER                                PAGE 20


                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
                THE!R INTERRECORD GAPS, ON THE FLY.

                1.   REWIND TAPE TO BOT.
                2.   WRITE ONE (1) RECORD TO GET TAPE OFF BOT
                3.   WRITE SIXTEEN (16) RECORDS WITH A PROGRESSIVE DELAY
                     OF FROM 0 TO 16 MILLISECONDS (APPROX) BETWEEN COMMANDS.
                4.   BACKSPACE 16 RECORDS AND ALLOW THE TAPE TO STOP.
                5.   READ FORWARD (NON-STOP) OVER THESE 16 RECORDS, EACH
                     TIME MONITORING THE TIME FROM THE END OF RECORD (DRY)
                     UNTIL THE FRAME COUNT NEXT GOES FROM 0 TO 1 (FC=1).
                6.   THE TIMES FROM DRY TO FC=1 IS THE GAP TIME AND IT
                     SHOULD REMAIN CONSISTANT FOR ALL RECORDS.
                7.   STOP
                **(SEE GTIMTBL IN DZTUG LISTING FOR GAP TIMES)**


                T22.  RESERVED FOR FUTURE USE*****

                T23.  DATA TIME AT 200 BPI:

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

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


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

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

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

T27. ERASE:

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

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

T30. TAPE MARK:

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

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

```
                        T31.  TAPE SPEED FORWARD:

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

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

                        T32.  TAPE SPEED REVERSE:

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

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

                    %
```

```
                                                                B  3
                                        .NLIST  MC
                                        .LIST   ME
                                        .ABS
                                        .MCALL  $CPVEC,$CPREG,$CATCH,$TYPE
                                        .TITLE  DZTUG-B TM02/TE16 DRIVE FUNCTION TIMER
                                        .SBTTL  STARTING INSTRUCTIONS
                                ;LOADING AND STARTING PROCEEDURE
                                ;       LOAD PROGRAM USING ABS LOADER
                                ;       LOAD ADDRESS 200
                                ;       SET SWITCH OPTIONS
                                ;       PRESS START

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

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

```
1027
1028                            ;       **********************************************************
1029                            ;                       ACT11 HOOKS
1030                            ;       **********************************************************
1031
1032            000764                          $SVPC=.
1033            000040                          .=40
1034  000040      000   DRIVE:    .BYTE   0       ;DRIVE # FOR XXDP LOAD MEDIUM
1035
1036            000041                          .=41
1037  000041      000   MEDIUM:   .BYTE   0       ;XXDP LOAD MEDIUM
1038                                              ;ASSEMBLE AS A 0
1039
1040            000042                          .=42
1041  000042    000000            .WORD   0       ;LOCATION INDICATOR - AUTOM/MAN  MODE
1042                                              ;ASSEMBLE AS A 0
1043
1044            000046                          .=46
1045  000046    013264            .WORD   $ENDAD  ;SET TO $ENDAD IN .$EOP
1046
1047            000052                          .=52
1048  000052    000000            .WORD   0       ;CHARACTERISTICS OF PROGRAM
1049                                              ;SET TO A 0
1050
1051            000764                          .=$SVPC         ;RESTORE PC
1052
1053                            ;       **********************************************************
1054
1055
```

```
1056
1057                            ;       ********************************************************************
1058                            ;                               ACT11 MODE INDICATORS
1059                            ;       ********************************************************************
1060
1061  000764  000000                    AUTOM:   .WORD   0              ;AUTOMATIC MODE INDICATOR
1062  000766     000                    ACT11M:  .BYTE   0              ;ACT11 AUTO MODE INDICATOR
1063  000767     000                    XXDPM:   .BYTE   0              ;XXDP AUTOM MODE INDICATOR
1064  000770     000                    ADUMPM:  .BYTE   0              ;ACT11 DUMP MODE INDICATOR
1065  000771     000                    XDUMPM:  .BYTE   0              ;XXDP DUMP MODE INDICATOR
1066
1067                            ;       ********************************************************
1068
```

```
1069
1070                            .MACRO  $CHNMODE
1071
1072                    ;       ************************************************************
1073
1074                    ;               MACRO FOR SIZING DRIVES AND SLAVES
1075                    ;               IN AUTOMATIC MODE.
1076
1077                    ;       +
1078
1079                    ,       +
1080
1081                            MOV     #-1,DRVNUM              ;INIT DRIVE NUMBER
1082                    NXTDRV: MOV     #-1,SLVNUM              ;INIT SLAVE NUMBER
1083                    1$:     MOV     #40,@CS2                ;INIT CONTROLLER
1084                            INC     DRVNUM                  ;STEP DRIVE #
1085                            CMP     #10,DRVNUM              ;ALL DRIVE TESTED?
1086                            BNE     2$                      ;BRANCH - IF NO
1087                            JMP     FINISH                  ;EXIT
1088                    2$:     MOV     DRVNUM,@CS2             ;LOAD DRIVE #
1089                            TST     @CS1                    ;ACCESS DRIVE
1090                            BIT     #10000,@CS2             ;NON-EXISTANT DRIVE?
1091                            BNE     1$                      ;BRANCH - IF YES
1092                    NXTSLV: INC     SLVNUM                  ;STEP SLAVE #
1093                            BNE     1$                      ;BRANCH IF NOT SLAVE 0
1094                            TST     DRVNUM                  ;DRIVE 0?
1095                            BNE     1$                      ;BRANCH - IF NO
1096                            TSTB    XXDPM                   ;XXDP?
1097                            BEQ     1$                      ;BRANCH - IF NO
1098                            INC     SLVNUM                  ;STEP TO SLAVE #1
1099                    1$:     CMP     #10,SLVNUM              ;ALL DRIVES TESTED?
1100                            BEQ     NXTDRV                  ;BRANCH - IF YES
1101                            MOV     SLVNUM,@TC              ;LOAD SLAVE UNIT?
1102                            BIT     #2000,@DT               ;SLAVE PRESENT?
1103                            BEQ     NXTSLV                  ;BRANCH - IF NO (SPR=0)
1104                            BIT     #140000,@DT             ;IS DRIVE A TAPE UNIT?
1105                            BEQ     NXTSLV                  ;BRANCH IF NO
1106
1107                    ;       ************************************************************
1108
1109                            .ENDM   $CHNMODE
```

```
1110
1111                                    .SBTTL  MACRO DEFINITIONS
1112                                    .MACRO  SAVE
1113                                    JSR     PC,.SAVE                ;SAVE REGISTERS ON THE STACK
1114                                    .ENDM   SAVE
1115                                    .MACRO  RESTORE
1116                                    JSR     PC,.RESTORE             ;RESTORE REGISTERS FROM THE STACK
1117                                    .ENDM   RESTORE
1118                                    .MACRO  INPUT
1119                                    JSR     PC,.INPUT               ;GET USER INPUT
1120                                    .ENDM   INPUT
1121                                    .MACRO  REWIND
1122                                    JSR     PC,.REWIND              ;REWIND SLAVE
1123                                    BVS     99$                     ;BRANCH IF ERROR ON REWIND
1124                                    .ENDM   REWIND
1125                                    .MACRO  TIMEON
1126                                    JSR     PC,TIMON                ;TURN TIMER ON
1127                                    .ENDM
1128                                    .MACRO  TIMCHK
1129                                    JMP     TIMER(R3)               ;GO TO TIMER & RETURN VIA R2
1130                                    .ENDM
1131                                    .MACRO  SETGO
1132                                    INC     (R5)                    ;SET 'GO' BIT
1133                                    .ENDM
1134
1135                                    .SBTTL  REGISTER ASSIGNMENTS
1136                                ;;DEFINITIONS AND REGISTER ASSIGNMENTS
1137                                ;;GENERAL REGISTER ASSIGNMENTS
1138            000000                      R0=%0
1139            000001                      R1=%1
1140            000002                      R2=%2
1141            000003                      R3=%3
1142            000004                      R4=%4
1143            000005                      R5=%5
1144            000006                      SP=%6
1145            000007                      PC=%7
1146            000000                      R10=%0
1147            000001                      R11=%1
1148            000002                      R12=%2
1149            000003                      R13=%3
1150            000004                      R14=%4
1151            000005                      R15=%5
1152
1153                                ;;REGISTER ADDRESSES
1154            177776                      PSW=    177776          ;;PROCESSER STATUS WORD
1155            177774                      SLR=    177774          ;;STACK LIMIT REGISTER (11/40,11/45)
1156            177772                      PIRQ=   177772          ;;PROGRAM INTERRUPT REQ. (11/45)
1157            177770                      UBREAK= 177770          ;;MICRO-BREAK REGISTER (11/45)
1158            177560                      TKS=    177560          ;;KEYBOARD CSR
1159            177562                      TKB=    177562          ;;KEYBOARD DATA BUFFER REGISTER
1160            177564                      TPS=    177564          ;;TELEPRINTER CSR
1161            177566                      TPB=    177566          ;;TELEPRINTER DATA BUFFER REGISTER
1162
1163                                ;;VECTOR ADDRESSES
1164            000004                      ERRVEC=4                ;;ADDRESS OF ERROR VECTOR
1165            000010                      RESVEC=10               ;;ADDRESS OF RESERVED INST. TRAP VECTOR
```

```
1166    000014                  TBITVEC=14              ;;ADDRESS OF 'T' BIT TRAP VECTOR
1167    000014                  TRTVEC=14               ;;ADDRESS OF 'TRACE' TRAP VECTOR
1168    000014                  BPTVEC=14               ;;ADDRESS OF 'BREAKPOINT' TRAP VECTOR
1169    000020                  IOTVEC=20               ;;ADDRESS OF IOT TRAP VECTOR
1170    000024                  PFVEC=24                ;;ADDRESS OF POWER FAIL TRAP VECTOR
1171    000030                  EMTVEC=30               ;;ADDRESS OF EMT VECTOR
1172    000034                  TRAPVEC=34              ;;ADDRESS OF TRAP VECTOR
1173    000060                  TKVEC=  60              ;;ADDRESS OF TTY KEYBOARD INT. VECTOR
1174    000064                  TPVEC=64                ;;ADDRESS OF TTY PRINTER INTERRUPT VECTOR
1175    000114                  PARVEC= 114             ;;ADDRESS OF MA/MF PARITY ERROR VECTOR
1176    000240                  PIRVEC=240              ;;ADDRESS OF PIRQ VECTOR
1177    000244                  FPEVEC=244              ;;ADDRESS OF FLOATING POINT INT. VECTOR
1178    000250                  MMVEC=250               ;;ADDRESS OF MEM MGMT ERROR TRAP VECTOR
1179
1180                    ;CLOCK ADDRESS AND VECTORS
1181    172540                  PLKCSR= 172540          ;KW11-P
1182    000104                  PLKVEC= 104
1183    177546                  LKS=    177546          ;KW11-L
1184    000100                  LKVEC=  100
1185    177514                  LPS=    177514          ;LP11
1186    177516                  LPB=    177516
1187
1188                    ;RH11,TM02/TE16 REGISTERS
1189    172440                  TMCS1=  172440
1190
1191                    ;TM02/TE16 INDEX VALUES
1192    000000                  CS1=    00                      ;CONTROL STATUS #1
1193    000002                  WC=     02
1194    000004                  BA=     04                      ;BUS ADDRESS REGISTER
1195    000006                  FC=     06                      ;FRAME COUNT
1196    000010                  CS2=    10                      ;CONTROL STATUS #2
1197    000012                  DS=     12                      ;DRIVE STATUS
1198    000014                  ER=     14                      ;ERROR REG #1
1199    000016                  AS=     16                      ;ATTENTION SUMMARY
1200    000022                  DB=     22                      ;DATA BUFFER REG
1201    000024                  MR=     24                      ;MAINTENANCE REG
1202    000026                  DT=     26                      ;DRIVE TYPE REG
1203    000030                  SN=     30                      ;SERIAL NUMBER REGISTER
1204    000032                  TC=     32                      ,TAPE CONTROL REG
1205
1206                            .SBTTL  TM02/TE16 REGISTER BITS
1207                    ;RHCS1-CS1(R5)
1208    000001                  GO=     1
1209    000000                  NOP=    0
1210    000002                  RWDOFF= 2
1211    000006                  RWD=    6
1212    000010                  DRYCLR= 10
1213    000026                  WFMK=   26
1214    000024                  ERASE=  24
1215    000030                  SPCFWD= 30
1216    000032                  SPCREV= 32
1217    000050                  WCHKF=  50
1218    000056                  WCHKR=  56
1219    000060                  WFWD=   60
1220    000070                  RDFWD=  70
1221    000076                  RDREV=  76
```

```
1222            000100                          IE=     100
1223            000200                          RDY=    200
1224            000400                          A16=    400
1225            001000                          A17=    1000
1226            002000                          PSEL=   2000
1227            004000                          DVA=    4000
1228            020000                          MCPE=   20000
1229            040000                          TRE=    40000
1230            100000                          SC=     100000
1231                                    ;RHCS2-CS2(R5)
1232            000000                          DV0=    0
1233            000001                          DV1=    1
1234            000002                          DV2=    2
1235            000003                          DV3=    3
1236            000004                          DV4=    4
1237            000005                          DV5=    5
1238            000006                          DV6=    6
1239            000007                          DV7=    7
1240            000010                          BAI=    10
1241            000020                          PAT=    20
1242            000040                          CLR=    40
1243            000100                          IR=     100
1244            000200                          OR=     200
1245            000400                          MDPE=   400
1246            001000                          MXF=    1000
1247            002000                          PGE=    2000
1248            004000                          NEM=    4000
1249            010000                          NED=    10000
1250            020000                          UPE=    20000
1251            040000                          WCE=    40000
1252            100000                          DLT=    100000
1253                                    ;RHDS-DS(R5)
1254            000001                          SLA=    1
1255            000002                          BOT=    2
1256            000004                          TMK=    4
1257            000010                          IDB=    10
1258            000020                          SDWN=   20
1259            000040                          PES=    40
1260            000100                          SSC=    100
1261            000200                          DRY=    200
1262            000400                          DPR=    400
1263            002000                          EOT=    2000
1264            004000                          WRL=    4000
1265            010000                          MOL=    10000
1266            020000                          PIP=    20000
1267            040000                          ERR=    40000
1268            100000                          ATA=    100000
1269                                    ;RHER-ER(R5)
1270            000001                          ILF=    1
1271            000002                          ILR=    2
1272            000004                          RMR=    4
1273
1274            000020                          FMT=    20
1275            000100                          INCVAE= 100
1276            000200                          PEFLRC= 200
1277            000400                          NSG=    400
```

```
1278            001000                          FCE=    1000
1279            002000                          CSITM=  2000
1280            004000                          NEF=    4000
1281            010000                          DTE=    10000
1282            020000                          OPI=    20000
1283            040000                          UNS=    40000
1284
1285                                    ;RHMR-MR(R5)
1286            000100                          OSC=    100
1287
1288                                    ;RHDT-DT(R5)
1289            002000                          SPR=    2000
1290            010000                          CH7=    10000
1291            040000                          TAP=    40000
1292
1293                                    ;RHTC-TC(R5)
1294            000300                          NORM11= 300
1295            000320                          CDM11=  320
1296            000000                          BPI200= 0
1297            000400                          BPI556= 000400
1298            001000                          BPI800= 001000
1299            002000                          PE1600= 002000
1300            100000                          ACCL=   100000
1301
1302                                    ;INSTRUCTION EQUATES
1303            104400                          HLT=    TRAP
1304            104000                          SCOPE=  EMT
1305            000004                          TYPE=   IOT
1306
1307                                    ;MISCELLANEOUS EQUATES
1308            005730                          OUTBUF=INIT             ;OTPUT BUFFER START AT BEG OF PROGRAM
1309            177400                          FRMCNT= -256.           ;FRAME COUNT
1310            177600                          WRDCNT= -128.           ;WORD COUNT
1311                                    ;ASCII EQUATES
1312            000003                          CNTRLC= 3               ;ASCII CODE FOR CONTROL C ( C)
1313            000011                          HT=     11              ;ASCII CODE FOR HORIZONTAL TAB
1314            000012                          LF=     12              ;ASCII CODE FOR LINE FEED
1315            000015                          CR=     15              ;ASCII CODE FOR CARRIAGE RETURN
1316            000017                          CNTRLO= 17              ;ASCII CODE FOR CONTROL O ( O)
1317            000025                          CNTRLU= 25              ;ASCII CODE FOR CONTROL U ( U)
```

```
1318                                    ;SETUP TRAP VECTORS
1319            000014                          .=TBITVEC
1320  000014    000016                          .WORD   .+2             ;SET 'T' TRAP TO TIMER ROUTINE
1321  000016    000000                          .WORD   HALT            ;PRIORITY LEVEL 7
1322  000020    002336                          .WORD   .TYPE           ;SET IOT TRAP TO .TYPE ROUTINE
1323  000022    000000                          .WORD   0               ;PRIOTITY LEVEL 0
1324  000024    000026                          .WORD   PFVEC+2         ;POWER FAIL TRAP TO HALT
1325  000026    000000                          .WORD   HALT            ;AT PFVEC+2
1326  000030    004132                          .WORD   .SCOPE          ;SET EMT TRAP TO .SCOPE ROUTINE
1327  000032    000340                          .WORD   340             ;PRIORITY LEVEL 7
1328  000034    003656                          .WORD   .HLT            ;SET TRAP TRAP TO .HLT ROUTINE
1329  000036    000340                          .WORD   340             ;PRIORITY LEVEL 7
1330            000060                          .=TKVEC
1331  000060    003612                          .WORD   TKISR
1332  000062    000340                          .WORD   340
1333
1334                                    ;SOFTWARE SWITCH REGISTER LOC. 176
1335            000176                          .=176
1336  000176    000000                  SWREG:  0                       ;SOFTWARE SWITCH REGISTER
1337
1338            000200                          .=200
1339
1340  000200    005067  000600                  CLR     PCNTR           ;INIT PROGRAM PASS INDICATOR
1341  000204    000137  005730                  JMP     @#INIT          ;GO TO START OF PROGRAM
1342
1343            000500                          .=500
1344            000600                  STKPTR= 600                     ;STACK
1345
1346            001000                          .=1000
1347                                    ;PROGRAM TAGS
1348  001000    177570                  SWR:    177570                  ;SWITCH REGISTER
1349
1350  001002    000000                  PAFLG:  .WORD   0               ;PASS INDICATOR
1351  001004    000000                  PCNTR:  .WORD   0               ;PASS COUNT INDICATOR
1352
1353  001006    000000                  SCPADR: .WORD   0
1354  001010      000                   DRVNUM: .BYTE   0               ;TM02 DRIVE UNDER TEST
1355  001011      000                   SLVNUM: .BYTE   0               ;TE16 SLAVE UNDER TEST
1356  001012    000000                  SLVPTR: .WORD   0               ;POINTER TO SLAVE TABLE (SLVTBL) BELOW
1357  001014    172440                  TMBASE: .WORD   TMCS1           ;BASE ADDRESS OF TM02/TE16 REGISTERS
1358  001016    000000                  ATIME:  .WORD   0                       ;CONTAINS 'TICK' COUNT
1359  001020    000020                  ATIMTBL:.BLKW   16.              ;EACH ENTRY CONTAINS TIME FOR FUNCTION
1360                                                                    ;ENTRIES ARE MADE BY 'SCOPE' ROUTINE
1361  001060    000020                  GAPTBL: .BLKW   16.             ;TIMES RECORDED BY 'GAP CONSISTANCY' TEST
1362  001120    000000                  DELTIM: .WORD   0               ;VARIBLE DELAY
1363  001122    000000                  OCTALO: .WORD   0
1364  001124      000                   GAP:    .BYTE   0               ;CONTAINS GAP # (USED FOR TST 021)
1365  001125      000                   ITCNT:  .BYTE   0               ;ITERATION COUNT
1366  001126      000                   TSTNUM: .BYTE   0               ;TEST #
1367  001127      000                   ERFLG:  .BYTE   0               ;ERROR FLAG
1368  001130      000                   PRGFLG: .BYTE   0               ;PROGRAM FLAG
1369  001131      000                   UNTFND: .BYTE   0               ;UNIT FOUND INDICATOR
1370  001132      000                   TYPFLG: .BYTE   0
1371  001133      000                   NRZFLG: .BYTE   0               ;INDICATES IF DRIVE IS NRZ ONLY.
1372  001134      000                   ASFLG:  .BYTE   0               ;1/0 = YES/NO.
1373            001136                          .EVEN
```

```
1374  001136  030460            DIGTAB:  "01
1375  001140  031462                     "23
1376  001142  032464                     "45
1377  001144  033466                     "67
1378  001146  034470                     "89
1379  001150  000006            ODIGITS: .BLKB   6              ;RESERVE SPACE FOR CONVERTED DIGITS
1380  001156     000                     .BYTE   0              ;TERMINATOR
1381          001160                     .EVEN
1382  001160  000010            DRVTBL:  .BLKB   8.             ;A 0/-1 = DRIVE NOT TO BE/TO BE TESTED
1383  001170  000100            SLVTBL:  .BLKB   64.            ;A 0/-1 = SLAVE NOT TO BE/TO BE TESTED
1384  001270  000110            INBUF:   .BLKB   72.            ;TELETYPE INPUT BUFFER
1385  001400  005015      000   CRLF:    .ASCIZ  <CR><LF>       ;MISCELLANEOUS ASCII CHARACTERS
1386  001403     134      000   BKSLSH:  .ASCIZ  '\'
1387  001405     060      000   ECHO:    .ASCIZ  '0'
1388  001407     007      000   BELL:    .ASCIZ  <7>
1389  001411     055      000   DASH:    .ASCIZ  '-'
1390  001413     040            SPACE2:  .ASCII  ' '
1391  001414  000040            SPACE:   .ASCIZ  ' '
1392  001416  004476      000   ANGTAB:  .ASCIZ  '>'<HT>
1393          001422                     .EVEN
```

```
1394                                            .SBTTL  TIME SPECIFICATION TABLE
1395                                    ;THE BELOW TABLE CONTAINS THE SPECIFIED FUNCTION TIMES IN TENS OF
1396                                    ;MICROCESONDS.  NOTE THAT WHEN TIMES ARE TYPED THAT THEY ARE TYPED IN
1397                                    ;MICROSECONDS (BY APPENDING A 0).
1398                                    ;FORMAT IS
1399                                    ;       .WORD   MAX,MIN         ;TIME IN MS     FUNCTION        TEST #
1400
1401  001422  000000  000000   STIMTBL: .WORD   0,0                     ;SPARE
1402  001426  036050  035230            .WORD   15400.,15000.           ;154.0-150.0    WRITE FROM BOT  TST001
1403  001432  001666  001546            .WORD   00950.,00870.           ;9.5-8.7        WRITE START     TST002
1404  001436  001572  001522            .WORD   00890.,00850.           ;8.9-8.5        WRITE SHUTDOWN  TST003
1405  001442  002506  001452            .WORD   01350.,00810.           ;13.5-8.1       WRITE STLDOWN   TST004
1406  001446  007164  006344            .WORD   03700.,03300.           ;37.0-33.0      READ FROM BOT   TST005
1407  001452  000500  000404            .WORD   00320.,00260.           ;3.2-2.6        READ START      TST006
1408  001456  000721  000651            .WORD   00465.,00425.           ;4.65-4.25      READ SHUTDOWN   TST007
1409  001462  002506  001452            .WORD   01350.,00810.           ;13.5-8.1       READ SETTLEDOWN TST010
1410  001466  000500  000404            .WORD   00320.,00260.           ;3.2-2.6        RD REV START    TST011
1411  001472  000562  000512            .WORD   00370.,00330.           ;3.7-3.3        RD REV SHTDWN   TST012
1412  001476  002506  001452            .WORD   01350.,00810.           ;13.5-8.1       RD REV STLDWN   TST013
1413  001502  003206  002056            .WORD   01670.,01070.           ;16.7-10.7      TRN RND DLY F-R TST014
1414  001506  003206  002056            .WORD   01670.,01070.           ;16.7-10.7      TRN RND DLY R-F TST015
1415  001512  002412  001666            .WORD   01290.,00950.           ;12.9-9.5       GAP SIZE STOP   TST016
1416  001516  002234  001522            .WORD   01180.,00850.           ;11.8-8.5       GAP SIZE STRT   TST017
1417  001522  002710  002532            .WORD   01480.,01370.           ;14.8-13.7      GAP SIZE INTER  TST020
1418  001526  002544  002330            .WORD   01380.,01240.           ;13.8-12.4      GAP CONSISANCY  TST021
1419  001532  000000  000000            .WORD   0,0                     ;0.0-0.0        DUMMY           TST022
1420  001536  004552  004406            .WORD   02410.,02310.           ;24.1-23.1      DAT TIME 200BPI TST023
1421  001542  004540  004374            .WORD   02400.,02300.           ;24.0-23.0      DAT TIME 556BPI TST024
1422  001546  004540  004374            .WORD   02400.,02300.           ;24 0-23.0      DAT TIME 800BPI TST025
1423  001552  004716  004552            .WORD   02510.,02410.           ;25.1-24.1      DAT TIME 1600PE TST026
1424  001556  023564  023254            .WORD   10100.,09900.           ;101.0-99.0     ERASE           TST027
1425  001562  024404  024074            .WORD   10500.,10300.           ;105.0-103.0    WRT FILE MARK   TST030
1426  001566  004336  004172            .WORD   02270.,02170.           ;22.7-21.7      READ 1" TAPE    TST031
1427  001572  004336  004172            .WORD   02270.,02170.           ;22.7-21.7      RD REV 1" TAPE  TST032
1428
1429                                    ;NOTE:  TEST 31 AND 32 REQUIRE PRERECORDED 800BPI SKEW TAPE.
```

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

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

001576  002602  002412         GTIMTBL:.WORD   01410.,01290.   ;14.1-12.9      GAP-0   0 MS
001602  002652  002506                 .WORD   01450.,01350.   ;14.5-13.5      GAP-1   1.0 MS
001606  002734  002532                 .WORD   01500.,01370.   ;15.0-13.7      GAP-2   2.0 MS
001612  002734  002532                 .WORD   01500.,01370.   ;15.0-13.7      GAP-3   3.0 MS
001616  002734  002424                 .WORD   01500.,01300.   ;15.0-13.0      GAP-4   4.0 MS
001622  002652  002260                 .WORD   01450.,01200.   ;14.5-12.0      GAP-5   5.0 MS
001626  002652  002260                 .WORD   01450.,01200.   ;14.5-12.0      GAP-6   6.0 MS
001632  002652  002260                 .WORD   01450.,01200.   ;14.5-12.0      GAP-7   7.0 MS
001636  002532  002260                 .WORD   01370.,01200.   ;13.7-12.0      GAP-10  8.0 MS
001642  002532  002260                 .WORD   01370.,01200.   ;13.7-12.0      GAP-11  9.0 MS
001646  002532  002260                 .WORD   01370.,01200.   ;13.7-12.0      GAP-12  10.0 MS
001652  002532  002260                 .WORD   01370.,01200.   ;13.7-12.0      GAP-13  11.0 MS
001656  002532  002260                 .WORD   01370.,01200.   ;13.7-12.0      GAP-14  12.0 MS
001662  002532  002260                 .WORD   01370.,01200.   ;13.7-12.0      GAP-15  13.1 MS
001666  002532  002260                 .WORD   01370.,01200.   ;13.7-12.0      GAP-16  14.1 MS
001672  002532  002260                 .WORD   01370.,01200.   ;13.7-12.0      GAP-17  15.1 MS
```

```
                                                                N  3
1453                                            .SBTTL  TEST HEADER POINTERS
1454                                    ;THE BELOW TABLE CONTAINS POINTERS TO EACH TEST'S DESCRIPTOR
1455  001676  000000                    NAMPTR: .WORD   0
1456  001700  015405                            .WORD   A.T001
1457  001702  015427                            .WORD   A.T002
1458  001704  015447                            .WORD   A.T003
1459  001706  015471                            .WORD   A.T004
1460  001710  015515                            .WORD   A.T005
1461  001712  015537                            .WORD   A.T006
1462  001714  015556                            .WORD   A.T007
1463  001716  015600                            .WORD   A.T010
1464  001720  015623                            .WORD   A.T011
1465  001722  015645                            .WORD   A.T012
1466  001724  015672                            .WORD   A.T013
1467  001726  015721                            .WORD   A.T014
1468  001730  015752                            .WORD   A.T015
1469  001732  016003                            .WORD   A.T016
1470  001734  016031                            .WORD   A.T017
1471  001736  016060                            .WORD   A.T020
1472  001740  016110                            .WORD   A.T021
1473  001742  000000                            .WORD   0                       ;DUMMY TEST
1474  001744  016133                            .WORD   A.T023
1475  001746  016157                            .WORD   A.T024
1476  001750  016203                            .WORD   A.T025
1477  001752  016227                            .WORD   A.T026
1478  001754  016254                            .WORD   A.T027
1479  001756  016276                            .WORD   A.T030
1480  001760  016321                            .WORD   A.T031
1481  001762  016343                            .WORD   A.T032
```

```
1482
1483                                    ;CHECK SWITCH REGISTER ROUTINE.  CHECKS FOR  G TO ALLOW CHANGING
1484                                    ;OF LOC.176.
1485                                    ;CALL IS BY WAY OF JSR  PC,CKSWR
1486                                    ;LOCATIONS USED:
1487  001764  000000                   TIB:    .WORD   0
1488  001766  000000                   TEMPST: .WORD   0
1489  001770  000000                   COUNT:  .WORD   0
1490  001772  000000                   RDSW:   .WORD   0
1491
1492  001774  022767  000176  176776  CKSWR:  CMP     #SWREG,SWR              ;SOFTWARE SWITCH REG PRESENT
1493  002002  001120                   BNE     OUT                     ;NO, GET OUT
1494  002004  016767  175552  177752          MOV     TKB,TIB         ;AND STRIP OFF
1495  002012  042767  177600  177744          BIC     #177600,TIB             ;THE GARBAGE
1496  002020  022767  000007  177736          CMP     #7,TIB          ;IS IT A < G>
1497  002026  001106                   BNE     OUT
1498  002030  000004  016365          CNTG:   TYPE,L. CNTG
1499  002034  000004  016372          CNTLU:  TYPE,L. SWR
1500  002040  017702  176734                  MOV     @SWR,R2
1501  002044  004767  000564                  JSR     PC,TYPOCT
1502  002050  000004  016401                  TYPE,L. NEW
1503
1504  002054  005067  177706                  CLR     TEMPST
1505  002060  012767  000007  177702          MOV     #7.COUNT
1506  002066  004767  000154          1$:     JSR     PC,TTIN         ;GO  READ A CHARACTER
1507  002072  042767  177600  177664          BIC     #177600,TIB     ;STRIP OFF GARBAGE
1508  002100  122767  000025  177656          CMPB    #25,TIB         ;IS IT A  U?
1509  002106  001001                   BNE     2$              ;BRANCH IF NOT
1510  002110  000751                   3$:     BR CNTLU                ;START OVER
1511  002112  122767  000015  177644  2$:     CMPB    #15,TIB         ;IS IT A <CR>?
1512  002120  001012                   BNE   . 4$              ;BRANCH IF NOT
1513  002122  012767  000200  177642          MOV     #200,RDSW
1514  002130  004767  000230                  JSR     PC,TCRLF        ;ECHO IT WITH <LF>
1515  002134  022767  000007  177626          CMP     #7,COUNT        ;WAS IT FIRST CHARACTER
1516  002142  001034                   BNE     7$              ;CHANGE SWR IF NOT FIRST ONE
1517  002144  000437                   8$:     BR      OUT             ;GET OUT
1518  002146  122767  000060  177610  4$:     CMPB    #60,TIB
1519  002154  003004                   BGT     5$
1520  002156  122767  000067  177600          CMPB    #67,TIB
1521  002164  002003                   BGE     6$
1522  002166  000004  016411          5$:     TYPE,L. QUEST
1523  002172  000746                   BR      3$              ;START OVER IF NOT LEGAL CHARACTER
1524  002174  006367  177566          6$:     ASL     TEMPST
1525  002200  006367  177562                  ASL     TEMPST
1526  002204  006367  177556                  ASL     TEMPST
1527  002210  142767  000060  177546          BICB    #60,TIB         ;GET NITTY-GRITTY
1528  002216  156767  177542  177542          BISB    TIB,TEMPST
1529  002224  005367  177540                  DEC     COUNT           ;ONLY WANT 6   DIGITS
1530  002230  001756                   BEQ     5$
1531  002232  000715                   BR      1$
1532  002234  016777  177526  176536  7$:     MOV     TEMPST,@SWR     ;CHANGE SWITCH REGISTER CONTENTS
1533  002242  000740                   BR      8$
1534  002244  000207                   OUT:    RTS     PC
1535
```

```
1536
1537                                    ;TTY READ SUBROUTINE**********
1538
1539  002246  005067  175306          TTIN:   CLR     TKS
1540  002252  005067  175304                  CLR     TKB
1541  002256  005067  177502                  CLR     TIB
1542  002262  005267  175272                  INC     TKS
1543  002266  105767  175266          TTIN1:  TSTB    TKS
1544  002272  100375                  TTIN1   BPL     TTIN1
1545  002274  016767  175262  177462          MOV     TKB,TIB
1546  002302  105767  175256          TTIN2:  TSTB    TPS
1547  002306  100375                  TTIN2   BPL     TTIN2
1548  002310  116767  177450  175250          MOVB    TIB,TPB
1549  002316  000207                          RTS     PC
1550
1551                                    .SBTTL  PROGRAM SUBROUTINES
1552                                    .SBTTL          TYPE SUBROUTINE
1553                                ;;ROUTINE TO TYPE ASCII MESSAGE. MESSAGE MUST TERMINATE WITH A 0 BYTE.
1554                                ;;THE ROUTINE WILL INSERT A NUMBER OF NULL CHARACTERS AFTER A LINE FEED.
1555                                ;;CALL: TYPE                    ;;A TRAP TYPE INSTRUCTION
1556                                ;;      MESADR                  ;;MESADR IS FIRST ADDRESS OF ASCIZ STRING
1557
1558                                ;;TAGS USED BY THE TYPE ROUTINE BELOW
1559          000011                        $HT=11                  ;;HORIZONTAL TAB
1560  002320     000                $NULL:  .BYTE   0               ;;CONTAINS NULL CHARACTER
1561  002321     002                $FILL:  .BYTE   2               ;;CONTAINS # OF FILLER CHARACTERS
1562  002322     000                $TPFLG: .BYTE   0               ;;CONTAINS TELEPRINTER AVAILABLE FLAG
1563                                                                ;;0/377 = AVAIL/NOT AVAIL
1564  002323     000                $TKFLG: .BYTE   0               ;;CONTAINS KEYBOARD AVAILABLE FLAG
1565  002324  177564                $TPS:   .WORD   177564          ;;ADDRESS OF TLEPRINTER STATUS REGISTER
1566  002326  177566                $TPB:   .WORD   177566          ;;ADDRESS OF TELEPRINTER DATA BUFFER
1567  002330     000                $CHARCNT:.BYTE  0               ;;CONTAINS # OF CHARS TYPED
1568  002331     000                $CNTRLO:.BYTE   0               ;;CONTAINS CONTROL O CHAR (IF TYPED)
1569  002332  005015     000        $CRLF:  .ASCIZ  <15><12>
1570          002336                        .EVEN
1571
1572  002336  010046                .TYPE:  MOV     R0,-(SP)        ;;SAVE R0
1573  002340  017600  000002                MOV     @2(SP),R0       ;;GET MESSAGE ADDRESS
1574  002344  062766  000002  000002        ADD     #2,2(SP)        ;;ADJUST RETURN PC
1575  002352  105067  177753                CLRB    $CNTRLO
1576
1577  002356  105767  177747          TYPE1:  TSTB    $CNTRLO         ;;BRANCH IF CONTROL O( O) WASN'T TYPED
1578  002362  001410                          BEQ     TYPE2
1579  002364  000004  002332          TCRLF:  TYPE,$CRLF              ;;TYPE <CR><LF>
1580  002370  105767  177376                  TSTB    RDSW
1581  002374  100006                          BPL     TYPE3
1582  002376  005067  177370                  CLR     RDSW
1583  002402  000207                          RTS     PC
1584  002404  112046                TYPE2:  MOVB    (R0)+,-(SP)     ;;PUSH CHARACTER TO BE TYPED ONTO STACK
1585  002406  001003                          BNE     TYPE4           ;;BRANCH IF NOT THE TERMINATOR
1586  002410  005726                          TST     (SP)+           ;;POP TERMINATOR CHAR OFF THE STACK
1587  002412  012600                TYPE3:  MOV     (SP)+,R0        ;;RESTORE R0
1588  002414  000002                          RTI                     ;;RETURN TO CALLER
1589
1590  002416  122716  000011          TYPE4:  CMPB    #$HT,(SP)       ;;BRANCH IF HORIZONTAL TAB <HT>
1591  002422  001445                          BEQ     9$
```

```
1592  002424  004767  000026          JSR     PC,5$           ;;TYPE CHARACTER
1593  002430  122726  000012    3$:   CMPB    #12,(SP)+       ;;CHECK IF CHARACTER WAS A LINE FEED
1594  002434  001350                  BNE     TYPE1           ;;BRANCH IF NOT LINE FEED
1595  002436  016746  177656          MOV     $NULL,-(SP)     ;;GET # OF FILLERS REQUIRED AND FILLER
1596                                                          ;;CHARACTER.
1597
1598  002442  105366  000001    4$:   DECB    1(SP)           ;;DECREMENT FILLERS REQ. COUNT
1599  002446  002770                  BLT     3$              ;;BRANCH IF NO MORE FILLERS ARE REQUIRED
1600  002450  004767  000002          JSR     PC,5$           ;;TYPE FILLER CHARACTER
1601  002454  000772                  BR      4$
1602
1603  002456  105777  177642    5$:   TSTB    @$TPS           ;;WAIT FOR OUTPUT DEVICE
1604  002462  100375                  BPL     .-4
1605  002464  122737  000017  002331        CMPB    #17,@#$CNTRLO   ;;CHECK IF CONTROL O WAS TYPED
1606  002472  001403                  BEQ     6$              ;;STOP TYPING MESSAGE IF  O WAS TYPED
1607  002474  116677  000002  177624        MOVB    2(SP),@$TPB     ;;OUTPUT CHARACTER
1608  002502  122766  000015  000002  6$:   CMPB    #15,2(SP)       ;;BRANCH IF NOT <CR>
1609  002510  001003                  BNE     7$
1610  002512  105067  177612          CLRB    $CHARCNT        ;;CLEAR CHARACTERS TYPED COUNT
1611  002516  000406                  BR      8$
1612  002520  122766  000012  000002  7$:   CMPB    #12,2(SP)       ;;BRANCH IF <LF> OR 'NULL'
1613  002526  002002                  BGE     8$
1614  002530  105267  177574          INCB    $CHARCNT        ;;INCREMENT CHARACTER TYPED COUNT
1615  002534  000207              8$:   RTS     PC
1616
1617                              ;;HORIZONTAL TAB <HT> PROCESSER
1618  002536  112716  000040        9$:   MOVB    #40,(SP)        ;;LOAD 'SPACE'
1619  002542  004767  177710        10$:  JSR     PC,5$           ;;TYPE 'SPACE'
1620  002546  132767  000007  177554        BITB    #7,$CHARCNT     ;;TYPE SPACES UNTIL A MULTIPLE
1621  002554  001372                  BNE     10$             ;;OF 8 CHARACTERS HAVE BEEN TYPED
1622  002556  105726                  TSTB    (SP)+           ;;POP SPACE
1623  002560  000676                  BR      TYPE1           ;;GET NEXT CHARACTER
1624
1625                              ;SUBROUTINE TO SAVE GENERAL REGISTERS ON THE STACK
1626                              ;CALL:  SAVE
1627  002562  010546              .SAVE:  MOV     R5,-(SP)                ;SAVE REGISTERS ON THE STACK
1628  002564  010446                      MOV     R4,-(SP)
1629  002566  010346                      MOV     R3,-(SP)
1630  002570  010246                      MOV     R2,-(SP)
1631  002572  010146                      MOV     R1,-(SP)
1632  002574  010046                      MOV     R0,-(SP)
1633  002576  016646  000014              MOV     14(SP),-(SP)            ;GET RETURN PC
1634  002602  000207                      RTS     PC                      ;RETURN
1635
1636                              ;SUBROUTINE TO RESTORE GENERAL REGISTERS FROM THE STACK
1637                              ;CALL:  RESTORE
1638  002604  012666  000014      .RESTORE:MOV    (SP)+,14(SP)            ;MOVE RETURN PC
1639  002610  012600                      MOV     (SP)+,R0                ;RESTORE REGISTERS
1640  002612  012601                      MOV     (SP)+,R1
1641  002614  012602                      MOV     (SP)+,R2
1642  002616  012603                      MOV     (SP)+,R3
1643  002620  012604                      MOV     (SP)+,R4
1644  002622  012605                      MOV     (SP)+,R5
1645  002624  000207                      RTS     PC                      ;RETURN
1646
1647                              ;SUBROUTINE TO CONVERT OCTAL DATA TO ASCII
```

```
                                                                E 4
 1648                                   ;CALL:   MOV     NUMBER,R2               ;MOVE NUMBER TO R2
 1649                                   ;        JSR     PC,CNVOCT
 1650
 1651  002626  110667  176300           CNVOCT:  MOVB    SP,TYPFLG               ;SET DO NOT TYPE FLAG
 1652  002632  000402                            BR      CNVTO
 1653
 1654                                           .SBTTL          OCTAL TO ASCII & TYPE ROUTINE
 1655                                   ;SUBROUTINE TO CONVERT OCTAL NUMBER TO ASCII AND TYPE IT OUT
 1656                                   ;CALL:   MOV     NUMBER,R2               ;PUT # IN R2
 1657                                   ;        JSR     PC,TYPOCT               ;CALL ROUTINE
 1658
 1659  002634  105037  001132           TYPOCT:  CLRB    @#TYPFLG                ;SET TYPE FLAG
 1660  002640                           CNVTO:
 1661  002640  004767  177716                    JSR     PC,.SAVE                ;SAVE REGISTERS ON THE STACK
 1662  002644  012704  001150                    MOV     #ODIGITS,R4             ;SET PTR TO OUTPUT
 1663  002650  005003                            CLR     R3                      ;R3 WILL CONTAIN OCTAL DIGIT
 1664  002652  010201                            MOV     R2,R1                   ;GET # TO BE TYPED
 1665  002654  006302                   1$:      ASL     R2                      ;SHIFT #
 1666  002656  006103                            ROL     R3
 1667  002660  012700  000006                    MOV     #6,R0                   ;SET DIGIT COUNTER
 1668  002664  000404                            BR      3$
 1669
 1670  002666  006302                   2$:      ASL     R2                      ;SHIFT # 3 PLACES LEFT
 1671  002670  006103                            ROL     R3
 1672  002672  005301                            DEC     R1
 1673  002674  001374                            BNE     2$
 1674  002676  012701  000003           3$:      MOV     #3,R1                   ;SET SHIFT COUNTER
 1675  002702  116324  001136                    MOVB    DIGTAB(R3),(R4)+        ;MOVE ASCII EQUIV TO OUTPUT
 1676  002706  005003                            CLR     R3
 1677  002710  005300                            DEC     R0                      ;DECREMENT DIGIT COUNT
 1678  002712  001365                            BNE     2$                      ;GET NEXT DIGIT
 1679  002714  105737  001132                    TSTB    @#TYPFLG                ;BRANCH IF ASCII IS
 1680  002720  001002                            BNE     4$                      ;NOT TO BE TYPED
 1681  002722  000004  001150                    TYPE,ODIGITS
 1682  002726                           4$:
 1683  002726  004767  177652                    JSR     PC,.RESTORE             ;RESTORE REGISTERS FROM THE STACK
 1684  002732  000207                            RTS     PC
 1685
 1686
 1687                                   ;SUBROUTINE TO CONVERT OCTAL DATA TO DECIMAL ASCII
 1688                                   ;CALL:   MOV     NUMBER,R2               ;MOVE NUMBER TO R2
 1689                                   ;        JSR     PC,CNVDEC
 1690
 1691  002734  110637  001132           CNVDEC:  MOVB    SP,@#TYPFLG             ;SET DO NOT TYPE FLAG
 1692  002740  000402                            BR      CNVTD
 1693                                           .SBTTL          OCTAL TO DECIMAL & TYPE ROUTINE
 1694                                   ;THIS ROUTINE CONVERTS AN OCTAL # TO DECIMAL ASCII AND TYPES IT OUT
 1695                                   ;CALL:   MOV     NUMBER,R2               ;PUT # IN R2
 1696                                   ;        JSR     PC,TYPDEC               ;CALL ROUTINE
 1697
 1698  002742  105037  001132           TYPDEC:  CLRB    @#TYPFLG                ;SET TYPE FLAG
 1699  002746                           CNVTD:
 1700  002746  004767  177610                    JSR     PC,.SAVE                ;SAVE REGISTERS ON THE STACK
 1701  002752  005000                            CLR     R0                      ;R0 IS INDEX TO DECIMAL CONSTANT
 1702  002754  012704  001150                    MOV     #ODIGITS,R4             ;SET OUTPUT PTR
 1703  002760  005003                   1$:      CLR     R3                      ;R3 CONTAINS DECIMAL DIGIT
```

```
1704  002762  166002  003042          2$:     SUB     DCONST(R0),R2           ;SUBTRACT DECIMAL CONSTANT UNTIL
1705  002766  103402                          BLO     3$                      ;INPUT # GOES NEGATIVE
1706  002770  005203                          INC     R3                      ;KEEPING TRACK OF SUBTRACTIONS
1707  002772  000773                          BR      2$
1708  002774  066002  003042          3$:     ADD     DCONST(R0),R2           ;ADD BACK CONSTANT WHEN NEGATIVE
1709  003000  116324  001136                  MOVB    DIGTAB(R3),(R4)+        ;MOVE ASCII EQUIVILENT
1710  003004  062700  000002                  ADD     #2,R0                   ;NEXT CONSTANT
1711  003010  005760  003042                  TST     DCONST(R0)              ;UNTIL ALL CONSTANTS DONE
1712  003014  001361                          BNE     1$
1713  003016  112724  000060                  MOVB    #'0,(R4)+               ;LAST DIGIT IS 0
1714  003022  105737  001132                  TSTB    @#TYPFLG                ;BRANCH IF ASCII IS
1715  003026  001002                          BNE     4$                      ;NOT TO BE TYPED
1716  003030  000004  001150                  TYPE,ODIGITS
1717  003034                          4$:
1718  003034  004767  177544                  JSR     PC,..RESTORE            ;RESTORE REGISTERS FROM THE STACK
1719  003040  000207                          RTS     PC
1720
1721  003042  023420          DCONST: .WORD   10000.
1722  003044  001750                  .WORD   1000.
1723  003046  000144                  .WORD   100.
1724  003050  000012                  .WORD   10.
1725  003052  000001                  .WORD   1.
1726  003054  000000                  .WORD   0                               ;TERMINATOR
1727
1728                                  .SBTTL          TYPE SPECIFIED TIMES ROUTINE
1729                          ;THIS SUBROUTINE OUTPUTS THE TIME SPECIFICATIONS FOR THE TEST
1730                          ;AND ALSO THE ACTUAL TIME RECORDED (ATIME)
1731                          ;FORMAT OF LINE TYPED
1732                          ;RANGE=<AAAAAA-BBBBBB>          ACTUAL=CCCCCC
1733                          ;WHERE:         AAAAAA IS MAXIMUM TIME FOR TEST (STIMTBL(TSTNUMX4)).
1734                          ;               BBBBBB IS MINIMUM TIME FOR TEST (STIMTBL(TSTNUMX4+2)).
1735                          ;               CCCCCC IS ACTUAL TIME RECORDED BY TEST (ATIME).
1736                          ;CALL:  MOVB    TEST NUMBER,R2  ;LOAD TEST NUMBER
1737                          ;       MOV     #TIME,@#ATIME           ;MOVE TIME TO ATIME
1738                          ;       JSR     PC,OUTSPC
1739  003056  010246          OUTSPC: MOV     R2,-(SP)                ;SAVE R2 & R3 ON THE STACK
1740  003060  010346                  MOV     R3,-(SP)
1741  003062  006302                  ASL     R2                      ;MULTIPLY TEST # TIMES 4
1742  003064  006302                  ASL     R2                      ,TO FORM INDEX INTO STIMTBL
1743  003066  010203                  MOV     R2,R3                   ;R3 CONTAINS INDEX INTO TABLE
1744  003070  000004  015365          TYPE,L.RNG
1745  003074  016302  001422          MOV     STIMTBL(R3),R2          ;GET MAXIMUM SPEC TIME
1746  003100  004767  177636          JSR     PC,TYPDEC               ;CONVERT TO DECIMAL & TYPE
1747  003104  000004  001411          TYPE,DASH
1748  003110  016302  001424          MOV     STIMTBL+2(R3),R2        ;GET MINIMUM TIME
1749  003114  004767  177622          JSR     PC,TYPDEC               ;CONVERT TO DECIMAL & TYPE
1750  003120  000004  001416          TYPE,ANGTAB
1751  003124  000004  015375          TYPE,L.ACT
1752  003130  013702  001016          MOV     @#ATIME,R2              ;GET ACTUAL TIME
1753  003134  004767  177602          JSR     PC,TYPDEC               ;CONVERT TO DECIMAL & TYPE
1754  003140  000004  001400          TYPE,CRLF
1755  003144  012603                  MOV     (SP)+,R3
1756  003146  012602                  MOV     (SP)+,R2
1757  003150  000207                  RTS     PC                      ;RETURN
1758
1759                                  .SBTTL          TYPE GAP TIMES SUBROUTINE
```

```
                        ;THIS SUBROUTINE IS USED TO TYPE THE SPECIFIED GAP SIZES (RECORDED IN
                        ;TST021).  IT IS CALLED BY THE GAPOK ROUTINE IF THE GAP SIZE IS OUT OF
                        ;RANGE VIA THE HLT ROUTINE (HLT+2).
                        ;CALL:  MOVB    #GAP,GAP                ;LOAD GAP # INTO GAP
                        ;       MOV     #TIME,ATIME             ;LOAD ACTUAL TIME INTO ATIME
                        ;       JSR     PC,OUTGAP
003152  010246          OUTGAP: MOV     R2,-(SP)                ;SAVE R2 AND R3
003154  010346                  MOV     R3,-(SP)
003156  116703  175742          MOVB    GAP,R3                  ;GET GAP #
003162  006303                  ASL     R3
003164  006303                  ASL     R3
003166  000004  015365          TYPE,L.RNG
003172  016302  001576          MOV     GTIMTBL(R3),R2          ;GET MAX TIME
003176  004767  177540          JSR     PC,TYPDEC               ;CONVERT TO DECIMAL & TYPE
003202  000004  001411          TYPE,DASH
003206  016302  001600          MOV     GTIMTBL+2(R3),R2        ;GET MIN TIME
003212  004767  177524          JSR     PC,TYPDEC               ;CONVERT TO DECIMAL & TYPE
003216  000004  001416          TYPE,ANGTAB                     ;TYPE <
003222  000004  015375          TYPE,L.ACT
003226  013702  001016          MOV     @#ATIME,R2              ;GET ACTUAL TIME
003232  004767  177504          JSR     PC,TYPDEC               ;CONVERT TO DECIMAL & TYPE
003236  000004  015054          TYPE,E.GAP
003242  113702  001124          MOVB    @#GAP,R2                ;GET GAP #
003246  004767  177362          JSR     PC,TYPOCT               ;TYPE GAP #
003252  000004  001400          TYPE,CRLF
003256  012603                  MOV     (SP)+,R3                ;RESTORE R3 AND R2
003260  012602                  MOV     (SP)+,R2
003262  000207                  RTS     PC

                                .SBTTL          ASCII TO OCTAL CONVERT SUBROUTINE
                        ;SUBROUTINE TO CONVERT ASCII DATA TO OCTAL.  CONVERTED OCTAL DATA
                        ;IS LEFT IN OCTALO <15-00>.
003264                  CNVTAO:
003264  004767  177272          JSR     PC,.SAVE                ;SAVE REGISTERS ON THE STACK
003270  012700  001270          MOV     #INBUF,R0               ;SET PTR TO ASCII DATA
003274  012701  001122          MOV     #OCTALO,R1              ;GET ADDRESS OF OCTAL DATA
003300  005011                  CLR     (R1)                    ;CLEAR OUT OLD OCTAL DATA
003302  005061  000002          CLR     2(R1)
003306  122710  000015  1$:     CMPB    #CR,(R0)                ;<CR> TERMINATES INPUT
003312  001414                  BEQ     3$
003314  112002                  MOVB    (R0)+,R2                ;GET 'OCTAL' DATA
003316  042702  177770          BIC     #177770,R2              ;STRIP UNUSED BITS
003322  012703  000003          MOV     #3,R3                   ;SET SHIFT COUNT
003326  006311          2$:     ASL     (R1)                    ;SHIFT LAST
003330  006161  000002          ROL     2(R1)                   ;OCTAL DIGIT
003334  005303                  DEC     R3
003336  001373                  BNE     2$
003340  050211                  BIS     R2,(R1)                 ;AND INSERT THIS DIGIT
003342  000761                  BR      1$                      ;GO GET NEXT DIGIT
003344                  3$:
003344  004767  177234          JSR     PC,.RESTORE             ;RESTORE REGISTERS FROM THE STACK
003350  000207                  RTS     PC

                                .SBTTL          PUBLISH SUBROUTINE
                        ;THE PUBLISH SUBROUTINE AVERAGES THE RECORDED TIMES FOR EACH TEST IT-
```

```
1816                                    ;ERATION (IF 16. ITERATIONS) AND PLACES THE AVERAGE RESULT IN 'ATIME'.
1817                                    ;IT TYPES THE NAME OF THE FUNCTION THAT WAS TIMED, THE TIME SPEC-
1818                                    ;IFICATION AND THE ACTUAL TIME .
1819
1820  003352                            PUBLISH:
1821  003352  004767  177204                    JSR     PC,.SAVE                ;SAVE REGISTERS ON THE STACK
1822  003356  012700  001020                    MOV     #ATIMTBL,R0     ;GET TABLE ADDRESS CONTAINING TIMES
1823  003362  113701  001125                    MOVB    @#ITCNT,R1      ;GET # OF ENTRIES (GIVEN BY ITERATION COUNT)
1824  003366  122701  000001                    CMPB    #1,R1                   ;BRANCH IF SINGLE ITERATION
1825  003372  001423                            BEQ     4$
1826  003374  005002                            CLR     R2                      ;CLEAR 'SUM' REGISTERS
1827  003376  005003                            CLR     R3
1828  003400  122701  000020                    CMPB    #16.,R1                 ;BRANCH IF 16. ITERATIONS
1829  003404  001402                            BEQ     1$
1830  003406  000000                            HALT                            ;ITERATION COUNT MUST BE 1 OR 16.
1831  003410  000777                            BR      .                       ;DO NOT CHANGE POSIT OF SW11
1832                                                                            ;WHEN TEST IS RUNNING.
1833
1834  003412  062002                    1$:     ADD     (R0)+,R2                ;SUM INDIVIDUAL TIMES
1835  003414  005503                            ADC     R3
1836  003416  005301                            DEC     R1
1837  003420  001374                            BNE     1$
1838
1839  003422  012700  000004            2$:     MOV     #4.R0
1840  003426  006203                    3$:     ASR     R3                      ;SHIFT TIME IN R3 & R2 4 PLACES
1841  003430  006002                            ROR     R2                      ;RIGHT = DIVIDE BY 16.
1842  003432  005300                            DEC     R0
1843  003434  001374                            BNE     3$
1844  003436  010237  001016                    MOV     R2,@#ATIME              ;MOVE AVERAGED TIMES
1845
1846  003442  113700  001126            4$:     MOVB    @#TSTNUM,R0             ;GET TEST #
1847  003446  006300                            ASL     R0
1848  003450  016067  001676  000002            MOV     NAMPTR(R0),5$           ;GET TEST NAME STRING ADDRESS
1849  003456  000004                            TYPE
1850  003460  000000                    5$:     .WORD   0
1851  003462  113702  001126                    MOVB    @#TSTNUM,R2             ;GET TEST #
1852  003466  004767  177364                    JSR     PC,OUTSPC               ;OUTPUT TIMES
1853  003472  004767  177106                    JSR     PC,.RESTORE             ;RESTORE REGISTERS FROM THE STACK
1854  003476  000207                            RTS     PC
1855
1856                                            .SBTTL          INPUT SUBROUTINE
1857                                    ;SUBROUTINE TO GET TTY INPUT
1858                                    ;CALL:  JSR     PC,.INPUT
1859                                    ;INPUT DATA IS RETUNED IN BUFFER BEGINNING AT INBUF.
1860
1861  003500  010046                    .INPUT: MOV     R0,-(SP)        ;SAVE R0 ON THE STACK
1862  003502  012700  001270            1$:     MOV     #INBUF,R0
1863  003506  105737  177560            2$:     TSTB    @#TKS
1864  003512  100375                            BPL     2$
1865
1866  003514  113746  177562                    MOVB    @#TKB,-(SP)     ;GET CHARACTER
1867  003520  042716  000200                    BIC     #200,(SP)
1868  003524  122716  000177                    CMPB    #177,(SP)       ;CHECK RUBOUT
1869  003530  001004                            BNE     3$
1870  003532  124026                            CMPB    -(R0),(SP)+     ;REMOVE CHARACTER FROM INPUT
1871  003534  000004  001403                    TYPE,BKSLSH
```

```
1872  003540  000762                          BR      2$                      ;WAIT FOR NEXT CHARACTER
1873  003542  122716  000025        3$:       CMPB    #CNTRLU,(SP)            ;CHECK CONTROL U ( U)
1874  003546  001004                          BNE     4$
1875  003550  005726                          TST     (SP)+
1876  003552  000004  001400                  TYPE,CRLF
1877  003556  000751                          BR      1$
1878  003560  111637  001405        4$:       MOVB    (SP),@#ECHO
1879  003564  111620                          MOVB    (SP),(R0)+
1880  003566  122726  000015                  CMPB    #CR,(SP)+
1881  003572  001403                          BEQ     5$
1882  003574  000004  001405                  TYPE,ECHO
1883  003600  000742                          BR      2$
1884  003602  000004  001400        5$:       TYPE,CRLF
1885  003606  012600                          MOV     (SP)+,R0
1886  003610  000207                          RTS     PC
1887
1888                                ;KEYBOARD INTERRUPT SERVICE ROUTINE
1889  003612  113746  177562        TKISR:    MOVB    @#TKB,-(SP)             ;GET TYPED CHARACTER
1890  003616  042716  000200                  BIC     #200,(SP)               ;STRIP PARITY BIT
1891  003622  122716  000017                  CMPB    #CNTRLO,(SP)            ;BRANCH IF NOT CONTROL O ( O)
1892  003626  001003                          BNE     1$
1893  003630  112667  176475                  MOVB    (SP)+,$CNTRLO           ;SET CONTROL O INDICATOR IN TYPE ROUTINE
1894  003634  000002                          RTI                             ;EXIT
1895
1896  003636  122726  000003        1$:       CMPB    #3,(SP)+                ;BRANCH IF NOT CONTROL C ( C)
1897  003642  001003                          BNE     2$
1898  003644  000005                          RESET
1899  003646  000137  005730                  JMP     @#INIT                  ;RESTART PROGRAM
1900  003652  000002                2$:       RTI                             ;EXIT
```

```
                                                                J  4
                                        .SBTTL          ERROR SERVICE ROUTINES
                                ;ROUTINE TO PROCESS ERROR TRAPS (TRAPS TO 4)
003654  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).

003656  004767  176112          .HLT:   JSR     PC,CKSWR                ;CHECK FOR CNTL G
003662  004767  176674                  JSR     PC,..SAVE               ;SAVE REGISTERS ON THE STACK
003666  110637  001127          1$:     MOVB    SP,@#ERFLG              ;SET ERROR FLAG
003672  032777  020000  175100          BIT     #SW13,@SWR              ;BRANCH IF NO TYPOUT
003700  001075                          BNE     4$
003702  000004  014655                  TYPE,E.HDR
003706  113702  001126                  MOVB    @#TSTNUM,R2             ;GET TEST #
003712  004767  176716                  JSR     PC,TYPOCT               ;AND TYPE IT
003716  016600  000016                  MOV     16(SP),R0               ;GET RETURN PC
003722  162700  000002                  SUB     #2,R0                   ;NOW PC OF HLT CALL
003726  111000                          MOVB    (R0),R0                 ;NOW HLT CALL ITSELF
003730  001417                          BEQ     2$                      ;BRANCH IF HLT
003732  000004  014740                  TYPE,E.HDR2
003736  122700  000002                  CMPB    #2,R0                   ;BRANCH IF NOT HLT+2
003742  001005                          BNE     10$
003744  004767  177202                  JSR     PC,OUTGAP               ;TYPE GAP SPECIFIED TIMES
003750  000004  001400                  TYPE,CRLF
003754  000447                          BR      4$
003756  004767  177074          10$:    JSR     PC,OUTSPC               ;TYPE SPECIFIED TIMES
003762  000004  001400                  TYPE,CRLF
003766  000442                          BR      4$
003770  016500  000014          2$:     MOV     ER(R5),R0
003774  032765  002000  000032          BIT     #PE1600,TC(R5)
004002  001403                          BEQ     20$
004004  042700  102100                  BIC     #102100,R0
004010  000402                          BR      21$
004012  042700  102300          20$:    BIC     #102300,R0
004016  005700                  21$:    TST     R0
004020  001003                          BNE     22$
004022  000004  014631                  TYPE,E.SFT              ;TYPE SOFT ERROR MESSAGE
004026  000434                          BR      6$

004030  000004  014665          22$:    TYPE,E.HDR1
004034  010500                          MOV     R5,R0                   ;GET FIRST ADDRESS OF REGS.
004036  012701  000007                  MOV     #7.,R1                  ;TYPE FIRST 7 REGS.
004042  012002                  3$:     MOV     (R0)+,R2                ;GET REG CONTENTS
004044  004767  176564                  JSR     PC,TYPOCT               ;AND TYPE IT
004050  000004  001413                  TYPE,SPACE2
004054  005301                          DEC     R1
004056  001371                          BNE     3$
004060  016502  000032                  MOV     TC(R5),R2               ;GET CONTENTS OF TC REGISTER
004064  004767  176544                  JSR     PC,TYPOCT
004070  000004  001400                  TYPE,CRLF

004074  032777  001000  174676  4$:     BIT     #SW09,@SWR              ;BRANCH IF NO RING THE BELL
004102  001402                          BEQ     5$
004104  000004  001407                  TYPE,BELL
```

```
1957  004110  005777  174664          5$:     TST     @SWR                    ;HALT ON ERROR?
1958  004114  100001                          BPL     6$
1959  004116  000000                          HALT
1960  004120  004767  175650          6$:     JSR     PC,CKSWR        ;CHECK FOR CNTL G
1961  004124  004767  176454                  JSR     PC,..RESTORE            ;RESTORE REGISTERS FROM THE STACK
1962  004130  000002                          RTI                             ;RETURN
1963
1964
```

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

004132  004767  175636          .SCOPE: JSR     PC,CKSWR                ;CHECK FOR CNTL G
004136  013705  001014                  MOV     @#TMBASE,R5             ;SET R5 TO FIRST TM REG
004142  032777  000400  174630          BIT     #SW08,@SWR              ;BRANCH IF SPECIFICATION LINE
004150  001404                          BEQ     10$                     ;NOT DESIRED ON EACH ITERATION
004152  113702  001126                  MOVB    @#TSTNUM,R2             ;GET TEST NUMBER
004156  004767  176674                  JSR     PC,OUTSPC               ;OUTPUT TIME  RECORDED
004162  032777  040000  174610  10$:    BIT     #SW14,@SWR              ;BRANCH IF CONTINUOS LOOP
004170  001417                          BEQ     2$                      ;NOT DESIRED
004172  004767  000546          1$:     JSR     PC,DELAY                ;DELAY 350 MS
004176  004767  000776                  JSR     PC,RHINIT               ;INIT
004202  105037  001127                  CLRB    @#ERFLG                 ;CLEAR ERROR FLAG
004206  016716  174574                  MOV     SCPADR,(SP)
004212  010501                          MOV     R5,R1
004214  062701  000012                  ADD     #DS,R1                  ;ADDRESS OF 'DS' REG IS IN R1
004220  010500                          MOV     R5,R0
004222  062700  000006                  ADD     #FC,R0                  ;ADDRESS OF 'FC' REG IS IN R0
004226  000002                          RTI

004230  105737  001127          2$:     TSTB    @#ERFLG                 ;BRANCH IF ERROR FLAG IS SET
004234  001006                          BNE     3$
004236  113700  001125                  MOVB    @#ITCNT,R0              ;GET ITERATION COUNT
004242  006300                          ASL     R0                      ;STORE TIME IN TABLE
004244  013760  001016  001020          MOV     @#ATIME,ATIMTBL(R0)
004252  105237  001125          3$:     INCB    @#ITCNT                 ;INCREMENT ITERATION COUNT
004256  032777  004000  174514          BIT     #SW11,@SWR              ;BRANCH IF SINGLE ITERATION DESIRED
004264  001004                          BNE     4$
004266  122737  000020  001125          CMPB    #16.,@#ITCNT            ;BRANCH IF ITERATIONS INCOMPLETE
004274  001336                          BNE     1$
004276  011637  001006          4$:     MOV     (SP),@#SCPADR           ;SET SCOPE ADDRESS TO NEXT TEST
004302  032777  002000  174470          BIT     #SW10,@SWR              ;BRANCH IF NO PUBLICATION DESIRED
004310  001002                          BNE     5$
004312  004767  177034                  JSR     PC,PUBLISH              ;GO PUBLISH TEST DATA
004316  105037  001125          5$:     CLRB    @#ITCNT                 ;RESET ITERATION COUNT
004322  105777  174452                  TSTB    @SWR                    ;BRANCH IF USER DOES NOT WANT TO
004326  001721                          BEQ     1$                      ;HALT ON A SELECTED TEST
004330  017746  174444                  MOV     @SWR,-(SP)              ;GET SWITCHES
004334  042716  177740                  BIC     #177740,(SP)            ;CLEAR ALL BUT TEST #
004340  005316                          DEC     (SP)                    ;FORM TEST # -1
004342  123726  001126                  CMPB    @#TSTNUM,(SP)+          ;BRANCH IF NOT AT TEST
004346  001311                          BNE     1$
```

```
2021  004350  000000                          HALT
2022  004352  004767  175416                  JSR     PC,CKSWR                ;CHECK FOR CNTL G
2023  004356  000705                          BR      1$
2024
2025                                          .SBTTL  TIMER SUBROUTINES
2026
2027                                  ;SUBROUTINE TO SYNCHRONIZE THE TIMER AND TURN IT ON.
2028                                  ;REGISTER 4 IS CLEARED, AND THE OSCILLATOR POLARITY IS MONITORED
2029                                  ;THE ROUTINE IS EXITED WHEN THE OSCILLATOR POLARITY CHANGES WITH R3
2030                                  ;SET TO INDICATE THE POLARITY OF THE OSCILLATOR.
2031                                  ;CALL:   JSR     PC,TIMON
2032                                  ;RETURNS:       R3 SET TO INDICATE LAST POLARITY (+24/-24=0/1)
2033                                  ;               R4 = 0
2034
2035  004360  005004                  TIMON:  CLR     R4                      ;CLEAR TIME COUNT
2036  004362  012703  000024                  MOV     #24,R3                  ;SET POLARITY TO '0' STATE
2037  004366  032765  000100  000024          BIT     #OSC,MR(R5)             ;BRANCH IF POLARITY IS '0'
2038  004374  001405                          BEQ     2$
2039  004376  032765  000100  000024  1$:     BIT     #OSC,MR(R5)             ;WAIT FOR OSCILLATOR TO RETURN
2040  004404  001374                          BNE     1$
2041  004406  000405                          BR      4$
2042
2043  004410  005403                  2$:     NEG     R3                      ;NEGATE PREV POLARITY INDICATOR
2044  004412  032765  000100  000024  3$:     BIT     #OSC,MR(R5)             ;WAIT FOR OSCILLATOR TO RETURN
2045  004420  001774                          BEQ     3$                      ;TO '1' STATE
2046  004422  000207                  4$:     RTS     PC
2047
2048                                  ;SUBROUTINE TO COUNT TIME
2049                                  ;EACH TIME THE OSCILLATOR TOGGLES (BIT <06> IN MR REG) REGISTER
2050                                  ;R4 IS INCREMENTED, AND THE REGISTER R3 IS NEGATED TO INDICATE
2051                                  ;THE LAST STATE OF THE OSCILLATOR.
2052                                  ;CALL    JMP     TIMER(R3)              ;R3 IS SET BY TIMON ROUTINE
2053                                  ;        R2=RETURN ADDRESS TO CALLER
2054                                  ;NOTE:  THE TIME TO EXECUTE THIS ROUTINE IS VERY CRITICAL.  IT MUST BE
2055                                  ;LESS THAN 40 US.
2056
2057                                  ;ENTER HERE VIA JMP  TIMER(R3) WHEN R3=-24 (PREV STATE=1)
2058  004424  032765  000100  000024  TIMER1: BIT     #OSC,MR(R5)             ;BRANCH IF CURRENT STATE IS '0'
2059  004432  001406                          BEQ     TIMER                   ,GO INCREMENT TIME
2060  004434  000112                          JMP     (R2)                    ;RETURN TO TEST
2061
2062          004450                          .=TIMER1+24
2063  004450  005403                  TIMER:  NEG     R3                      ;NEGATE PREV STATE INDICATOR
2064  004452  005204                          INC     R4                      ;INCREMENT 'TICK' COUNT
2065  004454  100401                          BMI     TIMERR                  ;BRANCH ON OVERFLOW
2066  004456  000112                          JMP     (R2)                    ;RETURN TO TEST
2067  004460  000004  014766          TIMERR: TYPE,E.TIMOV                    ;TYPE 'TIMER OVERFLOWED'
2068  004464  104400                          HLT                             ;REPORT HARDWARE ERROR
2069  004466  000177  174314                  JMP     @SCPADR                 ;RETURN TO BEGINNING OF TEST
2070
2071          004474                          .=TIMER+24
2072                                  ;ENTER HERE VIA JMP  TIMER(R3) WHEN R3=+24 (PREV STATE=0)
2073  004474  032765  000100  000024  TIMER0: BIT     #OSC,MR(R5)             ;BRANCH IF CURRENT STATE = '1'
2074  004502  001362                          BNE     TIMER
2075  004504  000112                          JMP     (R2)
2076
```

```
                                ;SUBROUTINE TO CHECK TIME RECORDED BY SUBTEST.
                                ;THIS SUBROUTINE COMPUTES THE ACTUAL TIME (IN MICROSECONDS) AND CHECKS
                                ;THAT THE TIME RECORDED BY THE SUBTEST IS CORRECT BY COMPARING THE TIME
                                ;WITH THE HIGH LIMIT (STIMTBL(R0)) AND THE LOW LIMIT (STIMTBL+2(R0)).
                                ;IF THE TIME IS OUT OF RANGE AN OUT OF RANGE ERROR TYPEOUT RESULTS.
                                ;THE SUBROUTINE IS ENTERED WITH:
                                ;       R4=TICK COUNT

004506                          TIMOK:
004506  004767  176050                  JSR     PC,.SAVE                ;SAVE REGISTERS ON THE STACK
004512  012700  000070                  MOV     #56.,R0                 ;GET TIME PER TICK
004516  010401                          MOV     R4,R1                   ;GET TICKS COUNT
004520  005002                          CLR     R2                      ;CLEAR SUMMING REGISTERS
004522  005003                          CLR     R3
004524  060002                  1$:     ADD     R0,R2                   ;MULTIPLY TIME PER TICK
004526  005503                          ADC     R3                      ;BY TICK COUNT
004530  005301                          DEC     R1
004532  001374                          BNE     1$
004534  010246                          MOV     R2,-(SP)                ;DIVIDE COUNT BY 10.

004536  010346                          MOV     R3,-(SP)
004540  012746  000012                  MOV     #10.,-(SP)
004544  004767  000262                  JSR     PC,DIVIDE
004550  005726                          TST     (SP)+                   ;DISCARD REMAINDER
004552  012637  001016                  MOV     (SP)+,@#ATIME           ;STORE QUOTIENT
004556  113700  001126                  MOVB    @#TSTNUM,R0             ;GET TEST #
004562  006300                          ASL     R0
004564  006300                          ASL     R0
004566  023760  001016  001422          CMP     @#ATIME,STIMTBL(R0)     ;CHECK THAT TIME IS WITHIN
004574  101004                          BHI     2$                      ;LIMITS SPECIFIED
004576  023760  001016  001424          CMP     @#ATIME,STIMTBL+2(R0)
004604  101001                          BHI     3$
004606  104401                  2$:     HLT+1                           ;CALL ERROR ROUTINE
004610                          3$:
004610  004767  175770                  JSR     PC,.RESTORE             ;RESTORE REGISTERS FROM THE STACK
004614  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

004616                          GAPOK:
004616  004767  175740                  JSR     PC,.SAVE                ;SAVE REGISTERS ON THE STACK
004622  012700  000070                  MOV     #56.,R0                 ;GET TIME PER TICK
004626  010401                          MOV     R4,R1                   ;GET TICK COUNT
004630  005002                          CLR     R2                      ;CLEAR SUMMING REGISTERS
004632  005003                          CLR     R3
004634  060002                  1$:     ADD     R0,R2                   ;MULTIPLY TICK COUNT
004636  005503                          ADC     R3                      ;BY TIME PER TICK
004640  005301                          DEC     R1
004642  001374                          BNE     1$
```

```
2133
2134  004644  010246                          MOV     R2,-(SP)                ;DIVIDE TIME BY 10.
2135  004646  010346                          MOV     R3,-(SP)
2136  004650  012746  000012                  MOV     #10.,-(SP)
2137  004654  004767  000152                  JSR     PC,DIVIDE
2138  004660  005726                          TST     (SP)+                   ;DISCARD REMAINDER
2139  004662  012637  001016                  MOV     (SP)+,@#ATIME           ;STORE QUOTIENT
2140  004666  113703  001124                  MOVB    @#GAP,R3                ;GET GAP #
2141  004672  006303                          ASL     R3                      ;MULTPLY BY 4
2142  004674  006303                          ASL     R3                      ;TO GET AT TABLE ENTRY
2143  004676  023763  001016  001576          CMP     @#ATIME,GTIMTBL(R3)     ;CHECK TIME (MAX)
2144  004704  101004                          BHI     2$
2145  004706  023763  001016  001600          CMP     @#ATIME,GTIMTBL+2(R3)   ;CHECK TIME (MIN)
2146  004714  101002                          BHI     3$
2147  004716  104402                  2$:     HLT+2                           ;REPORT OUT OF RANGE ERROR
2148  004720  000406                          BR      100$
2149  004722  032777  000400  174050  3$:     BIT     #SW08,@SWR              ;BRANCH IF TIMES NOT WANTED
2150  004730  001402                          BEQ     100$
2151  004732  004767  176214                  JSR     PC,OUTGAP               ;TYPE GAP TIMES
2152
2153  004736                          100$:
2154  004736  004767  175642                  JSR     PC,.RESTORE             ;RESTORE REGISTERS FROM THE STACK
2155  004742  000207                          RTS     PC                              ;RETURN TO TEST
2156
2157                                  .SBTTL          DELAY SUBROUTINES
2158                          ;THIS SUBROUTINE CAUSES A DELAY OF 350 MS.
2159  004744  004767  177410          DELAY:  JSR     PC,TIMON
2160  004750  010246                          MOV     R2,-(SP)                ;SAVE R2 ON THE STACK
2161  004752  012702  004762                  MOV     #2$,R2                  ;SET RETURN ADDRESS FOR TIMER
2162  004756                          1$:
2163  004756  000163  004450                  JMP     TIMER(R3)               ;GO TO TIMER & RETURN VIA R2
2164  004762  032704  004000          2$:     BIT     #4000,R4
2165  004766  001773                          BEQ     1$
2166  004770  012602                          MOV     (SP)+,R2                ;RESTORE R2
2167  004772  000207                          RTS     PC
2168
2169                          ;THIS SUBROUTINE ALLOWS A CALLER SPECIFIED DELAY (UP TO 65MS.)
2170                          ;CALL:  MOV     DELAY TIME,DELTIM       ;LOAD DELAY TIME (IN US)
2171                          ;       JSR     PC,DELAYV
2172  004774  005767  174120  DELAYV: TST     DELTIM                  ;BRANCH IF 0 DELAY
2173  005000  001413                          BEQ     3$
2174  005002  004767  177352                  JSR     PC,TIMON                ;TURN TIMER ON
2175  005006  010246                          MOV     R2,-(SP)                ;SAVE R2 ON THE STACK
2176  005010  012702  005020                  MOV     #2$,R2                  ;SET RETURN ADDRESS FROM TIMER
2177  005014                          1$:
2178  005014  000163  004450                  JMP     TIMER(R3)               ;GO TO TIMER & RETURN VIA R2
2179  005020  023704  001120          2$:     CMP     @#DELTIM,R4
2180  005024  101373                          BHI     1$
2181  005026  012602                          MOV     (SP)+,R2                ;RESTORE R2
2182  005030  000207                  3$:     RTS     PC
2183
2184                                  .SBTTL  DIVIDE SUBROUTINE
2185                          ;THIS SUBROUTINE DIVIDES A DOUBLE PRECISION # AND RETURNS THE RESULT
2186                          ;TO THE CALLER ON THE STACK.  BOTH DIVIDEND & DIVISOR MUST BE POSITIVE.
2187                          ;CALL:  MOV     LEAST SIGNIFICANT HALF DIVIDEND,-(SP)
2188                          ;       MOV     #MOST SIGNIFICANT HALF DIVIDEND,-(SP)
```

```
                                ;       MOV     #DIVISOR,-(SP)
                                ;       JSR     PC,DIVIDE
                                ;RETURN
                                ;       (SP)=REMAINDER ON STACK
                                ;       2(SP)=QUOTIENT

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

005032  005046                  DIVIDE: CLR     -(SP)                   ;SAVE LOC FOR SIGNS
005034  012746  000021                  MOV     #17.,-(SP)              ;SET ITERATION COUNT
005040  016601  000012                  MOV     12(SP),R1               ;GET LSH DIVIDEND
005044  016600  000010                  MOV     10(SP),R0               ;GET MSH DIVIDEND
005050  016602  000006                  MOV     6(SP),R2                ;GET DIVISOR
005054  005402                          NEG     R2                      ;NEGATE DIVISOR
005056  000241                          CLC                             ;CLEAR 'C' BIT IN PSW
005060  000405                          BR      2$
005062  006100                  1$:     ROL     R0                      ;ROTATE MSH DIVIDEND
005064  010003                          MOV     R0,R3                   ;SAVE IN R3
005066  060203                          ADD     R2,R3                   ;SUBTRACT DIVISOR FROM MSH DIVIDEND
005070  103001                          BCC     2$                      ;BRANCH IF DIVIDEND > DIVISOR
005072  010300                          MOV     R3,R0                   ;SAVE REMAINDER IN R0
005074  006101                  2$:     ROL     R1                      ;ROTATE LSH DIVIDEND
005076  005316                          DEC     (SP)                    ;DECREMENT ITERATION COUNT
005100  001370                          BNE     1$
005102  005726                          TST     (SP)+                   ;POP ITERATION COUNTER
005104  005726                          TST     (SP)+                   ;POP SIGN CORRECTION
005106  010166  000006                  MOV     R1,6(SP)                ;PUSH REMAINDER ON STACK
005112  010066  000004                  MOV     R0,4(SP)                ;PUSH QUOTIENT ONTO STACK
005116  012616                          MOV     (SP)+,(SP)
```

```
2218  005120  000207                        RTS     PC
2219
2220                                        .SBTTL  DRIVE SUBROUTINES
2221                                ;SUBROUTINE TO CHECK IF DRIVE IS AVAILABLE
2222                                ;CALL:  MOVB    DRIVE#,DRVNUM
2223                                ;       JSR     PC,DRVAVA
2224                                ;RETURN:        'C' BIT SET IF NOT AVAILABLE
2225  005122  113765  001010  000010  DRVAVA: MOVB    @#DRVNUM,CS2(R5)        ;LOAD DRIVE #
2226  005130  032765  040000  000026          BIT     #TAP,DT(R5)             ;CHECK IF TAPE UNIT
2227  005136  001003                          BNE     1$
2228  005140  004767  000034                  JSR     PC,RHINIT
2229  005144  000262                          SEV                             ;SET 'V' TO IND NOT AVAIL
2230  005146  000207                  1$:     RTS     PC                      ;RETURN
2231
2232                                ;SUBROUTINE TO CHECK IF TE16 SLAVE IS AVAILABLE FOR TEST
2233                                ;CALL:  MOVB    DRIVE #,@#DRVNUM        ;PASS DRIVE # VIA DRVNUM
2234                                ;       MOVB    SLAVE #,@#SLVNUM        ;PASS SLAVE # VIA SLVNUM
2235                                ;       JSR     PC,SLVAVA               ;CALL SUBROUTINE
2236  005150  113765  001010  000010  SLVAVA: MOVB    @#DRVNUM,CS2(R5)        ;LOAD DRIVE #
2237  005156  113765  001011  000032          MOVB    @#SLVNUM,TC(R5)         ;AND SLAVE #
2238  005164  032765  002000  000026          BIT     #SPR,DT(R5)             ;BRANCH IF SLAVE PRESENT
2239  005172  001001                          BNE     1$
2240  005174  000262                          SEV                             ;SET 'V' TO INDICATE NO SLAVE
2241  005176  000207                  1$:     RTS     PC
2242
2243                                ;SUBROUTINE TO  INITIALIZE RH CONTROLLER
2244                                ;CALL:  JSR PC,RHINIT
2245
2246  005200  012765  000040  000010  RHINIT: MOV     #40,CS2(R5)
2247  005206  113765  001010  000010          MOVB    @#DRVNUM,CS2(R5)
2248  005214  005046                          CLR     -(SP)
2249  005216  113716  001011                  MOVB    @#SLVNUM,(SP)
2250  005222  012665  000032                  MOV     (SP)+,TC(R5)            ;LOAD SLAVE # INTO TC REG
2251  005226  052765  000300  000032          BIS     #NORM11,TC(R5)
2252  005234  000207                          RTS     PC
2253
2254                                ;SUBROUTINE TO WAIT FOR DRIVE READY (DRY)
2255  005236  005027                  WAITRDY: CLR    (PC)+                   ;CLEAR WAIT TIMER
2256  005240  000000                  WAITTIM: .WORD  0
2257  005242  105765  000012          1$:     TSTB    DS(R5)                  ;WAIT FOR READY TO SET
2258  005246  100406                          BMI     2$
2259  005250  005267  177764                  INC     WAITTIM                 ;INCREMENT WAIT TIMER
2260  005254  001372                          BNE     1$                      ;BRANCH IF TIME HAS NOT EXPIRED
2261  005256  000004  015013                  TYPE,E.TIMEXP                   ;TYPE 'TIME EXPIRED WAITING FOR RDY'
2262  005262  000425                          BR      99$                     ;TAKE ERROR EXIT
2263  005264  032765  002000  000012  2$:     BIT     #EOT,DS(R5)             ;CHECK FOR END OF TAPE
2264  005272  001415                          BEQ     3$                      ;BRANCH IF NO EOT
2265  005274  000004  014060                  TYPE,M.NAM
2266  005300  000004  014376                  TYPE,M.EOT                      ;TYPE 'END OF TAPE'
2267  005304  004767  000032                  JSR     PC,.REWIND              ;REWIND SLAVE
2268  005310  102412                          BVS     99$                     ;BRANCH IF ERROR ON REWIND
2269  005312  004767  000106                  JSR     PC,WRITE                ;WRITE A RECORD
2270  005316  005215                          INC     (R5)                    ;SET 'GO' BIT
2271  005320  004767  177712                  JSR     PC,WAITRDY              ;WAIT FOR READY
2272  005324  000404                          BR      99$                     ;TAKE ERROR EXIT
2273  005326  032765  040000  000012  3$:     BIT     #ERR,DS(R5)             ;CHECK ERROR EXIT
```

```
                                                        E  5
 2274  005334  001401                    BEQ     100$
 2275  005336  000262            99$:    SEV
 2276  005340  000207            100$:   RTS     PC
 2277
```

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

```
2280                                    ;       MOVB    SLAVE #,@#SLVNUM
2281                                    ;       JSR     PC,.REWIND
2282                                    ;SUBROUTINE RETURNS TO CALLER WITH SELECTED SLAVE AT 'BOT', & 'V' SET IF
2283                                    ;AN ERROR OCCURS.
2284
2285  005342  004767  177632            .REWIND: JSR    PC,RHINIT               ;INITIALIZE CONTROLLER
```

```
2286  005346  004367  000206                JSR     R3,TMCMD                ;GO TO TM COMMAND SUBROUTINE
2287  005352  000000                        .WORD   0                       ;BUS ADDRESS (NOT USED)
2288  005354  000000                        .WORD   0                       ;WORD COUNT (NOT USED)
2289  005356  000000,                       .WORD   0                       ;FRAME COUNT (NOT USED)
2290  005360  000006'                       .WORD   RWD                     ;REWIND COMMAND
2291  005362  005215                        INC     (R5)                    ;SET 'GO' BIT
2292  005364  032765  000002  000012  1$:   BIT     #BOT,DS(R5)             ;BRANCH IF 'BOT' SET
2293  005372  001005                        BNE     2$
2294  005374  032765  040000  000012        BIT     #ERR,DS(R5)             ;CHECK ERROR BIT
2295  005402  001006                        BNE     99$                     ;BRANCH IF ERROR BIT SET
2296  005404  000767                        BR      1$
2297
2298  005406  032765  020000  000012  2$:   BIT     #PIP,DS(R5)             ;WAIT FOR TAPE MOTION TO STOP
2299  005414  001374                        BNE     2$
2300  005416  000401                        BR      100$
2301  005420  000262                99$:    SEV
2302  005422  000207                100$:   RTS     PC
2303
2304                                  ;SUBROUTINE TO WRITE 256. WORD RECORD
2305                                  ;CALL:  JSR     PC,WRITE
2306
2307  005424  004367  000130        WRITE:  JSR     R3,TMCMD                ;GO TO TM COMMAND SUBROUTINE
2308  005430  016420                        .WORD   WTBUF                   ;BUS ADDRESS
2309  005432  177600                        .WORD   WRDCNT                  ;WORD COUNT
2310  005434  177400                        .WORD   FRMCNT                  ;FRAME COUNT
2311  005436  000060                        .WORD   WFWD                    ;WRITE FORWARD COMMAND
2312  005440  000207                        RTS     PC
2313
2314                                  ;SUBROUTINE TO READ A 256. WORD RECORD.
2315                                  ;CALL:  JSR     PC,READ
2316
2317  005442  004337  005560        READ:   JSR     R3,@#TMCMD
2318  005446  016420                        .WORD   RDBUF                   ;ADDRESS OF READ BUFFER
2319  005450  177600                        .WORD   WRDCNT                  ;2'S COMPLEMENT OF WORD COUNT
2320  005452  177400                        .WORD   FRMCNT                  ;2'S COMPLEMENT OF FRAME COUNT
2321  005454  000070                        .WORD   RDFWD                   ;READ FORWARD COMMAND
2322  005456  000207                        RTS     PC
2323
2324                                  ;SUBROUTINE TO INITIATE READ REVERSE COMMAND
2325                                  ;CALL:  JSR     PC,REVRD
2326                                  ;
2327  005460  004367  000074        REVRD:  JSR     R3,TMCMD
2328  005464  017020                        .WORD   RDBUF+256.              ;ADDRESS OF READ REVERSE BUFFER
2329  005466  177600                        .WORD   WRDCNT                  ;2'S COMPLEMENT OF WORD COUNT
2330  005470  177400                        .WORD   FRMCNT                  ;2'S COMPLEMENT OF FRAME COUNT
2331  005472  000076                        .WORD   RDREV                   ;READ REVERSE COMMAND
2332  005474  000207                        RTS     PC
2333
2334                                  ;SUBROUTINE TO SPACE FORWARD 1 RECORD
2335  005476  012765  177777  000006  FWDSPC: MOV     #-1,FC(R5)              ;LOAD RECORD COUNT
2336  005504  012715  000031                MOV     #SPCFWD+1,(R5)          ;LOAD COMMAND
2337  005510  004767  177522                JSR     PC,WAITRDY              ;WAIT FOR READY
2338  005514  000207                        RTS     PC                      ;RETURN
2339
2340                                  ;SUBROUTINE TO WRITE A RECORD AND BACK SPACE OVER  THE RECORD.
2341  005516  004767  177702        WRT.BK: JSR     PC,WRITE                ;WRITE THE RECORD
```

```
2342  005522  005215                          INC     (R5)                    ;SET 'GO' BIT
2343  005524  004767  177506                  JSR     PC,WAITRDY
2344  005530  102412                          BVS     2$
2345  005532  012765  177777  000006          MOV     #-1,FC(R5)              ;LOAD RECORD COUNT
2346  005540  012715  000033                  MOV     #SPCREV+1,(R5)          ;LOAD COMMAND
2347  005544  004767  177466                  JSR     PC,WAITRDY
2348  005550  102402                          BVS     2$
2349  005552  004767  177166          1$:     JSR     PC,DELAY                ;WAIT FOR TAPE MOTION TO STOP
2350  005556  000207                  2$:     RTS     PC
2351
2352                                  ;SUBROUTINE TO LOAD A COMMAND
2353                                  ;CALL:  JSR     R3,TMCMD
2354                                  ;       .WORD   BUS ADDRESS
2355                                  ;       .WORD   WORD COUNT (2'S COMPLEMENT)
2356                                  ;       .WORD   FRAME COUNT (2'S COMPLEMENT)
2357                                  ;       .WORD   COMMAND
2358
2359  005560  012365  000004          TMCMD:  MOV     (R3)+,BA(R5)            ;LOAD BUS ADDRESS
2360  005564  012365  000002                  MOV     (R3)+,WC(R5)            ;LOAD WORD COUNT
2361  005570  012365  000006                  MOV     (R3)+,FC(R5)            ;LOAD FRAME COUNT
2362  005574  012315                          MOV     (R3)+,(R5)              ;LOAD COMMAND
2363  005576  000203                          RTS     R3                      ;RETURN
2364
2365                                          ;SUBROUTINE TO PRINT TE16 SERIAL NUMBER
2366                                          ;JSR    PC,SNPT
2367
2368  005600  016503  000030          SNPT:   MOV     SN(R5),R3
2369  005604  012701  001150                  MOV     #ODIGITS,R1
2370  005610  000303                          SWAB    R3
2371  005612  006003                          ROR     R3
2372  005614  006003                          ROR     R3
2373  005616  006003                          ROR     R3
2374  005620  006003                          ROR     R3                      ;GET FIRST DIGIT
2375  005622  042703  177760                  BIC     #177760,R3
2376  005626  052703  000260                  BIS     #260,R3
2377  005632  110321                          MOVB    R3,(R1)+                ;FILL FIRST DIGIT
2378  005634  016503  000030                  MOV     SN(R5),R3
2379  005640  000303                          SWAB    R3
2380  005642  042703  177760                  BIC     #177760,R3
2381  005646  052703  000260                  BIS     #260,R3
2382  005652  110321                          MOVB    R3,(R1)+                ;GET SECOND DIGIT
2383  005654  016503  000030                  MOV     SN(R5),R3
2384  005660  006003                          ROR     R3
2385  005662  006003                          ROR     R3
2386  005664  006003                          ROR     R3
2387  005666  006003                          ROR     R3
2388  005670  042703  177760                  BIC     #177760,R3
2389  005674  052703  000260                  BIS     #260,R3
2390  005700  110321                          MOVB    R3,(R1)+                ;GET THIRD DIGIT
2391  005702  016503  000030                  MOV     SN(R5),R3
2392  005706  042703  177760                  BIC     #177760,R3
2393  005712  052703  000260                  BIS     #260,R3
2394  005716  110321                          MOVB    R3,(R1)+                ;GET FOURTH DIGIT
2395  005720  105011                          CLRB    (R1)
2396  005722  000004  001150                  TYPE,ODIGITS                    ;TYPE SERIAL NUMBER
2397  005726  000207                          RTS     PC                      ;RETURN
```

```
                                                                J 5
2398
```

```
                                        .SBTTL  PROGRAM INITIAL!ZATION
005730  012706  000600          INIT:   MOV     #STKPTR,SP              ;SET STACK PTR
                                ;       ******* MODIFIED 11-8-77 **********************************
                                ;               DIAGNOSTIC SETUP FOR EXECUTION
                                ;               UNDER ACT11.
                                ;       +
                                ;       +
005734  004767  005732                  JSR     PC,CKMODE               ;CHECK FOR MODE OF OPERATION
005740  005767  173020                  TST     AUTOM                   ;AUTOMATIC MODE?
005744  001001                          BNE     1$                      ;BRANCH - IF YES
005746  000410                          BR      SUSWR                   ;CHECK SWITCH REGISTER IN DUMP MODE
005750  032737  020000  000052  1$:     BIT     #20000,@#52             ;MANUAL INTERVENTION?
005756  001404                          BEQ     SUSWR                   ;BRANCH - IF NO
005760  000004  014036                  TYPE,MSGB                       ;TYPE ABORT MESSAGE
005764  000167  005774                  JMP     ABORT                   ;AND ABORT THE PROGRAM
                                ;       ****************************************************************
005770  013746  000006          SUSWR:  MOV     @#6,-(SP)               ;SAVE VECTORS
005774  013746  000004                  MOV     @#4,-(SP)
006000  012737  006020  000004          MOV     #61$,@#4                        ;SET UP FOR TIMEOUT
006006  022777  177777  172764          CMP     #-1,@SWR                ;REFERENCE HARDWARE SWITCH REGISTER
006014  001402                          BEQ     60$
006016  000404                          BR      62$
006020  022626                  61$:    CMP     (SP)+,(SP)+             ;ADJUST STACK
006022  012767  000176  172750  60$:    MOV     #SWREG,SWR              ;POINT TO SOFTWARE SWITCH REG
006030  012637  000004          62$:    MOV     (SP)+,@#4               ;RESTORE VECTORS
006034  012637  000006                  MOV     (SP)+,@#6
006040  022737  000176  001000          CMP     #SWREG,@#SWR
006046  001002                          BNE     64$
006050  004767  173760                  JSR     PC,CNTLU
006054  105037  001130          64$:    CLRB    @#PRGFLG                ;CLEAR PROGRAM FLAG
006060  105037  001125                  CLRB    @#ITCNT                 ,CLEAR ITERATION COUNT
006064  105037  001126                  CLRB    @#TSTNUM                ;SET TEST # 0
006070  105037  001127                  CLRB    @#ERFLG                 ;CLEAR ERROR FLAG
006074  105067  173034                  CLRB    ASFLG                   ;CLEAR ASK FLAG
006100  012737  000006  000004          MOV     #ERRVEC+2,@#ERRVEC
006106  012737  000002  000006          MOV     #RTI,@#ERRVEC+2         ;CHECK IF 'LP' IS AVAILABLE
006114  005037  001270          2$:     CLR     @#INBUF
                                ;       ********* MODIFIED 11-8-77 ******************************
                                ;               IF IN ACT11 MODE INHIBIT TYPING PROGRAM
                                ;               IDENTIFICATION AND MANUAL INTERVENTION
006120  005767  172642                  TST     ACT11M                  ;AUTOMATIC MODE?
006124  001402                          BEQ     7$                      ;BRANCH - IF NO
006126  000167  000662                  JMP     SIZE                    ;START THERE
                                ;       ****************************************************************
```

```
2455
2456  006132                          7$:
2457
2458  006132  000004  001400                  TYPE,CRLF
2459  006136  000004  014060                  TYPE,M.NAM                      ;TYPE TITLE
2460  006142  000004  014116                  TYPE,I.REG                      ;ASK USER TO TYPE CONT BASE ADRS
2461  006146  004767  175326                  JSR     PC,.INPUT               ;GET USER INPUT
2462  006152  004767  175106            4$:   JSR     PC,CNVTAO               ;CONVERT ASCII TO OCTAL
2463  006156  013737  001122  001014          MOV     @#OCTALO,@#TMBASE       ;SET NEW ADDRESS
2464  006164  013705  001014            5$:   MOV     @#TMBASE,R5
2465
2466                                  ;ROUTINE TO CHECK IF CONTROLLER (RH11) IS AVAILAABLE
2467  006170  000261                          SEC                             ;SET 'C' IN PSW
2468  006172  005715                          TST     (R5)                    ;BRANCH IF CONTROLLER AVAIL
2469  006174  103003                          BCC     6$
2470  006176  000004  014435                  TYPE,E.NCON
2471  006202  000652                          BR      INIT
2472  006204  012737  003654  000004    6$:   MOV     #ERRTRP,@#ERRVEC        ;SET ERROR TRAP VECTOR
```

```
                                        ;ROUTINE TO GET TM02 DRIVES USER DESIRES TO TEST
006212  105037  001127      DRIVES: CLRB    @#ERFLG                 ;CLEAR ERROR FLAG
006216  012701  001160              MOV     #DRVTBL,R1              ;MARK ALL DRIVES AS NOT TO
006222  012700  000004              MOV     #4,R0                   ;BE TESTED. A '0' INDICATES
006226  005021              1$:     CLR     (R1)+                   ;THAT A DRIVE IS NOT TO BE
006230  005300                      DEC     R0                      ;TESTED
006232  001375                      BNE     1$
006234  000004  014163              TYPE,I.DRVS
006240  004767  175234              JSR     PC,.INPUT               ;GET USER INPUT
006244  012700  001270              MOV     #INBUF,R0
006250  122710  000101              CMPB    #'A,(R0)                ;AN 'A' SPECIFIES ALL
006254  001013                      BNE     3$                      ;DRIVES TO BE TESTED
006256  110667  172646              MOVB    SP,PRGFLG               ;SET FLAG TO IND ALL DRIVES
006262  012701  001160              MOV     #DRVTBL,R1              ;MARK ALL DRIVES TO BE TESTED
006266  012700  000004              MOV     #4,R0                   ;A '-1' INDICATES THAT A DRIVE
006272  012721  177777      2$:     MOV     #-1,(R1)+               ;IS TO BE TESTED
006276  005300                      DEC     R0
006300  001374                      BNE     2$
006302  000417                      BR      CHKDRV                  ;GO CHECK DRIVE AVAILABILITY

                                        ;GET USER SELECTED DRIVES AND MARK EACH DRIVE SELECTED TO BE TESTED
006304  122710  000015      3$:     CMPB    #CR,(R0)
006310  001414                      BEQ     CHKDRV
006312  121027  000054              CMPB    (R0),#',                ;CHECK IF 'COMMA'
006316  001001                      BNE     4$
006320  105720                      TSTB    (R0)+                   ;STEP PTR PAST 'COMMA'
006322  112001              4$:     MOVB    (R0)+,R1
006324  042701  177770              BIC     #177770,R1
006330  112761  177777  001160      MOVB    #-1,DRVTBL(R1)
006336  000240                      NOP
006340  000761                      BR      3$

                                        ;ASCERTAIN THAT DRIVES (TM02'S) SPECIFIED ARE AVAILABLE
006342  005000      CHKDRV: CLR     R0                              ;A 0/-1 INDICATES THAT THE
006344  105760  001160      1$:     TSTB    DRVTBL(R0)              ;DRIVE IS NOT/IS TO BE TESTED
006350  001005                      BNE     3$
006352  005200              2$:     INC     R0
006354  122700  000010              CMPB    #8.,R0
006360  001371                      BNE     1$
006362  000421                      BR      4$
006364  110037  001010      3$:     MOVB    R0,@#DRVNUM
006370  004737  005122              JSR     PC,@#DRVAVA             ;CHECK IF AVAILABLE
006374  102366                      BVC     2$                      ;'V' BIT SET INDICATES NOT AVAIL
006376  000004  014502              TYPE,E.NDRV
006402  116037  001136  014534      MOVB    DIGTAB(R0),@#E.NAVA     ;SET DRIVE # IN MESSAGE
006410  000004  014534              TYPE,E.NAVA
006414  110637  001127              MOVB    SP,@#ERFLG              ;SET 'ERROR' FLAG
006420  105060  001160              CLRB    DRVTBL(R0)              ;MARK DRIVE UNAVAILABLE
006424  000752                      BR      2$                      ;CHECK NEXT DRIVE
006426  105737  001127      4$:     TSTB    @#ERFLG                 ;GO GET SLAVES IF NO ERROR
006432  001403                      BEQ     SLAVES
006434  105737  001130              TSTB    @#PRGFLG                ;ASK USER TO RETYPE DRIVES IF
006440  001664                      BEQ     DRIVES                  ;'ALL' NOT SPECIFIED

                                        ;ROUTINE TO GET SLAVES (TE16'S) USER DESIRES TO TEST
006442  105037  001127      SLAVES: CLRB    @#ERFLG                 ;CLEAR 'ERROR' FLAG
```

```
2529  006446  012701  001170          MOV     #SLVTBL,R1              ;MARK ALL SLAVES (64.) AS NOT
2530  006452  012700  000040          MOV     #32.,R0                 ;TO BE TESTED. A 0 INDICATES THAT
2531  006456  005021          1$:     CLR     (R1)+                   ;A DRIVE'S SLAVE IS NOT TO BE
2532  006460  005300                  DEC     R0                      ;TESTED
2533  006462  001375                  BNE     1$
2534  006464  005000                  CLR     R0                      ;R0 = DRIVE # FOR SLAVES
2535  006466  012701  001170          MOV     #SLVTBL,R1              ;R1 POINTS TO DRIVE'S SLAVE
2536  006472  105760  001160  2$:     TSTB    DRVTBL(R0)              ;IF DRIVE IS TO BE TESTED
2537  006476  001007                  BNE     4$                      ;GO TO 4$ OTHERWISE
2538  006500  062701  000010  3$:     ADD     #8.,R1                  ;STEP SLAVE PTR TO NEXT DRIVE'S
2539  006504  005200                  INC     R0                      ;SLAVES AND INCREMENT DRIVE #
2540  006506  122700  000010          CMPB    #8.,R0                  ;CHECK ALL DRIVES
2541  006512  001367                  BNE     2$                      ;AND WHEN ALL DRIVES CHECKED
2542  006514  000454                  BR      CHKSLV                  ;GO CHECK SLAVE AVAILABILITY
2543
2544  006516  105737  001130  4$:     TSTB    @#PRGFLG                ;BRANCH IF USER SELECTED ALL
2545  006522  001020                  BNE     5$                      ;DRIVES
2546  006524  110067  172260          MOVB    R0,DRVNUM               ;GET DRIVE #
2547  006530  116037  001136  014244  MOVB    DIGTAB(R0),@#I.DRV      ;PREPARE USER ACTION MESSAGE
2548  006536  000004  014225          TYPE,I.SLVS
2549  006542  004767  174732          JSR     PC,.INPUT               ;GET USER INPUT
2550  006546  012703  001270          MOV     #INBUF,R3               ;SET PTR TO USER INPUT
2551  006552  122710  000101          CMPB    #'A,(R0)                ;BRANCH IF USER DOES NOT WANT
2552  006556  001015                  BNE     7$                      ;'ALL' SLAVES
2553  006560  110637  001130          MOVB    SP,@#PRGFLG             ;SET 'ALL' INDICATOR
2554  006564  012701  001170  5$:     MOV     #SLVTBL,R1              ;MARK ALL SLAVES FOR ALL
2555  006570  012700  000040          MOV     #32.,R0                 ;DRIVES AS TO BE TESTED
2556  006574  012721  177777  6$:     MOV     #-1,(R1)+
2557  006600  005300                  DEC     R0
2558  006602  001374                  BNE     6$
2559  006604  105737  001130          TSTB    @#PRGFLG                ;BRANCH IF ALL WAS SELECTED
2560  006610  001016                  BNE     CHKSLV
2561
2562  006612  122713  000015  7$:     CMPB    #CR,(R3)                ;GET USER SELECTED SLAVES FOR
2563  006616  001730                  BEQ     3$                      ;DRIVE
2564  006620  121327  000054          CMPB    (R3),#',                ;STEP PTR PAST 'COMMA'
2565  006624  001001                  BNE     8$
2566  006626  105723                  TSTB    (R3)+
2567  006630  112304  8$:     MOVB    (R3)+,R4                ,AND MARK SELECED SLAVE
2568  006632  042704  177770          BIC     #177770,R4              ;AS TO BE TESTED
2569  006636  060104                  ADD     R1,R4
2570  006640  112714  177777          MOVB    #-1,(R4)
2571  006644  000762                  BR      7$
2572
2573                          ;ASCERTAIN THAT SLAVES (TE16'S) SELECTED ARE AVAILABLE
2574  006646  005000  CHKSLV: CLR     R0                      ;R0 WILL CONTAIN THE DRIVE #
2575  006650  005001                  CLR     R1                      ;AND R1 THE SLAVE #
2576  006652  012702  001170          MOV     #SLVTBL,R2              ;SET PTR TO SLAVE TABLE
2577  006656  105760  001160  1$:     TSTB    DRVTBL(R0)              ;BRANCH IF DRIVE SELECTED
2578  006662  001007                  BNE     3$                      ;& AVAILABLE FOR TEST
2579  006664  005200  2$:     INC     R0                      ;INCREMENT DRIVE #
2580  006666  062702  000010          ADD     #8.,R2                  ;STEP SLAVE PTR TO NEXT DRIVE'S
2581  006672  022700  000010          CMP     #8.,R0                  ;SLAVES. BRANCH TO 1$ IF NOT ALL
2582  006676  001367                  BNE     1$                      ;DRIVES CHECKED OTHERWISE EXIT
2583  006700  000434                  BR      7$
2584
```

```
2585  006702  005001                  3$:     CLR     R1                      ;SET SLAVE # 0
2586  006704  105712                  4$:     TSTB    (R2)                    ;BRANCH IF DRIVE'S SLAVE IS SEL-
2587  006706  001006                          BNE     6$                      ;ECTED FOR TEST
2588  006710  005201                  5$:     INC     R1                      ;INCREMENT SLAVE #
2589  006712  005202                          INC     R2                      ;STEP PTR TO NEXT SLAVE
2590  006714  022701  000010                  CMP     #8.,R1                  ;GO TO 4$ IF ALL SLAVES NOT
2591  006720  001371                          BNE     4$                      ;CHECKED
2592  006722  000760                          BR      2$                      ;OTHERWISE GO TO 2$ ABOVE
2593
2594  006724  110037  001010          6$:     MOVB    R0,@#DRVNUM             ;PASS DRIVE & SLAVE #
2595  006730  110137  001011                  MOVB    R1,@#SLVNUM
2596  006734  004737  005150                  JSR     PC,@#SLVAVA             ;AND CHECK IF AVAILABLE
2597  006740  102363                          BVC     5$                      ;'V' BIT SET ON RETURN IND-
2598  006742  116037  001136  014524          MOVB    DIGTAB(R0),@#E.DRV      ;ICATES ERROR.  PREPARE ERROR
2599  006750  116137  001136  014534          MOVB    DIGTAB(R1),@#E.NAVA     ;MESSAGE
2600  006756  000004  014516                  TYPE,E.NSLV
2601  006762  110637  001127                  MOVB    SP,@#ERFLG              ;SET ERROR INDICATOR
2602  006766  105012                          CLRB    (R2)                    ;CLEAR SLAVE TABLE ENTRY
2603  006770  000747                          BR      5$                      ;GET NEXT SLAVE
2604
2605  006772  105737  001127          7$:     TSTB    @#ERFLG                 ;BRANCH IF NO ERROR
2606  006776  001403                          BEQ     100$
2607  007000  105737  001130                  TSTB    @#PRGFLG                ;BRANCH IF NOT 'ALL'
2608  007004  001616                          BEQ     SLAVES                  ;ASK USER TO RETYPE SLAVES
2609  007006  012737  003654  000004  100$:   MOV     #ERRTRP,@#ERRVEC
2610
2611  007014                          SIZE:
2612
2613                                  ;SCAN DRIVE AND SLAVE TABLE FOR DRIVE/SLAVE COMBINATION TO TEST
2614
2615  007014  105037  001010                  CLRB    @#DRVNUM                ;SET DRIVE AND SLAVE # 0
2616  007020  105037  001011                  CLRB    @#SLVNUM
2617  007024  012737  001170  001012          MOV     #SLVTBL,@#SLVPTR        ;SET PTR TO SLAVE TABLE
2618  007032  105037  001131                  CLRB    @#UNTFND                ;CLEAR 'UNIT FOUND' IND.
2619
2620
2621                                  ;       ************************************************************
2622
2623                                  ;               MACRO FOR SIZING DRIVES AND SLAVES
2624                                  ;               IN AUTOMATIC MODE.
2625
2626                                  ;       +
2627
2628                                  ;       +
2629
2630  007036  012767  177777  171744          MOV     #-1,DRVNUM              ;INIT DRIVE NUMBER
2631  007044  012767  177777  171737  NXTDRV: MOV     #-1,SLVNUM              ;INIT SLAVE NUMBER
2632  007052  012777  000040  170730  1$:     MOV     #40,@CS2                ;INIT CONTROLLER
2633  007060  005267  171724                  INC     DRVNUM                  ;STEP DRIVE #
2634  007064  022767  000010  171716          CMP     #10,DRVNUM              ;ALL DRIVE TESTED?
2635  007072  001002                          BNE     2$                      ;BRANCH - IF NO
2636  007074  000167  003746                  JMP     FINISH                  ;EXIT
2637  007100  016777  171704  170702  2$:     MOV     DRVNUM,@CS2             ;LOAD DRIVE #
2638  007106  005777  170666                  TST     @CS1                    ;ACCESS DRIVE
2639  007112  032777  010000  170670          BIT     #10000,@CS2             ;NON-EXISTANT DRIVE?
2640  007120  001354                          BNE     1$                      ;BRANCH - IF YES
```

```
 2641  007122  005267  171663          NXTSLV:  INC     SLVNUM                  ;STEP SLAVE #
 2642  007126  001010                           BNE     1$                      ;BRANCH IF NOT SLAVE 0
 2643  007130  005767  171654                   TST     DRVNUM                  ;DRIVE 0?
 2644  007134  001005                           BNE     1$                      ;BRANCH - IF NO
 2645  007136  105767  171625                   TSTB    XXDPM                   ;XXDP?
 2646  007142  001402                           BEQ     1$                      ;BRANCH - IF NO
 2647  007144  005267  171641                   INC     SLVNUM                  ;STEP TO SLAVE #1
 2648  007150  022767  000010  171633  1$:      CMP     #10,SLVNUM              ;ALL DRIVES TESTED?
 2649  007156  001732                           BEQ     NXTDRV                  ;BRANCH - IF YES
 2650  007160  016777  171625  170644          MOV     SLVNUM,@TC              ;LOAD SLAVE UNIT?
 2651  007166  032777  002000  170632          BIT     #2000,@DT               ;SLAVE PRESENT?
 2652  007174  001752                           BEQ     NXTSLV                  ;BRANCH - IF NO (SPR=0)
 2653  007176  032777  140000  170622          BIT     #140000,@DT             ;IS DRIVE A TAPE UNIT?
 2654  007204  001746                           BEQ     NXTSLV                  ;BRANCH IF NO
 2655
 2656                                  ;       ************************************************************
 2657
 2658  007206  113700  001010          BEGIN:  MOVB    @#DRVNUM,R0             ;GET DRIVE #
 2659  007212  113701  001011                  MOVB    @#SLVNUM,R1             ;AND SLAVE #
```

```
2660  007216  013702  001012          MOV     @#SLVPTR,R2             ;GET SLAVE PTR
2661  007222  105760  001160  1$:     TSTB    DRVTBL(R0)              ;BRANCH IF DRIVE AVAIL TO TEST
2662  007226  001011                  BNE     3$
2663  007230  005001                  CLR     R1                      ;CLEAR SLAVE #
2664  007232  062702  000010          ADD     #8.,R2                  ;AND STEP PTR TO NEXT DRIVE'S
2665  007236  005200          2$:     INC     R0                      ;SLAVES AND INCREMENT DRIVE #
2666  007240  022700  000010          CMP     #8.,R0                  ;EXIT TEST IF ALL DRIVES
2667  007244  001366                  BNE     1$                      ;CHECKED OTHERWISE CONTINUE
2668  007246  000137  013162          JMP     @#END                   ;SCAN FOR NEXT 'UNIT'
2669
2670  007252  105712          3$:     TSTB    (R2)                    ;BRANCH IF SLAVE ON DRIVE IS
2671  007254  001007                  BNE     4$                      ;AVAILABLE THERWISE STEP
2672  007256  005202                  INC     R2                      ;PTR TO NEXT SLAVE
2673  007260  005201                  INC     R1                      ;INCREMENT SLAVE #
2674  007262  122701  000010          CMPB    #8.,R1                  ;UNTIL ALL SLAVES CHECKED
2675  007266  001371                  BNE     3$                      ;WHEN ALL SLAVES CHECKED
2676  007270  005001                  CLR     R1                      ;SET SLAVE # 0
2677  007272  000761                  BR      2$                      ;AND CONTINUE SCAN
2678
2679  007274  110637  001131  4$:     MOVB    SP,@#UNTFND             ;INDICATE THAT A 'UNIT' IS FOUND
2680  007300  110037  001010          MOVB    R0,@#DRVNUM             ;SET DRIVE 3
```

```
 2681  007304  110137  001011                  MOVB    R1,@#SLVNUM             ;SET SLAVE #
 2682  007310  010237  001012                  MOV     R2,@#SLVPTR             ;SAVE SLAVE PTR
 2683
 2684  007314  105737  001134          5$:     TSTB    @#ASFLG
 2685  007320  001047                          BNE     7$
 2686  007322  112767  000001  171604          MOVB    #1,ASFLG
 2687  007330  105037  001130                  CLRB    @#PRGFLG        ;CLEAR PROGRAM INDICATOR
 2688
 2689                                  ;       ********* MODIFIED 8-11-77 ******************************
 2690
 2691                                  ;               CHECK FOR AUTOMATIC MODE
 2692
 2693  007334  005767  171424                  TST     AUTOM           ;AUTOMATIC MODE?
 2694  007340  001037                          BNE     TYPHDR          ;BRANCH - IF YES
 2695
 2696                                  ;       ************************************************************
 2697
 2698  007342  000004  014304                  TYPE,I.SKEW             ;ASK USER IF HE WANTS TO RUN SKEW TESTS
 2699  007346  004767  174126                  JSR     PC,.INPUT               ;GET USER INPUT
 2700  007352  012703  001270                  MOV     #INBUF,R3       ;GET REPLY
 2701  007356  122713  000060                  CMPB    #'0,(R3)        ;BRANCH IF 'NO' (0)
 2702  007362  001406                          BEQ     6$
 2703  007364  122713  000061                  CMPB    #'1,(R3)        ;CHECK IF 'YES' (1)
 2704  007370  001351                          BNE     5$              ;NEITHER SO ASK AGAIN
 2705  007372  111337  001130                  MOVB    (R3),@#PRGFLG   ;SET INDICATOR
 2706  007376  000420                          BR      7$
 2707
 2708  007400  105037  001133          6$:     CLRB    @#NRZFLG        ;CLEAR NRZ INDICATOR
 2709  007404  000004  014345                  TYPE,I.NRZ              ;ASK USER IF DRIVE 'NRZ' ONLY
 2710  007410  004767  174064                  JSR     PC,.INPUT               ;GET USER INPUT
 2711  007414  012703  001270                  MOV     #INBUF,R3       ;GET REPLY
 2712  007420  122713  000060                  CMPB    #'0,(R3)        ;BRANCH IF 'NO' (0)
 2713  007424  001405                          BEQ     7$
 2714  007426  122713  000061                  CMPB    #'1,(R3)        ;CHECK IF 'YES' (1)
 2715  007432  001362                          BNE     6$              ;ASK AGAIN IF NEITHER
 2716  007434  111337  001133                  MOVB    (R3),@#NRZFLG   ;SET INDICATOR
 2717
 2718  007440                          7$:
 2719
 2720  007440                          TYPHDR:
 2721
 2722  007440  052737  000100  177560          BIS     #100,@#TKS              ;SET KEYBOARD IE BIT
 2723  007446  000004  015064                  TYPE,L.HDR1
 2724  007452  116037  001136  015244          MOVB    DIGTAB(R0),@#L.DRV      ;SET DRIVE #
 2725  007460  116137  001136  015256          MOVB    DIGTAB(R1),@#L.SLV      ;AND SLAVE #
 2726  007466  112737  000071  015261          MOVB    #'9,@#L.CHAN            ;GET SLAVES CHANNEL TYPE
 2727  007474  032765  010000  000026          BIT     #CH7,DT(R5)
 2728  007502  001403                          BEQ     1$
 2729  007504  112737  000067  015261          MOVB    #'7,@#L.CHAN            ;SET 7 CHANNEL
 2730  007512  000004  015177          1$:     TYPE,L.HDR2
 2731  007516  004767  176056                  JSR     PC,SNPT         ;GO PRINT SERIAL NUMBER
 2732  007522  000004  015300                  TYPE,L.HDR3
 2733  007526  012737  007566  001006          MOV     #TST001,@#SCPADR        ;SET 'SCOPE' ADDRESS FOR FIRST TEST
 2734  007534  010500                          MOV     R5,R0
 2735  007536  062700  000006                  ADD     #FC,R0                  ;R0 CONTAINS ADDRESS OF FC REG
 2736  007542  010501                          MOV     R5,R1
```

```
 2737  007544  062701  000012                ADD     #DS,R1                  ;R1 CONTAINS ADDRESS OF DS REG
 2738  007550  012703  004450                MOV     #TIMER,R3               ;SET JUMP ADDRESS TO TIMER
 2739  007554  105737  001130                TSTB    @#PRGFLG                ;BRANCH IF NOT SKEW TESTS
 2740  007560  001402                        BEQ     TST001
 2741  007562  000137  013326                JMP     @#SKEWTST
```

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

                                ;THIS TEST MEASURES TIME FROM 'GO'=1 TO 'ACCL'=0.
007566  112737  000001  001126  TST001: MOVB    #1,@#TSTNUM             ;SET TEST #
007574  012702  007620          MOV     #1$,R2                  ;SET RETURN PC FROM TIMER
007600  004767  175536          JSR     PC,.REWIND              ;REWIND SLAVE
007604  102420                  BVS     99$                     ;BRANCH IF ERROR ON REWIND
007606  004767  175612          JSR     PC,WRITE                ;GO SETUP WRITE COMMAND
007612  004767  174542          JSR     PC,TIMON                ;TURN TIMER ON
007616  005215                  INC     (R5)                    ;SET 'GO' BIT

007620  005765  000032          1$:     TST     TC(R5)                  ;BRANCH WHEN 'ACCL'=0
007624  100002                  BPL     2$
007626  000163  004450          JMP     TIMER(R3)               ;GO TO TIMER & RETURN VIA R2

007632  004767  175400          2$:     JSR     PC,WAITRDY              ;WAIT FOR COMMAND TO FINISH
007636  102403                  BVS     99$                     ;BRANCH IF ERROR
007640  004767  174642          JSR     PC,TIMOK                ;GO CHECK TIME
007644  000401                  BR      100$
007646  104400                  99$:    HLT
007650  104000                  100$:   SCOPE

                                ;TEST 002 - WRITE START
                                ;THIS TST MEASURES TIME FROM 'GO'=1 TO 'ACCL'=0.
007652  112737  000002  001126  TST002: MOVB    #2,@#TSTNUM             ;SET TEST # 2
007660  004767  175540          JSR     PC,WRITE                ;INITIATE WRITE COMMAND
007664  012702  007676          MOV     #1$,R2                  ;SET RETURN PC FROM TIMER
007670  004767  174464          JSR     PC,TIMON
007674  005215                  INC     (R5)                    ;SET 'GO' BIT

007676  005765  000032          1$:     TST     TC(R5)                  ;BRANCH WHEN 'ACCL'=0
007702  100002                  BPL     2$
007704  000163  004450          JMP     TIMER(R3)               ;GO TO TIMER & RETURN VIA R2

007710  004767  175322          2$:     JSR     PC,WAITRDY              ,WAIT FOR READY
007714  102403                  BVS     99$                     ;BRANCH IF ERROR
007716  004767  174564          JSR     PC,TIMOK                ;GO CHECK TIME RECORDED
007722  000401                  BR      100$                    ;EXIT VIA SCOPE

007724  104400                  99$:    HLT                     ;REPORT ERROR
007726  104000                  100$:   SCOPE

                                ;TEST 003- WRITE SHUTDOWN
                                ;THIS TEST MEASURES TIME FROM 'FC REG'=0 TO 'SWDN'=1.
007730  112737  000003  001126  TST003: MOVB    #3,@#TSTNUM             ;SET TEST#3
007736  004767  175462          JSR     PC,WRITE                ;INITIATE WRITE COMMAND
007742  005215                  INC     (R5)                    ;SET 'GO' BIT

007744  005710                  1$:     TST     (R0)                    ;BRANCH WHEN WRITING FINISHED
007746  001404                  BEQ     2$
007750  032711  040000          BIT     #ERR,(R1)               ;MONITOR ERROR BIT
007754  001017                  BNE     99$
```

```
2798  007756  000772                          BR      1$
2799
2800  007760                          2$:
2801  007760  004767  174374                  JSR     PC,TIMON                ;TURN TIMER ON
2802  007764  010702                          MOV     PC,R2                   ;LOAD RETURN PC FROM TIMER
2803  007766  032711  000020          3$:     BIT     #SDWN,(R1)              ;BRANCH WHEN DS <SDWN> SETS
2804  007772  001002                          BNE     4$
2805  007774  000163  004450                  JMP     TIMER(R3)               ;GO TO TIMER & RETURN VIA R2
2806
2807  010000  004767  175232          4$:     JSR     PC,WAITRDY              ;WAIT FOR READY
2808  010004  102403                          BVS     99$
2809  010006  004767  174474                  JSR     PC,TIMOK                ;GO CHECK TIME RECORDED
2810  010012  000401                          BR      100$
2811  010014  104400                  99$:    HLT                             ;REPORT ERROR
2812  010016  104000                  100$:   SCOPE
2813
2814                                  ;TEST 004 - WRITE SETTLEDOWN
2815                                  ;THIS TEST MEASURES TIME FROM 'SWDN'=1 TO 'SWDN'=0.
2816  010020  112737  000004  001126 TST004: MOVB    #4,@#TSTNUM
2817  010026  004767  175372                  JSR     PC,WRITE
2818  010032  005215                          INC     (R5)                    ;SET 'GO' BIT
2819
2820  010034  005710                  1$:     TST     (R0)                    ;BRANCH WHEN WRITING FINISHED
2821  010036  001404                          BEQ     2$
2822  010040  032711  040000                  BIT     #ERP,(R1)               ;CHECK ERROR BIT
2823  010044  001026                          BNE     99$
2824  010046  000772                          BR      1$
2825
2826  010050  032711  000020          2$:     BIT     #SDWN,(R1)              ;WAIT FOR ASSERTION OF 'SDWN'
2827  010054  001004                          BNE     3$
2828  010056  032711  040000                  BIT     #ERR,(R1)               ;MONITOR ERROR BIT
2829  010062  001017                          BNE     99$
2830  010064  000771                          BR      2$
2831
2832  010066                          3$:
2833  010066  004767  174266                  JSR     PC,TIMON                ;TURN TIMER ON
2834  010072  010702                          MOV     PC,R2                   ;SET RETURN PC FROM TIMER
2835  010074  032711  000020                  BIT     #SDWN,(R1)              ;BRANCH WHEN SWDN CLEARS
2836  010100  001402                          BEQ     5$
2837  010102  000163  004450                  JMP     TIMER(R3)               ;GO TO TIMER & RETURN VIA R2
2838
2839  010106  004767  175124          5$:     JSR     PC,WAITRDY              ;WAIT FOR READY
2840  010112  102403                          BVS     99$
2841  010114  004767  174366                  JSR     PC,TIMOK
2842  010120  000401                          BR      100$
2843
2844  010122  104400                  99$:    HLT
2845  010124  104000                  100$:   SCOPE
2846
2847                                  ;TEST 005 - READ FROM BOT
2848                                  ;THIS TEST MEASURES TIME FROM 'GO'=1 TO 'ACCL'=0.
2849  010126  112737  000005  001126 TST005: MOVB    #5,@#TSTNUM             ;SET TEST #5
2850  010134  004767  175202                  JSR     PC,.REWIND              ;REWIND SLAVE
2851  010140  102422                          BVS     99$                     ;BRANCH IF ERROR ON REWIND
2852  010142  004767  175274                  JSR     PC,READ
2853  010146  012702  010160                  MOV     #1$,R2                  ;SET RETURN PC FROM TIMER
```

```
2854  010152  004767  174202            JSR     PC,TIMON                ;TURN TIMER ON
2855  010156  005215                    INC     (R5)                    ;SET 'GO' BIT
2856
2857  010160  005765  000032    1$:     TST     TC(R5)                  ;BRANCH WHEN 'ACCL' RESETS
2858  010164  100002                    BPL     2$
2859  010166  000163  004450            JMP     TIMER(R3)               ;GO TO TIMER & RETURN VIA R2
2860
2861  010172  004767  175040    2$:     JSR     PC,WAITRDY              ;WAIT FOR READY
2862  010176  102403                    BVS     99$                     ;BRANCH IF ERROR
2863  010200  004767  174302            JSR     PC,TIMOK                ;CHECK RECORDED TIME
2864  010204  000401                    BR      100$
2865
2866  010206  104400            99$:    HLT
2867  010210  104000            100$:   SCOPE
2868
2869                            ;TEST 006 - READ START
2870                            ;THIS TEST MEASURES TIME FROM 'GO'=1 TO 'ACCL'=0.
2871  010212  112737  000006  001126  TST006: MOVB    #6,@#TSTNUM             ;SET TEST #6
2872  010220  004767  175272            JSR     PC,WRT.BK               ;WRITE A RECORD & BACK SPACE
2873  010224  102422                    BVS     99$
2874  010226  004767  175210            JSR     PC,READ
2875  010232  012702  010244            MOV     #1$,R2                  ;SET RETURN PC FROM TIMER
2876  010236  004767  174116            JSR     PC,TIMON                ;TURN TIMER ON
2877  010242  005215                    INC     (R5)                    ;SET 'GO' BIT
2878
2879  010244  005765  000032    1$:     TST     TC(R5)                  ;BRANCH WHEN 'ACCL' RESETS
2880  010250  100002                    BPL     2$
2881  010252  000163  004450            JMP     TIMER(R3)               ;GO TO TIMER & RETURN VIA R2
2882
2883  010256  004767  174754    2$:     JSR     PC,WAITRDY
2884  010262  102403                    BVS     99$
2885  010264  004767  174216            JSR     PC,TIMOK
2886  010270  000401                    BR      100$
2887
2888  010272  104400            99$:    HLT
2889  010274  104000            100$:   SCOPE
2890
2891                            ;TEST 007 - READ SHUTDOWN
2892                            ;THIS TEST MEASURES TIME FROM 'FC REG'=FRAME COUNT TO 'SWDN'=1.
2893  010276  112737  000007  001126  TST007: MOVB    #7,@#TSTNUM             ;SET TEST #7
2894  010304  004767  175206            JSR     PC,WRT.BK               ;WRITE A RECORD & BACK SPACE
2895  010310  102430                    BVS     99$                     ;BRANCH IF ERROR
2896  010312  004767  175124            JSR     PC,READ
2897  010316  005215                    INC     (R5)                    ;SET 'GO' BIT
2898
2899  010320  022710  000400    1$:     CMP     #-FRMCNT,(R0)           ;WAIT FOR FRAME COUNT TO
2900  010324  001404                    BEQ     2$                      ;= # OF FRAMES WRITTEN
2901  010326  032711  040000            BIT     #ERR,(R1)               ;MONITOR ERROR BIT
2902  010332  001017                    BNE     99$
2903  010334  000771                    BR      1$
2904
2905  010336                    2$:
2906  010336  004767  174016            JSR     PC,TIMON                ;TURN TIMER ON
2907  010342  010702                    MOV     PC,R2                   ;SET RETURN PC FROM TIMER
2908  010344  032711  000020            BIT     #SDWN,(R1)              ;BRANCH WHEN SDWN SETS
2909  010350  001002                    BNE     3$
```

```
                                                                J  6
 2910  010352  000163  004450                 JMP     TIMER(R3)               ;GO TO TIMER & RETURN VIA R2
 2911
 2912  010356  004767  174654          3$:    JSR     PC,WAITRDY
 2913  010362  102403                        BVS     99$
 2914  010364  004767  174116                JSR     PC,TIMOK
 2915  010370  000401                        BR      100$
 2916
 2917  010372  104400                99$:    HLT                             ;REPORT ERROR
 2918  010374  104000                100$:   SCOPE
 2919
 2920                                ;TEST 010 - READ SETTLEDOWN
 2921                                ;THIS TEST MEASURES TIME FROM 'SWDN'=1 TO 'SWDN'=0.
 2922  010376  112737  000010  001126 TST010: MOVB    #10,@#TSTNUM            ;SET TEST #10
 2923  010404  012702  010462                MOV     #4$,R2                  ;SET RETURN PC FROM TIMER
 2924  010410  004767  175102                JSR     PC,WRT.BK               ;WRITE A RECORD & BACK SPACE
 2925  010414  102436                        BVS     99$
 2926  010416  004767  175020                JSR     PC,READ
 2927  010422  005215                        INC     (R5)                    ;SET 'GO' BIT
 2928
 2929  010424  105711                1$:     TSTB    (R1)                    ;WAIT FOR READY
 2930  010426  100404                        BMI     2$                      ;BRANCH WHEN SET
 2931  010430  032711  040000                BIT     #ERR,(R1)               ;CHECK ERROR BIT
 2932  010434  001026                        BNE     99$
 2933  010436  000772                        BR      1$
 2934
 2935  010440  032711  000020          2$:    BIT     #SDWN,(R1)              ;WAIT FOR ASSERTION OF 'SDWN'
 2936  010444  001004                        BNE     3$
 2937  010446  032711  040000                BIT     #ERR,(R1)               ;MONITOR ERROR BIT
 2938  010452  001017                        BNE     99$
 2939  010454  000771                        BR      2$
 2940
 2941  010456                          3$:
 2942  010456  004767  173676                JSR     PC,TIMON                ;TURN TIMER ON
 2943  010462  032765  000020  000012 4$:    BIT     #SDWN,DS(R5)            ;WAIT FOR NEGATION OF SDWN
 2944  010470  001402                        BEQ     5$
 2945  010472  000163  004450                JMP     TIMER(R3)               ;GO TO TIMER & RETURN VIA R2
 2946
 2947  010476  004767  174534          5$:    JSR     PC,WAITRDY
 2948  010502  102403                        BVS     99$
 2949  010504  004767  173776                JSR     PC,TIMOK
 2950  010510  000401                        BR      100$
 2951
 2952  010512  104400                99$:    HLT
 2953  010514  104000                100$:   SCOPE
 2954
 2955
 2956                                ;TEST 011-READ REVERSE START
 2957                                ;THIS TEST MEASURES TIME FROM 'GO'=1 TO 'ACCL'=0.
 2958  010516  112737  000011  001126 TST011: MOVB    #11,@#TSTNUM
 2959  010524  012702  010562                MOV     #1$,R2                  ;SET RETURN PC FROM TIMER
 2960  010530  004767  174670                JSR     PC,WRITE                ;WRITE A RECORD
 2961  010534  005215                        INC     (R5)                    ;SET 'GO' BIT
 2962  010536  004767  174474                JSR     PC,WAITRDY
 2963  010542  102422                        BVS     99$
 2964  010544  004767  174174                JSR     PC,DELAY                ;WAIT FOR TAPE MOTION TO STOP
 2965  010550  004767  174704                JSR     PC,REVRD
```

```
2966  010554  004767  173600            JSR     PC,TIMON                ;TURN TIMER ON
2967  010560  005215                    INC     (R5)                    ;SET 'GO' BIT
2968
2969  010562  005765  000032    1$:     TST     TC(R5)                  ;BRANCH WHEN 'ACCL' = 0
2970  010566  100002                    BPL     2$
2971  010570  000163  004450            JMP     TIMER(R3)               ;GO TO TIMER & RETURN VIA R2
2972
2973  010574  004767  174436    2$:     JSR     PC,WAITRDY
2974  010600  102403                    BVS     99$                     ;BRANCH IF ERROR
2975  010602  004767  173700            JSR     PC,TIMOK
2976  010606  000401                    BR      100$
2977
2978  010610  104400            99$:    HLT
2979  010612  104000            100$:   SCOPE
2980
2981                            ;TEST 012-READ REVERSE SHUTDOWN
2982                            ;THIS TEST MEASURES TIME FROM 'FC REG' = FRAME COUNT TO 'SDWN'=1.
2983  010614  112737  000012  001126 TST012: MOVB    #12,@#TSTNUM
2984  010622  012702  010672            MOV     #3$,R2                  ;SET RETURN PC FROM TIMER
2985  010626  004767  174572            JSR     PC,WRITE                ;WRITE A RECORD
2986  010632  005215                    INC     (R5)                    ;SET 'GO' BIT
2987  010634  004767  174376            JSR     PC,WAITRDY
2988  010640  102427                    BVS     99$
2989  010642  004767  174612            JSR     PC,REVRD
2990  010646  005215                    INC     (R5)                    ;SET 'GO' BIT
2991
2992  010650  022710  000400    1$:     CMP     #-FRMCNT,(R0)           ;BRANCH WHEN FRAME COUNT
2993  010654  001404                    BEQ     2$                      ;= # OF RECORD WRITTEN
2994  010656  032711  040000            BIT     #ERR,(R1)               ;MONITOR ERROR BIT IN 'DS' REG
2995  010662  001016                    BNE     99$
2996  010664  000771                    BR      1$
2997
2998  010666                    2$:
2999  010666  004767  173466            JSR     PC,TIMON                ;TURN TIMER ON
3000  010672  032711  000020    3$:     BIT     #SDWN,(R1)              ;BRANCH WHEN SDWN SETS
3001  010676  001002                    BNE     4$
3002  010700  000163  004450            JMP     TIMER(R3)               ;GO TO TIMER & RETURN VIA R2
3003
3004  010704  004767  174326    4$:     JSR     PC,WAITRDY              ;WAIT FOR READY
3005  010710  102403                    BVS     99$
3006  010712  004767  173570            JSR     PC,TIMOK
3007  010716  000401                    BR      100$
3008
3009  010720  104400            99$:    HLT
3010  010722  104000            100$:   SCOPE
3011
3012                            ;TEST 013-READ REVERSE SETTLEDOWN
3013                            ;THIS TEST MEASURES TIME FROM 'SDWN'=1 TO 'SDWN'=0.
3014  010724  112737  000013  001126 TST013: MOVB    #13,@#TSTNUM
3015  010732  012702  011016            MOV     #4$,R2                  ;SET RETURN PC FROM TIMER
3016  010736  004767  174462            JSR     PC,WRITE                ;WRITE A RECORD
3017  010742  005215                    INC     (R5)                    ;SET 'GO' BIT
3018  010744  004767  174266            JSR     PC,WAITRDY
3019  010750  102435                    BVS     99$
3020  010752  004767  174502            JSR     PC,REVRD
3021  010756  005215                    INC     (R5)                    ;SET 'GO' BIT
```

```
3022
3023  010760  105711                  1$:     TSTB    (R1)                    ;BRANCH WHEN
3024  010762  100404                          BMI     2$                      ;READY SETS
3025  010764  032711  040000                  BIT     #ERR,(R1)
3026  010770  001025                          BNE     99$
3027  010772  000772                          BR      1$
3028
3029  010774  032711  000020          2$:     BIT     #SDWN,(R1)
3030  011000  001004                          BNE     3$
3031  011002  032711  040000                  BIT     #ERR,(R1)
3032  011006  001016                          BNE     99$
3033  011010  000771                          BR      2$
3034
3035  011012                          3$:
3036  011012  004767  173342                  JSR     PC,TIMON                ;TURN TIMER ON
3037  011016  032711  000020          4$:     BIT     #SDWN,(R1)              ;BRANCH WHEN SWDN = 0
3038  011022  001402                          BEQ     5$
3039  011024  000163  004450                  JMP     TIMER(R3)               ;GO TO TIMER & RETURN VIA R2
3040
3041  011030  004767  174202          5$:     JSR     PC,WAITRDY              ;WAiT FOR READY
3042  011034  102403                          BVS     99$
3043  011036  004767  173444                  JSR     PC,TIMOK
3044  011042  000401                          BR      100$
3045
3046  011044  104400                  99$:    HLT
3047  011046  104000                  100$:   SCOPE
3048
3049                                  ;REWIND DRIVE
3050  011050                          A:
3051  011050  004767  174266                  JSR     PC,.REWIND              ;REWIND SLAVE
3052  011054  102401                          BVS     99$                     ;BRANCH IF ERROR ON REWIND
3053  011056  102002                          BVC     100$
3054  011060  104400                  99$:    HLT
3055  011062  000772                          BR      A
3056  011064                          100$:
3057
3058                                  ;TEST 014-TURN AROUND DELAY (FORWARD-REVERSE)
3059                                  ;THIS TEST MEASURES TIME FROM 'GO'=1 (READ REVERSE) TO 'ACCL'=0
3060  011064  112737  000014  001126  TST014: MOVB    #14,@#TSTNUM
3061  011072  012702  011124                  MOV     #2$,R2                  ;SET RETURN PC FROM TIMER
3062  011076  004767  174322                  JSR     PC,WRITE                ;WRITE A RECORD
3063  011102  005215                          INC     (R5)                    ;SET 'GO' BIT
3064  011104  004767  174126                  JSR     PC,WAITRDY
3065  011110  102420                          BVS     99$
3066
3067  011112  004767  174342          1$:     JSR     PC,REVRD                ;READ THE RECORD (REVERSE)
3068  011116  004767  173236                  JSR     PC,TIMON                ;TURN TIMER ON
3069  011122  005215                          INC     (R5)                    ;SET 'GO' BIT
3070
3071  011124  005765  000032          2$:     TST     TC(R5)                  ;WAIT FOR 'ACCL' = 0
3072  011130  100002                          BPL     3$
3073  011132  000163  004450                  JMP     TIMER(R3)               ;GO TO TIMER & RETURN VIA R2
3074
3075  011136  004767  174074          3$:     JSR     PC,WAITRDY
3076  011142  102403                          BVS     99$
3077  011144  004767  173336                  JSR     PC,TIMOK
```

```
3078  011150  000401                    BR      100$
3079
3080  011152  104400              99$:  HLT
3081  011154  104000              100$: SCOPE
3082
3083                                    ;TEST 015- TURN AROUND DELAY (REVERSE-FORWARD)
3084                                    ;THIS TEST MEASURES TIME FROM 'GO'=1 (READ) TO 'ACCL'=0.
3085  011156  112737  000015  001126  TST015: MOVB   #15,@#TSTNUM
3086  011164  012702  011232          MOV     #2$,R2          ;SET RETURN PC FROM TIMER
3087  011170  004767  174230          JSR     PC,WRITE        ;WRITE A RECORD
3088  011174  005215                  INC     (R5)            ;SET 'GO' BIT
3089  011176  004767  174034          JSR     PC,WAITRDY      ;WAIT FOR READY
3090  011202  102426                  BVS     99$
3091  011204  004767  174250          JSR     PC,REVRD        ;READ A RECORD IN THE
3092  011210  005215                  INC     (R5)            ;SET 'GO' BIT
3093
3094  011212  004767  174020          JSR     PC,WAITRDY
3095  011216  102420                  BVS     99$
3096
3097  011220  004767  174216          1$:     JSR     PC,READ         ;READ RECORD FORWARD
3098  011224  004767  173130          JSR     PC,TIMON        ;TURN TIMER ON
3099  011230  005215                  INC     (R5)            ;SET 'GO' BIT
3100
3101  011232  005765  000032          2$:     TST     TC(R5)          ;WAIT FOR 'ACCL' = 0
3102  011236  100002                  BPL     3$
3103  011240  000163  004450          JMP     TIMER(R3)       ;GO TO TIMER & RETURN VIA R2
3104
3105  011244  004767  173766          3$:     JSR     PC,WAITRDY
3106  011250  102403                  BVS     99$
3107  011252  004767  173230          JSR     PC,TIMOK
3108  011256  000401                  BR      100$
3109
3110  011260  104400              99$:  HLT
3111  011262  104000              100$: SCOPE
3112
3113                                    ;TEST 016-GAP SIZE (STOP HALF)
3114  011264  112737  000016  001126  TST016: MOVB   #16,@#TSTNUM
3115  011272  012702  011330          MOV     #1$,R2          ;SET RETURN PC FROM TIMER
3116  011276  004767  174122          JSR     PC,WRITE        ,WRITE A RECORD
3117  011302  005215                  INC     (R5)            ;SET 'GO' BIT
3118  011304  004767  173726          JSR     PC,WAITRDY
3119  011310  102421                  BVS     99$
3120  011312  004767  173426          JSR     PC,DELAY        ;DELAY 350 MS
3121  011316  004767  174136          JSR     PC,REVRD        ;READ REVERSE RECORD
3122  011322  004767  173032          JSR     PC,TIMON        ;TURN TIMER ON
3123  011326  005215                  INC     (R5)            ;SET 'GO' BIT
3124
3125  011330  005710                  1$:     TST     (R0)            ;WAIT FOR FRAME COUNT > 0
3126  011332  001002                  BNE     2$
3127  011334  000163  004450          JMP     TIMER(R3)       ;GO TO TIMER & RETURN VIA R2
3128
3129  011340  004767  173672          2$:     JSR     PC,WAITRDY      ;WAIT FOR READY BIT TO SET
3130  011344  102403                  BVS     99$
3131  011346  004767  173134          JSR     PC,TIMOK        ;CHECK TIME
3132  011352  000401                  BR      100$
3133
```

```
3134  011354  104400                    99$:    HLT
3135  011356  104000                    100$:   SCOPE
3136
3137                                    ;TEST 017-GAP SIZE (START HALF)
3138  011360  112737  000017  001126    TST017: MOVB    #17,@#TSTNUM
3139  011366  012702  011440            MOV     #1$,R2          ;SET RETURN PC FROM TIMER
3140  011372  004767  174026            JSR     PC,WRITE        ;WRITE A RECORD
3141  011376  005215                    INC     (R5)            ;SET 'GO' BIT
3142  011400  004767  173632            JSR     PC,WAITRDY      ;WAIT FOR READY
3143  011404  102427                    BVS     99$
3144  011406  004767  174046            JSR     PC,REVRD        ;READ REVERSE THE RECORD
3145  011412  005215                    INC     (R5)            ;SET 'GO' BIT
3146  011414  004767  173616            JSR     PC,WAITRDY      ;WAIT FOR READY
3147  011420  102421                    BVS     99$             ;BRANCH ON ERROR
3148  011422  004767  173316            JSR     PC,DELAY        ;WAIT FOR TAPE MOTION TO STOP
3149  011426  004767  174010            JSR     PC,READ         ;READ RECORD
3150  011432  004767  172722            JSR     PC,TIMON        ;TURN TIMER ON
3151  011436  005215                    INC     (R5)            ;SET 'GO' BIT
3152
3153  011440  005710                    1$:     TST     (R0)            ;WAIT FOR FRAME COUNT > 0
3154  011442  001002                    BNE     2$
3155  011444  000163  004450            JMP     TIMER(R3)       ;GO TO TIMER & RETURN VIA R2
3156
3157  011450  004767  173562            2$:     JSR     PC,WAITRDY      ;WAIT FOR READY
3158  011454  102403                    BVS     99$
3159  011456  004767  173024            JSR     PC,TIMOK        ;CHECK TIME
3160  011462  000401                    BR      100$
3161
3162  011464  104400                    99$:    HLT
3163  011466  104000                    100$:   SCOPE
3164
3165                                    ;TEST 020- GAP SIZE (INTERRECORD)
3166                                    ;THIS TEST MEASURES TIME FROM 'GO'=1 TO 'FC REG' >0.
3167  011470  112737  000020  001126    TST020: MOVB    #20.,@#TSTNUM
3168  011476  012702  011560            MOV     #1$,R2          ;SET RETURN PC FROM TIMER
3169  011502  004767  173716            JSR     PC,WRITE        ;WRITE A RECORD
3170  011506  005215                    INC     (R5)            ;SET 'GO' BIT
3171  011510  004767  173522            JSR     PC,WAITRDY      ;WAIT FOR READY
3172  011514  102433                    BVS     99$
3173  011516  004767  173702            JSR     PC,WRITE        ;WRITE SECOND RECORD
3174  011522  005215                    INC     (R5)            ;SET 'GO' BIT
3175  011524  004767  173506            JSR     PC,WAITRDY      ;WAIT FOR READY
3176  011530  102425                    BVS     99$
3177  011532  004767  173722            JSR     PC,REVRD        ;READ REVERSE SECOND RECORD
3178  011536  005215                    INC     (R5)            ;SET 'GO' BIT
3179  011540  004767  173472            JSR     PC,WAITRDY      ;WAIT FOR READY
3180  011544  102417                    BVS     99$
3181  011546  004767  173706            JSR     PC,REVRD        ;READ REVERSE FIRST RECORD
3182  011552  004767  172602            JSR     PC,TIMON        ;TURN TIMER ON
3183  011556  005215                    INC     (R5)            ;SET 'GO' BIT
3184
3185  011560  005710                    1$:     TST     (R0)            ;WAIT FOR FRAME COUNT > 0
3186  011562  001002                    BNE     2$
3187  011564  000163  004450            JMP     TIMER(R3)       ;GO TO TIMER & RETURN VIA R2
3188
3189  011570  004767  173442            2$:     JSR     PC,WAITRDY      ;WAIT FOR READY
```

```
3190  011574  102403                          BVS     99$
3191  011576  004767  172704                  JSR     PC,TIMOK
3192  011602  000401                          BR      100$
3193
3194  011604  104400                  99$:    HLT
3195  011606  104000                  100$:   SCOPE
3196
3197                                  ;TEST 021- GAP CONSISTANCY
3198                                  ;THIS TEST MEASURES TIME FROM 'GO'=1 TO 'FC REG' > 0.
3199                                  ;THE TEST REWINDS THE TAPE,WRITES 17 RECORDS WITH A DELAY FROM 1-16 MS
3200                                  ;BETWEEN EACH WRITE COMMAND. AFTER THE 17. RECORDS ARE WRITTTEN THE
3201                                  ;PROGRAM READ REVERSES 16 RECORDS. AT THIS POINT THE TAPE IS STOPPED BE-
3202                                  ;TWEEN THE FIRST AND SECOND RECORD. A READ COMMAND IS EXECUTED TO READ
3203                                  ;THE 16 RECORDS WITH THE TIME BETEWEN GO=1 TO FC > 0 STORED IN 'GAPTBL'
3204                                  ;FOR EACH RECORD READ. AFTER 16 RECORDS HAVE BEEN READ THE TIME IS VER-
3205                                  ;IFIED FOR EACH READ. AFTER ALL RECORD TIMES ARE VERIFIED THEY ARE AVER-
3206                                  ;AGED AND PLACED IN THE 'ATIMTBL' (BY SCOPE). THE ABOVE PROCESS IS RE-
3207                                  ;PEATED FOR EACH ITERATION.
3208
3209  011610  112737  000021  001126  TST021: MOVB    #21,@#TSTNUM
3210  011616  012702  011754                  MOV     #4$,R2          ;SET RETURN PC FROM TIMER
3211  011622  004767  173514                  JSR     PC,..REWIND     ;REWIND SLAVE
3212  011626  102530                          BVS     99$             ;BRANCH IF ERROR ON REWIND
3213  011630  005067  167264                  CLR     DELTIM          ;CLEAR VARIABLE DELAY TIME
3214  011634  012700  000021                  MOV     #17.,R0         ;SET # OF RECORDS TO WRITE
3215  011640  004767  173560          1$:     JSR     PC,WRITE        ;WRITE 17. RECORDS
3216  011644  005215                          INC     (R5)            ;SET 'GO' BIT
3217  011646  004767  173364                  JSR     PC,WAITRDY      ;WAIT FOR READY
3218  011652  102516                          BVS     99$
3219  011654  004767  173114                  JSR     PC,DELAYV       ;DELAY BEFORE WRITING NEXT REC.
3220  011660  062767  000022  167232          ADD     #18.,DELTIM     ;SET NEXT DELAY TIME
3221  011666  005300                          DEC     R0              ;DECREMENT RECORDS WRITTEN COUNT
3222  011670  001363                          BNE     1$
3223
3224  011672  012700  000021                  MOV     #17.,R0         ;SET # OF RECS. TO REVERSE READ
3225  011676  004767  173556          2$:     JSR     PC,REVRD        ;REVERSE READ 17. RECORDS
3226  011702  005215                          INC     (R5)            ;SET 'GO' BIT
3227  011704  004767  173326                  JSR     PC,WAITRDY      ;WAIT FOR READY
3228  011710  102477                          BVS     99$
3229  011712  005300                          DEC     R0              ;DECREMENT RECORD COUNT
3230  011714  001370                          BNE     2$
3231
3232  011716  012700  000020                  MOV     #16.,R0         ;SET # OF RECORDS TO READ
3233  011722  012701  001060                  MOV     #GAPTBL,R1      ;SET PTR TO GAP TABLE FOR TEST
3234  011726  004767  173510                  JSR     PC,READ         ;READ A RECORD
3235  011732  005215                          INC     (R5)            ;SET 'GO' BIT
3236
3237  011734  004767  173276          3$:     JSR     PC,WAITRDY      ;WAIT FOR READY
3238  011740  102463                          BVS     99$
3239  011742  004767  173474                  JSR     PC,READ         ;READ NEXT RECORD
3240  011746  004767  172406                  JSR     PC,TIMON        ;TURN TIMER ON
3241  011752  005215                          INC     (R5)            ;SET 'GO' BIT
3242
3243  011754  005765  000006          4$:     TST     FC(R5)          ;WAIT FOR FRAME COUNT > 0
3244  011760  001002                          BNE     5$
3245  011762  000163  004450                  JMP     TIMER(R3)       ;GO TO TIMER & RETURN VIA R2
```

```
3246
3247  011766  004767  173244          5$:     JSR     PC,WAITRDY              ;WAIT FOR READY
3248  011772  102446                          BVS     99$
3249  011774  010421                          MOV     R4,(R1)+                ;STORE TIME IN GAPTBL
3250  011776  005300                          DEC     R0                      ;DECREMENT # OF RECORDS READ
3251  012000  001355                          BNE     3$
3252
3253  012002  105037  001124                  CLRB    @#GAP                   ;SET GAP # 0
3254  012006  012700  000020                  MOV     #16.,R0
3255  012012  012701  001060                  MOV     #GAPTBL,R1
3256
3257  012016  012104                  6$:     MOV     (R1)+,R4                ;GET GAP TICK COUNT
3258  012020  004767  172572                  JSR     PC,GAPOK                ;CHECK TIME
3259  012024  105237  001124                  INCB    @#GAP                   ;INCREMENT GAP #
3260  012030  122737  000020  001124          CMPB    #16.,@#GAP              ;BRANCH IF ALL GAPS NOT CHECKED
3261  012036  001367                          BNE     6$
3262
3263  012040  012700  000020                  MOV     #16.,R0                 ;SETUP TO AVERAGE GAP SIZES
3264  012044  012701  001060                  MOV     #GAPTBL,R1              ;SET PTR TO TABLE
3265  012050  005002                          CLR     R2                      ;CLEAR 'SUM' REGISTERS
3266  012052  005003                          CLR     R3
3267  012054  062102                  7$:     ADD     (R1)+,R2                ;ADD ALL GAP SIZES TOGETHER
3268  012056  005503                          ADC     R3
3269  012060  005300                          DEC     R0
3270  012062  001374                          BNE     7$
3271  012064  012700  000004                  MOV     #4,R0                   ;NOW DIVIDE BY 16.
3272  012070  006203                  8$:     ASR     R3                      ;BY SHIFTING 4 PLACES RIGHT
3273  012072  006002                          ROR     R2
3274  012074  005300                          DEC     R0
3275  012076  001374                          BNE     8$
3276  012100  010204                          MOV     R2,R4                   ;MOVE AVERAGED TIMES TO R4
3277  012102  004767  172400                  JSR     PC,TIMOK                ;CHECK AVERAGED TIMES
3278  012106  000401                          BR      100$
3279
3280  012110  104400                  99$:    HLT
3281  012112  104000                  100$:   SCOPE
3282
3283                                  ;TEST 022-DUMMY TEST
3284                                  ;THIS TEST MEASURES NOTHING
3285  012114  112737  000022  001126  TST022: MOVB    #22,@#TSTNUM
3286
3287                                  ;TEST 023-DATA TIME (200BPI)
3288                                  ;THIS TEST MEASURES TIME FROM 'FC REG' CHANGES TO 'RDY'=1.
3289  012122  112737  000023  001126  TST023: MOVB    #23,@#TSTNUM
3290  012130  012702  012202                  MOV     #3$,R2                  ;SET RETURN PC FROM TIMER
3291  012134  004767  173202                  JSR     PC,.REWIND              ;REWIND SLAVE
3292  012140  102437                          BVS     99$                     ;BRANCH IF ERROR ON REWIND
3293  012142  004367  173412                  JSR     R3,TMCMD                ;WRITE 800 WORD RECORD
3294  012146  016420                          .WORD   WTBUF                   ;SET WRITE BUFFER ADDRESS
3295  012150  176340                          .WORD   -800.                   ;WORD COUNT
3296  012152  174700                          .WORD   -1600.                  ;FRAME COUNT
3297  012154  000060                          .WORD   WFWD                    ;WRITE COMMAND
3298  012156  005215                          INC     (R5)                    ;SET 'GO' BIT
3299
3300  012160  022710  174700          1$:     CMP     #-1600.,(R0)            ;WAIT FOR FRAME COUNT TO CHANGE
3301  012164  001004                          BNE     2$
```

```
 3302  012166  032711  040000              BIT     #ERR,(R1)               ;MONITOR ERROR BIT
 3303  012172  001022                      BNE     99$
 3304  012174  000771                      BR      1$
 3305
 3306  012176                      2$:
 3307  012176  004767  172156              JSR     PC,TIMON                ;TURN TIMER ON
 3308  012202  105711              3$:     TSTB    (R1)                    ;WAIT FOR READY TO SET
 3309  012204  100402                      BMI     4$
 3310  012206  000163  004450              JMP     TIMER(R3)               ;GO TO TIMER & RETURN VIA R2
 3311  012212  012700  000003      4$:     MOV     #3,R0                   ;SET TO DIVIDE BY 8
 3312  012216  006204              5$:     ASR     R4                      ;BY SHIFTING RIGHT 3 PLACES
 3313  012220  005300                      DEC     R0
 3314  012222  001375                      BNE     5$
 3315  012224  004767  173006              JSR     PC,WAITRDY
 3316  012230  102403                      BVS     99$
 3317  012232  004767  172250              JSR     PC,TIMOK                ;CHECK TIME
 3318  012236  000401                      BR      100$
 3319
 3320  012240  104400              99$:    HLT
 3321  012242  104000              100$:   SCOPE
 3322
 3323                              ;TEST 024-DATA TIME (556BPI)
 3324  012244  112737  000024  001126  TST024: MOVB    #24,@#TSTNUM
 3325  012252  012702  012332              MOV     #3$,R2                  ;SET RETURN PC FROM TIMER
 3326  012256  004767  173060              JSR     PC,.REWIND              ;REWIND SLAVE
 3327  012262  102442                      BVS     99$                     ;BRANCH IF ERROR ON REWIND
 3328  012264  052765  000700  000032      BIS     #BP1556+NORM11,TC(R5)   ;LOAD TAPE CONTROL REGISTER
 3329  012272  004367  173262              JSR     R3,TMCMD                ;WRITE 2224. WORD RECORD
 3330  012276  016420                      .WORD   WTBUF
 3331  012300  173520                      .WORD   -2224.
 3332  012302  167240                      .WORD   -4448.
 3333  012304  000060                      .WORD   WFWD
 3334  012306  005215                      INC     (R5)                    ;SET 'GO' BIT
 3335
 3336  012310  022710  167240      1$:     CMP     #-4448.,(R0)            ;BRANCH WHEN WRITING BEGINS
 3337  012314  001004                      BNE     2$
 3338  012316  032711  040000              BIT     #ERR,(R1)               ;MONITOR ERROR BIT
 3339  012322  001022                      BNE     99$
 3340  012324  000771                      BR      1$
 3341
 3342  012326                      2$:
 3343  012326  004767  172026              JSR     PC,TIMON                ;TURN TIMER ON
 3344  012332  105711              3$:     TSTB    (R1)                    ;BRANCH WHEN READY SETS
 3345  012334  100402                      BMI     4$
 3346  012336  000163  004450              JMP     TIMER(R3)               ;GO TO TIMER & RETURN VIA R2
 3347
 3348  012342  012700  000003      4$:     MOV     #3,R0                   ;SET SHIFT COUNT
 3349  012346  006204              5$:     ASR     R4
 3350  012350  005300                      DEC     R0
 3351  012352  001375                      BNE     5$
 3352  012354  004767  172656              JSR     PC,WAITRDY
 3353  012360  102403                      BVS     99$
 3354  012362  004767  172120              JSR     PC,TIMOK                ;CHECK TIME
 3355  012366  000401                      BR      100$
 3356
 3357  012370  104400              99$:    HLT
```

```
3358  012372  104000                    100$:   SCOPE
3359
3360                                    ;TEST 025-DATA TIME (800BPI)
3361  012374  112737  000025  001126    TST025: MOVB    #025,@#TSTNUM
3362  012402  012702  012462            MOV     #3$,R2                  ;SET RETURN PC FROM TIMER
3363  012406  004767  172730            JSR     PC,.REWIND              ;REWIND SLAVE
3364  012412  102442                    BVS     99$                     ;BRANCH IF ERROR ON REWIND
3365  012414  052765  001300  000032    BIS     #BPI800+NORM11,TC(R5)   ;SET 800 BPI
3366  012422  004367  173132            JSR     R3,TMCMD                ;WRITE 3200. WORD RECORD
3367  012426  016420                    .WORD   WTBUF
3368  012430  171600                    .WORD   -3200.
3369  012432  163400                    .WORD   -6400.
3370  012434  000060                    .WORD   WFWD
3371  012436  005215                    INC     (R5)                    ;SET 'GO' BIT
3372
3373  012440  022710  163400            1$:     CMP     #-6400.,(R0)            ;WAIT FOR WRITING TO START
3374  012444  001004                    BNE     2$
3375  012446  032711  040000            BIT     #ERR,(R1)               ;MONITOR ERROR BIT
3376  012452  001022                    BNE     99$
3377  012454  000771                    BR      1$
3378
3379  012456                            2$:
3380  012456  004767  171676            JSR     PC,TIMON                ;TURN TIMER ON
3381  012462  105711                    3$:     TSTB    (R1)            ;BRANCH WHEN READY SETS
3382  012464  100402                    BMI     4$
3383  012466  000163  004450            JMP     TIMER(R3)               ;GO TO TIMER & RETURN VIA R2
3384
3385  012472  012700  000003            4$:     MOV     #3,R0           ;SET SHIFT COUNT
3386  012476  006204                    5$:     ASR     R4
3387  012500  005300                    DEC     R0
3388  012502  001375                    BNE     5$
3389  012504  004767  172526            JSR     PC,WAITRDY
3390  012510  102403                    BVS     99$
3391  012512  004767  171770            JSR     PC,TIMOK                ;CHECK TIME
3392  012516  000401                    BR      100$
3393
3394  012520  104400                    99$:    HLT
3395  012522  104000                    100$:   SCOPE
3396
3397                                    ;TEST 026-DATA TIME (1600BPI)
3398  012524  112737  000026  001126    TST026: MOVB    #026,@#TSTNUM
3399  012532  105737  001133            TSTB    @#NRZFLG                ;BRANCH IF DRIVE 'NRZ ONLY'
3400  012536  001046                    BNE     TST027
3401  012540  012702  012620            MOV     #3$,R2                  ;SET RETURN PC FROM TIMER
3402  012544  004767  172572            JSR     PC,.REWIND              ;REWIND SLAVE
3403  012550  102437                    BVS     99$                     ;BRANCH IF ERROR ON REWIND
3404  012552  052765  002300  000032    BIS     #PE1600+NORM11,TC(R5)   ;SET 1600 BPI
3405  012560  004367  172774            JSR     R3,TMCMD                ;WRITE 3200. WORD RECORD
3406  012564  016420                    .WORD   WTBUF
3407  012566  171600                    .WORD   -3200.
3408  012570  163400                    .WORD   -6400.
3409  012572  000060                    .WORD   WFWD
3410  012574  005215                    INC     (R5)                    ;SET 'GO' BIT
3411
3412  012576  022710  163400            1$:     CMP     #-6400.,(R0)            ;BRANCH WHEN WRITING STARTS
3413  012602  001004                    BNE     2$
```

```
3414  012604  032711  040000          BIT     #ERR,(R1)               ;MONITOR ERROR BIT
3415  012610  001017                  BNE     99$
3416  012612  000771                  BR      1$
3417
3418  012614                  2$:
3419  012614  004767  171540          JSR     PC,TIMON                ;TURN TIMER ON
3420  012620  105711          3$:     TSTB    (R1)                    ;BRANCH WHEN READY SETS
3421  012622  100402                  BMI     4$
3422  012624  000163  004450          JMP     TIMER(R3)               ;GO TO TIMER & RETURN VIA R2
3423
3424  012630  006204          4$:     ASR     R4                      ;DIVIDE TIME BY 4
3425  012632  006204                  ASR     R4
3426  012634  004767  172376          JSR     PC,WAITRDY
3427  012640  102403                  BVS     99$
3428  012642  004767  171640          JSR     PC,TIMOK                ;CHECK TIME
3429  012646  000401                  BR      100$
3430
3431  012650  104400          99$:    HLT
3432  012652  104000          100$:   SCOPE
3433
3434                          ;TEST 027-ERASE
3435                          ;THIS TST MEASURES TIME FROM 'GO'=1 TO 'RDY'=1.
3436  012654  112737  000027  001126  TST027: MOVB    #27,@#TSTNUM
3437  012662  012702  012710          MOV     #1$,R2                  ;SET RETURN PC FROM TIMER
3438  012666  004337  005560          JSR     R3,@#TMCMD
3439  012672  000000                  .WORD   0
3440  012674  000000                  .WORD   0
3441  012676  000000                  .WORD   0
3442  012700  000024                  .WORD   ERASE
3443  012702  004767  171452          JSR     PC,TIMON                ;TURN TIMER ON
3444  012706  005215                  INC     (R5)                    ;SET 'GO' BIT
3445
3446  012710  105711          1$:     TSTB    (R1)                    ;BRANCH WHEN READY SETS
3447  012712  100402                  BMI     2$
3448  012714  000163  004450          JMP     TIMER(R3)               ;GO TO TIMER & RETURN VIA R2
3449
3450  012720  004767  172312          JSR     PC,WAITRDY
3451  012724  102403                  BVS     99$
3452  012726  004767  171554          JSR     PC,TIMOK
3453  012732  000401                  BR      100$
3454
3455  012734  104400          99$:    HLT
3456  012736  104000          100$:   SCOPE
3457
3458                          ;TEST-030 TAPE MARK
3459                          ;THIS TEST MEASURES TIME FROM 'GO'=1 TO 'RDY'=1.
3460  012740  112737  000030  001126  TST030: MOVB    #30,@#TSTNUM
3461  012746  012702  013010          MOV     #1$,R2                  ;SET RETURN PC FROM TIMER
3462  012752  004767  172446          JSR     PC,WRITE                ;WRITE A RECORD
3463  012756  005215                  INC     (R5)                    ;SET 'GO' BIT
3464  012760  004767  172252          JSR     PC,WAITRDY
3465  012764  102423                  BVS     99$
3466  012766  004337  005560          JSR     R3,@#TMCMD
3467  012772  000000                  .WORD   0
3468  012774  000000                  .WORD   0
3469  012776  000000                  .WORD   0
```

```
3470  013000  000026                    .WORD   WFMK
3471  013002  004767  171352            JSR     PC,TIMON                ;TURN TIMER ON
3472  013006  005215                    INC     (R5)                    ;SET 'GO' BIT
3473
3474  013010  105711            1$:     TSTB    (R1)                    ;BRANCH WHEN READY SETS
3475  013012  100402                    BMI     2$
3476  013014  000163  004450            JMP     TIMER(R3)               ;GO TO TIMER & RETURN VIA R2
3477
3478  013020  004767  172212    2$:     JSR     PC,WAITRDY
3479  013024  102403                    BVS     99$
3480  013026  004767  171454            JSR     PC,TIMOK
3481  013032  000401                    BR      100$
3482
3483  013034  104400            99$:    HLT
3484  013036                    100$:
3485  013036  004767  172300            JSR     PC,.REWIND              ;REWIND SLAVE
3486  013042  102774                    BVS     99$                     ;BRANCH IF ERROR ON REWIND
3487  013044  104000                    SCOPE
3488
```

```
 3489  013046  012700  000012          FINISH: MOV     #10.,R0                 ;SET LINE FEED COUNT
 3490  013052  000004  001400          1$:     TYPE,CRLF
 3491  013056  005300                          DEC     R0
 3492  013060  001374                          BNE     1$
 3493  013062  032777  000100  165710          BIT     #SW06,@SWR
 3494  013070  001410                          BEQ     2$
 3495  013072  113700  001010                  MOVB    @#DRVNUM,R0
 3496  013076  113701  001011                  MOVB    @#SLVNUM,R1
 3497  013102  113702  001012                  MOVB    @#SLVPTR,R2
 3498  013106  000137  007440                  JMP     @#TYPHDR
 3499  013112  105237  001011          2$:     INCB    @#SLVNUM                ;SET NEXT SLAVE #
 3500  013116  005237  001012                  INC     @#SLVPTR                ;AND ITS POINTER
 3501  013122  122737  000010  001011          CMPB    #8.,@#SLVNUM            ;BRANCH IF LAST SLAVE (7)
 3502  013130  001402                          BEQ     3$
 3503  013132  000137  007206                  JMP     @#BEGIN                 ;BEGIN TEST ON NEXT SLAVE
 3504  013136  105037  001011          3$:     CLRB    @#SLVNUM                ;SET SLAVE #0
 3505  013142  105237  001010                  INCB    @#DRVNUM                ;AND INCREMENT DRIVE #
 3506  013146  122737  000010  001010          CMPB    #8.,@#DRVNUM            ;AND CHECK IF LAST DRIVE
 3507  013154  001402                          BEQ     END
 3508  013156  000137  007206                  JMP     @#BEGIN
 3509
 3510  013162                          END:
 3511
 3512                                  ;       ********** MODIFIED 8-11-77 ********************************
 3513
 3514                                  ;               ACT11 END OF PASS
 3515
 3516                                  ;       +
 3517                                  ;       +
 3518
 3519  013162  005767  165576                  TST     AUTOM                   ;AUTOMATIC  MODE?
 3520  013166  001015                          BNE     2$                      ;BRANCH - IF YES
 3521  013170  105737  001131                  TSTB    @#UNTFND                ;BRANCH IF A UNIT WAS FOUND
 3522  013174  001004                          BNE     1$
 3523  013176  000004  014567                  TYPE,E.UNIT
 3524  013202  000137  005730                  JMP     @#INIT
 3525  013206  000000                  1$:     HALT
 3526  013210  004767  166560                  JSR     PC,CKSWR                ;CHECK FOR CNTL G
 3527  013214  000005                          RESET
 3528  013216  000137  005730                  JMP     @#INIT                  ;RESTART
 3529
 3530  013222  000004  014014          2$:     TYPE,MSGA                       ;PRINT END OF PASS
 3531  013226  005267  165552                  INC     PCNTR                   ;BUMP PASS COUNTER
 3532  013232  013702  001004                  MOV     @#PCNTR,R2              ;GET END OF PASS COUNT
 3533  013236  004767  167372                  JSR     PC,TYPOCT               ;AND TYPE IT
 3534  013242  005767  165534                  TST     PAFLG                   ;PASS INDICATOR SET?
 3535  013246  001002                          BNE     3$                      ;BRANCH - IF YES
 3536  013250  005267  165526                  INC     PAFLG                   ;SET PASS INDICATOR
 3537  013254  013704  000042          3$:     MOV     @#42,R4                 ;CONTENTS OF 42 TO R4
 3538  013260  001405                          BEQ     HERE                    ;BRANCH IF NOT AUTO MODE
 3539  013262  000005                          RESET                           ;CLEAR THE WORLD
 3540  013264  004714                  $ENDAD: JSR     PC,(R4)                 ;RETURN TO MONITOR
 3541  013266  000240                          NOP
 3542  013270  000240                          NOP
 3543  013272  000240                          NOP
 3544  013274                          HERE:
```

```
3545  013274  005767  165464                TST     AUTOM                   ;CHECK FOR AUTOM MODE
3546  013300  001010                       BNE     1$                      ;BRANCH - IF YES
3547  013302  032777  004000  165470       BIT     #4000,@SWR              ;SEE IF HALT ON PASS
3548  013310  001004                       BNE     1$                      ;BRANCH - IF NOT
3549  013312  000000                       HALT
3550  013314  004767  166454               JSR     PC,CKSWR                ;CHECK FOR CONTOL G
3551  013320  000005                       RESET                           ;CLEAR THE WORLD
3552  013322  000137  005730          1$:  JMP:    @#INIT                  ;RESTART
3553
3554                                  ;    ********************************************************************
3555
3556
```

```
                                        ;SKEW TAPE TIMING TESTS
                                        ;THE FOLLOWING TESTS REQUIRE A SPECIALLY WRITTEN 800 BPI SKEW TAPE
013326  012737  013334  001006  SKEWTST: MOV     #TST031,@#SCPADR        ;SET SCOPE POINTER

                                        ;TEST 031- SKEW TAPE SPEED TEST-FORWARD
                                        ;THIS TEST READS 32" OF TAPE (26400.-800. = 25600. FRAMES),THEN
                                        ;DIVIDES TIME BY 32. TO GET TIME TO READ 1" (800. FRAMES) OF TAPE.
013334  112737  000031  001126  TST031: MOVB    #31,@#TSTNUM
013342  012702  013420                  MOV     #2$,R2                  ;SET RETURN PC FROM TIMER
013346  004767  171770                  JSR     PC,..REWIND             ;REWIND SLAVE
013352  102441                          BVS     99$                     ;BRANCH IF ERROR ON REWIND
013354  052765  001300  000032          BIS     #BPI800+NORM11,TC(R5)   ;SET 800 BPI
013362  052765  000010  000010          BIS     #BAI,CS2(R5)            ;INHIBIT BUS ADDRESS INCREMENT
013370  004337  005560                  JSR     R3,@#TMCMD              ;READ 32" OF TAPE-FORWARD
013374  016420                          .WORD   RDBUF
013376  177777                          .WORD   -1.
013400  063440          10$:            .WORD   26400.                  ;FRAME COUNT
013402  000070                          .WORD   RDFWD
013404  005215                          INC     (R5)                    ;SET 'GO' BIT

013406  022710  001440          1$:     CMP     #800.,(R0)              ;WAIT FOR FIRST 800 FRAMES
013412  101375                          BHI     1$                      ;TO BE READ

013414  004767  170740                  JSR     PC,TIMON                ;TURN TIMER ON
013420  023710  013400          2$:     CMP     @#10$,(R0)              ;WAIT FOR READING TO FINISH
013424  103402                          BLO     3$
013426  000163  004450                  JMP     TIMER(R3)               ;GO TO TIMER & RETURN VIA R2

013432  012700  000005          3$:     MOV     #5,R0                   ;DIVIDE TIME BY 32.
013436  006204                  4$:     ASR     R4
013440  005300                          DEC     R0
013442  001375                          BNE     4$
013444  004767  171530                  JSR     PC,RHINIT               ;INIT DRIVE
013450  004767  171032                  JSR     PC,TIMOK                ;CHECK TIME
013454  000401                          BR      100$

013456  104400          99$:    HLT
013460  104000          100$:   SCOPE

                                        ;TEST 032-SKEW TAPE SPEED TEST-REVERSE
                                        ;THIS TEST READS FORWARD 40" (32000. FRAMES) OF TAPE, THEN READS REVERSE
                                        ;32" (26400.-800. = 25600. FRAMES) OF TAPE. THE TIME IS THEN DIVIDED BY
                                        ;32. TO GET TIME TO READ 1" (800. FRAMES) OF TAPE.
013462  112737  000032  001126  TST032: MOVB    #32,@#TSTNUM
013470  012702  013616                  MOV     #3$,R2                  ;SET RETURN PC FROM TIMER
013474  004767  171642                  JSR     PC,..REWIND             ;REWIND SLAVE
013500  102465                          BVS     99$                     ;BRANCH IF ERROR ON REWIND
013502  052765  001300  000032          BIS     #BPI800+NORM11,TC(R5)
013510  052765  000010  000010          BIS     #BAI,CS2(R5)
013516  004337  005560                  JSR     R3,@#TMCMD              ;READ FORWARD 32000. FRAMES
013522  016420                          .WORD   RDBUF
013524  177777                          .WORD   -1.                     ;WORD COUNT
013526  076400          10$:    .WORD   32000.                  ;FRAME COUNT
013530  000070                  .WORD   RDFWD                   ;READ FORWARD
013532  005215                  INC     (R5)                    ;SET 'GO' BIT
```

```
3613  013534  023710  013526          1$:     CMP     @#10$,(R0)
3614  013540  101375                          BHI     1$
3615
3616  013542  004767  171432                  JSR     PC,RHINIT               ;INIT DRIVE
3617  013546  004767  171172                  JSR     PC,DELAY                ;WAIT FOR TAPE MOTION TO STOP
3618  013552  052765  001300  000032          BIS     #BPI800+NORM11,TC(R5)   ;SET 800 BPI
3619  013560  052765  000010  000010          BIS     #BAI,CS2(R5)            ;INHIBIT BUS ADDRESS INCRENENT
3620  013566  004337  005560                  JSR     R3,@#TMCMD              ;READ REVERSE 32" OF TAPE
3621  013572  016420                          .WORD   RDBUF                   ;READ BUFFER
3622  013574  177777                          .WORD   -1.                     ;WORD COUNT
3623  013576  063440          11$:            .WORD   26400.                  ;FRAME COUNT
3624  013600  000076                          .WORD   RDREV                   ;READ REVERSE
3625  013602  005215                          INC     (R5)                    ;SET 'GO' BIT
3626
3627  013604  022710  001440          2$:     CMP     #800.,(R0)              ;WAIT FOR FIRST 800 FRAMES
3628  013610  101375                          BHI     2$                      ;TO BE READ
3629
3630  013612  004767  170542                  JSR     PC,TIMON                ;TURN TIMER ON
3631  013616  023710  013576          3$:     CMP     @#11$,(R0)              ;WAIT FOR ALL FRAMES TO BE READ
3632  013622  103402                          BLO     4$
3633  013624  000163  004450                  JMP     TIMER(R3)               ;GO TO TIMER & RETURN VIA R2
3634
3635  013630  012700  000005          4$:     MOV     #5,R0                   ;DIVIDE TIME BY 32.
3636  013634  006204          5$:             ASR     R4
3637  013636  005300                          DEC     R0
3638  013640  001375                          BNE     5$
3639  013642  004767  171332                  JSR     PC,RHINIT
3640  013646  004767  170634                  JSR     PC,TIMOK
3641  013652  000401                          BR      100$
3642
3643  013654  104400          99$:            HLT
3644  013656                  100$:
3645  013656  004767  171460                  JSR     PC,..REWIND             ;REWIND SLAVE
3646  013662  102774                          BVS     99$                     ;BRANCH IF ERROR ON REWIND
3647  013664  104000                          SCOPE
3648
3649  013666  000137  013046                  JMP     @#FINISH
3650
```

```
3651
3652                                    ;       *******INSERTED 11-8-77 **********************************
3653
3654                                    ;               CHECK FOR DUMP MODE OR AUTOMATIC MODE
3655                                    ;       +
3656                                    ;       +
3657
3658   013672  005067  165066          CKMODE: CLR     AUTOM                   ;INIT AUTO MODE
3659   013676  005737  000042                  TST     @#42                    ;AUTOMATIC MODE?
3660   013702  001417                          BEQ     2$                      ;BRANCH - IF NOT
3661   013704  005267  165054                  INC     AUTOM                   ;SET AUTO MODE INDICATOR
3662   013710  023737  000042  000046          CMP     @#42,@#46               ;ACT11 MODE?
3663   013716  001403                          BEQ     1$                      ;BRANCH - IF YES
3664   013720  105267  165043                  INCB    XXDPM                   ;INDICATE XXDP AUTO MODE
3665   013724  000416                          BR      5$                      ;AND EXIT
3666   013726  105267  165034          1$:     INCB    ACT11M                  ;INDICATE ACT11 AUTO MODE
3667   013732  052777  104100  165040          BIS     #104100,@SWR            ;CON: CYCLE,HALT ON ERROR,INHIBIT SUBTEST
3668   013740  000410                          BR      5$                      ;AND EXIT
3669   013742  105737  000041          2$:     TSTB    @#41                    ;MAN: MODE VIA ACT11/PAPER TAPE?
3670   013746  001003                          BNE     3$                      ;BRANCH - IF NOT
3671   013750  105267  165014                  INCB    ADUMPM                  ;INDICATE MAN: MODE VIA ACT11/PAPER TAPE
3672   013754  000402                          BR      5$                      ;AND EXIT
3673   013756  105267  165007          3$:     INCB    XDUMPM                  ;INDICATE MANUAL MODE VIA XXDP
3674   013762  000207                  5$:     RTS     PC                      ;RETURN
3675
3676                                    ;       ********************************************************************
3677
```

```
3678
3679                                    ;       ******** INSERTED 11-8-77 ********************************
3680                                    ;               DISCONTINUE TESTING FOR
3681                                    ;               ILLEGAL CONDITIONS
3682
3683  013764                            ABORT:
3684  013764  000005                           RESET                           ;INITIALIZE THE WORLD
3685  013766  000004  014036                   TYPE,MSGB                       ;TYPE ABORT MESSAGE
3686  013772  105767  164771                   TSTB    XXDPM                   ;XXDP AUTO MODE
3687  013776  001405                           BEQ     1$                      ;BRANCH - IF NOT
3688  014000  013700  000042                   MOV     @#42,R0                 ;GET MONITOR EXIT ADDRESS
3689  014004  005037  000042                   CLR     @#42                    ;USE AS ABORT FLAG
3690  014010  004710                           JSR     PC,(R0)                 ;EXIT TO XXDP MONITOR
3691  014012  000777                   1$:     BR      .                       ;AND HANG
3692
3693                                    ;       ********************************************************************************
3694
```

```
                                        .SBTTL    PROGRAM MESSAGES

                                    ;OPERATOR INSTRUCTIONS
```

```
                                                                    B  8
3699                                    ;       ******* INSERTED 11-8-77 **********************************
3700
3701  014014  005015  047105  020104  MSGA:   .ASCIZ  <CR><LF>'END OF PASS    '
3702  014022  043117  050040  051501
3703  014030  020123  020040  000040
3704  014036  005015  051120  043517  MSGB:   .ASCIZ  <CR><LF>'PROGRAM ABORTED'
3705  014044  040522  020115  041101
3706  014052  051117  042524  000104
```

```
3707
3708                            ;       **********************************************************************
3709
```

```
3710
3711
3712  014060  005015  055103  052524  M.NAM:   .ASCIZ  <CR><LF>'CZTUGC TM2/TE16 DR FCNT TMR'
3713  014066  041507  052040  031115
3714  014074  052057  030505  020066
3715  014102  051104  043040  047103
3716  014110  020124  046524  000122
3717  014116  005015  054524  042520  I.REG:   .ASCIZ  <CR><LF>'TYPE FIRST ADDRESS OF CONTROLLER  '
3718  014124  043040  051111  052123
3719  014132  040440  042104  042522
3720  014140  051523  047440  020106
3721  014146  047503  052116  047522
3722  014154  046114  051105  020040
3723  014162     000
3724  014163     124  050131  020105  I.DRVS:  .ASCIZ  %TYPE TM02 DRIVE #'S TO BE TESTED %
3725  014170  046524  031060  042040
3726  014176  044522  042526  021440
3727  014204  051447  052040  020117
3728  014212  042502  052040  051505
3729  014220  042524  020104     000
3730  014225     106  051117  052040  I.SLVS:  .ASCII  'FOR TM02 DRIVE '
3731  014232  030115  020062  051104
3732  014240  053111  020105
3733  014244  026460  052040  050131  I.DRV:   .ASCIZ  %0- TYPE SLAVE #'S TO BE TESTED %
3734  014252  020105  046123  053101
3735  014260  020105  023443  020123
3736  014266  047524  041040  020105
3737  014274  042524  052123  042105
3738  014302  000040
3739  014304  050123  042505  020104  I.SKEW:  .ASCIZ  'SPEED TESTS ONLY? (YES/NO = 1/0)'
3740  014312  042524  052123  020123
3741  014320  047117  054514  020077
3742  014326  054450  051505  047057
3743  014334  020117  020075  027461
3744  014342  024460     000
3745  014345     116  055122  047440  I.NRZ:   .ASCIZ  'NRZ ONLY? (YES/NO = 1/0)'
3746  014352  046116  037531  024040
3747  014360  042531  027523  047516
3748  014366  036440  030440  030057
3749  014374  000051
```

```
 3750  014376  005015  047105  020104  M.EOT:   .ASCIZ  <CR><LF>'END OF TAPE'<CR><LF>
 3751  014404  043117  052040  050101
 3752  014412  006505  000012
 3753
 3754                                  ;ERROR MESSAGES
 3755  014416  005015  051124  050101  E.TRP4:  .ASCIZ  <CR><LF>'TRAPPED TO 4'
 3756  014424  042520  020104  047524
 3757  014432  032040     000
 3758  014435     116  020117  047503  E.NCON:  .ASCIZ  'NO CONTROLLER AT ADDRESS SPECIFIED'<CR><LF>
 3759  014442  052116  047522  046114
 3760  014450  051105  040440  020124
 3761  014456  042101  051104  051505
 3762  014464  020123  050123  041505
 3763  014472  043111  042511  006504
 3764  014500  000012
 3765  014502  046524  031060  042040  E.NDRV:  .ASCIZ  'TM02 DRIVE '
 3766  014510  044522  042526  000040
 3767  014516  051104  053111  020105  E.NSLV:  .ASCII  'DRIVE '
```

```
3768  014524  020060  046123  053101  E.DRV:   .ASCII  'D SLAVE '
3769  014532  020105
3770  014534  020060  047516  020124  E.NAVA:  .ASCIZ  'D NOT AVAILABLE FOR TEST'<CR><LF>
3771  014542  053101  044501  040514
3772  014550  046102  020105  047506
3773  014556  020122  042524  052123
3774  014564  005015     000
3775  014567     116  020117  046524  E.UNIT:  .ASCIZ  'NO TM02/TE16 UNIT FOUND TO TEST'<CR><LF>
3776  014574  031060  052057  030505
3777  014602  020066  047125  052111
3778  014610  043040  052517  042116
3779  014616  052040  020117  042524
3780  014624  052123  005015     000
3781  014631     123  043117  020124  E.SFT:   .ASCIZ  'SOFT ERROR (DATA)'<CR><LF>
3782  014636  051105  047522  020122
3783  014644  042050  052101  024501
3784  014652  005015     000
3785  014655     124  051505  020124  E.HDR:   .ASCIZ  'TEST # '
3786  014662  020043     000
3787  014665     040  042504  044526  E.HDR1:  .ASCII  ' DEVICE ERROR'<CR><LF>
3788  014672  042503  042440  051122
3789  014700  051117  005015
3790  014704  051503  004461  041527           .ASCIZ  'CS1'<HT>'WC'<HT>'BA'<HT>'FC'<HT>'CS2'<HT>'DS'<HT>'ER'<HT>'TC'<CR><LF>
3791  014712  041011  004501  041506
3792  014720  041411  031123  042011
3793  014726  004523  051105  052011
3794  014734  006503  000012
3795  014740  047440  052125  047440  E.HDR2:  .ASCIZ  ' OUT OF RANGE ERROR'<CR><LF>
3796  014746  020106  040522  043516
3797  014754  020105  051105  047522
3798  014762  006522  000012
3799  014766  005015  044524  042515  E.TIMOV:         .ASCIZ  <CR><LF>'TIMER OVERFLOWED'<CR><LF>
3800  014774  020122  053117  051105
3801  015002  046106  053517  042105
3802  015010  005015     000
3803  015013     015  052012  046511  E.TIMEX:         .ASCIZ  <CR><LF>'TIME EXPIRED WAITING FOR RDY'<CR><LF>
3804  015020  020105  054105  044520
3805  015026  042522  020104  040527
3806  015034  052111  047111  020107
3807  015042  047506  020122  042122
3808  015050  006531  000012
3809  015054  043440  050101  021440  E.GAP:   .ASCIZ  ' GAP # '
3810  015062  000040
3811
3812                                   ;TIME DOCUMENT LINES
3813  015064  025052  025052  025052  L.HDR1:  .ASCIZ  '**********************************************************************************
3814  015072  025052  025052  025052
3815  015100  025052  025052  025052
3816  015106  025052  025052  025052
3817  015114  025052  025052  025052
3818  015122  025052  025052  025052
3819  015130  025052  025052  025052
3820  015136  025052  025052  025052
3821  015144  025052  025052  025052
3822  015152  025052  025052  025052
3823  015160  025052  025052  025052
```

```
3824  015166  025052  025052  025052
3825  015174  005015     000
3826  015177     052  052040  030115  L.HDR2:  .ASCII  '* TM02 DRIVE FUNCTION TIMES- DRIVE # '
3827  015204  020062  051104  053111
3828  015212  020105  052506  041516
3829  015220  044524  047117  052040
3830  015226  046511  051505  020055
3831  015234  051104  053111  020105
3832  015242  020043
3833  015244  020060  046123  053101  L.DRV:   .ASCII  '0 SLAVE # '
3834  015252  020105  020043
3835  015256  020060     040        L.SLV:   .ASCII  '0  '
3836  015261     071  041440  040510  L.CHAN:  .ASCIZ  '9 CHAN.  SER # '
3837  015266  027116  051440  051105
3838  015274  021440  000040
3839  015300  006440  025012  005015  L.HDR3:  .ASCII  ' '<CR><LF>'*'<CR><LF>
3840  015306  020052  052506  041516           .ASCIZ  '* FUNCTION'<HT><HT>'TIME(SPECIFICATION)'<HT>'TIME(ACTUAL)'<CR><LF>
3841  015314  044524  047117  004411
3842  015322  044524  042515  051450
3843  015330  042520  044503  044506
3844  015336  040503  044524  047117
3845  015344  004451  044524  042515
3846  015352  040450  052103  040525
3847  015360  024514  005015     000
3848
3849  015365     122  047101  042507  L.RNG:   .ASCIZ  'RANGE=<'
3850  015372  036075     000
3851  015375     101  052103  040525  L.ACT:   .ASCIZ  'ACTUAL='
3852  015402  036514     000
3853
3854                                  ;TEST DESCRIPTOR HEADERS
3855  015405     052  053440  044522  A.T001:  .ASCIZ  '* WRITE FROM BOT'<HT>
3856  015412  042524  043040  047522
3857  015420  020115  047502  004524
3858  015426     000
3859  015427     052  053440  044522  A.T002:  .ASCIZ  '* WRITE START'<HT><HT>
3860  015434  042524  051440  040524
3861  015442  052122  004411     000
3862  015447     052  053440  044522  A.T003:  .ASCIZ  '* WRITE SHUTDOWN'<HT>
3863  015454  042524  051440  052510
3864  015462  042124  053517  004516
3865  015470     000
3866  015471     052  053440  044522  A.T004:  .ASCIZ  '* WRITE SETTLEDOWN'<HT>
3867  015476  042524  051440  052105
3868  015504  046124  042105  053517
3869  015512  004516     000
3870  015515     052  051040  040505  A.T005:  .ASCIZ  '* READ FROM BOT'<HT><HT>
3871  015522  020104  051106  046517
3872  015530  041040  052117  004411
3873  015536     000
3874  015537     052  051040  040505  A.T006:  .ASCIZ  '* READ START'<HT><HT>
3875  015544  020104  052123  051101
3876  015552  004524  000011
3877  015556  020052  042522  042101  A.T007:  .ASCIZ  '* READ SHUTDOWN'<HT><HT>
3878  015564  051440  052510  042124
3879  015572  053517  004516  000011
```

```
3880  015600  020052  042522  042101  A.TO10:  .ASCIZ  '* READ SETTLEDOWN'<HT>
3881  015606  051440  052105  046124
3882  015614  042105  053517  004516
3883  015622     000
3884  015623     052  051040  040505  A.TO11:  .ASCIZ  '* READ REV START'<HT>
3885  015630  020104  042522  020126
3886  015636  052123  051101  004524
3887  015644     000
3888  015645     052  051040  040505  A.TO12:  .ASCIZ  '* READ REV SHUTDOWN'<HT>
3889  015652  020104  042522  020126
3890  015660  044123  052125  047504
3891  015666  047127  000011
3892  015672  020052  042522  042101  A.TO13:  .ASCIZ  '* READ REV SETTLEDOWN'<HT>
3893  015700  051040  053105  051440
3894  015706  052105  046124  042105
3895  015714  053517  004516     000
3896  015721     052  052040  051125  A.TO14:  .ASCIZ  '* TURN AROUND DELAY F-R'<HT>
3897  015726  020116  051101  052517
3898  015734  042116  042040  046105
3899  015742  054501  043040  051055
3900  015750  000011
3901  015752  020052  052524  047122  A.TO15:  .ASCIZ  '* TURN AROUND DELAY R-F'<HT>
3902  015760  040440  047522  047125
3903  015766  020104  042504  040514
3904  015774  020131  026522  004506
3905  016002     000
3906  016003     052  043440  050101  A.TO16:  .ASCIZ  '* GAP SIZE-STOP HALF'<HT>
3907  016010  051440  055111  026505
3908  016016  052123  050117  044040
3909  016024  046101  004506     000
3910  016031     052  043440  050101  A.TO17:  .ASCIZ  '* GAP SIZE-START HALF'<HT>
3911  016036  051440  055111  026505
3912  016044  052123  051101  020124
3913  016052  040510  043114  000011
3914  016060  020052  040507  020120  A.TO20:  .ASCIZ  '* GAP SIZE-INTERRECORD'<HT>
3915  016066  044523  042532  044455
3916  016074  052116  051105  042522
3917  016102  047503  042122  000011
3918  016110  020052  040507  020120  A.TO21:  .ASCIZ  '* GAP CONSISTANCY'<HT>
3919  016116  047503  051516  051511
3920  016124  040524  041516  004531
3921  016132     000
3922  016133     052  042040  052101  A.TO23:  .ASCIZ  '* DATA TIME-200BPI'<HT>
3923  016140  020101  044524  042515
3924  016146  031055  030060  050102
3925  016154  004511     000
3926  016157     052  042040  052101  A.TO24:  .ASCIZ  '* DATA TIME-556BPI'<HT>
3927  016164  020101  044524  042515
3928  016172  032455  033065  050102
3929  016200  004511     000
3930  016203     052  042040  052101  A.TO25:  .ASCIZ  '* DATA TIME-800BPI'<HT>
3931  016210  020101  044524  042515
3932  016216  034055  030060  050102
3933  016224  004511     000
3934  016227     052  042040  052101  A.TO26:  .ASCIZ  '* DATA TIME-1600BPI'<HT>
3935  016234  020101  044524  042515
```

```
  3936  016242  030455  030066  041060
  3937  016250  044520  000011
```

```
                                                                J  8
 3938  016254  020052  051105  051501  A.TO27: .ASCIZ  '* ERASE GAP TIME'<HT>
 3939  016262  020105  040507  020120
 3940  016270  044524  042515  000011
 3941  016276  020052  051127  052111  A.TO30: .ASCIZ  '* WRITE FILE MARK'<HT>
 3942  016304  020105  044506  042514
 3943  016312  046440  051101  004513
 3944  016320     000
 3945  016321     052  052040  050101  A.TO31: .ASCIZ  '* TAPE SPEED-FWD'<HT>
 3946  016326  020105  050123  042505
 3947  016334  026504  053506  004504
 3948  016342     000
 3949  016343     052  052040  050101  A.TO32: .ASCIZ  '* TAPE SPEED-REV'<HT>
 3950  016350  020105  050123  042505
 3951  016356  026504  042522  004526
 3952  016364     000
 3953
 3954  016365     015  057012  000107  L.CNTG: .ASCIZ  <CR><LF>' G'
 3955  016372  005015  053523  036522  L.SWR:  .ASCIZ  <CR><LF>'SWR='
 3956  016400     000
 3957  016401     040  047040  053505  L.NEW:  .ASCIZ  '  NEW= '
 3958  016406  020075     000
 3959  016411     015  037412  005015  L.QUEST:        .ASCIZ  <CR><LF>'?'<CR><LF>
 3960  016416     000
 3961          016420                          .EVEN
 3962          016420                          RDBUF=.
 3963          016420                          WTBUF=.
 3964  016420  000200                          .BLKW   128.
 3965          000001                          .END
```

```
A       011050            3050#   3055
ABORT   013764            2419    3683#
ACCL  = 100000            1300#
ACT11M  000766            1062#   2450    3666*
ADUMPM  000770            1064#   3671*
ANGTAB  001416            1392#   1750    1778
AS    = 000016            1199#
ASFLG   001134            1372#   2440*   2684    2686*
ATA   = 100000            1268#
ATIME   001016            1358#   1752    1780    1844*   2003    2101*   2105    2107    2139*   2143    2145
ATIMTB  001020            1359#   1822    2003*
AUTOM   000764            1061#   2413    2693    3519    3545    3658*   3661*
A.TO01  015405            1456    3855#
A.TO02  015427            1457    3859#
A.TO03  015447            1458    3862#
A.TO04  015471            1459    3866#
A.TO05  015515            1460    3870#
A.TO06  015537            1461    3874#
A.TO07  015556            1462    3877#
A.TO10  015600            1463    3880#
A.TO11  015623            1464    3884#
A.TO12  015645            1465    3888#
A.TO13  015672            1466    3892#
A.TO14  015721            1467    3896#
A.TO15  015752            1468    3901#
A.TO16  016003            1469    3906#
A.TO17  016031            1470    3910#
A.TO20  016060            1471    3914#
A.TO21  016110            1472    3918#
A.TO23  016133            1474    3922#
A.TO24  016157            1475    3926#
A.TO25  016203            1476    3930#
A.TO26  016227            1477    3934#
A.TO27  016254            1478    3938#
A.TO30  016276            1479    3941#
A.TO31  016321            1480    3945#
A.TO32  016343            1481    3949#
A16   = 000400            1224#
A17   = 001000            1225#
BA    = 000004            1194#   2359*
BAI   = 000010            1240#   3569    3605    3619
BEGIN   007206            2658#   3503    3508
BELL    001407            1388#   1956
BKSLSH  001403            1386#   1871
BOT   = 000002            1255#   2292
BPI200= 000000            1296#
BPI556= 000400            1297#   3328
BPI800= 001000            1298#   3365    3568    3604    3618
BPTVEC= 000014            1168#
CDM11 = 000320            1295#
CHKDRV  006342            2491    2495    2506#
CHKSLV  006646            2542    2560    2574#
CH7   = 010000            1290#   2727
CKMODE  013672            2412    3658#
CKSWR   001774            1492#   1910    1960    1981    2022    3526    3550
CLR   = 000040            1242#
```

```
CNTLU   002034          1499#   1510    2435
CNTRLC= 000003          1312#
CNTRLO= 000017          1316#   1891
CNTRLU= 000025          1317#   1873
CNVDEC  002734          1691#
CNVOCT  002626          1651#
CNVTAO  003264          1793#   2462
CNVTD   002746          1692    1699#
CNVTO   002640          1652    1660#
COUNT   001770          1489#   1505*   1515    1529*
CR    = 000015          1315#   1385    1799    1880    2494    2562    3701    3704    3712    3717    3750    3755    3758
                        3770    3775    3781    3787    3790    3795    3799    3803    3813    3839    3840    3954    3955
                        3959
CRLF    001400          1385#   1754    1785    1876    1884    1926    1929    1952    2458    3490
CSITM = 002000          1279#
CS1   = 000000          1192#   2638
CS2   = 000010          1196#   2225*   2236*   2246*   2247*   2632*   2637*   2639    3569*   3605*   3619*
DASH    001411          1389#   1747    1775
DB    = 000022          1200#
DCONST  003042          1704    1708    1711    1721#
DELAY   004744          1989    2159#   2349    2964    3120    3148    3617
DELAYV  004774          2172#   3219
DELTIM  001120          1362#   2172    2179    3213*   3220*
DIGTAB  001136          1374#   1675    1709    2517    2547    2598    2599    2724    2725
DIVIDE  005032          2099    2137    2197#
DLT   = 100000          1252#
DPR   = 000400          1262#
DRIVE   000040          1034#
DRIVES  006212          2474#   2525
DRVAVA  005122          2225#   2514
DRVNUM  001010          1354#   2225    2236    2247    2513*   2546*   2594*   2615*   2630*   2633*   2634    2637    2643
                        2658    2680*   3495    3505*   3506
DRVTBL  001160          1382#   2475    2486    2501*   2507    2520*   2536    2577    2661
DRY   = 000200          1261#
DRYCLR= 000010          1212#
DS    = 000012          1197#   1994    2257    2263    2273    2292    2294    2298    2737    2943
DT    = 000026          1202#   2226    2238    2651    2653    2727
DTE   = 010000          1281#
DVA   = 004000          1227#
DV0   = 000000          1232#
DV1   = 000001          1233#
DV2   = 000002          1234#
DV3   = 000003          1235#
DV4   = 000004          1236#
DV5   = 000005          1237#
DV6   = 000006          1238#
DV7   = 000007          1239#
ECHO    001405          1387#   1878*   1882
EMTVEC= 000030          1171#
END     013162          2668    3507    3510#
EOT   = 002000          1263#   2263
ER    = 000014          1198#   1931
ERASE = 000024          1214#   3442
ERFLG   001127          1367#   1912*   1991*   1999    2439*   2474*   2519*   2522    2528*   2601*   2605
ERR   = 040000          1267#   2273    2294    2796    2822    2828    2901    2931    2937    2994    3025    3031    3302
                        3338    3375    3414
```

```
ERRTRP  003654          1903#   2472    2609
ERRVEC= 000004          1164#   2441*   2442*   2472*   2609*
E.DRV   014524          2598*   3768#
E.GAP   015054          1782    3809#
E.HDR   014655          1915    3785#
E.HDR1  014665          1942    3787#
E.HDR2  014740          1922    3795#
E.NAVA  014534          2517*   2518    2599*   3770#
E.NCON  014435          2470    3758#
E.NDRV  014502          2516    3765#
E.NSLV  014516          2600    3767#
E.SFT   014631          1939    3781#
E.TIME  015013          2261    3803#
E.TIMO  014766          2067    3799#
E.TRP4  014416          3755#
E.UNIT  014567          3523    3775#
FC    = 000006          1195#   1996    2335*   2345*   2361*   2735    3243
FCE   = 001000          1278#
FINISH  013046          2636    3489#   3649
FMT   = 000020          1274#
FPEVEC= 000244          1177#
FRMCNT= 177400          1309#   2310    2320    2330    2899    2992
FWDSPC  005476          2335#
GAP     001124          1364#   1769    1783    2140    3253*   3259*   3260
GAPOK   004616          2123#   3258
GAPTBL  001060          1361#   3233    3255    3264
GO    = 000001          1208#
GTIMTB  001576          1437#   1773    1776    2143    2145
HERE    013274          3538    3544#
HLT   = 104400          1303#   2068    2109    2147    2765    2785    2811    2844    2866    2888    2917    2952    2978
                        3009    3046    3054    3080    3110    3134    3162    3194    3280    3320    3357    3394    3431
                        3455    3483    3593    3643
MT    = 000011          1313#   1392    3790    3840    3855    3859    3862    3866    3870    3874    3877    3880    3884
                        3888    3892    3896    3901    3906    3910    3914    3918    3922    3926    3930    3934    3938
                        3941    3945    3949
IDB   = 000010          1257#
IE    = 000100          1222#
ILF   = 000001          1270#
ILR   = 000002          1271#
INBUF   001270          1384#   1795    1862    2443*   2482    2550    2700    2711
INCVAE= 000100          1275#
INIT    005730          1308    1341    1899    2402#   2471    3524    3528    3552
IOTVEC= 000020          1169#
IR    = 000100          1243#
ITCNT   001125          1365#   1823    2001    2004*   2007    2013*   2437*
I.DRV   014244          2547*   3733#
I.DRVS  014163          2480    3724#
I.NRZ   014345          2709    3745#
I.REG   014116          2460    3717#
I.SKEW  014304          2698    3739#
I.SLVS  014225          2548    3730#
LF    = 000012          1314#   1385    3701    3704    3712    3717    3750    3755    3758    3770    3775    3781    3787
                        3790    3795    3799    3803    3813    3839    3840    3954    3955    3959
LKS   = 177546          1183#
LKVEC = 000100          1184#
LPB   = 177516          1186#
```

```
                                                                N  8
LPS    = 177514            1185#
L.ACT    015375            1751    1779    3851#
L.CHAN   015261            2726*   2729*   3836#
L.CNTG   016365            1498    3954#
L.DRV    015244            2724*   3833#
L.HDR1   015064            2723    3813#
L.HDR2   015177            2730    3826#
L.HDR3   015300            2732    3839#
L.NEW    016401            1502    3957#
L.QUES   016411            1522    3959#
L.RNG    015365            1744    1772    3849#
L.SLV    015256            2725*   3835#
L.SWR    016372            1499    3955#
MCPE   = 020000            1228#
MDPE   = 000400            1245#
MEDIUM   000041            1037#
MMVEC  = 000250            1178#
MOL    = 010000            1265#
MR     = 000024            1201#   2037    2039    2044    2058    2073
MSGA     014014            3530    3701#
MSGB     014036            2418    3685    3704#
MXF    = 001000            1246#
M.EOT    014376            2266    3750#
M.NAM    014060            2265    2459    3712#
NAMPTR   001676            1455#   1848
NED    = 010000            1249#
NEF    = 004000            1280#
NEM    = 004000            1248#
NOP    = 000000            1209#
NORM11 = 000300            1294#   2251    3328    3365    3404    3568    3604    3618
NRZFLG   001133            1371#   2708*   2716*   3399
NSG    = 000400            1277#
NXTDRV   007044            2631#   2649
NXTSLV   007122            2641#   2652    2654
OCTALO   001122            1363#   1796    2463
ODIGIT   001150            1379#   1662    1681    1702    1716    2369    2396
OPI    = 020000            1282#
OR     = 000200            1244#
OSC    = 000100            1286#   2037    2039    2044    2058    2073
OUT      002244            1493    1497    1517    1534#
OUTBUF = 005730            1308#
OUTGAP   003152            1767#   1925    2151
OUTSPC   003056            1739#   1852    1928    1986
PAFLG    001002            1350#   3534    3536*
PARVEC = 000114            1175#
PAT    = 000020            1241#
PCNTR    001004            1340*   1351#   3531*   3532
PEFLRC = 000200            1276#
PES    = 000040            1259#
PE1600 = 002000            1299#   1932    3404
PFVEC  = 000024            1170#   1324
PGE    = 002000            1247#
PIP    = 020000            1266#   2298
PIRQ   = 177772            1156#
PIRVEC = 000240            1176#
PLKCSR = 172540            1181#
```

```
PLKVEC= 000104  1182#
PRGFLG  001130  1368#  2436*  2485*  2524  2544  2553*  2559  2607  2687*  2705*  2739
PSEL  = 002000  1226#
PSW   = 177776  1154#
PUBLIS  003352  1820#  2012
RDBUF = 016420  2318  2328  3571  3607  3621  3962#
RDFWD = 000070  1220#  2321  3574  3610
RDREV = 000076  1221#  2331  3624
RDSW    001772  1490#  1513*  1580  1582*
RDY   = 000200  1223#
READ    005442  2317#  2852  2874  2896  2926  3097  3149  3234  3239
RESVEC= 000010  1165#
REVRD   005460  2327#  2965  2989  3020  3067  3091  3121  3144  3177  3181  3225
RHINIT  005200  1990  2228  2246#  2285  3589  3616  3639
RMR   = 000004  1272#
RWD   = 000006  1211#  2290
RWDOFF= 000002  1210#
SC    = 100000  1230#
SCOPE = 104000  1304#  2766  2786  2812  2845  2867  2889  2918  2953  2979  3010  3047  3081
                3111  3135  3163  3195  3281  3321  3358  3395  3432  3456  3487  3594  3647
SCPADR  001006  1353#  1992  2009*  2069  2733*  3559*
SDWN  = 000020  1258#  2803  2826  2835  2908  2935  2943  3000  3029  3037
SIZE    007014  2452  2611#
SKEWTS  013326  2741  3559#
SLA   = 000001  1254#
SLAVES  006442  2523  2528#  2608
SLR   = 177774  1155#
SLVAVA  005150  2236#  2596
SLVNUM  001011  1355#  2237  2249  2595*  2616*  2631*  2641*  2647*  2648  2650  2659  2681*  3496
                3499*  3501  3504*
SLVPTR  001012  1356#  2617*  2660  2682*  3497  3500*
SLVTBL  001170  1383#  2529  2535  2554  2576  2617
SN    = 000030  1203#  2368  2378  2383  2391
SNPT    005600  2368#  2731
SPACE   001414  1391#
SPACE2  001413  1390#  1947
SPCFWD= 000030  1215#  2336
SPCREV= 000032  1216#  2346
SPR   = 002000  1289#  2238
SSC   = 000100  1260#
STIMTB  001422  1401#  1745  1748  2105  2107
STKPTR= 000600  1344#  2402
SUSWR   005770  2415  2417  2423#
SWR     001000  1348#  1492  1500  1532*  1913  1954  1957  1983  1987  2005  2010  2014  2016
                2149  2426  2430*  2433  3493  3547  3667*
SWREG   000176  1336#  1492  2430  2433
SW06  = 000100  1021#  3493
SW07  = 000200  1020#
SW08  = 000400  1019#  1983  2149
SW09  = 001000  1018#  1954
SW10  = 002000  1017#  2010
SW11  = 004000  1016#  2005
SW13  = 020000  1015#  1913
SW14  = 040000  1014#  1987
SW15  = 100000  1013#
TAP   = 040000  1291#  2226
```

```
TBITVE= 000014           1166#  1319
TC    = 000032           1204#  1932  1950  2237*  2250*  2251*  2650*  2757  2776  2857  2879  2969  3071
                         3101   3328*  3365*  3404*  3568*  3604*  3618*
TCRLF   002364           1514   1579#
TEMPST  001766           1488#  1504*  1524*  1525*  1526*  1528*  1532
TIB     001764           1487#  1494*  1495*  1496   1507*  1508   1511  1518  1520  1527*  1528  1541*  1545*
                         1548
TIMER   004450           2059   2063#  2071   2074   2163   2178   2738  2759  2778  2805   2837  2859   2881
                         2910   2945   2971   3002   3039   3073   3103  3127  3155  3187   3245  3310   3346
                         3383   3422   3448   3476   3583   3633
TIMERR  004460           2065   2067#
TIMERO  004474           2073#
TIMER1  004424           2058#  2062
TIMOK   004506           2085#  2763   2782   2809   2841   2863   2885  2914  2949  2975   3006  3043   3077
                         3107   3131   3159   3191   3277   3317   3354  3391  3428  3452   3480  3590   3640
TIMON   004360           2035#  2159   2174   2754   2773   2801   2833  2854  2876  2906   2942  2966   2999
                         3036   3068   3098   3122   3150   3182   3240  3307  3343  3380   3419  3443   3471
                         3580   3630
TKB   = 177562           1159#  1494   1540*  1545   1866   1889
TKISR   003612           1331   1889#
TKS   = 177560           1158#  1539*  1542*  1543   1863   2722*
TKVEC = 000060           1173#  1330
TMBASE  001014           1357#  1982   2463*  2464
TMCMD   005560           2286   2307   2317   2327   2359#  3293  3329  3366  3405  3438  3466  3570  3606
                         3620
TMCS1 = 172440           1189#  1357
TMK   = 000004           1256#
TPB   = 177566           1161#  1548*
TPS   = 177564           1160#  1546
TPVEC = 000064           1174#
TRAPVE= 000034           1172#
TRE   = 040000           1229#
TRTVEC= 000014           1167#
TSTNUM  001126           1366#  1846   1851   1916   1985   2019   2102  2438*  2749*  2770*  2790*  2816*  2849*
                         2871*  2893*  2922*  2958*  2983*  3014*  3060*  3085*  3114*  3138*  3167*  3209*  3285*
                         3289*  3324*  3361*  3398*  3436*  3460*  3564*  3600*
TST001  007566           2733   2740   2749#
TST002  007652           2770#
TST003  007730           2790#
TST004  010020           2816#
TST005  010126           2849#
TST006  010212           2871#
TST007  010276           2893#
TST010  010376           2922#
TST011  010516           2958#
TST012  010614           2983#
TST013  010724           3014#
TST014  011064           3060#
TST015  011156           3085#
TST016  011264           3114#
TST017  011360           3138#
TST020  011470           3167#
TST021  011610           3209#
TST022  012114           3285#
TST023  012122           3289#
TST024  012244           3324#
```

```
TST025  012374            3361#
TST026  012524            3398#
TST027  012654            3400    3436#
TST030  012740            3460#
TST031  013334            3559    3564#
TST032  013462            3600#
TTIN    002246            1506    1539#
TTIN1   002266            1543#   1544
TTIN2   002302            1546#   1547
TYPDEC  002742            1698#   1746    1749    1753    1774    1777    1781
TYPE  = 000004            1305#   1498    1499    1502    1522    1579    1681    1716    1744    1747    1750    1751    1754
                          1772    1775    1778    1779    1782    1785    1849    1871    1876    1882    1884    1915    1922
                          1926    1929    1939    1942    1947    1952    1956    2067    2261    2265    2266    2396    2418
                          2458    2459    2460    2470    2480    2516    2518    2548    2600    2698    2709    2723    2730
                          2732    3490    3523    3530    3685
TYPE1   002356            1577#   1594    1623
TYPE2   002404            1578    1584#
TYPE3   002412            1581    1587#
TYPE4   002416            1585    1590#
TYPFLG  001132            1370#   1651*   1659*   1679    1691*   1698*   1714
TYPHDR  007440            2694    2720#   3498
TYPOCT  002634            1501    1659#   1784    1917    1946    1951    3533
UBREAK= 177770            1157#
UNS   = 040000            1283#
UNTFND  001131            1369#   2618*   2679*   3521
UPE   = 020000            1250#
WAITRD  005236            2255#   2271    2337    2343    2347    2761    2780    2807    2839    2861    2883    2912    2947
                          2962    2973    2987    3004    3018    3041    3064    3075    3089    3094    3105    3118    3129
                          3142    3146    3157    3171    3175    3179    3189    3217    3227    3237    3247    3315    3352
                          3389    3426    3450    3464    3478
WAITTI  005240            2256#   2259*
WC    = 000002            1193#   2360*
WCE   = 040000            1251#
WCHKF = 000050            1217#
WCHKR = 000056            1218#
WFMK  = 000026            1213#   3470
WFWD  = 000060            1219#   2311    3297    3333    3370    3409
WRDCNT= 177600            1310#   2309    2319    2329
WRITE   005424            2269    2307#   2341    2753    2771    2791    2817    2960    2985    3016    3062    3087    3116
                          3140    3169    3173    3215    3462
WRL   = 004000            1264#
WRT.BK  005516            2341#   2872    2894    2924
WTBUF = 016420            2308    3294    3330    3367    3406    3963#
XDUMPM  000771            1065#   3673*
XXDPM   000767            1063#   2645    3664*   3686
$CHARC  002330            1567#   1610*   1614*   1620
$CNTRL  002331            1568#   1575*   1577    1605    1893*
$CRLF   002332            1569#   1579
$ENDAD  013264            1045    3540#
$FILL   002321            1561#
$HT   = 000011            1559#   1590
$NULL   002320            1560#   1595
$SVPC = 000764            1032#   1051
$TKFLG  002323            1564#
$TPB    002326            1566#   1607*
$TPFLG  002322            1562#
```

```
$TPS    002324          1565#   1603
.       = 017020        1024#   1032    1033#   1036#   1040#   1044#   1047#   1051#   1319#   1320    1330#   1335#   1338#
                        1343#   1346#   1359#   1361#   1373#   1379#   1381#   1382#   1383#   1384#   1393#   1570#   1604
                        1831    2062#   2071#   3691    3961#   3962    3963    3964#
. HLT   003656          1328    1910#
. INPUT 003500          1861#   2461    2481    2549    2699    2710
. RESTO 002604          1638#   1683    1718    1811    1853    1961    2111    2154
. REWIN 005342          2267    2285#   2751    2850    3051    3211    3291    3326    3363    3402    3485    3566    3602
                        3645
. SAVE  002562          1627#   1661    1700    1794    1821    1911    2086    2124
. SCOPE 004132          1326    1981#
. TYPE  002336          1322    1572#
```

CZTUGC.P11 02-DEC-77 09:46 CROSS REFERENCE TABLE -- MACRO NAMES

```
INPUT    1118#  2461  2481  2549  2699  2710
RESTOR   1115#  1682  1717  1810  1853  1961  2110  2153
REWIND   1121#  2267  2751  2850  3050  3211  3291  3326  3363  3402  3484  3566  3602  3644
SAVE     1112#  1660  1699  1793  1820  2085  2123
SETGO    1131#  2270  2291  2342  2877  2897  2927  2961  2967  2986  2990  3017  3021  3063  3069
         3088   3092  3099  3117  3123  3141  3145  3151  3170  3174  3178  3183  3216  3226  3235
         3241   3298  3334  3371  3410  3444  3463  3472  3575  3611  3625
TIMCHK   1128#  2162  2177  2759  2778  2805  2837  2859  2881  2910  2945  2971  3002  3039  3073
         3103   3127  3155  3187  3245  3310  3346  3383  3422  3448  3476  3583  3633
TIMEON   1125#  2174  2754  2800  2832  2854  2876  2905  2941  2966  2998  3035  3068  3098  3122
         3150   3182  3240  3306  3342  3379  3418  3443  3471  3580  3630
$CATCH    995#  1024
$CHNMO   1070#  2620
$CPREG    995#  1136
$CPVEC    995#  1163
$TYPE     995#  1553


. ABS.   017020      000


 ERRORS DETECTED:   0

 CZTUGC,CZTUGC.SEQ/SOL/NL:TOC/DOC=CZTUGC.SML/ML:CZTUGC.P11
 RUN-TIME:  4 7 .8 SECONDS
 RUN-TIME RATIO:  440/13=33.5
 CORE USED:   8K  (15 PAGES)

 DOCUMENT PAGES:  109
```

```
■y *■z P          ¢Σ      S
  ¢  &8  (v  ■y  y■        q■z          N
```