

TM03, TE16  
TU77

TM03/TE16, TU77 DFT  
CZTEEDO

AH-A804D-MC  
FICHE 1 OF 1

OCT 1983  
COPYRIGHT © 77-83  
MADE IN USA



The main body of the document is a large grid of data, likely a technical drawing or a set of tables. The grid is composed of approximately 10 columns and 15 rows of cells. Each cell contains text, possibly representing a specific data point or a small diagram. The text is very small and difficult to read, but it appears to be organized in a structured manner. There are some dark, irregular shapes on the right side of the grid, which could be artifacts from the scanning process or damage to the original document. A small white square is visible at the bottom center of the grid.

.REM %

IDENTIFICATION

PRODUCT CODE: AC-A803D-MC  
PRODUCT NAME: CZTEEDO TM03-TE16/TU77 DRIVE FUNCTION TIMER  
PRODUCT DATE: 11- JULY - 1983  
MAINTAINER: TAPE DIAGNOSTIC GROUP  
AUTHOR: J. G. ADAMS

REVISED TO REV C MIKE PAGE  
20 MAR 79  
;+C SHOWS CODE ADDED TO REV C.

REVISED TO REV D B. LEBLANC  
01 - MAY - 1983  
;BL FIXED FOR M8940 ECO

THE INFORMATION IN THIS DOCUMENT IS SUBJECT TO CHANGE WITHOUT NOTICE AND SHOULD NOT BE CONSTRUED AS A COMMITMENT BY DIGITAL EQUIPMENT CORPORATION. DIGITAL EQUIPMENT CORPORATION ASSUMES NO RESPONSIBILITY FOR ANY ERRORS THAT MAY APPEAR IN THIS DOCUMENT.

THE SOFTWARE DESCRIBED IN THIS DOCUMENT IS FURNISHED UNDER A LICENSE AND MAY ONLY BE USED OR COPIED IN ACCORDANCE WITH THE TERMS OF SUCH LICENSE.

DIGITAL EQUIPMENT CORPORATION ASSUMES NO RESPONSIBILITY FOR THE USE OR RELIABILITY OF ITS SOFTWARE ON EQUIPMENT THAT IS NOT SUPPLIED BY DIGITAL.

COPYRIGHT (C) 1977,1983 BY DIGITAL EQUIPMENT CORPORATION

TM03 DRIVE FUNCTION TIMER  
TABLE OF CONTENTS

PAGE 2

TABLE OF CONTENTS

ABSTRACT

CHAPTER 1 REQUIREMENTS

1.1 EQUIPMENT

1.2 MEMORY STORAGE

1.3 PRELIMINARY PROGRAMS

CHAPTER 2 LOADING AND STARTING PROCEDURE

2.1 ACT11 OPERATION

CHAPTER 3 SWITCH SETTINGS

CHAPTER 4 ERRORS

4.1 ERROR TYPEOUT FORMAT (HARDWARE)

4.2 ERROR TYPEOUT FORMAT (FUNCTION OUT OF RANGE)

CHAPTER 5 SUBROUTINE ABSTRACTS

CHAPTER 6 MISCELLANEOUS

6.1 STACK POINTER

6.2 EXECUTION TIME

CHAPTER 7 PROGRAM DESCRIPTION

7.1 FUNCTION TIME DOCUMENT

7.2 TEST SEQUENCE / RELATED ADJUSTMENTS / ASSOCIATED HARDWARE

7.3 SUBTEST DESCRIPTIONS

TMO3 DRIVE FUNCTION TIMER  
ABSTRACT

PAGE 3

ABSTRACT

PROGRAM DZTEE MEASURES THE TIME REQUIRED AND GAP  
SIZES PRODUCED BY THE TMO3-TE16/TU77 MAGTAPE  
DRIVE/SLAVE.

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

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

DEVICE ERRORS ARE CHECKED AND PRINTED AS THEY  
OCCUR. IF AN ERROR IS DATA RELATED(PARITY; ETC)  
THEY ARE PRINTED AS SOFT ERROR<sup>c</sup>.

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

TM03 DRIVE FUNCTION TIMER  
REQUIREMENTS

PAGE 4

CHAPTER 1  
REQUIREMENTS

PDP-11 FAMILY CENTRAL PROCESSOR WITH 4K MEMORY WITH UP TO 64 TM03-TE16/TU77  
CONTROLLER/MAGTAPE STATIONS.

1.1 OPTIONAL EQUIPMENT USED

1. NONE

1.2 STORAGE

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

1.3 PRELIMINARY PROGRAMS (TO ASSURE HARDWARE OPERATION)

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

TM03 DRIVE FUNCTION TIMER  
LOADING AND STARTING PROCEDURE

PAGE 5

CHAPTER 2

LOADING AND STARTING PROCEDURE

2.0 LOAD & START PROCEDURE:

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

THE PROGRAM WILL THEN REQUEST THE FIRST BUS ADDRESS OF THE RHXX  
CONTROLLER, TM03 DRIVES TO BE TESTED, TE16/TU77 SLAVES TO BE TESTED,  
AND IF SPEED TESTS ARE TO BE RUN. IN ADDITON TO EACH REQUEST A  
DEFAULT ANSWER WILL BE TYPED. TO INVOKE THE DEFAULT TYPE A  
CARRIAGE RETURN.

THE REQUESTS & THEIR DEFAULTS ARE:

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

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

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

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

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

2.1 RESTART PROCEDURE

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

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

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

## 2.2 AUTOMATIC MODE OPERATION

IF THE PROGRAM IS LOADED AND RUN IN AUTOMATIC (CHAIN) MODE  
DEFAULT RESPONSES TO OPERATOR REQUESTS ARE USED, AND ALL AVAIL-  
ABLE TM03-TE16/TU77 COMBINATIONS ARE TESTED. ADDITIONALLY THE SOFTWARE  
SWR IS INVOKED WITH A SWITCH SETTING OF 000000 IF LOADED VIA ACT11.  
NO OPERATOR INTERVENTION IS REQUIRED

\*\* EXCEPTION: IF LOADED VIA TMDP TM03 DRIVE 0 TE16/TU77 SLAVE 0 IS  
NOT TESTED.

\*\*NOTE: IN ORDER TO CHANGE THE DEFAULT SETTING OF THE SOFTWARE  
SWR, SET LOC: 176(SWREG:) TO THE DESIRED SETTING.

TM03 DRIVE FUNCTION TIMER  
SWITCH SETTINGS

PAGE 6

CHAPTER 3  
SWITCH SETTINGS

CONTROL :

- 1) CONTROL G <^G>:  
SELECTS THE SOFTWARE SWR AND ALLOWS NEW SWITCH SETTINGS.  
THE MACHINE WILL THEN TYPE: SWR=XXXXXXNEW=  
WHERE: XXXXXX IS THE OCTAL CONTENTS OF THE SOFTWARE SWR.  
AFTER THE ''NEW=''  
OF THE FOLLOWING AT THE TTY:  
A) TYPE A NUMBER TO BE LOADED INTO THE SOFTWARE SWR.  
B) IF A <CR> IS THE FIRST KEY DEPRESSED THE SOFTWARE SWITCH  
REGISTER CONTENTS WILL NOT BE CHANGED.
- 2) CONTROL A <^A>:  
ALTERNATES USAGE OF THE SWR BETWEEN HARDWARE & SOFTWARE.
- 3) CONTROL C <^C>:  
RESTARTS PROGRAM AT 200
- 4) CONTROL U <^U>:  
DELETES ALL CHARACTERS TYPED IN RESPONSE TO A REQUEST.

SW15  
(100000)

HALT ON ERROR THIS SWITCH WHEN SET WILL HALT THE  
PROCESSOR WHEN AN ERROR IS DETECTED.

THE PC+2 AND PSW AT THE TIME OF THE  
ERROR IS STORED ON THE STACK. PRESSING  
CONTINUE WILL CAUSE THE ERROR TO BE  
TYPED (IF SELECTED) AND FURTHER TESTING  
RESUMED.

SW14  
(040000)

LOOP SUBTEST

THIS SWITCH WHEN SET LOOPS THE CURRENT  
SUBTEST REGARDLESS OF ERROR CONDITION.

SW13  
(020000)

INHIBIT ERROR  
TYPEOUT

THIS SWITCH WHEN SET INHIBITS ERROR  
TYPEOUT.

SW11  
(004000)

INHIBIT SUB-  
TEST ITERATION

THIS SWITCH WHEN SET CAUSES EACH SUBTEST  
TO BE EXECUTED ONLY ONCE.

SW10  
(002000)

INHIBIT  
FUNCTION TIME  
PUBLICATION

THIS SWITCH WHEN SET WILL INHIBIT THE  
PRINTING OF THE FUNCTION TIMES. (SEE  
CHAPTER 8.)

SW09  
(001000)

RING BELL  
ON ERROR

THIS SWITCH WHEN SET WILL RING THE BELL  
ON THE TTY WHEN AN ERROR IS DETECTED.

SW06  
(000100)

CONTINUOUS  
CYCLE

THIS SWITCH WHEN SET WILL CAUSE THE  
PROGRAM TO RUN CONTINUOUSLY UNTIL  
STOPPED BY THE OPERATOR.

SW5-0

TEST SELECT

RUN SUBTEST SELECTED

NOTE:

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

DO NOT INHIBIT SUBTEST ITERATIONS WHEN PROGRAM IS RUNNING.

TM03 DRIVE FUNCTION TIMER  
ERRORS

PAGE 7

CHAPTER 4  
ERRORS

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

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

TEST # XXXXXX DEVICE ERROR

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

WHERE:

XXXXXX = TEST NUMBER  
AAAAAA-III III = 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

TM03 DRIVE FUNCTION TIMER  
SUBROUTINE ABSTRACTS

PAGE 8

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

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

5.2 PUBLISH

THE PUBLISH ROUTINE IS CALLED FROM THE SCOPE ROUTINE IF SW10 IS EQUAL TO 0 (PUBLISH TIME DOCUMENT). THE ROUTINE WILL PRINT A THE TIME RECORDED BY THE SUBTEST.

TM03 DRIVE FUNCTION TIMER  
SUBROUTINE ABSTRACTS

PAGE 9

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.

TMO3 DRIVE FUNCTION TIMER  
MISCELLANEOUS

PAGE 10

CHAPTER 6  
MISCELLANEOUS

6.1 STACK POINTER

THE STACK POINTER IS INITAILLY SET TO 500.

6.2 EXECUTION TIME

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

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

TM03 DRIVE FUNCTION TIMER  
PROGRAM DESCRIPTION

CHAPTER 7  
PROGRAM DESCRIPTION

7.1 SAMPLE TIME DOCUMENT

TYPE FIRST ADDRESS OF CONTROLLER:172440  
TYPE TM03 DRIVE #'S TO BE TESTED:ALL 0  
FOR TM03 DRIVE 0- TYPE SLAVE #'S TO BE TESTED:ALL 0  
TAPE SPEED TESTS ONLY? (YES/NO):NO

\*\*\*\*\*

\* TM03 DRIVE FUNCTION TIMES- DRIVE # 0 SLAVE # 7 TE16 SER. # 5009

* FUNCTION	TIME(SPECIFICATION)	TIME(ACTUAL)
* WRITE FROM BOT	RANGE=<176000-172000>	ACTUAL=174740
* WRITE START	RANGE=<009500-008700>	ACTUAL=009120
* WRITE SHUTDOWN	RANGE=<008500-007500>	ACTUAL=008840
* WRITE SETTLEDOWN	RANGE=<013500-007300>	ACTUAL=010970
* READ FROM BOT	RANGE=<045000-041000>	ACTUAL=043580
* READ START	RANGE=<003200-002400>	ACTUAL=002740
* READ SHUTDOWN	RANGE=<004100-003050>	ACTUAL=004360
* READ SETTLEDOWN	RANGE=<013500-007300>	ACTUAL=010970
* READ REV START	RANGE=<003200-002400>	ACTUAL=002740
* READ REV SHUTDOWN	RANGE=<003700-003300>	ACTUAL=003520
* READ REV SETTLEDOWN	RANGE=<013500-007300>	ACTUAL=010970
* TURN AROUND DELAY F-R	RANGE=<016700-010700>	ACTUAL=013600
* TURN AROUND DELAY R-F	RANGE=<016700-010700>	ACTUAL=013660
* GAP SIZE-STOP HALF	RANGE=<012900-009500>	ACTUAL=012200
* GAP SIZE-START HALF	RANGE=<011800-008500>	ACTUAL=010520
* GAP SIZE-INTERRECORD	RANGE=<014300-012600>	ACTUAL=014500
* GAP CONSISANCY	RANGE=<014000-011800>	ACTUAL=013040-
* DATA TIME-800BPI	RANGE=<024000-022000>	ACTUAL=023400
* DATA TIME-1600BPI	RANGE=<025100-024100>	ACTUAL=024470
* ERASE GAP TIME	RANGE=<101000-098000>	ACTUAL=099510
* WRITE FILE MARK	RANGE=<104000-102000>	ACTUAL=103990

TM03 DRIVE FUNCTION TIMER

7.1.1 SAMPLE TIME DOCUMENT FOR TAPE SPEED TESTS

TYPE FIRST ADDRESS OF CONTROLLER 172440:  
TYPE TM03 DRIVE #'S TO BE TESTED:ALL 0  
FOR TM03 DRIVE 0- TYPE SLAVE #'S TO BE TESTED:ALL 7  
SPEED TESTS ONLY? (YES/NO):NO Y

\*\*\*\*\*

\*TM03 DRIVE FUNCTION TIMES- DRIVE # 0 SLAVE # 7 TE16 SERIAL # 5009

*FUNCTION	TIME(SPECIFICATION)	TIME(ACTUAL)
*TAPE SPEED FWD	RANGE=<022700-021700>	ACTUAL=022500
*TAPE SPEED REV	RANGE=<022700-021700>	ACTUAL=022500

TM03 DRIVE FUNCTION TIMER

7.2 TEST SEQUENCE WITH RELATED ADJUSTMENTS AND ASSOCIATED HARDWARE

TEST NO./NAME	RELATED ADJUSTMENTS	ASSOCIATED HARDWARE
1. WRITE FROM BOT	*NONE	*M8931/M8940 ROM*M8933 ACCL CNTR
2. WRITE START	* ''	* '' * ''
3. WRITE SHUTDOWN	* ''	* '' * ''
4. WRITE SETTLEDOWN	* ''	*M8916/M8940 (TU77) SETTLEDOWN O
5. READ FROM BOT	* ''	*M8931/M8940 ROM*M8933 ACCL CNTR
6. READ START	* ''	* '' * ''
7. READ SHUTDOWN	* ''	* '' * ''
10. READ SETTLEDOWN	* ''	*M8916/M8940 (TU77) SETTLEDOWN O
11. READ REVERSE START	* ''	*M8931/M8940 ROM*M8933 ACCL CNTR
12. READ REVERSE SHUTDOWN	* ''	* '' * ''
13. READ REVERSE SETTLEDOWN	* ''	*M8916/M8940 (TU77) SETTLEDOWN O
14. TURN AROUND F-R	* ''	*M8931/M8940 ROM*M8933 ACCL CNTR
15. TURN AROUND R-F	* ''	* '' * ''
16. GAP SIZE-STOP HALF	*FWRD/REV SPEED-START/STOP-RAMPS	*CAPSTAN SERVO LOOP
17. GAP SIZE-START HALF	*SAME AS IN TEST 16	* '' '' ''
20. GAP SIZE INTERRECORD	*FWD/REV SPEED	* '' '' ''

TMO3 DRIVE FUNCTION TIMER

PAGE 14

- |                        |                     |                                  |
|------------------------|---------------------|----------------------------------|
| 21. GAP CONSISTENCY    | *SAME AS IN TEST 16 | *WRITE CLOCK                     |
| 22. DATA TIME 800 BPI  | *NONE               | * " "                            |
| 23. DATA TIME 1600 BPI | * "                 | * " "                            |
| 24. ERASE GAP TIME     | * "                 | *M8931/M8940 ROM*M8933 ACCL CNTR |
| 25. WRITE FILE MARK    | * "                 | * " " * " " "                    |
| 26. TAPE SPEED-FORWARD | *FWD SPEED          | *CAPSTAN SERVO LOOP              |
| 27. TAPE SPEED-REVERSE | *REVERSE SPEED      | *CAPSTAN SF' JO LOOP             |

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

TM03 DRIVE FUNCTION TIMER

PAGE 15

7.3 SUBTEST DESCRIPTIONS:

THE FIRST THIRTEEN (13) TESTS (T1 - T15) ARE CHECKS OF THE ROM CIRCUITS IN THE TE16 (M8931/M8940), THE ACCL COUNTER IN THE TM03 (M8933), AND THE SETTLEDOWN ONE SHOT (M8916/M8940 (TU77)).

T1. WRITE FROM BOT:

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

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

T2. WRITE START:

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

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

T3. WRITE SHUTDOWN:

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

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

T4. WRITE SETTLEDOWN:

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

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

T5. READ FROM BOT

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

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

T6. READ START

THIS TEST MEASURES THE SAME DELAY AS IN T2.

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

T7. READ SHUTDOWN:

THIS TEST MEASURES THE SAME DELAY AS IN T3.

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

T10. READ SETTLEDOWN:

THIS TEST MEASURES THE SAME DELAY AS IN T4.

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

TM03 DRIVE FUNCTION TIMER

PAGE 17

T11. READ REVERSE START:

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

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

T12. READ REVERSE SHUTDOWN

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

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

T13. READ REVERSE SETTLEDOWN:

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

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

T14. TURN AROUND DELAY-FORWARD TO REVERSE

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

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

TM03 DRIVE FUNCTION TIMER

PAGE 18

T15. TURN AROUND DELAY-REVERSE TO FORWARD

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

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

TM03 DRIVE FUNCTION TIMER

PAGE 19

GAP MEASUREMENTS:

THE PREVIOUS THIRTEEN (13) TESTS WERE MEASUREMENTS OF LOGIC DELAYS PERFORMED BY THE TM03 OR TE16/TU77 IN ORDER TO ALLOW FOR PROPER ACCELERATION AND DECELERATION OF TAPE ACCORDING TO THE DESIRED INTERCORD GAP (.6 INCHES). THIS TEST, HOWEVER, WILL MEASURE THE PHYSICAL SIZE OF THE INTERCORD GAP THAT EXISTS ON TAPE AS A RESULT OF THE START/STOP TIMES OF THE CAPSTAN ITSELF. BECAUSE THE INTERCORD 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 ACCELERATION AND DECELERATION CURVES ARE NOT IN EFFECT. THE VALUE HERE SHOULD ACTUALLY BE LESS THAN THE SUM OF T16 AND T17.

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

TM03 DRIVE FUNCTION TIMER

PAGE 20

T21. GAP CONSISTENCY:

NOW THAT WE HAVE ESTABLISHED THAT THE INTERRECORD GAP IS THE PROPER SIZE, LET US DETERMINE THE CONSISTENCY OF THE GAP UNDER VARIOUS COMMAND EXECUTION TIMES. BY WRITING A SERIES OF RECORDS, EACH WITH A DIFFERENT DELAY BETWEEN EXECUTION, WE CAN ESTABLISH THE CONSISTENCY OF THE GAPS BY READING THESE RECORDS AND MONITORING THEIR INTERRECORD GAPS, ON THE FLY.

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

T22. DATA TIME AT 800 BPI:

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

1. REWIND TO BOT AND ALLOW TAPE TO STOP
2. WRITE A RECORD AT 800 BPI.
3. MONITOR DRY (BIT 7 OF DS) FOR EACH RECORD
4. THE TIME FROM FC=FC+1 TO DRY WILL BE THE TIME REQUIRED FOR 1 INCH AT THE SELECTED DENSITY
5. STOP

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

TM03 DRIVE FUNCTION TIMER

PAGE 21

T24. ERASE:

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

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

T25. TAPE MARK:

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

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

TM03 DRIVE FUNCTION TIMER

PAGE 22

T26. TAPE SPEED FORWARD:

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

1. ASSURE TAPE IS STOPPED AT BOT.
2. ISSUE A READ FORWARD (800 BPI, NORMAL)
3. MONITOR FC FOR FC = 800(10)
4. MONITOR FC FOR FC = 26400(10)
5. TIME FROM FC = 800 TO FC = 26400 IS THE TIME REQUIRED  
FOR TAPE TO TRAVEL 32 INCHES
6. DIVIDE THE TIME FOR 32 INCHES BY 32.
7. THE RESULT IS AN AVERAGE SPEED FOR 1 INCH.
8. STOP.

T27. TAPE SPEED REVERSE:

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

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

z

1013  
1014  
1015  
1016  
1017  
1018  
1019  
1020  
1021  
1022  
1023  
1024  
1025  
1026  
1027  
1028  
1029  
1030  
1031  
1032  
1033  
1034  
1035  
1036  
1037  
1038  
1039  
1040  
1041  
1042  
1043  
1044  
1045  
1046  
1047  
1048  
1049  
1050  
1051  
1052  
1053  
1054  
1055  
1056  
1057  
1058  
1059  
1060  
1061  
1062  
1063  
1064  
1065  
1066  
1067  
1068

100000  
040000  
020000  
004000  
002000  
001000  
000400  
000200  
000100

```
.LIST BIN,LOC,SEQ
.NLIST MC
.NLIST TOC
.LIST ME
.ENABLE ABS,AMA
.MCALL $CPVEC,$CPREG,$SCATCH,$STYPE,..$ACT11,..$EOP,$CHAIN
.TITLE CZTEED0 TM03-TE16/TU77 DFT
:DRIVE FUNCTION TIMER
.SBTTL STARTING INSTRUCTIONS
:LOADING AND STARTING PROCEDURE
LOAD PROGRAM USING ABS LOADER
LOAD ADDRESS 200
SET SWITCH OPTIONS
PRESS START

:RESTART PROCEDURE
LOAD ADDRESS 210
SET SWITCH OPTIONS
PRESS START

:SWITCH REGISTER SWITCH ASSIGNMENTS
SW15= 100000 ;HALT ON ERROR
SW14= 040000 ;LOOP SUBTEST
SW13= 020000 ;INHIBIT ERROR TYPE OUT
SW11= 004000 ;INHIBIT SUBTEST ITERATION
SW10= 002000 ;INHIBIT PUBLISHING TIME SPECIFICATION
SW09= 001000 ;RING BELL ON ERROR
SW08= 000400
SW07= 00200 ;NOT USED
SW06= 000100 ;CONTINUOUS CYCLE
SW05-SW00 ;RUN TEST SELECTED
**NOTE: IF <SW15-SW00> = 177777 AT STARTUP USE SOFTWARE
SWITCH REGISTER.

:CONSOLE COMMANDS
CONTROL C ;RESTART PROGRAM (SAME AS START @ 200)
CONTROL G ;SET NEW SOFTWARE SWITCH REGISTER
CONTROL U ;DELETE LINE TYPED
RUBOUT (DELETE) ;DELETE LAST CHAR TYPED

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

.SBTTL MACRO DEFINITIONS
.MACRO SAVE
JSR PC,SAVE ;SAVE REGISTERS ON THE STACK
.ENDM
.MACRO RESTORE
JSR PC,RESTORE ;RESTORE REGISTERS FROM THE STACK
.ENDM
.MACRO INPUT
```

```
1069 JSR PC, INPUT ;GET USER INPUT
1070 .ENDM INPUT
1071 .MACRO REWIND
1072 JSR PC, REWIND ;REWIND SLAVE
1073 BVS 99$ ;BRANCH IF ERROR ON REWIND
1074 .ENDM REWIND
1075 .MACRO TIMEON
1076 JSR PC, TIMON ;TURN TIMER ON
1077 .ENDM TIMEON
1078 .MACRO TIMCHK
1079 JMP TIMER(R3) ;GO TO TIMER & RETURN VIA R2
1080 .ENDM TIMCHK
1081 .MACRO SETGO
1082 INC (R5) ;SET 'GO' BIT
1083 .ENDM SETGO
1084
1085
1086
```

.SBTTL REGISTER ASSIGNMENTS  
::DEFINITIONS AND REGISTER ASSIGNMENTS  
::GENERAL REGISTER ASSIGNMENTS

```
(1) 000000 R0=X0
(1) 000001 R1=X1
(1) 000002 R2=X2
(1) 000003 R3=X3
(1) 000004 R4=X4
(1) 000005 R5=X5
(1) 000006 SP=X6
(1) 000007 PC=X7
(1) 000000 R10=X0
(1) 000001 R11=X1
(1) 000002 R12=X2
(1) 000003 R13=X3
(1) 000004 R14=X4
(1) 000005 R15=X5
```

::REGISTER ADDRESSES

```
(1) 177776 PSW= 177776 ;:PROCESSOR STATUS WORD
(1) 177774 SLR= 177774 ;:STACK LIMIT REGISTER (11/40,11/45)
(1) 177772 PIRQ= 177772 ;:PROGRAM INTERRUPT REQ. (11/45)
(1) 177770 UBREAK= 177770 ;:MICRO-BREAK REGISTER (11/45)
(1) 177560 TKS= 177560 ;:KEYBOARD CSR
(1) 177562 TKB= 177562 ;:KEYBOARD DATA BUFFER REGISTER
(1) 177564 TPS= 177564 ;:TELEPRINTER CSR
(1) 177566 TPB= 177566 ;:TELEPRINTER DATA BUFFER REGISTER
```

::VECTOR ADDRESSES

```
(1) 000004 ERRVEC=4 ;:ADDRESS OF ERROR VECTOR
(1) 000010 RESVEC=10 ;:ADDRESS OF RESERVED INST. TRAP VECTOR
(1) 000014 TBITVEC=14 ;:ADDRESS OF 'T' BIT TRAP VECTOR
(1) 000014 TRTVEC=14 ;:ADDRESS OF 'TRACE' TRAP VECTOR
(1) 000014 BPTVEC=14 ;:ADDRESS OF 'BREAKPOINT' TRAP VECTOR
(1) 000020 IOTVEC=20 ;:ADDRESS OF IOT TRAP VECTOR
(1) 000024 PFVEC=24 ;:ADDRESS OF POWER FAIL TRAP VECTOR
(1) 000030 EMTVEC=30 ;:ADDRESS OF EMT VECTOR
(1) 000034 TRAPVEC=34 ;:ADDRESS OF TRAP VECTOR
(1) 000060 TKVEC= 60 ;:ADDRESS OF TTY KEYBOARD INT. VECTOR
(1) 000064 TPVEC=64 ;:ADDRESS OF TTY PRINTER INTERRUPT VECTOR
```

CZTEEDO TMO3-TE16/TU77 DFT  
CZTEED.P11 06-JUL-83 14:42

MACY11 30(1046) 06-JUL-83 21:05  
REGISTER ASSIGNMENTS

C 3  
PAGE 25-3

SEQ 0028

(1) 000114  
(1) 000240  
(1) 000244  
(1) 000250  
(1)

PARVEC= 114  
PIRVEC=240  
FPEVEC=244  
MMVEC=250

:::ADDRESS OF MA/MF PARITY ERROR VECTOR  
:::ADDRESS OF PIRQ VECTOR  
:::ADDRESS OF FLOATING POINT INT. VECTOR  
:::ADDRESS OF MEM MGMT ERROR TRAP VECTOR

1089  
1090  
1091  
1092  
1093  
1094  
1095  
1096  
1097  
1098  
1099  
1100  
1101  
1102  
1103  
1104  
1105  
1106  
1107  
1108  
1109  
1110  
1111  
1112  
1113  
1114  
1115  
1116  
1117  
1118  
1119  
1120  
1121  
1122  
1123  
1124  
1125  
1126  
1127  
1128  
1129  
1130  
1131  
1132  
1133  
1134  
1135  
1136  
1137  
1138  
1139  
1140  
1141  
1142  
1143  
1144

172440  
  
000000  
000002  
000004  
000006  
000010  
000012  
000014  
000016  
000022  
000024  
000026  
000030  
000032  
  
  
000001  
000000  
000002  
000006  
000010  
000026  
000024  
000030  
000032  
000050  
000056  
000060  
000070  
000076  
000100  
000200  
000400  
001000  
002000  
004000  
020000  
040000  
100000  
  
000000  
000001  
000002  
000003  
000004  
000005  
000006  
000007  
000010  
000020  
000040  
000100

:RH, TM03-TE16/TU77 REGISTERS  
TMCS1= 172440

:TM03-TE16/TU77 INDEX VALUES

CS1= 00  
WC= 02  
BA= 04  
FC= 06  
CS2= 10  
DS= 12  
ER= 14  
AS= 16  
DB= 22  
MR= 24  
DT= 26  
SN= 30  
TC= 32

:CONTROL STATUS #1  
:BUS ADDRESS REGISTER  
:FRAME COUNT  
:CONTROL STATUS #2  
:DRIVE STATUS  
:ERROR REG #1  
:ATTENTION SUMMARY  
:DATA BUFFER REG  
:MAINTENANCE REG  
:DRIVE TYPE REG  
:SERIAL NUMBER REGISTER  
:TAPE CONTROL REG

.SBTTL TM03-TE16/TU77 REGISTER BITS  
:RHCS1-CS1(R5)

GO= 1  
NOP= 0  
RWDOFF= 2  
RWD= 6  
DRYCLR= 10  
WFMK= 26  
ERASE= 24  
SPCFWD= 30  
SPCREV= 32  
WCHKF= 50  
WCHKR= 56  
WFWD= 60  
RDFWD= 70  
RDREV= 76  
IE= 100  
RDY= 200  
A16= 400  
A17= 1000  
PSEL= 2000  
DVA= 4000  
MCPE= 20000  
TRE= 40000  
SC= 100000

:RHCS2-CS2(R5)

DV0= 0  
DV1= 1  
DV2= 2  
DV3= 3  
DV4= 4  
DV5= 5  
DV6= 6  
DV7= 7  
BAI= 10  
PAT= 20  
CLR= 40  
IR= 100

1145	000200	OR=	200	
1146	000400	MDPE=	400	
1147	001000	MXF=	1000	
1148	002000	PGE=	2000	
1149	004000	NEM=	4000	
1150	010000	NED=	10000	
1151	020000	UPE=	20000	
1152	040000	WCE=	40000	
1153	100000	DLT=	100000	
1154		;RHDS-DS(R5)		
1155	000001	SLA=	1	
1156	000002	BOT=	2	
1157	000004	TMK=	4	
1158	000010	IDB=	10	
1159	000020	SDWN=	20	
1160	000040	PES=	40	
1161	000100	SSC=	100	
1162	000200	DRY=	200	
1163	000400	DPR=	400	
1164	002000	EOT=	2000	
1165	004000	WRL=	4000	
1166	010000	MOL=	10000	
1167	020000	PIP=	20000	
1168	040000	ERR=	40000	
1169	100000	ATA=	100000	
1170		;RHER-ER(R5)		
1171	000001	ILF=	1	
1172	000002	ILR=	2	
1173	000004	RMR=	4	
1174				
1175	000020	FMT=	20	
1176	000100	INCVAE=	100	
1177	000200	PEFLRC=	200	
1178	000400	NSG=	400	
1179	001000	FCE=	1000	
1180	002000	CSITM=	2000	
1181	004000	NEF=	4000	
1182	010000	DTE=	10000	
1183	020000	OPI=	20000	
1184	040000	UNS=	40000	
1185	075027	HRDERR=	UNS!OPI!DTE!NEF!FCE!FMT!RMR!ILR!ILF	;HARDERROR BITS
1186				
1187		;RHMR-MR(R5)		
1188	000100	OSC=	100	
1189				
1190		;RHDT-DT(R5)		
1191	002000	SPR=	2000	
1192	010000	CH7=	10000	
1193	040000	TAP=	40000	
1194				
1195		;RHTC-TC(R5)		
1196	001700	NORM11=	1700	
1197	000320	CDM11=	320	
1198	000000	BPI200=	0	
1199	000400	BPI556=	000400	
1200	001000	BPI800=	001000	

1201	002300	PE1600= 002300	
1202	100000	ACCL= 100000	
1203			
1204			
1205			
1206		: INSTRUCTION EQUATES	
1207	104400	HLT= TRAP	
1208	104000	SCOPE= EMT	
1209	000004	TYPE= IOT	
1210			
1211		: MISCELLANEOUS EQUATES	
1212	006344	OUTBUF=INIT	: OUTPUT BUFFER STARTS AT BEG OF PROGRAM
1213	177400	FRMCNT= -256.	: FRAME COUNT
1214	177600	WRDCNT= -128.	: WORD COUNT
1215		: ASCII EQUATES	
1216	000001	CNTRLA= 1	: ASCII CODE FOR CONTROL A (^A)
1217	000003	CNTRLC= 3	: ASCII CODE FOR CONTROL C (^C)
1218	000007	CNTRLG= 7	: ASCII CODE FOR CONTROL G (^G)
1219	000011	HT= 11	: ASCII CODE FOR HORIZONTAL TAB
1220	000012	LF= 12	: ASCII CODE FOR LINE FEED
1221	000015	CR= 15	: ASCII CODE FOR CARRIAGE RETURN
1222	000017	CNTRLO= 17	: ASCII CODE FOR CONTROL O (^O)
1223	000025	CNTRLU= 25	: ASCII CODE FOR CONTROL U (^U)

```
1226                                     ;SETUP TRAP VECTORS
1227                                     . =TBITVEC
1228 000014 000016                       .WORD .+2                       ;SET 'T' TRAP TO TIMER ROUTINE
1229 000016 000000                       .WORD HALT                       ;PRIORITY LEVEL 7
1230 000020 002636                       .WORD .TYPE                       ;SET IOT TRAP TO .TYPE ROUTINE
1231 000022 000000                       .WORD 0                           ;PRIORITY LEVEL 0
1232 000024 000026                       .WORD PFVEC+2                     ;POWER FAIL TRAP TO HALT
1233 000026 000000                       .WORD HALT                       ;AT PFVEC+2
1234 000030 004566                       .WORD .SCOPE                      ;SET EMT TRAP TO .SCOPE ROUTINE
1235 000032 000340                       .WORD 340                         ;PRIORITY LEVEL 7
1236 000034 004252                       .WORD .HLT                        ;SET TRAP TRAP TO .HLT ROUTINE
1237 000036 000340                       .WORD 340                         ;PRIORITY LEVEL 7
1238
(1)                                     ;ACT11 HOOK *****
(1)                                     $SVPC=.
(1)                                     . =42
(1) 000042 000000                       .WORD 0
(1)                                     . =46
(1) 000046 013554                       .WORD $ENDAD                       ;SET LOCATION 46
(1)                                     . =52
(1) 000052 000000                       .WORD 0                           ;SET LOCATION 52 = 0
(1)                                     . = $SVPC                       ;RESTORE LOCATION CTR
(1)
1239                                     . =TKVEC
1240 000060 004126                       .WORD TKISR
1241 000062 000200                       .WORD 200
1242
1243                                     ;SOFTWARE SWITCH REGISTER LOC. 176
1244                                     . =176
1245 000176 000000                       SWREG: .WORD 0                   ;SOFTWARE SWITCH REGISTER
1246
1247                                     . =200
1248 000200 000137 006344                 JMP @#INIT                       ;GO TO START OF PROGRAM
1249                                     . =210
1250 000210 000137 007464                 JMP @#RSTRT                       ;RESTART ADDRESS
1251
1252                                     . =500
1253 000600 000600                       STKPTR= 600                       ;STACK
1254
1255                                     . =1000
1256                                     ;PROGRAM TAGS
1257 001000 177570                       SWR: 177570                       ;SWITCH REGISTER
1258 001002 000000                       SCPADR: .WORD 0
1259 001004 000000                       DRVNUM: .BYTE 0
1260 001005 000000                       SLVNUM: .BYTE 0
1261 001006 000000                       SLVPTR: .WORD 0
1262 001010 172440                       TMBASE: .WORD TMCS1
1263 001012 000000                       OSCTIM: .WORD 0
1264 001014 000000                       GAPDEL: .WORD 0
1265 001016 000000                       ATIME: .WORD 0
1266 001020 000020                       ATIMTBL: .BLKW 16.
1267
1268 001060 000020                       GAPTRL: .BLKW 16.
1269 001120 000000                       DELTIM: .WORD 0
1270 001122 000000                       OCTALO: .WORD 0
1271 001124 000000                       GAP: .BYTE 0

;TM03 DRIVE UNDER TEST
;TE16/TU77 SLAVE UNDER TEST
;POINTER TO SLAVE TABLE (SLVTBL) BELOW
;BASE ADDRESS OF TM03-TE16/TU77 REGISTERS
;US/TICK (56/80 FOR TE16/TU77)
;TICKS/MS (18/6 FOR TE16/TU77)
;CONTAINS 'TICK' COUNT
;EACH ENTRY CONTAINS TIME FOR FUNCTION
;ENTRIES ARE MADE BY 'SCOPE' ROUTINE
;TIMES RECORDED BY 'GAP CONSISTANCY' TEST
;VARIABLE DELAY
;CONTAINS GAP # (USED FOR TST 021)
```

1272	001125	000	ITCNT:	.BYTE	0	: ITERATION COUNT
1273	001126	000	TSTNUM:	.BYTE	0	: TEST #
1274	001127	000	ERFLG:	.BYTE	0	: ERROR FLAG
1275	001130	000	SKEWFLG:	.BYTE	0	: 0/1 = DO NOT/DO SKEW (SPEED) TESTS
1276	001131	000	PRGFLG:	.BYTE	0	: PROGRAM FLAG
1277	001132	000	UNTFND:	.BYTE	0	: UNIT FOUND INDICATOR
1278	001133	000	TYPFLG:	.BYTE	0	
1279	001134	000	PSCNT:	.BYTE	0	: CONTAINS PASS COUNT
1280	001135	000	ASFLG:	.BYTE	0	: 1/0 = YES/NO.
1281	001136	000	TE16:	.BYTE	0	: 0/1 = TE16/TU77
1282		001140		.EVEN		
1283	001140	030460	DIGTAB:	'01		
1284	001142	031462		'23		
1285	001144	032464		'45		
1286	001146	033466		'67		
1287	001150	034470		'89		
1288	001152	000006	ODIGITS:	.BLKB	6	: RESERVE SPACE FOR CONVERTED DIGITS
1289	001160	000		.BYTE	0	: TERMINATOR
1290		001162		.EVEN		
1291	001162	000010	DRVTBL:	.BLKB	8.	: A 0/-1 = DRIVE NOT TO BE/TO BE TESTED
1292	001172	000100	SLVTBL:	.BLKB	64.	: A 0/-1 = SLAVE NOT TO BE/TO BE TESTED
1293	001272	000110	INBUF:	.BLKB	72.	: TELETYPE INPUT BUFFER
1294	001402	005015	CRLF:	.ASCIZ	<CR><LF>	: MISCELLANEOUS ASCII CHARACTERS
1295	001405	134	BKSLSH:	.ASCIZ	'\'	
1296	001407	060	ECHO:	.ASCIZ	'0'	
1297	001411	007	BELL:	.ASCIZ	<7>	
1298	001413	055	DASH:	.ASCIZ	'-'	
1299	001415	040	SPACE2:	.ASCII	':'	
1300	001416	000040	SPACE:	.ASCIZ	':'	
1301	001420	004476	ANGTAB:	.ASCIZ	'>'<HT>	
1302		001424		.EVEN		

1304  
 1305  
 1306  
 1307  
 1308  
 1309  
 1310  
 1311 001424 000000 000000  
 1312 001430 036050 035230  
 1313 001434 001666 001546  
 1314 001440 001522 001356  
 1315 001444 002506 001332  
 1316 001450 007164 006344  
 1317 001454 000500 000360  
 1318 001460 000632 000454  
 1319 001464 002506 001332  
 1320 001470 000500 000360  
 1321 001474 000562 000512  
 1322 001500 002506 001332  
 1323 001504 003206 002056  
 1324 001510 003206 002056  
 1325 001514 002412 001666  
 1326 001520 002234 001522  
 1327 001524 002626 002354  
 1328 001530 002570 002234  
 1329 001534 004540 004230  
 1330 001540 004716 004552  
 1331 001544 023564 023110  
 1332 001550 024240 023730  
 1333 001554 004336 004172  
 1334 001560 004336 004172  
 1335  
 1336  
 1337  
 1338  
 1339  
 1340  
 1341  
 1342  
 1343  
 1344 001564 002602 002412  
 1345 001570 002652 002506  
 1346 001574 002734 002532  
 1347 001600 003016 002424  
 1348 001604 003016 002304  
 1349 001610 002734 002176  
 1350 001614 002652 002176  
 1351 001620 002652 002176  
 1352 001624 002570 002176  
 1353 001630 002570 002176  
 1354 001634 002570 002176  
 1355 001640 002570 002176  
 1356 001644 002570 002176  
 1357 001650 002570 002176  
 1358 001654 002570 002176  
 1359 001660 002570 002176

.SBTTL TE16 TIME SPECIFICATION TABLE  
 :THE BELOW TABLE CONTAINS THE TE16 SPECIFIED FUNCTION TIMES IN TENS OF  
 :MICROSECONDS. NOTE THAT WHEN TIMES ARE TYPED THAT THEY ARE TYPED IN  
 :MICROSECONDS (BY APPENDING A 0).  
 :FORMAT IS

:WORD	MAX,MIN	:TIME IN MS	FUNCTION	TEST #
TE16TTBL: .WORD	0,0	:SPARE		
.WORD	15400.,15000.	:154.0-150.0	WRITE FROM BOT	TST001
.WORD	00950.,00870.	:9.5-8.7	WRITE START	TST002
.WORD	00850.,00750.	:8.9-8.5	WRITE SHUTDOWN	TST003
.WORD	01350.,00730.	:13.5-7.3	WRITE STLDOWN	TST004
.WORD	03700.,03300.	:37.0-33.0	READ FROM BOT	TST005
.WORD	00320.,00240.	:3.2-2.4	READ START	TST006
.WORD	00410.,00300.	:4.1-3.00	READ SHUTDOWN	TST007
.WORD	01350.,00730.	:13.5-7.3	READ SETTLEDOWN	TST010
.WORD	00320.,00240.	:3.2-2.4	RD REV START	TST011
.WORD	00370.,00330.	:3.7-3.3	RD REV SHTDWN	TST012
.WORD	01350.,00730.	:13.5-7.3	RD REV STLDWN	TST013
.WORD	01670.,01070.	:16.7-10.7	TRN RND DLY F-R	TST014
.WORD	01670.,01070.	:16.7-10.7	TRN RND DLY R-F	TST015
.WORD	01290.,00950.	:12.9-9.5	GAP SIZE STOP	TST016
.WORD	01180.,00850.	:11.8-8.5	GAP SIZE STRT	TST017
.WORD	01430.,01260.	:14.3-12.6	GAP SIZE INTER	TST020
.WORD	01400.,01180.	:14.0-11.8	GAP CONSISANCY	TST021
.WORD	02400.,02200.	:24.0-22.0	DAT TIME 800BPI	TST022
.WORD	02510.,02410.	:25.1-24.1	DAT TIME 1600PE	TST023
.WORD	10100.,09800.	:101.0-98.0	ERASE	TST024
.WORD	10400.,10200.	:104.0-102.0	WRT FILE MARK	TST025
.WORD	02270.,02170.	:22.7-21.7	TAPE SPEED FWD	TST026
.WORD	02270.,02170.	:22.7-21.7	TAPE SPEED REV	TST027

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

.SBTTL TE16 GAP TIME SPECIFICATION TABLE  
 :THIS TABLE CONTAINS THE TE16 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)
TE16GTBL: .WORD	01410.,01290.	:14.10-12.9	GAP-0	0 MS
.WORD	01450.,01350.	:14.5-13.5	GAP-1	1.0 MS
.WORD	01500.,01370.	:15.0-13.7	GAP-2	2.0 MS
.WORD	01550.,01300.	:15.5-13.0	GAP-3	3.0 MS
.WORD	01550.,01220.	:15.5-12.2	GAP-4	4.0 MS
.WORD	01500.,01150.	:15.0-11.5	GAP-5	5.0 MS
.WORD	01450.,01150.	:14.5-11.5	GAP-6	6.0 MS
.WORD	01450.,01150.	:14.5-11.5	GAP-7	7.0 MS
.WORD	01400.,01150.	:14.0-11.5	GAP-10	8.0 MS
.WORD	01400.,01150.	:14.0-11.5	GAP-11	9.0 MS
.WORD	01400.,01150.	:14.0-11.5	GAP-12	10.0 MS
.WORD	01400.,01150.	:14.0-11.5	GAP-13	11.1 MS
.WORD	01400.,01150.	:14.0-11.5	GAP-14	12.1 MS
.WORD	01400.,01150.	:14.0-11.5	GAP-15	13.1 MS
.WORD	01400.,01150.	:14.0-11.5	GAP-16	14.1 MS
.WORD	01400.,01150.	:14.0-11.5	GAP-17	15.1 MS

1361  
 1362  
 1363  
 1364  
 1365  
 1366  
 1367  
 1368 001664 000000 000000  
 1369 001670 012606 011571  
 1370 001674 000520 000460  
 1371 001700 000346 000313  
 1372 001704 003410 001717  
 1373 001710 001315 001112  
 1374 001714 000111 000067  
 1375 001720 000111 000067  
 1376 001724 003410 001717  
 1377 001730 000111 000067  
 1378 001734 000101 000057  
 1379 001740 003410 001717  
 1380 001744 003500 002006  
 1381 001750 003510 002017  
 1382 001754 001510 001231  
 1383 001760 000642 000505  
 1384 001764 001012 000646  
 1385 001770 001034 000641  
 1386 001774 001515 001434  
 1387 002000 001536 001434  
 1388 002004 007020 006476  
 1389 002010 007176 006642  
 1390 002014 001560 001320  
 1391 002020 001560 001320

.SBTTL TU77 TIME SPECIFICATION TABLE  
 ;THE BELOW TABLE CONTAINS THE TU77 SPECIFIED FUNCTION TIMES IN TENS OF  
 ;MICROSECONDS. NOTE THAT WHEN TIMES ARE TYPED THAT THEY ARE TYPED IN  
 ;MICROSECONDS (BY APPENDING A 0).  
 ;FORMAT IS

.WORD	MAX,MIN	:TIME IN MS	FUNCTION	TEST #
TU77TTBL: .WORD	0,0	:SPARE		
.WORD	5510.,4985.	:55.1-49.85	WRITE FROM BOT	TST001
.WORD	00336.,00304.	:3.36-3.04	WRITE START	TST002
.WORD	00230.,00203.	:2.3-2.03	WRITE SHUTDOWN	TST003
.WORD	01800.,00975.	:18.0-9.75	WRITE STLDOWN	TST004
.WORD	00717.,00586.	:7.17-5.86	READ FROM BOT	TST005
.WORD	00073.,00055.	:.73-.55	READ START	TST006
.WORD	00073.,00055.	:.73-.55	READ SHUTDOWN	TST007
.WORD	01800.,00975.	:18.0-9.75	READ SETTLEDOWN	TST010
.WORD	00073.,00055.	:0.73-0.55	RD REV START	TST011
.WORD	00065.,00047.	:0.65-0.47	RD REV SHTDWN	TST012
.WORD	01800.,00975.	:18.0-9.75	RD REV STLDWN	TST013
.WORD	01856.,01030.	:18.56-10.3	TRN RND DLY F-R	TST014
.WORD	01864.,01039.	:18.64-10.39	TRN RND DLY R-F	TST015
.WORD	0840.,00665.	:8.4-6.65	GAP SIZE STOP	TST016
.WORD	0418.,00325.	:4.18-3.25	GAP SIZE STRT	TST017
.WORD	0522.,0422.	:5.22-4.22	GAP SIZE INTER	TST020
.WORD	0540.,0417.	:5.40-4.17	GAP CONSISANCY	TST021
.WORD	0845.,0796.	:8.45-7.96	DAT TIME 800BPI	TST022
.WORD	0862.,0796.	:8.62-7.96	DAT TIME 1600PE	TST023
.WORD	3600.,03390.	:36.00-33.90	ERASE	TST024
.WORD	3710.,3490.	:37.10-34.90	WRT FILE MARK	TST025
.WORD	00880.,00720.	:8.8-7.2	TAPE SPEED FWD	TST026
.WORD	00880.,00720.	:8.8-7.2	TAPE SPEED REV	TST027

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

.SBTTL TU77 GAP TIME SPECIFICATION TABLE  
 ;THIS TABLE CONTAINS THE TU77 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.

1392  
 1393  
 1394  
 1395  
 1396  
 1397  
 1398  
 1399  
 1400  
 1401 002024 000770 000676  
 1402 002030 001013 000717  
 1403 002034 001033 000732  
 1404 002040 001050 000742  
 1405 002044 001062 000746  
 1406 002050 001067 000742  
 1407 002054 001072 000736  
 1408 002060 001072 000724  
 1409 002064 001066 000704  
 1410 002070 001065 000660  
 1411 002074 001046 000627  
 1412 002100 001027 000572  
 1413 002104 000776 000632  
 1414 002110 000776 000632  
 1415 002114 000776 000632  
 1416 002120 000776 000632

.WORD	MAX,MIN(10)	:TIME IN MS(10)	GAP #	DELAY IN MS(10)
TU77GTBL: .WORD	0504.,0446.	:5.04-4.46	GAP-0	0 MS
.WORD	0523.,0463.	:5.23-4.63	GAP-1	0.24 MS
.WORD	0539.,0474.	:5.39-4.74	GAP-2	0.48 MS
.WORD	0552.,0482.	:5.52-4.82	GAP-3	0.72 MS
.WORD	0562.,0486.	:5.62-4.86	GAP-4	0.96 MS
.WORD	0567.,0482.	:5.67-4.82	GAP-5	1.20 MS
.WORD	0570.,0478.	:5.70-4.78	GAP-6	1.44 MS
.WORD	0570.,0468.	:5.70-4.68	GAP-7	1.68 MS
.WORD	0566.,0452.	:5.66-4.52	GAP-10	1.92 MS
.WORD	0565.,0432.	:5.65-4.32	GAP-11	2.16 MS
.WORD	0550.,0407.	:5.50-4.07	GAP-12	2.40 MS
.WORD	0535.,0378.	:5.35-3.78	GAP-13	2.64 MS
.WORD	0510.,0410.	:5.10-4.10	GAP-14	2.88 MS
.WORD	0510.,0410.	:5.10-4.10	GAP-15	3.12 MS
.WORD	0510.,0410.	:5.10-4.10	GAP-16	3.36 MS
.WORD	0510.,0410.	:5.10-4.10	GAP-17	3.60 MS

1418  
 1419  
 1420  
 1421  
 1422  
 1423  
 1424 002124  
 1425 002124 000000 000000  
 1426 002130 036050 035230  
 1427 002134 001666 001546  
 1428 002140 001522 001356  
 1429 002144 002506 001332  
 1430 002150 007164 006344  
 1431 002154 000500 000360  
 1432 002160 000632 000454  
 1433 002164 002506 001332  
 1434 002170 000500 000360  
 1435 002174 000562 000512  
 1436 002200 002506 001332  
 1437 002204 003206 002056  
 1438 002210 003206 002056  
 1439 002214 002412 001666  
 1440 002220 002234 001522  
 1441 002224 002626 002354  
 1442 002230 002506 002234  
 1443 002234 004540 004230  
 1444 002240 004716 004552  
 1445 002244 023564 023110  
 1446 002250 024240 023730  
 1447 002254 004336 004172  
 1448 002260 004336 004172  
 1449  
 1450  
 1451  
 1452  
 1453  
 1454  
 1455  
 1456 002264 002544 002354  
 1457 002270 002652 002506  
 1458 002274 002722 002506  
 1459 002300 002710 002474  
 1460 002304 002570 002260  
 1461 002310 002556 002114  
 1462 002314 002532 002114  
 1463 002320 002532 002114  
 1464 002324 002506 002176  
 1465 002330 002474 002176  
 1466 002334 002474 002176  
 1467 002340 002474 002176  
 1468 002344 002474 002176  
 1469 002350 002474 002176  
 1470 002354 002474 002176  
 1471 002360 002474 002176  
 1472 002364

.SBITL TIME SPECIFICATION TABLE  
 :THE BELOW TABLE WILL CONTAIN THE SPECIFIED FUNCTION TIMES IN TENS OF  
 :MICROSECONDS. NOTE THAT WHEN TIMES ARE TYPED THAT THEY ARE TYPED IN  
 :MICROSECONDS (BY APPENDING A 0).  
 :FORMAT IS

STTBL:	WORD	MAX,MIN	TIME IN MS	FUNCTION	TEST #
STIMTBL:	WORD	0 0	:SPARE		
	WORD	15400.,15000.	:154.0-150.0	WRITE FROM BOT	TST001
	WORD	00950.,00870.	:9.5-8.7	WRITE START	TST002
	WORD	00850.,00750.	:8.9-8.5	WRITE SHUTDOWN	TST003
	WORD	01350.,00730.	:13.5-7.3	WRITE STLDOWN	TST004
	WORD	03700.,03300.	:37.0-33.0	READ FROM BOT	TST005
	WORD	00320.,00240.	:3.2-2.4	READ START	TST006
	WORD	00410.,00300.	:4.1-3.00	READ SHUTDOWN	TST007
	WORD	01350.,00730.	:13.5-7.3	READ SETTLEDOWN	TST010
	WORD	00320.,00240.	:3.2-2.4	RD REV START	TST011
	WORD	00370.,00330.	:3.7-3.3	RD REV SHTDWN	TST012
	WORD	01350.,00730.	:13.5-7.3	RD REV STLDWN	TST013
	WORD	01670.,01070.	:16.7-10.7	TRN RND DLY F-R	TST014
	WORD	01670.,01070.	:16.7-10.7	TRN RND DLY R-F	TST015
	WORD	01290.,00950.	:12.9-9.5	GAP SIZE STOP	TST016
	WORD	01180.,00850.	:11.8-8.5	GAP SIZE STRT	TST017
	WORD	01430.,01260.	:14.3-12.6	GAP SIZE INTER	TST020
	WORD	01350.,01180.	:13.5-11.8	GAP CONSISANCY	TST021
	WORD	02400.,02200.	:24.0-22.0	DAT TIME 800BPI	TST022
	WORD	02510.,02410.	:25.1-24.1	DAT TIME 1600PE	TST023
	WORD	10100.,09800.	:101.0-98.0	ERASE	TST024
	WORD	10400.,10200.	:104.0-102.0	WRT FILE MARK	TST025
	WORD	02270.,02170.	:22.7-21.7	TAPE SPEED FWD	TST026
	WORD	02270.,02170.	:22.7-21.7	TAPE SPEED REV	TST027

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

.SBTTL GAP TIME SPECIFICATION TABLE  
 :THIS TABLE WILL CONTAIN 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.

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

ENDTBL:

1474  
1475  
1476 002364 016456  
1477 002366 016506  
1478 002370 016530  
1479 002372 016550  
1480 002374 016572  
1481 002376 016616  
1482 002400 016640  
1483 002402 016657  
1484 002404 016701  
1485 002406 016724  
1486 002410 016746  
1487 002412 016773  
1488 002414 017022  
1489 002416 017053  
1490 002420 017104  
1491 002422 017132  
1492 002424 017161  
1493 002426 017211  
1494 002430 017234  
1495 002432 017260  
1496 002434 017305  
1497 002436 017327  
1498 002440 017352  
1499 002442 017374  
1500  
1501  
1502 002444 010050  
1503 002446 010344  
1504 002450 010430  
1505 002452 010506  
1506 002454 010576  
1507 002456 010704  
1508 002460 010770  
1509 002462 011054  
1510 002464 011154  
1511 002466 011274  
1512 002470 011372  
1513 002472 011502  
1514 002474 011642  
1515 002476 011734  
1516 002500 012042  
1517 002502 012136  
1518 002504 012246  
1519 002506 012366  
1520 002510 012672  
1521 002512 013022  
1522 002514 013152  
1523 002516 013272  
1524 002520 013626  
1525 002522 013764

.SB1TL TEST HEADER POINTERS  
;THE BELOW TABLE CONTAINS POINTERS TO EACH TEST'S DESCRIPTOR  
NAMPTR:

.WORD A.T000  
.WORD A.T001  
.WORD A.T002  
.WORD A.T003  
.WORD A.T004  
.WORD A.T005  
.WORD A.T006  
.WORD A.T007  
.WORD A.T010  
.WORD A.T011  
.WORD A.T012  
.WORD A.T013  
.WORD A.T014  
.WORD A.T015  
.WORD A.T016  
.WORD A.T017  
.WORD A.T020  
.WORD A.T021  
.WORD A.T022  
.WORD A.T023  
.WORD A.T024  
.WORD A.T025  
.WORD A.T026  
.WORD A.T027

;TABLE OF TEST STARTING ADDRESSES  
TSTBL:

.WORD TST000  
.WORD TST001  
.WORD TST002  
.WORD TST003  
.WORD TST004  
.WORD TST005  
.WORD TST006  
.WORD TST007  
.WORD TST010  
.WORD TST011  
.WORD TST012  
.WORD TST013  
.WORD TST014  
.WORD TST015  
.WORD TST016  
.WORD TST017  
.WORD TST020  
.WORD TST021  
.WORD TST022  
.WORD TST023  
.WORD TST024  
.WORD TST025  
.WORD TST026  
.WORD TST027



```

(1) 002712 012600          TYPE3: MOV      (SP)+,R0          ;;RESTORE R0
(1) 002714 000002          RTI                          ;;RETURN TO CALLER
(1)
(1) 002716 122716 000011  TYPE4: CMPB     #SHT,(SP)          ;;BRANCH IF HORIZONTAL TAB <HT>
(1) 002722 001445          BEQ      9$
(1) 002724 004737 002756  JSR      PC,5$          ;;TYPE CHARACTER
(1) 002730 122726 000012  3$:  CMPB     #12,(SP)+          ;;CHECK IF CHARACTER WAS A LINE FEED
(1) 002734 001350          BNE     TYPE1          ;;BRANCH IF NOT LINE FEED
(1) 002736 013746 002616  MOV      $NULL,-(SP)      ;;GET # OF FILLERS REQUIRED AND FILLER
(1)                          ;;CHARACTER.
(1)
(1) 002742 105366 000001  4$:  DECB     1(SP)          ;;DECREMENT FILLERS REQ. COUNT
(1) 002746 002770          BLT     3$              ;;BRANCH IF NO MORE FILLERS ARE REQUIRED
(1) 002750 004737 002756  JSR      PC,5$          ;;TYPE FILLER CHARACTER
(1) 002754 000772          BR      4$
(1)
(1) 002756 105777 177640  5$:  TSTB     @STPS          ;;WAIT FOR OUTPUT DEVICE
(1) 002762 100375          BPL     -4
(1) 002764 122737 000017 002627  CMPB     #17,@#SCNTRLO    ;;CHECK IF CONTROL O WAS TYPED
(1) 002772 001403          BEQ     6$              ;;STOP TYPING MESSAGE IF ^O WAS TYPED
(1) 002774 116677 000002 177622  MOVB     2(SP),@STPB      ;;OUTPUT CHARACTER
(1) 003002 122766 000015 000002  6$:  CMPB     #15,2(SP)      ;;BRANCH IF NOT <CR>
(1) 003010 001003          BNE     7$
(1) 003012 105037 002626  CLRB     $CHARCNT        ;;CLEAR CHARACTERS TYPED COUNT
(1) 003016 000406          BR      8$
(1) 003020 122766 000012 000002  7$:  CMPB     #12,2(SP)      ;;BRANCH IF <LF> OR 'NULL'
(1) 003026 002002          BGE     8$
(1) 003030 105237 002626  INCB     $CHARCNT        ;;INCREMENT CHARACTER TYPED COUNT
(1) 003034 000207          8$:  RTS      PC
(1)
(1)                          ;;HORIZONTAL TAB <HT> PROCESSER
(1) 003036 112716 000040  9$:  MOVB     #40,(SP)        ;;LOAD 'SPACE'
(1) 003042 004737 002756  10$: JSR      PC,5$          ;;TYPE 'SPACE'
(1) 003046 132737 000007 002626  BITB     #7,$CHARCNT      ;;TYPE SPACES UNTIL A MULTIPLE
(1) 003054 001372          BNE     10$            ;;OF 8 CHARACTERS HAVE BEEN TYPED
(1) 003056 105726          TSTB     (SP)+          ;;POP SPACE
(1) 003060 000676          BR      TYPE1          ;;GET NEXT CHARACTER
1549
1550          ;SUBROUTINE TO SAVE GENERAL REGISTERS ON THE STACK
1551          ;CALL:  SAVE
1552 003062 010546          .SAVE: MOV      R5,-(SP)          ;SAVE REGISTERS ON THE STACK
1553 003064 010446          MOV      R4,-(SP)
1554 003066 010346          MOV      R3,-(SP)
1555 003070 010246          MOV      R2,-(SP)
1556 003072 010146          MOV      R1,-(SP)
1557 003074 010046          MOV      R0,-(SP)
1558 003076 016646 000014          MOV      14(SP),-(SP)      ;GET RETURN PC
1559 003102 000207          RTS      PC              ;RETURN
1560
1561          ;SUBROUTINE TO RESTORE GENERAL REGISTERS FROM THE STACK
1562          ;CALL:  RESTORE
1563 003104 012666 000014          .RESTORE:MOV     (SP)+,14(SP)      ;MOVE RETURN PC
1564 003110 012600          MOV      (SP)+,R0        ;RESTORE REGISTERS
1565 003112 012601          MOV      (SP)+,R1
1566 003114 012602          MOV      (SP)+,R2
1567 003116 012603          MOV      (SP)+,R3

```

```
1568 003120 012604      MOV      (SP)+,R4
1569 003122 012605      MOV      (SP)+,R5
1570 003124 000207      RTS      PC                      ;RETURN
1571
1572      ;SUBROUTINE TO CONVERT OCTAL DATA TO ASCII
1573      ;CALL: MOV      NUMBER,R2      ;MOVE NUMBER TO R2
1574      ;      JSR      PC,CNVOCT
1575
1576 003126 110637 001133  CNVOCT: MOVB      SP,TYPFLG      ;SET DO NOT TYPE FLAG
1577 003132 000402      BR      CNVTO
1578
1579      .SBTTL      OCTAL TO ASCII & TYPE ROUTINE
1580      ;SUBROUTINE TO CONVERT OCTAL NUMBER TO ASCII AND TYPE IT OUT
1581      ;CALL: MOV      NUMBER,R2      ;PUT # IN R2
1582      ;      JSR      PC,TYPOCT      ;CALL ROUTINE
1583
1584 003134 105037 001133  TYPOCT: CLRB      @#TYPFLG      ;SET TYPE FLAG
1585 003140  CNVTO:
1586 (1) 003140 004737 003062      JSR      PC,SAVE      ;SAVE REGISTERS ON THE STACK
1587 003144 012704 001152      MOV      #ODIGITS,R4      ;SET PTR TO OUTPUT
1588 003150 005003      CLR      R3      ;R3 WILL CONTAIN OCTAL DIGIT
1589 003152 010201      MOV      R2,R1      ;GET # TO BE TYPED
1590 003154 006302      1$: ASL      R2      ;SHIFT #
1591 003156 006103      ROL      R3
1592 003160 012700 000006      MOV      #6,R0      ;SET DIGIT COUNTER
1593 003164 000404      BR      3$
1594 003166 006302      2$: ASL      R2      ;SHIFT # 3 PLACES LEFT
1595 003170 006103      ROL      R3
1596 003172 005301      DEC      R1
1597 003174 001374      BNE      2$
1598 003176 012701 000003      3$: MOV      #3,R1      ;SET SHIFT COUNTER
1599 003202 116324 001140      MOVB      DIGTAB(R3),(R4)+      ;MOVE ASCII EQUIV TO OUTPUT
1600 003206 005003      CLR      R3
1601 003210 005300      DEC      R0      ;DECREMENT DIGIT COUNT
1602 003212 001365      BNE      2$      ;GET NEXT DIGIT
1603 003214 105737 001133      TSTB      @#TYPFLG      ;BRANCH IF ASCII IS
1604 003220 001002      BNE      4$      ;NOT TO BE TYPED
1605 003222 000004 001152      TYPE,ODIGITS
1606 003226 4$:
1607 (1) 003226 004737 003104      JSR      PC,RESTORE      ;RESTORE REGISTERS FROM THE STACK
1608 003232 000207      RTS      PC
1609
1610      ;SUBROUTINE TO CONVERT OCTAL DATA TO DECIMAL ASCII
1611      ;CALL: MOV      NUMBER,R2      ;MOVE NUMBER TO R2
1612      ;      JSR      PC,CNVDEC
1613
1614 003234 110637 001133  CNVDEC: MOVB      SP,@#TYPFLG      ;SET DO NOT TYPE FLAG
1615 003240 000402      BR      CNVTD
1616
1617      .SBTTL      OCTAL TO DECIMAL & TYPE ROUTINE
1618      ;THIS ROUTINE CONVERTS AN OCTAL # TO DECIMAL ASCII AND TYPES IT OUT
1619      ;CALL: MOV      NUMBER,R2      ;PUT # IN R2
1620      ;      JSR      PC,TYPDEC      ;CALL ROUTINE
1621 003242 105037 001133  TYPDEC: CLRB      @#TYPFLG      ;SET TYPE FLAG
```

1622 003246  
(1) 003246 004737 003062  
1623 003252 005000  
1624 003254 012704 001152  
1625 003260 005003  
1626 003262 166002 003342  
1627 003266 103402  
1628 003270 005203  
1629 003272 000773  
1630 003274 066002 003342  
1631 003300 116324 001140  
1632 003304 062700 000002  
1633 003310 005760 003342  
1634 003314 001361  
1635 003316 112724 000060  
1636 003322 105737 001133  
1637 003326 001002  
1638 003330 000004 001152  
1639 003334  
(1) 003334 004737 003104  
1640 003340 000207  
1641  
1642 003342 023420  
1643 003344 001750  
1644 003346 000144  
1645 003350 000012  
1646 003352 000001  
1647 003354 000000  
1648  
1649  
1650  
1651  
1652  
1653  
1654  
1655  
1656  
1657  
1658  
1659  
1660 003356 010246  
1661 003360 010346  
1662 003362 006302  
1663 003364 006302  
1664 003366 010203  
1665 003370 000004 016436  
1666 003374 016302 002124  
1667 003400 004737 003242  
1668 003404 000004 001413  
1669 003410 016302 002126  
1670 003414 004737 003242  
1671 003420 000004 001420  
1672 003424 000004 016446  
1673 003430 013702 001016  
1674 003434 004737 003242  
1675 003440 000004 001402

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

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

.SBTTL TYPE SPECIFIED TIMES ROUTINE
;THIS SUBROUTINE OUTPUTS THE TIME SPECIFICATIONS FOR THE TEST
;AND ALSO THE ACTUAL TIME RECORDED (ATIME)
;FORMAT OF LINE TYPED
;RANGE=<AAAAAA-BBBBBB> ACTUAL=CCCCC
;WHERE: AAAAAA IS MAXIMUM TIME FOR TEST (STIMTBL(TSTNUMX4)).
        BBBBBB IS MINIMUM TIME FOR TEST (STIMTBL(TSTNUMX4+2)).
        CCCCCC IS ACTUAL TIME RECORDED BY TEST (ATIME).
;CALL: MOVB TEST NUMBER,R2 ;LOAD TEST NUMBER
        MOV #TIME,@#ATIME ;MOVE TIME TO ATIME
        JSR PC,OUTSPC
OUTSPC: MOV R2,-(SP) ;SAVE R2 & R3 ON THE STACK
        MOV R3,-(SP)
        ASL R2 ;MULTIPLY TEST # TIMES 4
        ASL R2 ;TO FORM INDEX INTO STIMTBL
        MOV R2,R3 ;R3 CONTAINS INDEX INTO TABLE
        TYPE,L,RNG
        MOV STIMTBL(R3),R2 ;GET MAXIMUM SPEC TIME
        JSR PC,TYPDEC ;CONVERT TO DECIMAL & TYPE
        TYPE,DASH
        MOV STIMTBL+2(R3),R2 ;GET MINIMUM TIME
        JSR PC,TYPDEC ;CONVERT TO DECIMAL & TYPE
        TYPE,ANGTAB
        TYPE,L,ACT
        MOV @#ATIME,R2 ;GET ACTUAL TIME
        JSR PC,TYPDEC ;CONVERT TO DECIMAL & TYPE
        TYPE,CRLF
```

1676 003444 012603  
 1677 003446 012602  
 1678 003450 000207  
 1679  
 1680  
 1681  
 1682  
 1683  
 1684  
 1685  
 1686  
 1687  
 1688 003452 010246  
 1689 003454 010346  
 1690 003456 113703 001124  
 1691 003462 006303  
 1692 003464 006303  
 1693 003466 000004 016436  
 1694 003472 016302 002264  
 1695 003476 004737 003242  
 1696 003502 000004 001413  
 1697 003506 016302 002266  
 1698 003512 004737 003242  
 1699 003516 000004 001420  
 1700 003522 000004 016446  
 1701 003526 013702 001016  
 1702 003532 004737 003242  
 1703 003536 000004 016113  
 1704 003542 113702 001124  
 1705 003546 004737 003134  
 1706 003552 000004 001402  
 1707 003556 012603  
 1708 003560 012602  
 1709 003562 000207  
 1710  
 1711  
 1712  
 1713  
 1714 003564  
 (1) 003564 004737 003062  
 1715 003570 012700 001272  
 1716 003574 012701 001122  
 1717 003600 005011  
 1718 003602 005061 000002  
 1719 003606 122710 000015  
 1720 003612 001414  
 1721 003614 112002  
 1722 003616 042702 177770  
 1723 003622 012703 000003  
 1724 003626 006311  
 1725 003630 006161 000002  
 1726 003634 005303  
 1727 003636 001373  
 1728 003640 050211  
 1729 003642 000761  
 1730 003644

```

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

.SBTTL      TYPE GAP TIMES SUBROUTINE
:THIS SUBROUTINE IS USED TO TYPE THE SPECIFIED GAP SIZES (RECORDED IN
:TST021). IT IS CALLED BY THE GAPOK ROUTINE IF THE GAP SIZE IS OUT OF
:RANGE VIA THE HLT ROUTINE (HLT+2).
:CALL:  MOVB  #GAP,GAP      ;LOAD GAP # INTO GAP
:      MOV   #TIME,ATIME    ;LOAD ACTUAL TIME INTO ATIME
:      JSR   PC,OUTGAP
OUTGAP:  MOV   R2,-(SP)      ;SAVE R2 AND R3
        MOV   R3,-(SP)
        MOVB  GAP,R3        ;GET GAP #
        ASL  R3
        ASL  R3
        TYPE,L.RNG
        MOV  GTIMTBL(R3),R2 ;GET MAX TIME
        JSR  PC,TYPDEC      ;CONVERT TO DECIMAL & TYPE
        TYPE,DASH
        MOV  GTIMTBL+2(R3),R2 ;GET MIN TIME
        JSR  PC,TYPDEC      ;CONVERT TO DECIMAL & TYPE
        TYPE,ANGTAB
        TYPE,L.ACT
        MOV  @ATIME,R2      ;GET ACTUAL TIME
        JSR  PC,TYPDEC      ;CONVERT TO DECIMAL & TYPE
        TYPE,E.GAP
        MOVB  @GAP,R2       ;GET GAP #
        JSR  PC,TYPDEC      ;TYPE GAP #
        TYPE,CRLF
        MOV  (SP)+,R3       ;RESTORE R3 AND R2
        MOV  (SP)+,R2
        RTS  PC

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

```

(1) 003644 004737 003104  
1731 003650 000207

JSR PC,RESTORE ;RESTORE REGISTERS FROM THE STACK  
RTS PC ;RETURN

1732  
1733  
1734  
1735  
1736  
1737  
1738

.SBTTL PUBLISH SUBROUTINE  
:THE PUBLISH SUBROUTINE AVERAGES THE RECORDED TIMES FOR EACH TEST IT-  
:ERATION (IF 16. ITERATIONS) AND PLACES THE AVERAGE RESULT IN 'ATIME'.  
:IT TYPES THE NAME OF THE FUNCTION THAT WAS TIMED,THE TIME SPEC-  
:IFICATION AND THE ACTUAL TIME .

1739 003652  
(1) 003652 004737 003062  
1740 003656 012700 001020  
1741 003662 113701 001125  
1742 003666 122701 000001  
1743 003672 001423  
1744 003674 005002  
1745 003676 005003  
1746 003700 122701 000020  
1747 003704 001402  
1748 003706 000000  
1749 003710 000777

PUBLISH:  
JSR PC,SAVE ;SAVE REGISTERS ON THE STACK  
MOV #ATIMTBL,R0 ;GET TABLE ADDRESS CONTAINING TIMES  
MOVB @#ITCNT,R1 ;GET # OF ENTRIES (GIVEN BY ITERATION COUNT)  
CMPB #1,R1 ;BRANCH IF SINGLE ITERATION  
BEQ 4\$  
CLR R2 ;CLEAR 'SUM' REGISTERS  
CLR R3  
CMPB #16.,R1 ;BRANCH IF 16. ITERATIONS  
BEQ 1\$  
HALT  
BR . ;ITERATION COUNT MUST BE 1 OR 16.  
;DO NOT CHANGE POSIT OF SW11  
;WHEN TEST IS RUNNING.

1750  
1751  
1752 003712 062002  
1753 003714 005503  
1754 003716 005301  
1755 003720 001374  
1756

1\$: ADD (R0)+,R2 ;SUM INDIVIDUAL TIMES  
ADC R3  
DEC R1  
BNE 1\$

1757 003722 012700 000004  
1758 003726 006203  
1759 003730 006002  
1760 003732 005300  
1761 003734 001374  
1762 003736 010237 001016  
1763

2\$: MOV #4,R0  
3\$: ASR R3 ;SHIFT TIME IN R3 & R2 4 PLACES  
ROR R2 ;RIGHT = DIVIDE BY 16.  
DEC R0  
BNE 3\$

1764 003742 113700 001126  
1765 003746 006300  
1766 003750 016037 002364 003760  
1767 003756 000004  
1768 003760 000000  
1769 003762 113702 001126

MOV R2,@ATIME ;MOVE AVERAGED TIMES

1770 003766 004737 003356  
1771 003772 004737 003104  
1772 003776 000207

4\$: MOVB @#TSTNUM,R0 ;GET TEST #  
ASL R0  
MOV #NAMPTR(R0),5\$ ;GET TEST NAME STRING ADDRESS  
TYPE

1773  
1774  
1775  
1776  
1777  
1778

5\$: .WORD 0  
MOVB @#TSTNUM,R2 ;GET TEST #  
JSR PC,OUTSPC ;OUTPUT TIMES  
JSR PC,RESTORE ;RESTORE REGISTERS FROM THE STACK  
RTS PC

1779 004000 010046  
1780 004002 012700 001272  
1781 004006 105737 177560  
1782 004012 100375  
1783  
1784 004014 113746 177562

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

.INPUT: MOV R0,-(SP) ;SAVE R0 ON THE STACK  
1\$: MOV #INBUF,R0  
2\$: TSTB @#TKS  
BPL 2\$  
MOVB @#TKB,-(SP) ;GET CHARACTER

```
1785 004020 042716 000200      BIC      #200,(SP)
1786 004024 122716 000177      CMPS     #177,(SP)      ;CHECK RUBOUT
1787 004030 001004              BNE      3$
1788 004032 124026              CMPB     -(RO),(SP)+    ;REMOVE CHARACTER FROM INPUT
1789 004034 000004 001405      TYPE,BKSLSH
1790 004040 000762              BR       2$            ;WAIT FOR NEXT CHARACTER
1791 004042 122716 000025      3$:     CMPB     #CNTRLU,(SP) ;CHECK CONTROL U (^U)
1792 004046 001004              BNE      4$
1793 004050 005726              TST     (SP)+
1794 004052 000004 001402      TYPE,CRLF
1795 004056 000751              BR       1$
1796 004060 122716 000003      4$:     CMPB     #CNTRLC,(SP) ;BRANCH IF NOT CONTROL C
1797 004064 001003              BNE      40$
1798 004066 000005              RESET
1799 004070 000137 006344      JMP      @#INIT        ;RESTART I/O
1800 004074 111637 001407      40$:    MOV      (SP),@#ECHO  ;RESTART PROGRAM
1801 004100 111620              MOV      (SP),(RO)+
1802 004102 122726 000015      CMPB     #CR,(SP)+
1803 004106 001403              BEQ      5$
1804 004110 000004 001407      TYPE,ECHO
1805 004114 000734              BR       2$
1806 004116 000004 001402      5$:     TYPE,CRLF
1807 004122 012600              MOV      (SP)+,RO
1808 004124 000207              RTS      PC

;KEYBOARD INTERRUPT SERVICE ROUTINE
1810
1811 004124 113746 177562      TKISR:  MOV      @#TKB,-(SP) ;GET TYPED CHARACTER
1812 004132 042716 000200      BIC      #200,(SP)      ;STRIP PARITY BIT
1813 004136 122716 000017      CMPB     #CNTRLO,(SP)   ;BRANCH IF NOT CONTROL O (^O)
1814 004142 001002              BNE      1$
1815 004144 111637 002627      MOV      (SP),#CNTRLO  ;SET CONTROL O INDICATOR IN TYPE ROUTINE
1816
1817 004150 122716 000003      1$:     CMPB     #3,(SP)        ;BRANCH IF NOT CONTROL C (^C)
1818 004154 001007              BNE      2$
1819 004156 023727 000042 013554      CMP      @#42,#$ENDAD   ;INHIBIT ^C IF ACT11 QV OR AA
1820 004164 001403              BEQ      2$
1821 004166 000005              RESET
1822 004170 000137 006344      JMP      @#INIT        ;RESTART PROGRAM
1823
1824 004174 122716 000001      2$:     CMPB     #CNTRLA,(SP)   ;BRANCH IF NOT ^A
1825 004200 001011              BNE      3$
1826 004202 022737 000176 001000      CMP      #SWREG,SWR     ;BRANCH IF HARDWARE SWR IS INVOKED
1827 004210 001010              BNE      4$
1828 004212 012737 177570 001000      MOV      #177570,SWR    ;INVOKE HARDWARE SWR
1829 004220 000004 014364      TYPE,M.HSWR
1830 004224 122716 000007      3$:     CMPB     #CNTRLG,(SP)   ;BRANCH IF NOT ^G
1831 004230 001005              BNE      5$
1832 004232 012737 000176 001000      4$:     MOV      #SWREG,SWR  ;INVOKE SOFTWARE SWR
1833 004240 004737 002526      JSR      PC,GTSWR      ;GET NEW SWITCH REGISTER
1834 004244 005726              TST     (SP)+          ;POP CHARACTER OFF THE STACK
1835 004246 000002              RTI                    ;RETURN TO CALLER
```

```
1837          .SBITL          ERROR SERVICE ROUTINES
1838          ;ROUTINE TO PROCESS ERROR TRAPS (TRAPS TO 4)
1839 004250 000000          ERRTRP: HALT
1840
1841          ;ERROR SERVICE ROUTINE
1842          ;THIS ROUTINE PROCESSES TWO TYPES OF ERRORS (OUT OF RANGE AND HARDWARE)
1843          ;THE CALLS FOR AN OUT OF RANGE ERROR ARE <HLT+1>,<HLT+2> AND, FOR A
1844          ;HARDWARE ERROR THE CALL IS <HLT>.
1845
1846 004252 004737 003062          .HLT: JSR PC,SAVE          ;SAVE REGISTERS ON THE STACK
1847 004256 110637 001127          1$:  MOVB SP,@WERFLG          ;SET ERROR FLAG
1848 004262 032777 020000 174510  BIT #SW13,@SWR          ;BRANCH IF NO TYP0UT
1849 004270 001121          BNE 4$
1850 004272 000004 015207          TYPE,E.HDR
1851 004276 113702 001126          MOVB @#TSTNUM,R2          ;GET TEST #
1852 004302 004737 003134          JSR PC,TYPOCT          ;AND TYPE IT
1853 004306 016600 000016          MOV 16(SP),R0          ;GET RETURN PC
1854 004312 162700 000002          SUB #2,R0          ;NOW PC OF HLT CALL
1855 004316 111000          MOVB (R0),R0          ;NOW HLT CALL ITSELF
1856 004320 001443          BEQ 2$          ;BRANCH IF HLT
1857 004322 000004 015272          TYPE,E.HDR2
1858 004326 122737 000005 001126  CMPB #5,@#TSTNUM          ;SEE IF IT IS TEST 5
1859 004334 001006          BNE 99$          ;CONTINUE IF NOT TEST 5
1860 004336 122737 000001 001136  CMPB #1,@#TE16          ;CHECK DRIVE TYPE
1861 004344 001002          BNE 99$          ;BRANCH IF NOT TU77
1862 004346 000004 015320          TYPE,E.7T5          ;TYP TU77 SPECIFIC MESSAGE
1863 004352 122737 000016 001126  99$: CMPB #16,@#TSTNUM          ;SEE IF IT IS TEST 16
1864 004360 001006          BNE 98$          ;CONTINUE IF NOT TEST 16
1865 004362 122737 000001 001136  CMPB #1,@#TE16          ;CHECK DRIVE TYPE
1866 004370 001002          BNE 98$          ;BRANCH IF NOT TU77
1867 004372 000004 015567          TYPE,E.7T16          ;TYP TU77 SPECIFIC MESSAGE
1868 004376 122700 000002          98$: CMPB #2,R0          ;BRANCH IF NOT HLT+2
1869 004402 001005          BNE 10$
1870 004404 004737 003452          JSR PC,OUTGAP          ;TYPE GAP SPECIFIED TIMES
1871 004410 000004 001402          TYPE,CRLF
1872 004414 000447          BR 4$
1873 004416 004737 003356          10$: JSR PC,OUTSPC          ;TYPE SPECIFIED TIMES
1874 004422 000004 001402          TYPE,CRLF
1875 004426 000442          BR 4$
1876 004430 016500 000014          2$: MOV ER(R5),R0
1877 004434 032765 002300 000032  BIT #PE1600,TC(R5)
1878 004442 001403          BEQ 20$
1879 004444 042700 102100          BIC #102100,R0
1880 004450 000402          BR 21$
1881 004452 042700 102300          20$: BIC #102300,R0
1882 004456 005700          21$: TST R0
1883 004460 001003          BNE 22$
1884 004462 000004 015163          TYPE,E.SFT          ;TYPE SOFT ERROR MESSAGE
1885 004466 000434          BR 6$
1886
1887 004470 000004 015217          22$: TYPE,E.HDR1
1888 004474 010500          MOV R5,R0          ;GET FIRST ADDRESS OF REGS.
1889 004476 012701 000007          MOV #7,R1          ;TYPE FIRST 7 REGS.
1890 004502 012002          3$: MOV (R0)+,R2          ;GET REG CONTENTS
1891 004504 004737 003134          JSR PC,TYPOCT          ;AND TYPE IT
1892 004510 000004 001415          TYPE,SPACE2
```

1893	004514	005301			DEC	R1		
1894	004516	001371			BNE	3\$		
1895	004520	016502	000032		MOV	TC(R5),R2	;GET CONTENTS OF TC REGISTER	
1896	004524	004737	003134		JSR	PC,TYPE		
1897	004530	000004	001402		TYPE,CRLF			
1898								
1899	004534	032777	001000	174236	4\$:	BIT	#SW09,@SWR	;BRANCH IF NO RING THE BELL
1900	004542	001402			BEO	5\$		
1901	004544	000004	001411		TYPE,BELL			
1902	004550	005777	174224		5\$:	TST	@SWR	;HALT ON ERROR?
1903	004554	100001			BPL	6\$		
1904	004556	000000			HALT			
1905	004560				6\$:			
(1)	004560	004737	003104		JSR	PC,RESTORE	;RESTORE REGISTERS FROM THE STACK	
1906	004564	000002			RTI		;RETURN	
1907								
1908								

```

1910 .SBITL SCOPE SUBROUTINE
1911 :SCOPE ROUTINE
1912 :THIS ROUTINE IS ENTERED UPON COMPLETION OF EACH SUBTEST
1913 :THE SCOPE ROUTINE:
1914 :REPEATS TEST IF SW14 IS SET
1915 :STORES ACTUAL TIME FOR FUNCTION IN TIME TABLE (ATIMTBL)
1916 :PUBLISHES TIME IF SW10=0
1917 :UPDATES ITERATION COUNT AND IF ITERATIONS COMPLETE CONTINUES
1918 :TO NEXT TEST, OTHERWISE REPEATS TEST.
1919 :DELAYS BEFORE CONTINUING OR REPEATING TEST.
1920 :INITIALIZES DRIVE
1921 :RETURNS: R5=BASE ADDRESS OF TM03 REGISTERS (ADDRESS OF CS1)
1922 : R1='DS' REG ADDRESS
1923 : R0='FC' REG ADDRESS
1924 :
1925 004566 013705 001010 .SCOPE: MOV @TMBASE,R5 ;SET R5 TO FIRST TM REG
1926 004572 032777 040000 174200 BIT #SW14,@SWR ;BRANCH IF CONTINUOUS LOOP
1927 004600 001432 BEQ 2$ ;NOT DESIRED
1928 004602 017701 174172 1$: MOV @SWR,R1 ;GET SWITCHES
1929 004606 042701 177740 BIC #177740,R1 ;CLEAR ALL BUT TEST #
1930 004612 001406 BEQ 11$ ;BRANCH IF ALL SELECTED
1931 004614 120137 001126 CMPB R1,@TSTNUM ;BRANCH IF RUNNING SELECTED TEST
1932 004620 001403 BEQ 11$
1933 004622 012737 010050 001002 11$: MOV #TST000,SCPADR ;RESTART AT TST000
1934 004630 004737 005360 JSR PC,DELAY ;DELAY 350 MS
1935 004634 004737 005614 JSR PC,RHINIT ;INIT
1936 004640 105037 001127 CLRB @WERFLG ;CLEAR ERROR FLAG
1937 004644 013716 001002 MOV SCPADR,(SP)
1938 004650 010501 MOV R5,R1
1939 004652 062701 000012 ADD #DS,R1 ;ADDRESS OF 'DS' REG IS IN R1
1940 004656 010500 MOV R5,R0
1941 004660 062700 000006 ADD #FC,R0 ;ADDRESS OF 'FC' REG IS IN R0
1942 004664 000002 RTI
1943
1944 004666 105737 001127 2$: TSTB @WERFLG ;BRANCH IF ERROR FLAG IS SET
1945 004672 001006 BNE 3$
1946 004674 113700 001125 MOVB @ITCNT,R0 ;GET ITERATION COUNT
1947 004700 006300 ASL R0 ;STORE TIME IN TABLE
1948 004702 013760 001016 001020 MOV @ATIME,ATIMTBL(R0)
1949 004710 105237 001125 3$: INCB @ITCNT ;INCREMENT ITERATION COUNT
1950 004714 105737 001134 TSTB @PSCNT ;INHIBIT ITERATIONS ON
1951 004720 001410 BEQ 4$ ;ON FIRST PASS
1952 004722 032777 004000 174050 BIT #SW11,@SWR ;BRANCH IF SINGLE ITERATION DESIRED
1953 004730 001004 BNE 4$
1954 004732 122737 000020 001125 CMPB #16,@ITCNT ;BRANCH IF ITERATIONS INCOMPLETE
1955 004740 001320 BNE 1$
1956 004742 032777 000037 174030 4$: BIT #37,@SWR ;IF TEST SELECTED IS TEST 0
1957 004750 001002 BNE 42$ ;TREAT AS ALL TESTS
1958 004752 011637 001002 40$: MOV (SP),@SCPADR ;SET SCOPE ADDRESS TO NEXT TEST
1959 004756 032777 002000 174014 42$: BIT #SW10,@SWR ;BRANCH IF NO PUBLICATION DESIRED
1960 004764 001002 BNE 5$
1961 004766 004737 003652 JSR PC,PUBLISH ;GO PUBLISH TEST DATA
1962 004772 105037 001125 5$: CLRB @ITCNT ;RESET ITERATION COUNT
1963 004776 000701 BR 1$
1964
1965 .SBITL TIMER SUBROUTINES

```

```

1966
1967 ;SUBROUTINE TO SYNCHRONIZE THE TIMER AND TURN IT ON.
1968 ;REGISTER 4 IS CLEARED, AND THE OSCILLATOR POLARITY IS MONITORED
1969 ;THE ROUTINE IS EXITED WHEN THE OSCILLATOR POLARITY CHANGES WITH R3
1970 ;SET TO INDICATE THE POLARITY OF THE OSCILLATOR.
1971 ;CALL: JSR PC,TIMON
1972 ;RETURNS: R3 SET TO INDICATE LAST POLARITY (+24/-24=0/1)
1973 ; R4 = 0
1974
1975 005000 005004 TIMON: CLR R4 ;CLEAR TIME COUNT
1976 005002 012703 000024 MOV #24,R3 ;SET POLARITY TO '0' STATE
1977 005006 032765 000100 000024 BIT #OSC,MR(R5) ;BRANCH IF POLARITY IS '0'
1978 005014 001405 BEQ 2$
1979 005016 032765 000100 000024 1$: BIT #OSC,MR(R5) ;WAIT FOR OSCILLATOR TO RETURN
1980 005024 001374 BNE 1$
1981 005026 000405 BR 4$
1982
1983 005030 005403 2$: NEG R3 ;NEGATE PREV POLARITY INDICATOR
1984 005032 032765 000100 000024 3$: BIT #OSC,MR(R5) ;WAIT FOR OSCILLATOR TO RETURN
1985 005040 001774 BEQ 3$ ;TO '1' STATE
1986 005042 000207 4$: RTS PC
1987
1988 ;SUBROUTINE TO COUNT TIME
1989 ;EACH TIME THE OSCILLATOR TOGGLES (BIT <06> IN MR REG) REGISTER
1990 ;R4 IS INCREMENTED, AND THE REGISTER R3 IS NEGATED TO INDICATE
1991 ;THE LAST STATE OF THE OSCILLATOR.
1992 ;CALL JMP TIMER(R3) ;R3 IS SET BY TIMON ROUTINE
1993 ; R2=RETURN ADDRESS TO CALLER
1994 ;NOTE: THE TIME TO EXECUTE THIS ROUTINE IS VERY CRITICAL. IT MUST BE
1995 ;LESS THAN 40 US.
1996
1997 ;ENTER HERE VIA JMP TIMER(R3) WHEN R3=-24 (PREV STATE=1)
1998 005044 032765 000100 000024 TIMER1: BIT #OSC,MR(R5) ;BRANCH IF CURRENT STATE IS '0'
1999 005052 001406 BEQ TIMER ;GO INCREMENT TIME
2000 005054 000112 JMP (R2) ;RETURN TO TEST
2001
2002 ;.=TIMER1+24
2003 005070 005403 TIMER: NEG R3 ;NEGATE PREV STATE INDICATOR
2004 005072 005204 INC R4 ;INCREMENT 'TICK' COUNT
2005 005074 100401 BMI TIMERR ;BRANCH ON OVERFLOW
2006 005076 000112 JMP (R2) ;RETURN TO TEST
2007 005100 000004 016025 TIMERR: TYPE,E.TIMOV ;TYPE 'TIMER OVERFLOWED'
2008 005104 104400 HLT ;REPORT HARDWARE ERROR
2009 005106 000177 173670 JMP @SCPADR ;RETURN TO BEGINNING OF TEST
2010
2011 ;.=TIMER+24
2012 ;ENTER HERE VIA JMP TIMER(R3) WHEN R3=+24 (PREV STATE=0)
2013 005114 032765 000100 000024 TIMERO: BIT #OSC,MR(R5) ;BRANCH IF CURRENT STATE = '1'
2014 005122 001374 BNE TIMEP
2015 005124 000112 JMP (R2)
2016
2017 ;SUBROUTINE TO CHECK TIME RECORDED BY SUBTEST.
2018 ;THIS SUBROUTINE COMPUTES THE ACTUAL TIME (IN MICROSECONDS) AND CHECKS
2019 ;THAT THE TIME RECORDED BY THE SUBTEST IS CORRECT BY COMPARING THE TIME
2020 ;WITH THE HIGH LIMIT (STIMTBL(R0)) AND THE LOW LIMIT (STIMTBL+2(R0)).
2021 ;IF THE TIME IS OUT OF RANGE AN OUT OF RANGE ERROR TYPEOUT RESULTS.

```

```

2022 ;THE SUBROUTINE IS ENTERED WITH:
2023 ; R4=TICK COUNT
2024
2025 TIMOK:
(1) 005126 004737 003062 JSR PC,SAVE ;SAVE REGISTERS ON THE STACK
2026 005132 013700 001012 MOV OSCTIM,R0 ;GET TIME PER TICK
2027 005136 010401 MOV R4,R1 ;GET TICKS COUNT
2028 005140 005002 CLR R2 ;CLEAR SUMMING REGISTERS
2029 005142 005003 CLR R3
2030 005144 060002 1$: ADD R0,R2 ;MULTIPLY TIME PER TICK
2031 005146 005503 ADC R3 ;BY TICK COUNT
2032 005150 005301 DEC R1
2033 005152 001374 BNE 1$
2034 005154 010246 MOV R2,-(SP) ;DIVIDE COUNT BY 10.
2035
2036 005156 010346 MOV R3,-(SP)
2037 005160 012746 000012 MOV #10,-(SP)
2038 005164 004737 005446 JSR PC,DIVIDE
2039 005170 005726 TST (SP)+ ;DISCARD REMAINDER
2040 005172 012637 001016 MOV (SP)+,@#ATIME ;STORE QUOTIENT
2041 005176 113700 001126 MOVB @#TSTNUM,R0 ;GET TEST #
2042 005202 006300 ASL R0
2043 005204 006300 ASL R0
2044 005206 023760 001016 002124 CMP @#ATIME,STIMTBL(R0) ;CHECK THAT TIME IS WITHIN
2045 005214 101004 BHI 2$ ;LIMITS SPECIFIED
2046 005216 023760 001016 002126 CMP @#ATIME,STIMTBL+2(R0)
2047 005224 101001 BHI 3$
2048 005226 104401 2$: HLT+1 ;CALL ERROR ROUTINE
2049 005230 3$:
(1) 005230 004737 003104 JSR PC,.RESTORE ;RESTORE REGISTERS FROM THE STACK
2050 005234 000207 RTS PC ;RETURN
2051
2052 ;SUBROUTINE TO CHECK INDIVIDUAL GAP TIMES (PRODUCED BY TST021)
2053 ;SUBROUTINE COMPUTES THE ACTUAL TIME (IN MICROSECONDS) AND CHECKS
2054 ;THAT THE GAP TIME RECORDED BY THE SUBTEST (TST021) BY COMPARING THE
2055 ;TIME WITH THE MAX LIMIT (GTIMTBL-GAPTBL(R1)) AND THE MIN LIMIT
2056 ;:(GTIMTBL+2-GAPTBL(R1)).
2057 ;CALL: MOV #TICK COUNT,R4 ;R4 CONTAINS TICK COUNT
2058 ; MOVB #GAP,@#GAP ;LOCATION GAP CONTAINS GAP #
2059 ; JSR PC,GAPOK
2060
2061 GAPOK:
(1) 005236 004737 003062 JSR PC,SAVE ;SAVE REGISTERS ON THE STACK
2062 005242 013700 001012 MOV @#OSCTIM,R0 ;GET TIME PER TICK
2063 005246 010401 MOV R4,R1 ;GET TICK COUNT
2064 005250 005002 CLR R2 ;CLEAR SUMMING REGISTERS
2065 005252 005003 CLR R3
2066 005254 060002 1$: ADD R0,R2 ;MULTIPLY TICK COUNT
2067 005256 005503 ADC R3 ;BY TIME PER TICK
2068 005260 005301 DEC R1
2069 005262 001374 BNE 1$
2070
2071 005264 010246 MOV R2,-(SP) ;DIVIDE TIME BY 10.
2072 005266 010346 MOV R3,-(SP)
2073 005270 012746 000012 MOV #10,-(SP)
2074 005274 004737 005446 JSR PC,DIVIDE

```

```
2075 005300 005726          TST      (SP)+          ;DISCARD REMAINDER
2076 005302 012637 001016    MOV      (SP)+,@#ATIME  ;STORE QUOTIENT
2077 005306 113703 001124    MOVB    @#GAP,R3       ;GET GAP #
2078 005312 006303          ASL      R3             ;MULTPLY BY 4
2079 005314 006303          ASL      R3             ;TO GET AT TABLE ENTRY
2080 005316 023763 001016 002264  CMP      @#ATIME,GTIMTBL(R3) ;CHECK TIME (MAX)
2081 005324 101004          BHI     2$             ;CHECK TIME (MIN)
2082 005326 023763 001016 002266  CMP      @#ATIME,GTIMTBL+2(R3)
2083 005334 101001          BHI     3$             ;REPORT OUT OF RANGE ERROR
2084 005336 104402          HLT+2                ;BRANCH IF TIMES NOT WANTED
2085 005340 032777 002000 173432 2$:      BIT      #SW10,@SWR
2086 005346 001001          BNE     100$          ;
2087 005350 000240          NOP
2088
2089 005352          100$:
(1) 005352 004737 003104    JSR     PC,..RESTORE  ;RESTORE REGISTERS FROM THE STACK
2090 005356 000207          RTS     PC            ;RETURN TO TEST
2091
2092          .SBTTL          DELAY SUBROUTINES
2093          ;THIS SUBROUTINE CAUSES A DELAY OF 115 MS.
2094 005360 004737 005000    DELAY: JSR     PC,TIMON
2095 005364 010246          MOV     R2,-(SP)      ;SAVE R2 ON THE STACK
2096 005366 012702 005376    MOV     #2$,R2        ;SET RETURN ADDRESS FOR TIMER
2097 005372          1$:
(1) 005372 000163 005070    JMP     TIMER(R3)     ;GO TO TIMER & RETURN VIA R2
2098 005376 032704 004000    2$:      BIT     #4000,R4
2099 005402 001773          BEQ     1$
2100 005404 012602          MOV     (SP)+,R2     ;RESTORE R2
2101 005406 000207          RTS     PC
2102
2103          ;THIS SUBROUTINE ALLOWS A CALLER SPECIFIED DELAY.
2104          ;CALL: MOV     DELAY TIME,DELTIM ;LOAD DELAY TIME (# OF TICKS)
2105          ;      JSR     PC,DELAYV
2106 005410 005737 001120    DELAYV: TST    DELTIM ;BRANCH IF 0 DELAY
2107 005414 001413          BEQ     3$
2108 005416 004737 005000    JSR     PC,TIMON     ;TURN TIMER ON
2109 005422 010246          MOV     R2,-(SP)     ;SAVE R2 ON THE STACK
2110 005424 012702 005434    MOV     #2$,R2        ;SET RETURN ADDRESS FROM TIMER
2111 005430          1$:
(1) 005430 000163 005070    JMP     TIMER(R3)     ;GO TO TIMER & RETURN VIA R2
2112 005434 023704 001120    2$:      CMP     @#DELTIM,R4
2113 005440 101373          BHI     1$
2114 005442 012602          MOV     (SP)+,R2     ;RESTORE R2
2115 005444 000207          3$:      RTS     PC
2116
2117          .SBTTL          DIVIDE SUBROUTINE
2118          ;THIS SUBROUTINE DIVIDES A DOUBLE PRECISION # AND RETURNS THE RESULT
2119          ;TO THE CALLER ON THE STACK. BOTH DIVIDEND & DIVISOR MUST BE POSITIVE.
2120          ;CALL: MOV     LEAST SIGNIFICANT HALF DIVIDEND,-(SP)
2121          ;      MOV     #MOST SIGNIFICANT HALF DIVIDEND,-(SP)
2122          ;      MOV     #DIVISOR,-(SP)
2123          ;      JSR     PC,DIVIDE
2124          ;RETURN
2125          ;      (SP)=REMAINDER ON STACK
2126          ;      2(SP)=QUOTIENT
2127
```

```

2128 ;NOTE: THIS SUBROUTINE DESTROYS PREVIOUS CONTENTS OF R0,R1,R2 & R3.
2129
2130 005446 005046 DIVIDE: CLR -(SP) ;SAVE LOC FOR SIGNS
2131 005450 012746 000021 MOV #17, -(SP) ;SET ITERATION COUNT
2132 005454 016601 000012 MOV 12(SP),R1 ;GET LSH DIVIDEND
2133 005460 016600 000010 MOV 10(SP),R0 ;GET MSH DIVIDEND
2134 005464 016602 000006 MOV 6(SP),R2 ;GET DIVISOR
2135 005470 005402 NEG R2 ;NEGATE DIVISOR
2136 005472 000241 CLC ;CLEAR 'C' BIT IN PSW
2137 005474 000405 BR 2$
2138 005476 006100 1$: ROL R0 ;ROTATE MSH DIVIDEND
2139 005500 010003 MOV R0,R3 ;SAVE IN R3
2140 005502 060203 ADD R2,R3 ;SUBTRACT DIVISOR FROM MSH DIVIDEND
2141 005504 103001 BCC 2$ ;BRANCH IF DIVIDEND > DIVISOR
2142 005506 010300 MOV R3,R0 ;SAVE REMAINDER IN R0
2143 005510 006101 2$: ROL R1 ;ROTATE LSH DIVIDEND
2144 005512 005316 DEC (SP) ;DECREMENT ITERATION COUNT
2145 005514 001370 BNE 1$
2146 005516 005726 TST (SP)+ ;POP ITERATION COUNTER
2147 005520 005726 TST (SP)+ ;POP SIGN CORRECTION
2148 005522 010166 000006 MOV R1,6(SP) ;PUSH REMAINDER ON STACK
2149 005526 010066 000004 MOV R0,4(SP) ;PUSH QUOTIENT ONTO STACK
2150 005532 012616 MOV (SP)+,(SP)
2151 005534 000207 RTS PC
2152
2153 ;SBTTL DRIVE SUBROUTINES
2154 ;SUBROUTINE TO CHECK IF DRIVE IS AVAILABLE
2155 ;CALL: MOV B DRIVE#,DRVNUM
2156 ; JSR PC,DRVAVA
2157 ;RETURN: 'C' BIT SET IF NOT AVAILABLE
2158 005536 113765 001004 000010 DRVAVA: MOV B @#DRVNUM,CS2(R5) ;LOAD DRIVE #
2159 005544 032765 040000 000026 BIT #TAP,DT(R5) ;CHECK IF TAPE UNIT
2160 005552 001003 BNE 1$
2161 005554 004737 005614 JSR PC,RHINIT
2162 005560 000262 SEV ;SET 'V' TO IND NOT AVAIL
2163 005562 000207 1$: RTS PC ;RETURN
2164
2165 ;SUBROUTINE TO CHECK IF TE16/TU77 SLAVE IS AVAILABLE FOR TEST
2166 ;CALL: MOV B DRIVE #,@#DRVNUM ;PASS DRIVE # VIA DRVNUM
2167 ; MOV B SLAVE #,@#SLVNUM ;PASS SLAVE # VIA SLVNUM
2168 ; JSR PC,SLVAVA ;CALL SUBROUTINE
2169 005564 113765 001004 000010 SLVAVA: MOV B @#DRVNUM,CS2(R5) ;LOAD DRIVE #
2170 005572 113765 001005 000032 MOV B @#SLVNUM,TC(R5) ;AND SLAVE #
2171 005600 032765 002000 000026 BIT #SPR,DT(R5) ;BRANCH IF SLAVE PRESENT
2172 005606 001001 BNE 1$
2173 005610 000262 SEV ;SET 'V' TO INDICATE NO SLAVE
2174 005612 000207 1$: RTS PC
2175
2176 ;SUBROUTINE TO INITIALIZE RH CONTROLLER
2177 ;CALL: JSR PC,RHINIT
2178
2179 005614 012765 000040 000010 RHINIT: MOV #40,CS2(R5)
2180 005622 113765 001004 000010 MOV B @#DRVNUM,CS2(R5)
2181 005630 005046 CLR -(SP)
2182 005632 113716 001005 MOV B @#SLVNUM,(SP)
2183 005636 012665 000032 MOV (SP)+,TC(R5) ;LOAD SLAVE # INTO TC REG

```

```

2184 005642 052765 001700 000032      BIS      #NORM11,TC(R5)
2185 005650 000207      RTS      PC
2186
2187      ;SUBROUTINE TO WAIT FOR DRIVE READY (DRY)
2188 005652 005027      WAITRDY:CLR      (PC)+      ;CLEAR WAIT TIMER
2189 005654 000000      WAITTIM: .WORD 0
2190 005656 105765 000012      1$: TSTB      DS(R5)      ;WAIT FOR READY TO SET
2191 005662 100406      BMI      2$
2192 005664 005237 005654      INC      WAITTIM      ;INCREMENT WAIT TIMER
2193 005670 001372      BNE      1$      ;BRANCH IF TIME HAS NOT EXPIRED
2194 005672 000004 016052      TYPE,E.TIMEXP      ;TYPE 'TIME EXPIRED WAITING FOR RDY'
2195 005676 000425      BR      99$      ;TAKE ERROR EXIT
2196 005700 032765 002000 000012 2$: BIT      #EOT,DS(R5)      ;CHECK FOR END OF TAPE
2197 005706 001415      BEQ      3$      ;BRANCH IF NO EOT
2198 005710 000004 014204      TYPE,M.NAM
2199 005714 000004 014723      TYPE,M.EOT      ;TYPE 'END OF TAPE'
2200 005720 004737 005756      JSR      PC,.REWIND      ;REWIND SLAVE
(1) 005724 102412      BVS      99$      ;BRANCH IF ERROR ON REWIND
2201 005726 004737 006040      JSR      PC,WRITE      ;WRITE A RECORD
2202 005732 005215      INC      (R5)      ;SET 'GO' BIT
2203 005734 004737 005652      JSR      PC,WAITRDY      ;WAIT FOR READY
2204 005740 000404      BR      99$      ;TAKE ERROR EXIT
2205 005742 032765 040000 000012 3$: BIT      #ERR,DS(R5)      ;CHECK ERROR EXIT
2206 005750 001401      BEQ      100$
2207 005752 000262      99$: SEV
2208 005754 000207      100$: RTS      PC
2209      ;SUBROUTINE TO REWIND A UNIT (DRIVE/SLAVE COMBINATION)
2210      ;CALL      MOVB      DRIVE #,@#DRVNUM
2211      ;      MOVB      SLAVE #,@#SLVNUM
2212      ;      JSR      PC,.REWIND
2213      ;SUBROUTINE RETURNS TO CALLER WITH SELECTED SLAVE AT 'BOT', & 'V' SET IF
2214      ;AN ERROR OCCURS.
2215
2216 005756 004737 005614      .REWIND:JSR      PC,RHINIT      ;INITIALIZE CONTROLLER
2217 005762 004337 006174      JSR      R3,TCMD      ;GO TO TM COMMAND SUBROUTINE
2218 005766 000000      .WORD 0      ;BUS ADDRESS (NOT USED)
2219 005770 000000      .WORD 0      ;WORD COUNT (NOT USED)
2220 005772 000000      .WORD 0      ;FRAME COUNT (NOT USED)
2221 005774 000006      .WORD RWD      ;REWIND COMMAND
2222 005776 005215      INC      (R5)      ;SET 'GO' BIT
2223 006000 032765 000002 000012 1$: BIT      #BOT,DS(R5)      ;BRANCH IF 'BOT' SET
2224 006006 001005      BNE      2$
2225 006010 032765 040000 000012      BIT      #ERR,DS(R5)      ;CHECK ERROR BIT
2226 006016 001006      BNE      99$      ;BRANCH IF ERROR BIT SET
2227 006020 000767      BR      1$
2228
2229 006022 032765 020000 000012 2$: BIT      #PIP,DS(R5)      ;WAIT FOR TAPE MOTION TO STOP
2230 006030 001374      BNE      2$
2231 006032 000401      BR      100$
2232 006034 000262      99$: SEV
2233 006036 000207      100$: RTS      PC
2234
2235      ;SUBROUTINE TO WRITE 256. WORD RECORD
2236      ;CALL: JSR      PC,WRITE
2237
2238 006040 004337 006174      WRITE: JSR      R3,TCMD      ;GO TO TM COMMAND SUBROUTINE
  
```

```
2239 006044 017450          .WORD  WTBUF          ;BUS ADDRESS
2240 006046 177600          .WORD  WRDCNT         ;WORD COUNT
2241 006050 177400          .WORD  FRMCNT         ;FRAME COUNT
2242 006052 000060          .WORD  WFWFWD        ;WRITE FORWARD COMMAND
2243 006054 000207          RTS      PC
2244
2245          ;SUBROUTINE TO READ A 256. WORD RECORD.
2246          ;CALL: JSR      PC,READ
2247
2248 006056 004337 006174      READ: JSR      R3,@TMCMD
2249 006062 017450          .WORD  RDBUF          ;ADDRESS OF READ BUFFER
2250 006064 177600          .WORD  WRDCNT         ;2'S COMPLEMENT OF WORD COUNT
2251 006066 177400          .WORD  FRMCNT         ;2'S COMPLEMENT OF FRAME COUNT
2252 006070 000070          .WORD  RDFWD         ;READ FORWARD COMMAND
2253 006072 000207          RTS      PC
2254
2255          ;SUBROUTINE TO INITIATE READ REVERSE COMMAND
2256          ;CALL: JSR      PC,REVRD
2257
2258 006074 004337 006174      REVRD: JSR      R3,TMCMD
2259 006100 020050          .WORD  RDBUF+256.    ;ADDRESS OF READ REVERSE BUFFER
2260 006102 177600          .WORD  WRDCNT         ;2'S COMPLEMENT OF WORD COUNT
2261 006104 177400          .WORD  FRMCNT         ;2'S COMPLEMENT OF FRAME COUNT
2262 006106 000076          .WORD  RDREV         ;READ REVERSE COMMAND
2263 006110 000207          RTS      PC
2264
2265          ;SUBROUTINE TO SPACE FORWARD 1 RECORD
2266 006112 012765 177777 000006  FWDSPC: MOV     #-1,FC(R5) ;LOAD RECORD COUNT
2267 006120 012715 000031          MOV     #SPCFWD+1,(R5) ;LOAD COMMAND
2268 006124 004737 005652          JSR     PC,WAITRDY    ;WAIT FOR READY
2269 006130 000207          RTS      PC          ;RETURN
2270
2271          ;SUBROUTINE TO WRITE A RECORD AND BACK SPACE OVER THE RECORD.
2272 006132 004737 006040      WRT.BK: JSR     PC,WRITE ;WRITE THE RECORD
2273 006136 005215          INC     (R5)          ;SET 'GO' BIT
2274 006140 004737 005652          JSR     PC,WAITRDY
2275 006144 102412          BVS    2$
2276 006146 012765 177777 000006  MOV     #-1,FC(R5) ;LOAD RECORD COUNT
2277 006154 012715 000033          MOV     #SPCREV+1,(R5) ;LOAD COMMAND
2278 006160 004737 005652          JSR     PC,WAITRDY
2279 006164 102402          BVS    2$
2280 006166 004737 005360      1$: JSR     PC,DELAY    ;WAIT FOR TAPE MOTION TO STOP
2281 006172 000207      2$: RTS      PC
2282
2283          ;SUBROUTINE TO LOAD A COMMAND
2284          ;CALL: JSR      R3,TMCMD
2285          ;      .WORD  BUS ADDRESS
2286          ;      .WORD  WORD COUNT (2'S COMPLEMENT)
2287          ;      .WORD  FRAME COUNT (2'S COMPLEMENT)
2288          ;      .WORD  COMMAND
2289
2290 006174 012365 000004      TMCMD: MOV     (R3)+,BA(R5) ;LOAD BUS ADDRESS
2291 006200 012365 000002          MOV     (R3)+,WC(R5) ;LOAD WORD COUNT
2292 006204 012365 000006          MOV     (R3)+,FC(R5) ;LOAD FRAME COUNT
2293 006210 012315          MOV     (R3)+,(R5)   ;LOAD COMMAND
2294 006212 000203          RTS      R3          ;RETURN
```

```
2295
2296                :SUBROUTINE TO PRINT SERIAL NUMBER
2297                :JSR      PC,SNPT
2298
2299 006214 016503 000030      SNPT:  MOV      SN(R5),R3
2300 006220 012701 001152      MOV      #ODIGITS,R1
2301 006224 000303           SWAB     R3
2302 006226 006003           ROR      R3
2303 006230 006003           ROR      R3
2304 006232 006003           ROR      R3
2305 006234 006003           ROR      R3           :GET FIRST DIGIT
2306 006236 042703 177760      BIC      #177760,R3
2307 006242 052703 000260      BIS      #260,R3
2308 006246 110321           MOVB     R3,(R1)+       :FILL FIRST DIGIT
2309 006250 016503 000030      MOV      SN(R5),R3
2310 006254 000303           SWAB     R3
2311 006256 042703 177760      BIC      #177760,R3
2312 006262 052703 000260      BIS      #260,R3
2313 006266 110321           MOVB     R3,(R1)+       :GET SECOND DIGIT
2314 006270 016503 000030      MOV      SN(R5),R3
2315 006274 006003           ROR      R3
2316 006276 006003           ROR      R3
2317 006300 006003           ROR      R3
2318 006302 006003           ROR      R3
2319 006304 042703 177760      BIC      #177760,R3
2320 006310 052703 000260      BIS      #260,R3
2321 006314 110321           MOVB     R3,(R1)+       :GET THIRD DIGIT
2322 006316 016503 000030      MOV      SN(R5),R3
2323 006322 042703 177760      BIC      #177760,R3
2324 006326 052703 000260      BIS      #260,R3
2325 006332 110321           MOVB     R3,(R1)+       :GET FOURTH DIGIT
2326 006334 105011           CLRB    (R1)
2327 006336 000004 001152      TYPE,ODIGITS       :TYPE SERIAL NUMBER
2328 006342 000207           RTS      PC           :RETURN
2329
```

```
2331          .SBITL PROGRAM INITIALIZATION
2332 006344 012706 000600 INIT: MOV #STKPTR,SP ;SET STACK PTR
2333 006350 005037 001272 CLR @#INBUF
2334
2335 006354 013746 000006 MOV @#6,-(SP) ;SAVE VECTORS
2336 006360 013746 000004 MOV @#4,-(SP)
2337 006364 012737 006404 000004 MOV #61$,@#4 ;SET UP FOR TIMEOUT
2338 006372 022777 177777 172400 CMP #-1,@SWR ;REFERENCE HARDWARE SWITCH REGISTER
2339 006400 001402 BEQ 60$
2340 006402 000404 BR 62$
2341 006404 022626 61$: CMP (SP)+,(SP)+ ;ADJUST STACK
2342 006406 012737 000176 001000 60$: MOV #SWREG,SWR ;POINT TO SOFTWARE SWITCH REG
2343 006414 012637 000004 62$: MOV (SP)+,@#4 ;RESTORE VECTORS
2344 006420 012637 000006 MOV (SP)+,@#6
2345 006424 105037 001131 CLRB @#PRGFLG ;CLEAR PROGRAM FLAG
2346 006430 105037 001135 CLRB @#ASFLG ;CLEAR ASK FLAG
2347 006434 105037 001134 CLRB @#PSCNT ;SET PASS COUNT = 0
2348 006440 005027 CLR (PC)+ ;:CLEAR CHAIN INDICATOR
(1) 006442 000000 CHNFLG: .WORD 0 ;:CHAIN MODE INDICATOR
(1) ;:1/0 = CHAIN/NOT CHAIN MODE
(1) ;:BRANCH IF IN DUMP MODE
(1) 006444 005737 000042 TST @#42
(1) 006450 001407 BEQ 50$
(1) 006452 012737 000176 001000 MOV #SWREG,SWR ;:INVOKE SOFTWARE SWR
(1) 006460 005237 006442 INC CHNFLG ;:SET CHNFLG = CHAIN MODE
(1) 006464 000137 006470 JMP 1$ ;:GO TO CHAIN ADDRESS
(1) 006470 50$:
2349 006470 122737 000006 000041 1$: CMPB #6,@#41 ;BRANCH IF NOT LOADED VIA TMDP
2350 006476 001003 BNE 2$
2351 006500 000004 014414 TYPE,I.REM ;ADVISE USER TO REMOVE TMDP
2352 006504 000000 HALT
2353 006506 000004 014204 2$: TYPE,M.NAM ;TYPE TITLE
2354 006512 005737 006442 TST CHNFLG ;SEE IF CHAIN MODE
2355 006516 001025 BNE 5$ ;IF SO: BR
2356 006520 105037 014204 CLRB M.NAM ;DO NOT TYPE TITLE ON RESTART
2357 006524 000004 014467 TYPE,I.REG ;ASK USER TO TYPE CONT BASE ADRS
2358 006530 013702 001010 MOV @#TMBASE,R2 ;GET CURRENT CONT BASE ADDRESS
2359 006534 004737 003134 JSR PC,TYPOCT ;AND TYPE IT
2360 006540 000004 001416 TYPE,SPACE
2361 006544 004737 004000 JSR PC,.INPUT ;GET USER INPUT
2362 006550 122737 000015 001272 CMPB #CR,@#INBUF ;DO NOT CHANGE CURRENT VALUE
2363 006556 001405 BEQ 5$ ;IF USER TYPES <CR>
2364 006560 004737 003564 4$: JSR PC,CNVTAO ;CONVERT ASCII TO OCTAL
2365 006564 013737 001122 001010 MOV @#OCTALO,@#TMBASE ;SET NEW ADDRESS
2366 006572 013705 001010 5$: MOV @#TMBASE,R5
2367
2368 ;ROUTINE TO CHECK IF CONTROLLER (RH11) IS AVAILAABLE
2369 006576 000261 SEC ;SET 'C' IN PSW
2370 006600 005715 TST (R5) ;BRANCH IF CONTROLLER AVAIL
2371 006602 103003 BCC 6$
2372 006604 000004 014762 TYPE,E.NCON
2373 006610 000655 BR INIT
2374 006612 012737 004250 000004 6$: MOV #ERRTRP,@#ERRVEC ;SET ERROR TRAP VECTOR
```

```
2376 ;ROUTINE TO GET TM03 DRIVES USER DESIRES TO TEST
2377 DRIVES: CLR @MERFLG ;CLEAR ERROR FLAG
2378 MOV #DRVTBL,R1 ;MARK ALL DRIVES AS NOT TO
2379 MOV #4,R0 ;BE TESTED. A '0' INDICATES
2380 1$: CLR (R1)+ ;THAT A DRIVE IS NOT TO BE
2381 DEC R0 ;TESTED
2382 BNE 1$
2383 TST CHNFLG ;BRANCH IF IN CHAIN MODE
2384 BNE 2$
2385 TYPE,I.DRVS
2386 JSR PC,,INPUT ;GET USER INPUT
2387 MOV #INBUF,R0
2388 CMPB #'A,(R0) ;IF USER RESPONDS WITH 'A' OR
2389 BEQ 2$ ;<CR> THEN ALL AVAILABLE DRIVES
2390 CMPB #CR,(R0) ;ARE TO BE TESTED
2391 BNE 4$
2392 2$: MOVB SP,PRGFLG ;SET FLAG TO IND ALL DRIVES
2393 MOV #DRVTBL,R1 ;MARK ALL DRIVES TO BE TESTED
2394 MOV #4,R0 ;A '-1' INDICATES THAT A DRIVE
2395 3$: MOV #-1,(R1)+ ;IS TO BE TESTED
2396 DEC R0
2397 BNE 3$
2398 BR CHKDRV ;GO CHECK DRIVE AVAILABILITY
2399
2400 ;GET USER SELECTED DRIVES AND MARK EACH DRIVE SELECTED TO BE TESTED
2401 4$: CMPB #CR,(R0)
2402 BEQ CHKDRV
2403 CMPB (R0),#',
2404 BNE 5$ ;CHECK IF 'COMMA'
2405 TSTB (R0)+ ;STEP PTR PAST 'COMMA'
2406 5$: MOVB (R0)+,R1
2407 BIC #177770,R1
2408 MOVB #-1,DRVTBL(R1)
2409 NOP
2410 BR 4$
2411
2412 ;ASCERTAIN THAT DRIVES (TM03'S) SPECIFIED ARE AVAILABLE
2413 CHKDRV: CLR R0 ;A (0) IN DRVTBL(R0) INDICATES
2414 1$: TSTB DRVTBL(R0) ;THE DRIVE IS NOT TO BE TESTED
2415 BNE 3$ ;A '1' INDICATES TO BE TESTED
2416 2$: INC R0
2417 CMPB #8,,R0
2418 BNE 1$
2419 BR 5$
2420 3$: MOVB R0,@DRVNUM ;GET DRIVE #
2421 JSR PC,@DRVAVA ;AND CHECK IF AVAILABLE
2422 BVC 2$ ;'V' BIT SET INDICATES NOT AVAIL
2423 TSTB @PRGFLG ;DO NOT TYPE NOT AVAILABLE
2424 BNE 4$ ;MESSAGE IF ALL SELECTED
2425 TYPE,E.NDRV
2426 MOVB DIGTAB(R0),@E.NAVA ;SET DRIVE # IN MESSAGE
2427 TYPE,E.NAVA
2428 MOVB SP,@MERFLG ;SET 'ERROR' FLAG
2429 4$: CLRB DRVTBL(R0) ;MARK DRIVE UNAVAILABLE
2430 BR 2$ ;CHECK NEXT DRIVE
2431 5$: TSTB @MERFLG ;GO GET SLAVES IF NO ERROR
```

```
2432 007062 001256          BNE      DRIVES          ;ELSE ASK USER TO RETYPE DRIVES
2433
2434          ;ROUTINE TO GET SLAVES (TE16/TU77'S) USER DESIRES TO TEST
2435 007064 105037 001127  SLAVES: CLR      @WERFLG          ;CLEAR ERROR INDICATOR
2436 007070 012701 001172  MOV      #SLVTBL,R1          ;MARK ALL SLAVES (64.) AS NOT
2437 007074 012700 000040  MOV      #32.,R0            ;TO BE TESTED.A 0 INDICATES THAT
2438 007100 005021          1$: CLR      (R1)+          ;A DRIVE'S SLAVE IS NOT TO BE
2439 007102 005300          DEC      R0                ;TESTED
2440 007104 001375          BNE      1$
2441 007106 012701 001172  MOV      #SLVTBL,R1          ;R1 POINTS TO DRIVE'S SLAVE
2442 007112 105760 001162  2$: TSTB   DRVTBL(R0)        ;BRANCH IF DRIVE IS TO BE TESTED
2443 007116 001007          BNE      4$                ;B IS AVAILABLE
2444 007120 062701 000010  3$: ADD      #8.,R1          ;STEP SLAVE PTR TO NEXT DRIVE'S
2445 007124 005200          INC      R0                ;SLAVES AND INCREMENT DRIVE #
2446 007126 122700 000010  CMPB    #8.,R0            ;CHECK ALL DRIVES
2447 007132 001367          BNE      2$                ;AND WHEN ALL DRIVES CHECKED
2448 007134 000457          BR       CHKSLV           ;GO CHECK SLAVE AVAILABILITY
2449
2450 007136 105737 001131  4$: TSTB   @NPRGFLG        ;BRANCH IF USER SELECTED ALL
2451 007142 001021          BNE      5$                ;DRIVES
2452 007144 110037 001004  MOVB    R0,DRVNUM          ;GET DRIVE #
2453 007150 116037 001140  MOVB    DIGTAB(R0),@#1.DRV ;PREPARE USER ACTION MESSAGE
2454 007156 000004 014604  TYPE,I.SLVS
2455 007162 004737 004000  JSR     PC,,INPUT          ;GET USER INPUT
2456 007166 012703 001272  MOV      #INBUF,R3          ;SET PTR TO USER INPUT
2457 007172 122713 000101  CMPB    #'A',(R3)          ;AN 'A' OR <CR> AS FIRST CHAR
2458 007176 001403          BEQ     5$                ;INDICATES TEST ALL SLAVES
2459 007200 122713 000015  CMPB    #CR,(R3)
2460 007204 001015          BNE      7$
2461 007206 110637 001131  5$: MOVB    SP,@NPRGFLG      ;SET 'ALL' INDICATOR
2462 007212 012701 001172  MOV      #SLVTBL,R1          ;MARK ALL SLAVES FOR ALL
2463 007216 012700 000040  MOV      #32.,R0            ;DRIVES AS TO BE TESTED
2464 007222 012721 177777  6$: MOV      #-1,(R1)+
2465 007226 005300          DEC      R0
2466 007230 001374          BNE      6$
2467 007232 105737 001131  TSTB    @NPRGFLG          ;BRANCH IF ALL WAS SELECTED
2468 007236 001016          BNE      CHKSLV
2469
2470 007240 122713 000015  7$: CMPB    #CR,(R3)          ;GET USER SELECTED SLAVES FOR
2471 007244 001725          BEQ     3$                ;DRIVE
2472 007246 121327 000054  CMPB    (R3),#'          ;STEP PTR PAST 'COMMA'
2473 007252 001001          BNE      8$
2474 007254 105723          TSTB   (R3)+
2475 007256 112304          8$: MOVB    (R3)+,R4          ;AND MARK SELECED SLAVF
2476 007260 042704 177770  BIC     #177770,R4          ;AS TO BE TESTED
2477 007264 060104          ADD     R1,R4
2478 007266 112714 177777  MOVB    #-1,(R4)
2479 007272 000762          BR      7$
2480
2481          ;ASCERTAIN THAT SLAVES (TE16/TU77'S) SELECTED ARE AVAILABLE
2482 007274 005000  CHKSLV: CLR      R0          ;R0 WILL CONTAIN THE DRIVE #
2483 007276 005001          CLR     R1                ;AND R1 THE SLAVE #
2484 007300 012702 001172  MOV      #SLVTBL,R2          ;SET PTR TO SLAVE TABLE
2485 007304 105760 001162  1$: TSTB   DRVTBL(R0)        ;BRANCH IF DRIVE SELECTED
2486 007310 001020          BNE      3$                ;B AVAILABLE FOR TEST
2487 007312 005200          2$: INC     R0                ;INCREMENT DRIVE #
```

```

2488 007314 105760 001161      TSTB   <DRVTBL-1>(R0)      ;++C WAS PREVIOUS DRIVE SELECTED
2489 007320 001003      BNE    9$                  ;++C BRANCH IF AVAIL.
2490 007322 062702 000010      ADD    #8.,R2              ;++C ADJUST SLAVE POINTER
2491 007326 000405      BR     10$
2492 007330 105737 001131      9$:   TSTB   @#PRGFLG        ;++C WAS ALL SELECTED
2493 007334 001002      BNE    10$
2494 007336 062702 000010      ADD    #8.,R2              ;++C ADJUST SLAVE POINTER
2495 007342 022700 000010      10$:  CMP    #8.,R0            ;SLAVES. BRANCH TO 1$ IF NOT ALL
2496 007346 001356      BNE    1$                  ;DRIVES CHECKED OTHERWISE EXIT
2497 007350 000437      BR     8$
2498
2499 007352 005001      3$:   CLR    R1                ;SET SLAVE # 0
2500 007354 105712      4$:   TSTB   (R2)              ;BRANCH IF DRIVE'S SLAVE IS SEL-
2501 007356 001006      BNE    6$                  ;ECTED FOR TEST
2502 007360 005201      5$:   INC    R1                ;INCRMENT SLAVE #
2503 007362 005202      INC    R2                ;STEP PTR TO NEXT SLAVE
2504 007364 022701 000010      CMP    #8.,R1            ;GO TO 4$ IF ALL SLAVES NOT
2505 007370 001371      BNE    4$                  ;CHECKED
2506 007372 000747      BR     2$                  ;OTHERWISE GO TO 2$ ABOVE
2507
2508 007374 110037 001004      6$:   MOVB   R0,@#DRVNUM      ;PASS DRIVE & SLAVE #
2509 007400 110137 001005      MOVB   R1,@#SLVNUM
2510 007404 004737 005564      JSR    PC,@#SLVAVA
2511 007410 102363      BVC    5$                  ;AND CHECK IF AVAILABLE
2512 007412 105737 001131      TSTB   @#PRGFLG          ;'V' SET INDICATES ERROR
2513 007416 001012      BNE    7$                  ;DO NOT TYPE ERROR MSG IF ALL
2514 007420 116037 001140 015051      MOVB   DIGTAB(R0),@#E.DRV ;SLAVES SELECTED
2515 007426 116137 001140 015061      MOVB   DIGTAB(R1),@#E.NAVA ;ICATES ERROR. PREPARE ERROR
2516 007434 000004 015043      TYPE ,E.NSLV             ;MESSAGE
2517 007440 110637 001127      MOVB   SP,@#ERRFLG
2518 007444 105012      7$:   CLRB   (R2)              ;SET ERROR INDICATOR
2519 007446 000744      BR     5$                  ;CLEAR SLAVE TABLE ENTRY
2520
2521 007450 105737 001127      8$:   TSTB   @#ERRFLG        ;GET NEXT SLAVE
2522 007454 001203      BNE    SLAVES
2523 007456 012737 004250 000004 100$:  MOV    #ERRTRP,@#ERRVEC ;BRANCH IF ERROR
2524
2525
2526
2527
2528 007464 012706 000600      ;SCAN DRIVE AND SLAVE TABLE FOR DRIVE/SLAVE COMBINATION TO TEST.
2529 007470 105037 001004      ;RESTART ADDRESS--PROGRAM STARTS HERE WHEN START ADDRESS = 210 AND
2530 007474 105037 001005      ;AFTER ALL SELECTED DRIVE/SLAVE COMBINATIONS HAVE BEEN TESTED.
2531 007500 012737 001172 001006      RSTRT: MOV    #600,SP        ;SET STACK PTR
2532 007506 105037 001132      CLRB   @#DRVNUM          ;SET DRIVE AND SLAVE # 0
2533
2534
2535 007512 113700 001004      CLRB   @#SLVNUM
2536 007516 113701 001005      MOV    #SLVTBL,@#SLVPTR ;SET PTR TO SLAVE TABLE
2537 007522 013702 001006      CLRB   @#UNTFND         ;CLEAR 'UNIT FOUND' IND
2538 007526 122737 000006 000041      ;PROGRAM RESTARTS HERE AFTER A DRIVE/SLAVE HAS BEEN TESTED.
2539 007534 001001      BEGIN: MOVB   @#DRVNUM,R0 ;GET DRIVE #
2540 007536 105012      MOVB   @#SLVNUM,R1      ;AND SLAVE #
2541
2542 007540 105760 001162      MOV    @#SLVPTR,R2      ;GET SLAVE PTR
2543 007544 001011      CMPB   #6,@#41         ;BRANCH IF LOADED VIA TMDP
2544
2545
2546
2547
2548
2549
2550
2551
2552
2553
2554
2555
2556
2557
2558
2559
2560
2561
2562
2563
2564
2565
2566
2567
2568
2569
2570
2571
2572
2573
2574
2575
2576
2577
2578
2579
2580
2581
2582
2583
2584
2585
2586
2587
2588
2589
2590
2591
2592
2593
2594
2595
2596
2597
2598
2599
2600
2601
2602
2603
2604
2605
2606
2607
2608
2609
2610
2611
2612
2613
2614
2615
2616
2617
2618
2619
2620
2621
2622
2623
2624
2625
2626
2627
2628
2629
2630
2631
2632
2633
2634
2635
2636
2637
2638
2639
2640
2641
2642
2643
2644
2645
2646
2647
2648
2649
2650
2651
2652
2653
2654
2655
2656
2657
2658
2659
2660
2661
2662
2663
2664
2665
2666
2667
2668
2669
2670
2671
2672
2673
2674
2675
2676
2677
2678
2679
2680
2681
2682
2683
2684
2685
2686
2687
2688
2689
2690
2691
2692
2693
2694
2695
2696
2697
2698
2699
2700
2701
2702
2703
2704
2705
2706
2707
2708
2709
2710
2711
2712
2713
2714
2715
2716
2717
2718
2719
2720
2721
2722
2723
2724
2725
2726
2727
2728
2729
2730
2731
2732
2733
2734
2735
2736
2737
2738
2739
2740
2741
2742
2743
2744
2745
2746
2747
2748
2749
2750
2751
2752
2753
2754
2755
2756
2757
2758
2759
2760
2761
2762
2763
2764
2765
2766
2767
2768
2769
2770
2771
2772
2773
2774
2775
2776
2777
2778
2779
2780
2781
2782
2783
2784
2785
2786
2787
2788
2789
2790
2791
2792
2793
2794
2795
2796
2797
2798
2799
2800
2801
2802
2803
2804
2805
2806
2807
2808
2809
2810
2811
2812
2813
2814
2815
2816
2817
2818
2819
2820
2821
2822
2823
2824
2825
2826
2827
2828
2829
2830
2831
2832
2833
2834
2835
2836
2837
2838
2839
2840
2841
2842
2843
2844
2845
2846
2847
2848
2849
2850
2851
2852
2853
2854
2855
2856
2857
2858
2859
2860
2861
2862
2863
2864
2865
2866
2867
2868
2869
2870
2871
2872
2873
2874
2875
2876
2877
2878
2879
2880
2881
2882
2883
2884
2885
2886
2887
2888
2889
2890
2891
2892
2893
2894
2895
2896
2897
2898
2899
2900
2901
2902
2903
2904
2905
2906
2907
2908
2909
2910
2911
2912
2913
2914
2915
2916
2917
2918
2919
2920
2921
2922
2923
2924
2925
2926
2927
2928
2929
2930
2931
2932
2933
2934
2935
2936
2937
2938
2939
2940
2941
2942
2943
2944
2945
2946
2947
2948
2949
2950
2951
2952
2953
2954
2955
2956
2957
2958
2959
2960
2961
2962
2963
2964
2965
2966
2967
2968
2969
2970
2971
2972
2973
2974
2975
2976
2977
2978
2979
2980
2981
2982
2983
2984
2985
2986
2987
2988
2989
2990
2991
2992
2993
2994
2995
2996
2997
2998
2999
3000

```

```
2544 007546 005001          CLR      R1          ;CLEAR SLAVE #
2545 007550 062702 000010    ADD      #8.,R2      ;AND STEP PTR TO NEXT DRIVE'S
2546 007554 005200          INC      R0          ;SLAVES AND INCREMENT DRIVE #
2547 007556 022700 000010    CMP      #8.,R0      ;EXIT TEST IF ALL DRIVES
2548 007562 001366          BNE     1$          ;CHECKED OTHERWISE CONTINUE
2549 007564 000137 013502    JMP      @WEND       ;SCAN FOR NEXT 'UNIT'
2550
2551 007570 105712          3$:     TSTB     (R2)   ;BRANCH IF SLAVE ON DRIVE IS
2552 007572 001007          BNE     4$          ;AVAILABLE THERWISE STEP
2553 007574 005202          INC      R2          ;PTR TO NEXT SLAVE
2554 007576 005201          INC      R1          ;INCREMENT SLAVE #
2555 007600 122701 000010    CMPB   #8.,R1      ;UNTIL ALL SLAVES CHECKED
2556 007604 001371          BNE     3$          ;WHEN ALL SLAVES CHECKED
2557 007606 005001          CLR      R1          ;SET SLAVE # 0
2558 007610 000761          BR      2$          ;AND CONTINUE SCAN
2559
2560 007612 110637 001132    4$:     MOVB     SP,@UNTFND ;INDICATE THAT A 'UNIT' IS FOUND
2561 007616 110037 001004    MOVB     R0,@DRVNUM   ;SET DRIVE #
2562 007622 110137 001005    MOVB     R1,@SLVNUM   ;SET SLAVE #
2563 007626 010237 001006    MOV      R2,@SLVPTR   ;SAVE SLAVE PTR
2564
2565 007632 105737 001135    5$:     TSTB     @WASFLG   ;
2566 007636 001034          BNE     7$          ;
2567 007640 112737 000001 001135    MOVB     #1,ASFLG     ;
2568 007646 005737 006442    TST      CHNFLG      ;BRANCH IF IN CHAIN MODE
2569 007652 001026          BNE     7$          ;
2570 007654 105037 001130    CLRB    @SKEWFLG     ;++B CLEAR SKEW (SPEED) TESTS SELECTED FLAG
2571 007660 000004 014670    TYPE,I.SPD          ;ASK USER IF HE WANTS TO RUN SPEED TESTS
2572 007664 004737 004000    JSR     PC,,INPUT    ;GET USER INPUT
2573 007670 012703 001272    MOV      #INBUF,R3   ;GET REPLY
2574 007674 122713 000015    CMPB    #CR,(R3)     ;DO NOT DO SKEW TESTS IF <CR> IS FIRST
2575 007700 001405          BEQ     6$          ;
2576 007702 132713 000001    BITB    #1,(R3)     ;BRANCH IF 'N'
2577 007706 001402          BEQ     6$          ;
2578 007710 111337 001130    MOVB    (R3),@SKEWFLG ;SET INDICATOR
2579 007714 022737 000176 001000 6$:     CMP      #SWREG,SWR   ;BRANCH IF SOFTWARE SWR
2580 007722 001002          BNE     7$          ;NOT INVOKED
2581 007724 004737 00252'    JSR     PC,GTSWR     ;GET SWITCH REGISTER
2582 007730
2583
2584
2585 007730 013705 001010    7$:     ;ROUTINE TO ASCERTAIN SLAVE TYPE AND LOAD APPROPRIATE SPECIFICATION
2586 007734 004737 005614    ;TABLES (TE16 OR TU77) INTO STIMTBL AND GTIMTBL.
2587 007740 012704 000240    MCV     @TMBASE,R5   ;GET BASE ADDRESS OF REGISTERS
2588 007744 012703 002124    JSR     PC,RHINIT    ;INIT DRIVE/SLAVE
2589 007750 012702 001424    MOV     #ENDTBL-STTBL,R4 ;GET TABLE LENGTH
2590 007754 105037 001136    MOV     #STIMTBL,R3  ;AND STARTING ADDRESS OF TABLE
2591 007760 032765 000004 000026    MOV     #TE16TTBL,R2 ;GET ADDRESS OF TE16 TIME TABLE
2592 007766 001012          CLRB    @TE16       ;SET FLAG = TE16
2593 007770 012737 000070 001012    BIT     #4,DT(R5)    ;BRANCH IF TU77
2594 007776 012737 000022 001014    BNE     9$          ;
2595 010004 112223          MOV     #56.,OSCTIM  ;SET US/TICK = 56
2596 010006 005304          MOV     #18.,GAPDEL  ;SET 18 TICKS/MS
2597 010010 001375          MOVB    (R2)+,(R3)+  ;MOVE TE16 TIME AND GAP TABLES
2598 010012 000416          DEC     R4          ;INTO STIMTBL & GTIMTBL
2599          BNE     8$          ;
          BR      11$     ;EXIT ROUTINE
```

```

2600 010014 012702 001664          9$:  MOV    #TU77TTBL,R2    :GET ADDRESS OF TU77 TIME TABLE
2601 010020 112737 000001 001136  MOVB   #1,@#TE16        :SET FLAG = TU77
2602 010026 012737 000120 001012  MOV    #80,,OSCTIM     :SET US/TICK = 80
2603 010034 012737 000003 001014  MOV    #3,GAPDEL       :SET 3 TICKS PER .25MS
2604 010042 112223          10$: MOVB   (R2)+,(R3)+     :MOVE TU77 TIME AND GAP TABLES
2605 010044 005304          DEC    R4              :INTO STIMTBL & GTIMTBL
2606 010046 001375          BNE   10$
2607 010050          11$:
2608
2609          :NOTE THIS IS NOT A TEST
2610          :INITIALIZE PROGRAM FLAGS
2611 010050 105037 001126          TST00: CLRB   @#TSTNUM    :SET TEST # 0
2612 010054 013705 001010          MOV    @#TMBASE,R5     :SET ADDRESS OF FIRST TMO3 REG
2613 010060 010500          MOV    R5,R0           :RO CONTAINS ADDRESS OF FC REG
2614 010062 062700 000006          ADD    #FC,R0
2615 010066 010501          MOV    R5,R1           :R1 CONTAINS ADDRESS OF DS REG
2616 010070 062701 000012          ADD    #DS,R1
2617 010074 012703 005070          MOV    #TIMER,R3      :SET JUMP ADDRESS TO TIMER
2618 010100 105037 001125          CLRB   @#ITCNT        :CLEAR SUBTEST ITERATION COUNT
2619 010104 052737 000100 177560  BIS    #100,@#TKS     :SET KEYBOARD IE BIT
2620
2621          :GET USER RUN PROCEDURE
2622          :IF SWR<05::00> IS NOT 0 THEN RUN TEST IN SWR<05::00>
2623          :OTHERWISE RUN ALL TESTS
2624
2625 010112 004737 005756          JSR    PC,,REWIND      :REWIND SLAVE
2626 (1) 010116 102504          BVS    99$            :BRANCH IF ERROR ON REWIND
2627 010120 105737 001130          TSTB   @#SKEWFLG     :++B BRANCH IF SWEW (SPEED) TEST SELECTED
2628 010124 001006          BNE   10$
2629 010126 004737 006040          JSR    PC,WRITE       :WRITE A RECORD
2630 010132 005215          INC    (R5)          :SET 'GO' BIT
2631 010134 004737 005652          JSR    PC,WAITRDY    :WAIT FOR READY
2632 010140 102473          BVS    99$
2633 010142 117702 170632          10$: MOVB   @SWR,R2        :GET SWITCHES
2634 010144 042702 177740          BIC    #177740,R2    :CLEAR ALL BUT TEST #
2635 010152 001421          BEQ    2$            :B BRANCH IF TEST 0 WAS SELECTED
2636 010154 000004 015207          TYPE,E.HDR          :TYPE TEST #
2637 010160 004737 003134          JSR    PC,TYPOCT
2638 010164 006302          ASL    R2            :FORM INDEX VALUE
2639 010166 016237 002364 010176  MOV    #NAMPTR(R2),1$ :GET ADDRESS OF TEST'S NAME
2640 010174 000004          TYPE                                :AND TYPE IT
2641 010176 000000          1$: .WORD 0
2642 010200 000004 001402          TYPE,CRLF
2643 010204 016237 002444 001002  MOV    TSTTBL(R2),@#SCPADR :SET SCOPE ADDRESS FOR TEST
2644 010212 000172 002444          JMP    @TSTTBL(R2)    :GO TO TEST
2645 010216 032777 002000 170554  2$: BIT    #SW10,@SWR    :BRANCH IF TIMES NOT TO BE TYPED
2646 010224 001034          BNE   5$
2647 010226 000004 016123          TYPE,L.HDR1
2648 010232 113702 001004          MOVB   DRVNUM,R2     :GET DRIVE #
2649 010236 113704 001005          MOVB   SLVNUM,R4     :AND SLAVE #
2650 010242 116237 001140 016305  MOVB   DIGTAB(R2),@#L.DRV :SET DRIVE AND SLAVE #'S
2651 010250 116437 001140 016317  MOVB   DIGTAB(R4),@#L.SLV :INTO L.HDR2 MESSAGE
2652 010256 000004 016240          TYPE,L.HDR2
2653 010262 105737 001136          TSTB   @#TE16        :BRANCH IF NOT TE16
2654 010266 001003          BNE   3$
2655 010270 000004 016323          TYPE,L.TE16         :TYPE 'TE16'

```



```
2668 .SBITL START OF TESTS
2669 :TEST 001 - WRITE FROM BOT
2670 :THIS TEST WILL MEASURE ACCELERATION DELAY REQUIRED TO
2671 :MOVE THE TAPE APPROXIMATELY SEVEN (7) INCHES FORWARD
2672 :FROM DEAD STOP BEFORE STARTING TO TRANSFER DATA.
2673
2674 :THIS TEST MEASURES TIME FROM 'GO'=1 TO 'ACCL'=0.
2675 010344 112737 000001 001126 TST001: MOVB #1,@TSTNUM ;SET TEST #
2676 010352 012702 010376 ;MOV #1$,R2 ;SET RETURN PC FROM TIMER
2677 010356 004737 005756 ;JSR PC,.REWIND ;REWIND SLAVE
(1) 010362 102420 ;BVS 99$ ;BRANCH IF ERROR ON REWIND
2678 010364 004737 006040 ;JSR PC,WRITE ;GO SETUP WRITE COMMAND
2679 010370 004737 005000 ;JSR PC,TIMON ;TURN TIMER ON
2680 010374 005215 ;INC (R5) ;SET 'GO' BIT
2681
2682 010376 005765 000032 1$: TST TC(R5) ;BRANCH WHEN 'ACCL'=0
2683 010402 100002 ;BPL 2$
2684 010404 000163 005070 ;JMP TIMER(R3) ;GO TO TIMER & RETURN VIA R2
2685
2686 010410 004737 005652 2$: JSR PC,WAITRDY ;WAIT FOR COMMAND TO FINISH
2687 010414 102403 ;BVS 99$ ;BRANCH IF ERROR
2688 010416 004737 005126 ;JSR PC,TIMOK ;GO CHECK TIME
2689 010422 000401 ;BR 100$
2690 010424 104400 99$: HLT
2691 010426 104000 100$: SCOPE
2692
2693 :TEST 002 - WRITE START
2694 :THIS TST MEASURES TIME FROM 'GO'=1 TO 'ACCL'=0.
2695 010430 112737 000002 001126 TST002: MOVB #2,@TSTNUM ;SET TEST # 2
2696 010436 004737 006040 ;JSR PC,WRITE ;INITIATE WRITE COMMAND
2697 010442 012702 010454 ;MOV #1$,R2 ;SET RETURN PC FROM TIMER
2698 010446 004737 005000 ;JSR PC,TIMON
2699 010452 005215 ;INC (R5) ;SET 'GO' BIT
2700
2701 010454 005765 000032 1$: TST TC(R5) ;BRANCH WHEN 'ACCL'=0
2702 010460 100002 ;BPL 2$
2703 010462 000163 005070 ;JMP TIMER(R3) ;GO TO TIMER & RETURN VIA R2
2704
2705 010466 004737 005652 2$: JSR PC,WAITRDY ;WAIT FOR READY
2706 010472 102403 ;BVS 99$ ;BRANCH IF ERROR
2707 010474 004737 005126 ;JSR PC,TIMOK ;GO CHECK TIME RECORDED
2708 010500 000401 ;BR 100$ ;EXIT VIA SCOPE
2709
2710 010502 104400 99$: HLT ;REPORT ERROR
2711 010504 104000 100$: SCOPE
2712
2713 :TEST 003- WRITE SHUTDOWN
2714 :THIS TEST MEASURES TIME FROM 'FC REG'=0 TO 'SWDN'=1.
2715 010506 112737 000003 001126 TST003: MOVB #3,@TSTNUM ;SET TEST#3
2716 010514 004737 006040 ;JSR PC,WRITE ;INITIATE WRITE COMMAND
2717 010520 005215 ;INC (R5) ;SET 'GO' BIT
2718
2719 010522 005710 1$: TST (R0) ;BRANCH WHEN WRITING FINISHED
2720 010524 001404 ;BEQ 2$
2721 010526 032711 040000 ;BIT #ERR,(R1) ;MONITOR ERROR BIT
2722 010532 001017 ;BNE 99$
```

```
2723 010534 000772          BR      1$
2724
2725 010536 000772          2$:
(1) 010536 004737 005000      JSR    PC,TIMON      ;TURN TIMER ON
2726 010542 010702          MOV    PC,R2        ;LOAD RETURN PC FROM TIMER
2727 010544 032711 000020      3$:  BIT    #SDWN,(R1) ;BRANCH WHEN DS <SDWN> SETS
2728 010550 001002          BNE    4$
2729 010552 000163 005070      JMP    TIMER(R3)    ;GO TO TIMER & RETURN VIA R2
2730
2731 010556 004737 005652      4$:  JSR    PC,WAITRDY ;WAIT FOR READY
2732 010562 102403          BVS    99$
2733 010564 004737 005126      JSR    PC,TIMOK     ;GO CHECK TIME RECORDED
2734 010570 000401          BR     100$
2735 010572 104400          99$:  HLT
2736 010574 104000          100$: SCOPE        ;REPORT ERROR
2737
2738          ;TEST 004 - WRITE SETTLEDOWN
2739          ;THIS TEST MEASURES TIME FROM 'SWDN'=1 TO 'SWDN'=0.
2740 010576 112737 000004 001126  TST004: MOVB   #4,#TSTNUM
2741 010604 004737 006040      JSR    PC,WRITE
2742 010610 005215          INC    (R5)         ;SET 'GO' BIT
2743
2744 010612 005710          1$:  TST    (R0)       ;BRANCH WHEN WRITING FINISHED
2745 010614 001404          BEQ    2$
2746 010616 032711 040000      BIT    #ERR,(R1)   ;CHECK ERROR BIT
2747 010622 001026          BNE    99$
2748 010624 000772          BR     1$
2749
2750 010626 032711 000020      2$:  BIT    #SDWN,(R1) ;WAIT FOR ASSERTION OF 'SDWN'
2751 010632 001004          BNE    3$
2752 010634 032711 040000      BIT    #ERR,(R1)   ;MONITOR ERROR BIT
2753 010640 001017          BNE    99$
2754 010642 000771          BR     2$
2755
2756 010644          3$:
(1) 010644 004737 005000      JSR    PC,TIMON     ;TURN TIMER ON
2757 010650 010702          MOV    PC,R2        ;SET RETURN PC FROM TIMER
2758 010652 032711 000020      BIT    #SDWN,(R1)  ;BRANCH WHEN SWDN CLEARS
2759 010656 001402          BEQ    5$
2760 010660 000163 005070      JMP    TIMER(R3)   ;GO TO TIMER & RETURN VIA R2
2761
2762 010664 004737 005652      5$:  JSR    PC,WAITRDY ;WAIT FOR READY
2763 010670 102403          BVS    99$
2764 010672 004737 005126      JSR    PC,TIMOK
2765 010676 000401          BR     100$
2766
2767 010700 104400          99$:  HLT
2768 010702 104000          100$: SCOPE
2769
2770          ;TEST 005 - READ FROM BOT
2771          ;THIS TEST MEASURES TIME FROM 'GO'=1 TO 'ACCL'=0.
2772 010704 112737 000005 001126  TST005: MOVB   #5,#TSTNUM
2773 010712 004737 005756      JSR    PC,.REWIND  ;SET TEST #5
(1) 010716 102422          BVS    99$         ;REWIND SLAVE
2774 010720 004737 006056      JSR    PC,READ     ;BRANCH IF ERROR ON REWIND
2775 010724 012702 010736      MOV    #1$,R2     ;SET RETURN PC FROM TIMER
```

```

2776 010730 004737 005000      JSR    PC,TIMON      ;TURN TIMER ON
2777 010734 005215              INC    (R5)          ;SET 'GO' BIT
2778                                ;
2779 010736 005765 000032      1$:   TST    TC(R5)      ;BRANCH WHEN 'ACCL' RESETS
2780 010742 100002              BPL    2$            ;
2781 010744 000163 005070      JMP    TIMER(R3)     ;GO TO TIMER & RETURN VIA R2
2782                                ;
2783 010750 004737 005652      2$:   JSR    PC,WAITRDY  ;WAIT FOR READY
2784 010754 102403              BVS    99$          ;BRANCH IF ERROR
2785 010756 004737 005126      JSR    PC,TIMOK      ;CHECK RECORDED TIME
2786 010762 000401              BR     100$         ;
2787                                ;
2788 010764 104400              99$:   HLT                    ;
2789 010766 104000              100$:  SCOPE          ;
2790                                ;
2791                                ;TEST 006 - READ START
2792                                ;THIS TEST MEASURES TIME FROM 'GO'=1 TO 'ACCL'=0.
2793 010770 112737 000006 001126  TST006: MOVB   #6,#TSTNUM  ;SET TEST #6
2794 010776 004737 006132      JSR    PC,WRT.BK     ;WRITE A RECORD & BACK SPACE
2795 011002 102422              BVS    99$          ;
2796 011004 004737 006056      JSR    PC,READ       ;
2797 011010 012702 011022      MOV    #1$,R2        ;SET RETURN PC FROM TIMER
2798 011014 004737 005000      JSR    PC,TIMON      ;TURN TIMER ON
2799 011020 005215              INC    (R5)          ;SET 'GO' BIT
2800                                ;
2801 011022 005765 000032      1$:   TST    TC(R5)      ;BRANCH WHEN 'ACCL' RESETS
2802 011026 100002              BPL    2$            ;
2803 011030 000163 005070      JMP    TIMER(R3)     ;GO TO TIMER & RETURN VIA R2
2804                                ;
2805 011034 004737 005652      2$:   JSR    PC,WAITRDY  ;
2806 011040 102403              BVS    99$          ;BRANCH IF ERROR
2807 011042 004737 005126      JSR    PC,TIMOK      ;CHECK RECORDED TIME
2808 011046 000401              BR     100$         ;
2809                                ;
2810 011050 104400              99$:   HLT                    ;
2811 011052 104000              100$:  SCOPE          ;
2812                                ;
2813                                ;TEST 007 - READ SHUTDOWN
2814                                ;THIS TEST MEASURES TIME FROM 'FC REG'=FRAME COUNT TO 'SWDN'=1.
2815 011054 112737 000007 001126  TST007: MOVB   #7,#TSTNUM  ;SET TEST #7
2816 011062 004737 006132      JSR    PC,WRT.BK     ;WRITE A RECORD & BACK SPACE
2817 011066 102430              BVS    99$          ;BRANCH IF ERROR
2818 011070 004737 006056      JSR    PC,READ       ;
2819 011074 005215              INC    (R5)          ;SET 'GO' BIT
2820                                ;
2821 011076 022710 000400      1$:   CMP    #-FRMCNT,(R0) ;WAIT FOR FRAME COUNT TO
2822 011102 001404              BEQ    2$            ;= # OF FRAMES WRITTEN
2823 011104 032711 040000      BIT    #ERR,(R1)     ;MONITOR ERROR BIT
2824 011110 001017              BNE    99$          ;
2825 011112 000771              BR     1$            ;
2826                                ;
2827 011114 004737 005000      2$:   JSR    PC,TIMON      ;TURN TIMER ON
2828 011120 010702              MOV    PC,R2         ;SET RETURN PC FROM TIMER
2829 011122 032711 000020      BIT    #SDWN,(R1)    ;BRANCH WHEN SDWN SETS
2830 011126 001002              BNE    3$            ;

```

```

2831 011130 000163 005070          JMP    TIMER(R3)          ;GO TO TIMER & RETURN VIA R2
2832
2833 011134 004737 005652          3$:   JSR    PC,WAITRDY
2834 011140 102403                    BVS    99$
2835 011142 004737 005126          JSR    PC,TIMOK
2836 011146 000401                    BR     100$
2837
2838 011150 104400                    99$:   HLT
2839 011152 104000                    100$:  SCOPE          ;REPORT ERROR
2840
2841
2842
2843 011154 112737 000010 001126 ;TEST 010 - READ SETTLEDOWN
2844 011162 012702 011240          ;THIS TEST MEASURES TIME FROM 'SWDN'=1 TO 'SWDN'=0.
2845 011166 004737 006132          TST010: MOVB   #10,#TSTNUM ;SET TEST #10
2846 011172 102436                    MOV    #4$,R2           ;SET RETURN PC FROM TIMER
2847 011174 004737 006056          JSR    PC,WRT.BK       ;WRITE A RECORD & BACK SPACE
2848 011200 005215                    BVS    99$
2849
2850 011202 105711                    JSR    PC,READ
2851 011204 100404                    INC    (R5)            ;SET 'GO' BIT
2852 011206 032711 040000          1$:   TSTB   (R1)       ;WAIT FOR READY
2853 011212 001026                    BMI    2$             ;BRANCH WHEN SET
2854 011214 000772                    BIT    #ERR,(R1)     ;CHECK ERROR BIT
2855
2856 011216 032711 000020          2$:   BNE    #SDWN,(R1) ;WAIT FOR ASSERTION OF 'SDWN'
2857 011222 001004                    BNE    3$
2858 011224 032711 040000          BIT    #ERR,(R1)     ;MONITOR ERROR BIT
2859 011230 001017                    BNE    99$
2860 011232 000771                    BR     2$
2861
2862 011234
2863 (1) 011234 004737 005000          3$:   JSR    PC,TIMON
2864 011240 032765 000020 000012 4$:   BIT    #SDWN,DS(R5) ;TURN TIMER ON
2865 011246 001402                    BEQ    5$             ;WAIT FOR NEGATION OF SDWN
2866 011250 000163 005070          JMP    TIMER(R3)     ;GO TO TIMER & RETURN VIA R2
2867 011254 004737 005652          5$:   JSR    PC,WAITRDY
2868 011260 102403                    BVS    99$
2869 011262 004737 005126          JSR    PC,TIMOK
2870 011266 000401                    BR     100$
2871
2872 011270 104400                    99$:   HLT
2873 011272 104000                    100$:  SCOPE
2874
2875
2876
2877
2878 011274 112737 000011 001126 ;TEST 011-READ REVERSE START
2879 011302 012702 011340          ;THIS TEST MEASURES TIME FROM 'GO'=1 TO 'ACCL'=0.
2880 011306 004737 006040          TST011: MOVB   #11,#TSTNUM ;SET RETURN PC FROM TIMER
2881 011312 005215                    MOV    #1$,R2         ;WRITE A RECORD
2882 011314 004737 005652          JSR    PC,WRITE
2883 011320 102422                    INC    (R5)           ;SET 'GO' BIT
2884 011322 004737 005360          JSR    PC,WAITRDY
2885 011326 004737 006074          JSR    PC,DELAY
2886
2887
2888
2889
2890
2891
2892
2893
2894
2895
2896
2897
2898
2899
2900
2901
2902
2903
2904
2905
2906
2907
2908
2909
2910
2911
2912
2913
2914
2915
2916
2917
2918
2919
2920
2921
2922
2923
2924
2925
2926
2927
2928
2929
2930
2931
2932
2933
2934
2935
2936
2937
2938
2939
2940
2941
2942
2943
2944
2945
2946
2947
2948
2949
2950
2951
2952
2953
2954
2955
2956
2957
2958
2959
2960
2961
2962
2963
2964
2965
2966
2967
2968
2969
2970
2971
2972
2973
2974
2975
2976
2977
2978
2979
2980
2981
2982
2983
2984
2985
2986
2987
2988
2989
2990
2991
2992
2993
2994
2995
2996
2997
2998
2999
3000
  
```

```

2886 011332 004737 005000      JSR    PC,TIMON      ;TURN TIMER ON
2887 011336 005215              INC    (R5)          ;SET 'GO' BIT
2888
2889 011340 005765 000032      1$:   TST    TC(R5)      ;BRANCH WHEN 'ACCL' = 0
2890 011344 100002              BPL    2$
2891 011346 000163 005070      JMP    TIMER(R3)     ;GO TO TIMER & RETURN VIA R2
2892
2893 011352 004737 005652      2$:   JSR    PC,WAITRDY   ;BRANCH IF ERROR
2894 011356 102403              BVS    99$
2895 011360 004737 005126      JSR    PC,TIMOK
2896 011364 000401              BR     100$
2897
2898 011366 104400      99$:   HLT
2899 011370 104000      100$:  SCOPE
2900
2901      ;TEST 012-READ REVERSE SHUTDOWN
2902      ;THIS TEST MEASURES TIME FROM 'FC REG' = FRAME COUNT TO 'SDWN'=1.
2903 011372 112737 000012 001126  TST012: MOVB   #12,#TSTNUM
2904 011400 012702 011450      MOV    #3$,R2      ;SET RETURN PC FROM TIMER
2905 011404 004737 006040      JSR    PC,WRITE     ;WRITE A RECORD
2906 011410 005215              INC    (R5)         ;SET 'GO' BIT
2907 011412 004737 005652      JSR    PC,WAITRDY
2908 011416 102427              BVS    99$
2909 011420 004737 006074      JSR    PC,REVRD
2910 011424 005215              INC    (R5)         ;SET 'GO' BIT
2911
2912 011426 022710 000400      1$:   CMP    #-FRMCNT,(R0) ;BRANCH WHEN FRAME COUNT
2913 011432 001404              BEQ    2$           ;= # OF RECORD WRITTEN
2914 011434 032711 040000      BIT    #ERR,(R1)   ;MONITOR ERROR BIT IN 'DS' REG
2915 011440 001016              BNE    99$
2916 011442 000771              BR     1$
2917
2918 011444              2$:
2919 (1) 011444 004737 005000      JSR    PC,TIMON     ;TURN TIMER ON
2920 011450 032711 000020      3$:   BIT    #SDWN,(R1) ;BRANCH WHEN SDWN SETS
2921 011454 001002              BNE    4$
2922 011456 000163 005070      JMP    TIMER(R3)   ;GO TO TIMER & RETURN VIA R2
2923
2924 011462 004737 005652      4$:   JSR    PC,WAITRDY   ;WAIT FOR READY
2925 011466 102403              BVS    99$
2926 011470 004737 005126      JSR    PC,TIMOK
2927 011474 000401              BR     100$
2928
2929 011476 104400      99$:   HLT
2930 011500 104000      100$:  SCOPE
2931
2932      ;TEST 013-READ REVERSE SETTLEDOWN
2933      ;THIS TEST MEASURES TIME FROM 'SDWN'=1 TO 'SDWN'=0.
2934 011502 112737 000013 001126  TST013: MOVB   #13,#TSTNUM
2935 011510 012702 011574      MOV    #4$,R2      ;SET RETURN PC FROM TIMER
2936 011514 004737 006040      JSR    PC,WRITE     ;WRITE A RECORD
2937 011520 005215              INC    (R5)         ;SET 'GO' BIT
2938 011522 004737 005652      JSR    PC,WAITRDY
2939 011526 102435              BVS    99$
2940 011530 004737 006074      JSR    PC,REVRD
2941 011534 005215              INC    (R5)         ;SET 'GO' BIT

```

```

2941
2942 011536 105711      1$:   TSTB   (R1)           ;BRANCH WHEN
2943 011540 100404      BMI    2$           ;READY SETS
2944 011542 032711 040000  BIT    #ERR,(R1)
2945 011546 001025      BNE    99$
2946 011550 000772      BR     1$
2947
2948 011552 032711 000020  2$:   BIT    #SDWN,(R1)
2949 011556 001004      BNE    3$
2950 011560 032711 040000  BIT    #ERR,(R1)
2951 011564 001016      BNE    99$
2952 011566 000771      BR     2$
2953
2954 011570      3$:
   (1) 011570 004737 005000  JSR    PC,TIMON      ;TURN TIMER ON
2955 011574 032711 000020  4$:   BIT    #SDWN,(R1)  ;BRANCH WHEN SWDN = 0
2956 011600 001402      BEQ    5$
2957 011602 000163 005070  JMP    TIMER(R3)     ;GO TO TIMER & RETURN VIA R2
2958
2959 011606 004737 005652  5$:   JSR    PC,WAITRDY   ;WAIT FOR READY
2960 011612 102403      BVS    99$
2961 011614 004737 005126  JSR    PC,TIMOK
2962 011620 000401      BR     100$
2963
2964 011622 104400      99$:   HLT
2965 011624 104000      100$:  SCOPE
2966
2967      ;REWIND DRIVE
2968 011626      A:
   (1) 011626 004737 005756  JSR    PC,.REWIND    ;REWIND SLAVE
   (1) 011632 102401      BVS    99$           ;BRANCH IF ERROR ON REWIND
2969 011634 102002      BVC    100$
2970 011636 104400      99$:   HLT
2971 011640 000772      BR     A
2972 011642      100$:
2973
2974      ;TEST 014-TURN AROUND DELAY (FORWARD-REVERSE)
2975      ;THIS TEST MEASURES TIME FROM 'GO'=1 (READ REVERSE) TO 'ACCL'=0
2976 011642 112737 000014 001126 TST014: MOVB   #14,#TSTNUM
2977 011650 012702 011702  MOV    #2$,R2       ;SET RETURN PC FROM TIMER
2978 011654 004737 006040  JSR    PC,WRITE     ;WRITE A RECORD
2979 011660 005215      INC    (R5)         ;SET 'GO' BIT
2980 011662 004737 005652  JSR    PC,WAITRDY
2981 011666 102420      BVS    99$
2982
2983 011670 004737 006074  1$:   JSR    PC,REVRD    ;READ THE RECORD (REVERSE)
2984 011674 004737 005000  JSR    PC,TIMON     ;TURN TIMER ON
2985 011700 005215      INC    (R5)         ;SET 'GO' BIT
2986
2987 011702 005765 000032  2$:   TST    TC(R5)     ;WAIT FOR 'ACCL' = 0
2988 011706 100002      BPL    3$
2989 011710 000163 005070  JMP    TIMER(R3)    ;GO TO TIMER & RETURN VIA R2
2990
2991 011714 004737 005652  3$:   JSR    PC,WAITRDY
2992 011720 102403      BVS    99$
2993 011722 004737 005126  JSR    PC,TIMOK

```

```
2994 011726 000401 BR 100$
2995
2996 011730 104400 99$: HLT
2997 011732 104000 100$: SCOPE
2998
2999 ;TEST 015- TURN AROUND DELAY (REVERSE-FORWARD)
3000 ;THIS TEST MEASURES TIME FROM 'GO'=1 (READ) TO 'ACCL'=0.
3001 011734 112737 000015 001126 TST015: MOVB #15,2@TSTNUM
3002 011742 012702 012010 MOV #2$,R2 ;SET RETURN PC FROM TIMER
3003 011746 004737 006040 JSR PC,WRITE ;WRITE A RECORD
3004 011752 005215 INC (R5) ;SET 'GO' BIT
3005 011754 004737 005652 JSR PC,WAITRDY ;WAIT FOR READY
3006 011760 102426 BVS 99$
3007 011762 004737 006074 JSR PC,REVRD ;READ A RECORD IN THE
3008 011766 005215 INC (R5) ;SET 'GO' BIT
3009
3010 011770 004737 005652 JSR PC,WAITRDY
3011 011774 102420 BVS 99$
3012
3013 011776 004737 006056 1$: JSR PC,READ ;READ RECORD FORWARD
3014 012002 004737 005000 JSR PC,TIMON ;TURN TIMER ON
3015 012006 005215 INC (R5) ;SET 'GO' BIT
3016
3017 012010 005765 000032 2$: TST TC(R5) ;WAIT FOR 'ACCL' = 0
3018 012014 100002 BPL 3$
3019 012016 000163 005070 JMP TIMER(R3) ;GO TO TIMER & RETURN VIA R2
3020
3021 012022 004737 005652 3$: JSR PC,WAITRDY
3022 012026 102403 BVS 99$
3023 012030 004737 005126 JSR PC,TIMOK
3024 012034 000401 BR 100$
3025
3026 012036 104400 99$: HLT
3027 012040 104000 100$: SCOPE
3028
3029 ;TEST 016-GAP SIZE (STOP HALF)
3030 012042 112737 000016 001126 TST016: MOVB #16,2@TSTNUM
3031 012050 012702 012106 MOV #1$,R2 ;SET RETURN PC FROM TIMER
3032 012054 004737 006040 JSR PC,WRITE ;WRITE A RECORD
3033 012060 005215 INC (R5) ;SET 'GO' BIT
3034 012062 004737 005652 JSR PC,WAITRDY
3035 012066 102421 BVS 99$
3036 012070 004737 005360 JSR PC,DELAY ;DELAY 350 MS
3037 012074 004737 006074 JSR PC,REVRD ;READ REVERSE RECORD
3038 012100 004737 005000 JSR PC,TIMON ;TURN TIMER ON
3039 012104 005215 INC (R5) ;SET 'GO' BIT
3040
3041 012106 005710 1$: TST (R0) ;WAIT FOR FRAME COUNT > 0
3042 012110 001002 BNE 2$
3043 012112 000163 005070 JMP TIMER(R3) ;GO TO TIMER & RETURN VIA R2
3044
3045 012116 004737 005652 2$: JSR PC,WAITRDY ;WAIT FOR READY BIT TO SET
3046 012122 102403 BIS 99$
3047 012124 004737 005126 JSR PC,TIMOK ;CHECK TIME
3048 012130 000401 BR 100$
3049
```

```

3050 012132 104400          99$:   HLT
3051 012134 104000          100$:  SCOPE
3052
3053
3054 012136 112737 000017 001126 ;TEST 017-GAP SIZE (START HALF)
3055 012144 012702 012216 ;TST017: MOVB #17,20TSTNUM
3056 012150 004737 006040      MOV #1$,R2      ;SET RETURN PC FROM TIMER
3057 012154 005215          JSR PC,WRITE   ;WRITE A RECORD
3058 012156 004737 005652      INC (R5)       ;SET 'GO' BIT
3059 012162 102427          JSR PC,WAITRDY ;WAIT FOR READY
3060 012164 004737 006074      BVS 99$
3061 012170 005215          JSR PC,REVRD   ;READ REVERSE THE RECORD
3062 012172 004737 005652      INC (R5)       ;SET 'GO' BIT
3063 012176 102421          JSR PC,WAITRDY ;WAIT FOR READY
3064 012200 004737 005360      BVS 99$       ;BRANCH ON ERROR
3065 012204 004737 006056      JSR PC,DELAY   ;WAIT FOR TAPE MOTION TO STOP
3066 012210 004737 005000      JSR PC,READ    ;READ RECORD
3067 012214 005215          JSR PC,TIMON   ;TURN TIMER ON
3068
3069 012216 005710          1$:   TST      (R0)      ;WAIT FOR FRAME COUNT > 0
3070 012220 001002          BNE      2$
3071 012222 000163 005070      JMP      TIMER(R3) ;GO TO TIMER & RETURN VIA R2
3072
3073 012226 004737 005652          2$:   JSR      PC,WAITRDY ;WAIT FOR READY
3074 012232 102403          BVS      99$
3075 012234 004737 005126      JSR      PC,TIMOK ;CHECK TIME
3076 012240 000401          BR
3077
3078 012242 104400          99$:   HLT
3079 012244 104000          100$:  SCOPE
3080
3081
3082 ;TEST 020- GAP SIZE (INTERRECORD)
3083 012246 112737 000020 001126 ;THIS TEST MEASURES TIME FROM 'GO'=1 TO 'FC REG' >0.
3084 012254 012702 012336 ;TST020: MOVB #20,20TSTNUM
3085 012260 004737 006040      MOV #1$,R2      ;SET RETURN PC FROM TIMER
3086 012264 005215          JSR PC,WRITE   ;WRITE A RECORD
3087 012266 004737 005652      INC (R5)       ;SET 'GO' BIT
3088 012272 102433          JSR PC,WAITRDY ;WAIT FOR READY
3089 012274 004737 006040      BVS 99$
3090 012300 005215          JSR PC,WRITE   ;WRITE SECOND RECORD
3091 012302 004737 005652      INC (R5)       ;SET 'GO' BIT
3092 012306 102425          JSR PC,WAITRDY ;WAIT FOR READY
3093 012310 004737 006074      BVS 99$
3094 012314 005215          JSR PC,REVRD   ;READ REVERSE SECOND RECORD
3095 012316 004737 005652      INC (R5)       ;SET 'GO' BIT
3096 012322 102417          JSR PC,WAITRDY ;WAIT FOR READY
3097 012324 004737 006074      BVS 99$
3098 012330 004737 005000      JSR PC,REVRD   ;READ REVERSE FIRST RECORD
3099 012334 005215          JSR PC,TIMON   ;TURN TIMER ON
3100
3101 012336 005710          1$:   TST      (R0)      ;WAIT FOR FRAME COUNT > 0
3102 012340 001002          BNE      2$
3103 012342 000163 005070      JMP      TIMER(R3) ;GO TO TIMER & RETURN VIA R2
3104
3105 012346 004737 005652          2$:   JSR      PC,WAITRDY ;WAIT FOR READY

```

3106 012352 102403  
 3107 012354 004737 005126  
 3108 012360 000401  
 3109  
 3110 012362 104400  
 3111 012364 104000  
 3112  
 3113  
 3114  
 3115  
 3116  
 3117  
 3118  
 3119  
 3120  
 3121  
 3122  
 3123  
 3124

BVS 99\$  
 JSR PC,TIMOK  
 BR 100\$

99\$: HLT  
 100\$: SCOPE

:TEST 021- GAP CONSISTANCY  
 :THIS TEST MEASURES TIME FROM 'GO'=1 TO 'FC REG' > 0.  
 :THE TEST REWINDS THE TAPE,WRITES 17 RECORDS WITH A DELAY FROM 1-16 MS  
 :BETWEEN EACH WRITE COMMAND. AFTER THE 17. RECORDS ARE WRITTEN THE  
 :PROGRAM READ REVERSES 16 RECORDS. AT THIS POINT THE TAPE IS STOPPED BE-  
 :TWEEN THE FIRST AND SECOND RECORD. A READ COMMAND IS EXECUTED TO READ  
 :THE 16 RECORDS WITH THE TIME BETEWEN GO=1 TO FC > 0 STORED IN 'GAPTBL'  
 :FOR EACH RECORD READ. AFTER 16 RECORDS HAVE BEEN READ THE TIME IS VER-  
 :IFIED FOR EACH READ. AFTER ALL RECORD TIMES ARE VERIFIED THEY ARE AVER-  
 :AGED AND PLACED IN THE 'ATIMTBL' (BY SCOPE). THE ABOVE PROCESS IS RE-  
 :PEATED FOR EACH ITERATION.

3125 012366 112737 000021 001126  
 3126 012374 012702 012532  
 3127 012400 004737 005756  
 (1) 012404 102530  
 3128 012406 005037 001120  
 3129 012412 012700 000021  
 3130 012416 004737 006040  
 3131 012422 005215  
 3132 012424 004737 005652  
 3133 012430 102516  
 3134 012432 004737 005410  
 3135 012436 063737 001014 001120  
 3136 012444 005300  
 3137 012446 001363  
 3138  
 3139 012450 012700 000021  
 3140 012454 004737 006074  
 3141 012460 005215  
 3142 012462 004737 005652  
 3143 012466 102477  
 3144 012470 005300  
 3145 012472 001370  
 3146  
 3147 012474 012700 000020  
 3148 012500 012701 001060  
 3149 012504 004737 006056  
 3150 012510 005215  
 3151  
 3152 012512 004737 005652  
 3153 012516 102463  
 3154 012520 004737 006056  
 3155 012524 004737 005000  
 3156 012530 005215  
 3157  
 3158 012532 005765 000006  
 3159 012536 001002  
 3160 012540 000163 005070

TST021: MOV #21, @TSTNUM  
 MOV #4\$,R2 ;SET RETURN PC FROM TIMER  
 JSR PC,REWIND ;REWIND SLAVE  
 BVS 99\$ ;BRANCH IF ERROR ON REWIND  
 CLR DELTIM ;CLEAR VARIABLE DELAY TIME  
 MOV #17,RO ;SET # OF RECORDS TO WRITE  
 1\$: JSR PC,WRITE ;WRITE 17. RECORDS  
 INC (R5) ;SET 'GO' BIT  
 JSR PC,WAITRDY ;WAIT FOR READY  
 BVS 99\$  
 JSR PC,DELAYV ;DELAY BEFORE WRITING NEXT REC.  
 ADD GAPDEL,DELTIM ;ADD 1MS TO DELAY TIME  
 DEC RO ;DECREMENT RECORDS WRITTEN COUNT  
 BNE 1\$  
 2\$: MOV #17,RO ;SET # OF RECS. TO REVERSE READ  
 JSR PC,REVRD ;REVERSE READ 17. RECORDS  
 INC (R5) ;SET 'GO' BIT  
 JSR PC,WAITRDY ;WAIT FOR READY  
 BVS 99\$  
 DEC RO ;DECREMENT RECORD COUNT  
 BNE 2\$  
 3\$: MOV #16,RO ;SET # OF RECORDS TO READ  
 MOV #GAPTBL,R1 ;SET PTR TO GAP TABLE FOR TEST  
 JSR PC,READ ;READ A RECORD  
 INC (R5) ;SET 'GO' BIT  
 JSR PC,WAITRDY ;WAIT FOR READY  
 BVS 99\$  
 JSR PC,READ ;READ NEXT RECORD  
 JSR PC,TIMON ;TURN TIMER ON  
 INC (R5) ;SET 'GO' BIT  
 4\$: TST FC(R5) ;WAIT FOR FRAME COJNT > 0  
 BNE 5\$  
 JMP TIMER(R3) ;GO TO TIMER & RETURN VIA R2

```

3161
3162 012544 004737 005652      5$:   JSR   PC, WAITRDY      ;WAIT FOR READY
3163 012550 102446                BVS   99$
3164 012552 010421                MOV   R4, (R1)+          ;STORE TIME IN GAPTBL
3165 012554 005300                DEC   R0                 ;DECREMENT # OF RECORDS READ
3166 012556 001355                BNE   3$
3167
3168 012560 105037 001124        CLRB  @#GAP              ;SET GAP # 0
3169 012564 012700 000020        MOV   #16., R0
3170 012570 012701 001060        MOV   #GAPTBL, R1
3171
3172 012574 012104                6$:   MOV   (R1)+, R4       ;GET GAP TICK COUNT
3173 012576 004737 005236        JSR   PC, GAPOK         ;CHECK TIME
3174 012602 105237 001124        INCB  @#GAP             ;INCREMENT GAP #
3175 012606 122737 000020 001124  CMPB  #16., @#GAP       ;BRANCH IF ALL GAPS NOT CHECKED
3176 012614 001367                BNE   6$
3177
3178 012616 012700 000020        MOV   #16., R0          ;SETUP TO AVERAGE GAP SIZES
3179 012622 012701 001060        MOV   #GAPTBL, R1      ;SET PTR TO TABLE
3180 012626 005002                CLR   R2                ;CLEAR 'SUM' REGISTERS
3181 012630 005003                CLR   R3
3182 012632 062102                7$:   ADD   (R1)+, R2       ;ADD ALL GAP SIZES TOGETHER
3183 012634 005503                ADC   R3
3184 012636 005300                DEC   R0
3185 012640 001374                BNE   7$
3186 012642 012700 000004        MOV   #4., R0           ;NOW DIVIDE BY 16.
3187 012646 006203                8$:   ASR   R3                ;BY SHIFTING 4 PLACES RIGHT
3188 012650 006002                ROR   R2
3189 012652 005300                DEC   R0
3190 012654 001374                BNE   8$
3191 012656 010204                MOV   R2, R4            ;MOVE AVERAGED TIMES TO R4
3192 012660 004737 005126        JSR   PC, TIMOK         ;CHECK AVERAGED TIMES
3193 012664 000401                BR    100$
3194
3195 012666 104400                99$:  HLT
3196 012670 104000                100$: SCOPE
3197
3198
3199
3200 ;TEST 022-DATA TIME (800BPI)
3201 ;THIS TEST MEASURES THE TIME FROM FC REG >-6400 TO 'RDY' = 1.
3202 012672 112737 000022 001126  TST022: MOVB  #022, @#TSTNUM
3203 012700 012702 012760                MOV   #3$, R2           ;SET RETURN PC FROM TIMER
3204 012704 004737 005756                JSR   PC, .REWIND       ;REWIND SLAVE
(1) 012710 102442                BVS   99$               ;BRANCH IF ERROR ON REWIND
3205 012712 052765 001700 000032  BIS   #NORM11, TC(R5)   ;SET 800 BPI
3206 012720 004337 006174                JSR   R3, TMCMD         ;WRITE 3200. WORD RECORD
3207 012724 017450                .WORD WTBUF
3208 012726 171600                .WORD -3200.
3209 012730 163400                .WORD -6400.
3210 012732 000060                .WORD WFD
3211 012734 005215                INC   (R5)              ;SET 'GO' BIT
3212
3213 012736 022710 163400                1$:  CMP   #-6400., (R0)     ;WAIT FOR WRITING TO START
3214 012742 001004                BNE   2$
3215 012744 032711 040000                BIT   #ERR, (R1)       ;MONITOR ERROR BIT

```

3216	012750	001022			BNE	99\$		
3217	012752	000771			BR	1\$		
3218								
3219	012754			2\$:				
(1)	012754	004737	005000		JSR	PC,TIMON		:TURN TIMER ON
3220	012760	105711		3\$:	TSTB	(R1)		:BRANCH WHEN READY SETS
3221	012762	100402			BMI	4\$		
3222	012764	000163	005070		JMP	TIMER(R3)		:GO TO TIMER & RETURN VIA R2
3223								
3224	012770	012700	000003	4\$:	MOV	#3,R0		:SET SHIFT COUNT
3225	012774	006204		5\$:	ASR	R4		
3226	012776	005300			DEC	R0		
3227	013000	001375			BNE	5\$		
3228	013002	004737	005652		JSR	PC,WAITRDY		
3229	013006	102403			BVS	99\$		
3230	013010	004737	005126		JSR	PC,TIMOK		:CHECK TIME
3231	013014	000401			BR	100\$		
3232								
3233	013016	104400		99\$:	HLT			
3234	013020	104000		100\$:	SCOPE			
3235								
3236								
3237								
3238	013022	112737	000023	001126	TST023:	MOV	#023,#TSTNUM	
3239	013030	012702	013116		MOV	#3\$,R2		:SET RETURN PC FROM TIMER
3240	013034	004737	005756		JSR	PC,.REWIND		:REWIND SLAVE
(1)	013040	102442			BVS	99\$		:BRANCH IF ERROR ON REWIND
3241	013042	042765	003700	000032	BIC	#3700,TC(R5)		:CLEAR CURRENT DENSITY
3242	013050	052765	002300	000032	BIS	#PE1600,TC(R5)		:SET 1600 BPI
3243	013056	004337	006174		JSR	R3,TMCMD		:WRITE 3200. WORD RECORD
3244	013062	017450			.WORD	WTBUF		
3245	013064	171600			.WORD	-3200.		
3246	013066	163400			.WORD	-6400.		
3247	013070	000060			.WORD	WFWD		
3248	013072	005215			INC	(R5)		:SET 'GO' BIT
3249								
3250	013074	022710	163400	1\$:	CMR	#-6400.,(R0)		:BRANCH WHEN WRITING STARTS
3251	013100	001004			BNE	2\$		
3252	013102	032711	040000		BIT	#ERR,(R1)		:MONITOR ERROR BIT
3253	013106	001017			BNE	99\$		
3254	013110	000771			BR	1\$		
3255								
3256	013112			2\$:				
(1)	013112	004737	005000		JSR	PC,TIMON		:TURN TIMER ON
3257	013116	105711		3\$:	TSTB	(R1)		:BRANCH WHEN READY SETS
3258	013120	100402			BMI	4\$		
3259	013122	000163	005070		JMP	TIMER(R3)		:GO TO TIMER & RETURN VIA R2
3260								
3261	013126	006204		4\$:	ASR	R4		:DIVIDE TIME BY 4
3262	013130	006204			ASR	R4		
3263	013132	004737	005652		JSR	PC,WAITRDY		
3264	013136	102403			BVS	99\$		
3265	013140	004737	005126		JSR	PC,TIMOK		:CHECK TIME
3266	013144	000401			BR	100\$		
3267								
3268	013146	104400		99\$:	HLT			

```
3269 013150 104000 100$: SCOPE
3270
3271 :TEST 024-ERASE
3272 :THIS TEST MEASURES TIME FROM 'GO'=1 TO 'RDY'=1.
3273 013152 112737 000024 001126 TST024: MOVB #24,@TSTNUM
3274 013160 012702 013242 MOV #2$,R2 ;SET RETURN PC FROM TIMER
3275 013164 004737 005756 JSR PC,REWIND ;REWIND SLAVE
(1) 013170 102436 BVS 99$ ;BRANCH IF ERROR ON REWIND
3276 013172 004737 005614 JSR PC,RHINIT ;SET NRZ
3277 013176 004737 006040 JSR PC,WRITE ;WRITE A RECORD
3278 013202 005215 INC (R5) ;SET 'GO' BIT
3279 013204 004737 005652 JSR PC,WAITRDY
3280 013210 102426 BVS 99$
3281 013212 012737 013220 001002 MOV #1$,@NSCPADR
3282 013220 004337 006174 1$: JSR R3,@TMCMD
3283 013224 000000 .WORD 0
3284 013226 000000 .WORD 0
3285 013230 000000 .WORD 0
3286 013232 000024 .WORD ERASE
3287 013234 004737 005000 JSR PC,TIMON ;TURN TIMER ON
3288 013240 005215 INC (R5) ;SET 'GO' BIT
3289
3290 013242 105711 2$: TSTB (R1) ;BRANCH WHEN READY SETS
3291 013244 100402 BMI 3$
3292 013246 000163 005070 JMP TIMER(R3) ;GO TO TIMER & RETURN VIA R2
3293
3294 013252 004737 005652 3$: JSR PC,WAITRDY
3295 013256 102403 BVS 99$
3296 013260 004737 005126 JSR PC,TIMOK
3297 013264 000401 BR 100$
3298
3299 013266 104400 99$: HLT
3300 013270 104000 100$: SCOPE
3301
3302 :TEST 025 TAPE MARK
3303 :THIS TEST MEASURES TIME FROM 'GO'=1 TO 'RDY'=1.
3304 013272 112737 000025 001126 TST025: MOVB #25,@TSTNUM
3305 013300 012702 013342 MOV #1$,R2 ;SET RETURN PC FROM TIMER
3306 013304 004737 006040 JSR PC,WRITE ;WRITE A RECORD
3307 013310 005215 INC (R5) ;SET 'GO' BIT
3308 013312 004737 005652 JSR PC,WAITRDY
3309 013316 102423 BVS 99$
3310 013320 004337 006174 JSR R3,@TMCMD
3311 013324 000000 .WORD 0
3312 013326 000000 .WORD 0
3313 013330 000000 .WORD 0
3314 013332 000026 .WORD WFMK
3315 013334 004737 005000 JSR PC,TIMON ;TURN TIMER ON
3316 013340 005215 INC (R5) ;SET 'GO' BIT
3317
3318 013342 105711 1$: TSTB (R1) ;BRANCH WHEN READY SETS
3319 013344 100402 BMI 2$
3320 013346 000163 005070 JMP TIMER(R3) ;GO TO TIMER & RETURN VIA R2
3321
3322 013352 004737 005652 2$: JSR PC,WAITRDY
3323 013356 102403 BVS 99$
```

3324 013360 004737 005126  
3325 013364 000401  
3326  
3327 013366 104400  
3328 013370  
(1) 013370 004737 005756  
(1) 013374 102774  
3329 013376 104000  
3330

JSR PC.TIMOK  
BR 100\$  
99\$: HLT  
100\$: JSR PC..REWIND  
BVS 99\$  
SCOPE

:REWIND SLAVE  
:BRANCH IF ERROR ON REWIND

```
3332 013400 032777 002000 165372 FINISH: BIT #SW10,@SWR ;DO NOT SPACE PAPER
3333 013406 001011 BNE 2$ ;IF USER SELECTED NO OUTPUT
3334 013410 005737 006442 TST CHNFLG ;OR IF IN CHAIN MODE
3335 013414 001006 BNE 2$
3336 013416 012700 000012 MOV #10.,R0 ;SET LINE FEED COUNT
3337 013422 000004 001402 1$: TYPE,CRLF
3338 013426 005300 DEC R0
3339 013430 001374 BNE 1$
3340
3341
3342 013432 105237 001005 2$: INCB @#SLVNUM ;SET NEXT SLAVE #
3343 013436 005237 001006 INC @#SLVPTR ;AND ITS POINTER
3344 013442 122737 000010 001005 CMPB #8.,@#SLVNUM ;BRANCH IF LAST SLAVE (7)
3345 013450 001402 BEQ 3$
3346 013452 000137 007512 JMP @#BEGIN ;BEGIN TEST ON NEXT SLAVE
3347 013456 105037 001005 3$: CLRB @#SLVNUM ;SET SLAVE #0
3348 013462 105237 001004 INCB @#DRVNUM ;AND INCREMENT DRIVE #
3349 013466 122737 000010 001004 CMPB #8.,@#DRVNUM ;AND CHECK IF LAST DRIVE
3350 013474 001402 BEQ END
3351 013476 000137 007512 JMP @#BEGIN
3352
3353 013502 105737 001132 END: TSTB @#UNTFND ;BRANCH IF A UNIT WAS FOUND
3354 013506 001004 BNE 1$
3355 013510 000004 015114 TYPE,E.UNIT
3356 013514 000137 006344 JMP @#INIT
3357 013520 105237 001134 1$: INCB @#PSCNT ;INCREMENT PASS COUNT
3358 013524 000004 014345 TYPE,M.EOP
3359 013530 113702 001134 MOVB @#PSCNT,R2 ;GET PASSCOUNT
3360 013534 004737 003134 JSR PC,TYPOCT ;AND TYPE IT
3361 013540 000004 001402 TYPE,CRLF
3362 013544 013700 000042 MOV @#42,R0 ;GET ACT11 RETURN ADDRESS
(1) 013550 001405 BEQ HERE ;BRANCH IF NOT ACT11
(1) 013552 000005 RESET
(1) 013554 004710 SENDAD: JSR PC,(R0)
(1) 013556 000240 NOP
(1) 013560 000240 NOP
(1) 013562 000240 NOP
(1) 013564 000240 HERE: NOP
3363 013566 005737 006442 TST CHNFLG ;BRANCH IF CHAIN MODE
3364 013572 001004 BNE 1$
3365 013574 032777 000100 165176 BIT #SW06,@SWR ;BRANCH IF NOT CONTINOUS LOOP
3366 013602 001402 BEQ 2$
3367 013604 000137 007464 1$: JMP @#RSTRT ;RESTART
3368 013610 000000 2$: HALT
3369 013612 000005 RESET
3370 013614 000137 006344 JMP @#INIT ;RESTART
```

```

3372 ;SKEW TAPE TIMING TESTS
3373 ;THE FOLLOWING TESTS REQUIRE A SPECIALLY WRITTEN 800 BPI SKEW TAPE
3374 013620 012737 013626 001002 SKEWTST:MOV #TST026,@#SCPADR ;SET SCOPE POINTER
3375
3376 ;TEST 026- SKEW TAPE SPEED TEST-FORWARD
3377 ;THIS TEST READS 32" OF TAPE (26400.-800. = 25600. FRAMES), THEN
3378 ;DIVIDES TIME BY 32. TO GET TIME TO READ 1" (800. FRAMES) OF TAPE.
3379 013626 112737 000026 001126 TST026: MOV #26,@#TSTNUM
3380 013634 012702 013722 MOV #2$,R2 ;SET RETURN PC FROM TIMER
3381 013640 004737 005756 JSR PC,.REWIND ;REWIND SLAVE
(1) 013644 102445 BVS 99$ ;BRANCH IF ERROR ON REWIND
3382 013646 052765 001700 000032 BIS #NORM11,TC(R5) ;SET 800 BPI
3383 013654 052765 000010 000010 BIS #BAI,CS2(R5) ;INHIBIT BUS ADDRESS INCREMENT
3384 013662 004337 006174 JSR R3,@#TMCMD ;READ 32" OF TAPE-FORWARD
3385 013666 017450 .WORD RDBUF
3386 013670 177777 .WORD -1.
3387 013672 063440 10$: .WORD 26400. ;FRAME COUNT
3388 013674 000070 .WORD RDFWD
3389 013676 005215 INC (R5) ;SET 'GO' BIT
3390
3391 013700 032765 075027 000014 1$: BIT #HRDERR,ER(R5) ;BRANCH IF ANY HARD ERROR BIT SETS
3392 013706 001024 BNE 99$
3393 013710 022710 001440 CMP #800.,(R0) ;WAIT FOR FIRST 800 FRAMES
3394 013714 101371 BHI 1$ ;TO BE READ
3395
3396 013716 004737 005000 JSR PC,TIMON ;TURN TIMER ON
3397 013722 023710 013672 2$: CMP @#10$, (R0) ;WAIT FOR READING TO FINISH
3398 013726 103402 BLO 3$
3399 013730 000163 005070 JMP TIMER(R3) ;GO TO TIMER & RETURN VIA R2
3400
3401 013734 012700 000005 3$: MOV #5,R0 ;DIVIDE TIME BY 32.
3402 013740 006204 4$: ASR R4
3403 013742 005300 DEC R0
3404 013744 001375 BNE 4$
3405 013746 004737 005614 JSR PC,RHINIT ;INIT DRIVE
3406 013752 004737 005126 JSR PC,TIMOK ;CHECK TIME
3407 013756 000401 BR 100$
3408
3409 013760 104400 99$: HLT
3410 013762 104000 100$: SCOPE
3411
3412 ;TEST 027-SKEW TAPE SPEED TEST-REVERSE
3413 ;THIS TEST READS FORWARD 40" (32000. FRAMES) OF TAPE, THEN READS REVERSE
3414 ;32" (26400.-800. = 25600. FRAMES) OF TAPE. THE TIME IS THEN DIVIDED BY
3415 ;32. TO GET TIME TO READ 1" (800. FRAMES) OF TAPE.
3416 013764 112737 000027 001126 TST027: MOV #27,@#TSTNUM
3417 013772 012702 014130 MOV #3$,R2 ;SET RETURN PC FROM TIMER
3418 013776 004737 005756 JSR PC,.REWIND ;REWIND SLAVE
(1) 014002 102471 BVS 99$ ;BRANCH IF ERROR ON REWIND
3419 014004 052765 001700 000032 BIS #NORM11,TC(R5)
3420 014012 052765 000010 000010 BIS #BAI,CS2(R5)
3421 014020 004337 006174 JSR R3,@#TMCMD ;READ FORWARD 32000. FRAMES
3422 014024 017450 .WORD RDBUF
3423 014026 177777 .WORD -1. ;WORD COUNT
3424 014030 076400 10$: .WORD 32000. ;FRAME COUNT
3425 014032 000070 .WORD RDFWD ;READ FORWARD
  
```

3426	014034	005215			INC	(R5)		:SET 'GO' BIT
3427								
3428	014036	032711	040000		1\$:	BIT	#ERR,(R1)	:BRANCH IF ERROR BITS SETS
3429	014042	001051				BNE	99\$	
3430	014044	023710	014030			CMP	@#10\$, (R0)	
3431	014050	101372				BHI	1\$	
3432								
3433	014052	004737	005614			JSR	PC,RHINIT	:INIT DRIVE
3434	014056	004737	005360			JSR	PC,DELAY	:WAIT FOR TAPE MOTION TO STOP
3435	014062	052765	000010	000010		BIS	#BAI,CS2(R5)	:INHIBIT BUS ADDRESS INCREMENT
3436	014070	004337	006174			JSR	R3,@#TMCMD	:READ REVERSE 32" OF TAPE
3437	014074	017450				.WORD	RDBUF	:READ BUFFER
3438	014076	177777				.WORD	-1.	:WORD COUNT
3439	014100	063440			11\$:	.WORD	26400.	:FRAME COUNT
3440	014102	000076				.WORD	RDREV	:READ REVERSE
3441	014104	005215				INC	(R5)	:SET 'GO' BIT
3442								
3443	014106	032765	075027	000014	2\$:	BIT	#HRDERR,ER(R5)	:EXIT TEST IF ERROR BIT SETS
3444	014114	001024				BNE	99\$	
3445	014116	022710	001440			CMP	#800.,(R0)	:WAIT FOR FIRST 800 FRAMES
3446	014122	101371				BHI	2\$	:TO BE READ
3447								
3448	014124	004737	005000			JSR	PC,TIMON	:TURN TIMER ON
3449	014130	023710	014100		3\$:	CMP	@#11\$, (R0)	:WAIT FOR ALL FRAMES TO BE READ
3450	014134	103402				BLO	4\$	
3451	014136	000163	005070			JMP	TIMER(R3)	:GO TO TIMER & RETURN VIA R2
3452								
3453	014142	012700	000005		4\$:	MOV	#5,R0	:DIVIDE TIME BY 32.
3454	014146	006204			5\$:	ASR	R4	
3455	014150	005300				DEC	R0	
3456	014152	001375				BNE	5\$	
3457	014154	004737	005614			JSR	PC,RHINIT	
3458	014160	004737	005126			JSR	PC,TIMOK	
3459	014164	000401				BR	100\$	
3460								
3461	014166	104400			99\$:	HLT		
3462	014170				100\$:			
(1)	014170	004737	005756			JSR	PC,.REWIND	:REWIND SLAVE
(1)	014174	102774				BVS	99\$	:BRANCH IF ERROR ON REWIND
3463	014176	104000				SCOPE		
3464								
3465	014200	000137	013400			JMP	@#FINISH	
3466								
3467								
3468								

```
3470          .SBITL      PROGRAM MESSAGES
3471          :OPERATOR INSTRUCTIONS
3472 014204 005015 046524 031460 M.NAM: .ASCII <CR><LF>'TM03-TE16/TU77 DRIVE FUNCTION TIMER (CZTEEDO)';++B
      014212 052055 030505 027466
      014220 052524 033467 042040
      014226 044522 042526 043040
      014234 047125 052103 047511
      014242 020116 044524 042515
      014250 020122 041450 052132
      014256 042505 030104 051
3473 014263 015 052012 050131 .ASCIZ <CR><LF>'TYPE <CR> TO TERMINATE RESPONSE & ^C TO RESTART'
      014270 020105 041474 037122
      014276 052040 020117 042524
      014304 046522 047111 052101
      014312 020105 042522 050123
      014320 047117 042523 023040
      014326 057040 020103 047524
      014334 051040 051505 040524
      01342 052122 000
3474 01345 015 042412 042116 M.EOP: .ASCIZ <CR><LF>'END OF PASS '
      014352 047440 020106 040520
      014360 051523 000040
3475 014364 005015 040510 042122 M.HSWR: .ASCIZ <CR><LF>'HARDWARE SWR IN USE'<CR><LF>
      014372 040527 042522 051440
      014400 051127 044440 020116
      014406 051525 006505 000012
3476 014414 005015 042522 047515 I.REM: .ASCIZ <CR><LF>'REMOVE TMDP FROM SLAVE 0. CLEAR LOC. 40.'
      014422 042526 052040 042115
      014430 020120 051106 046517
      014436 051440 040514 042526
      014444 030040 020056 046103
      014452 040505 020122 047514
      014460 027103 032040 027060
      014466 000
3477 014467 015 052012 050131 I.REG: .ASCIZ <CR><LF>'TYPE FIRST ADDRESS OF CONTROLLER '
      014474 020105 044506 051522
      014502 020124 042101 051104
      014510 051505 020123 043117
      014516 041440 047117 051124
      014524 046117 042514 020122
      014532 000040
3478 014534 054524 042520 052040 I.DRVS: .ASCIZ %TYPE TM03 DRIVE #'S TO BE TESTED: ALL %
      014542 030115 020063 051104
      014550 053111 020105 023443
      014556 020123 047524 041040
      014564 020105 042524 052123
      014572 042105 020072 040440
      014600 046114 000040
3479 014604 047506 020122 046524 I.SLVS: .ASCII 'FOR TM03 DRIVE '
      014612 031460 042040 044522
      014620 042526 040
3480 014623 060 020055 054524 I.DR": .ASCIZ %0- TYPE SLAVE #'S TO BE TESTED: ALL %
      014630 042520 051440 040514
      014636 042526 021440 051447
      014644 052040 020117 042502
      014652 052040 051505 042524
```

	014660	035104	040440	046114	
3481	014666	000040			
	014670	050123	042505	020104	I.SPD: .ASCIZ 'SPEED TESTS? (YES/NO): NO '
	014676	042524	052123	037523	
	014704	024040	042531	027523	
	014712	047516	035051	047040	
3482	014720	020117	000		
	014723	015	042412	042116	M.EOT: .ASCIZ <CR><LF>'END OF TAPE'<CR><LF>
	014730	047440	020106	040524	
	014736	042520	005015	000	
3483					
3484					:ERROR MESSAGES
3485	014743	015	052012	040522	E.TRP4: .ASCIZ <CR><LF>'TRAPPED TO 4'
	014750	050120	042105	052040	
	014756	020117	000064		
3486	014762	047516	041440	047117	E.NCON: .ASCIZ 'NO CONTROLLER AT ADDRESS SPECIFIED'<CR><LF>
	014770	051124	046117	042514	
	014776	020122	052101	040440	
	015004	042104	042522	051523	
	015012	051440	042520	044503	
	015020	044506	042105	005015	
	015026	000			
3487	015027	124	030115	020063	E.NDRV: .ASCIZ 'TM03 DRIVE '
	015034	051104	053111	020105	
	015042	000			
3488	015043	104	044522	042526	E.NSLV: .ASCII 'DRIVE '
	015050	040			
3489	015051	060	051440	040514	E.DRV: .ASCII '0 SLAVE '
	015056	042526	040		
3490	015061	060	047040	052117	E.NAVA: .ASCIZ '0 NOT AVAILABLE FOR TEST'<CR><LF>
	015066	040440	040526	046111	
	015074	041101	042514	043040	
	015102	051117	052040	051505	
3491	015110	006524	000012		
	015114	047516	052040	030115	E.UNIT: .ASCIZ 'NO TM03-TE16/TU77 UNIT FOUND TO TEST'<CR><LF>
	015122	026463	042524	033061	
	015130	052057	033525	020067	
	015136	047125	052111	043040	
	015144	052517	042116	052040	
	015152	020117	042524	052123	
	015160	005015	000		
3492	015163	123	043117	020124	E.SFT: .ASCIZ 'SOFT ERROR (DATA)'<CR><LF>
	015170	051105	047522	020122	
	015176	042050	052101	024501	
	015204	005015	000		
3493	015207	124	051505	020124	E.HDR: .ASCIZ 'TEST # '
	015214	020043	000		
3494	015217	040	042504	044526	E.HDR1: .ASCII ' DEVICE ERROR'<CR><LF>
	015224	042503	042440	051122	
	015232	051117	005015		
3495	015236	051503	004461	041527	.ASCIZ 'CS1'<HT>'WC'<HT>'BA'<HT>'FC'<HT>'CS2'<HT>'DS'<HT>'ER'<HT>'TC'<CR><LF>
	015244	041011	004501	041506	
	015252	041411	031123	042011	
	015260	004523	051105	052011	
	015266	006503	000012		
3496	015272	047440	052125	047440	E.HDR2: .ASCIZ ' OUT OF RANGE ERROR'<CR><LF>

	015300	020106	040522	043516	
	015306	020105	051105	047522	
	015314	006522	000012		
3497	015320	044440	020106	044124	E.77T5: .ASCII ' IF THE M8940 IS REV J3 , L OR HIGHER THEN CHECK ACTUAL VALUE '<CR><LF>
	015326	020105	034115	032071	
	015334	020060	051511	051040	
	015342	053105	045040	020063	
	015350	020054	020114	051117	
	015356	044040	043511	042510	
	015364	020122	044124	047105	
	015372	041440	042510	045503	
	015400	040440	052103	040525	
	015406	020114	040526	052514	
	015414	020105	005015		
3498	015420	040440	040507	047111	.ASCII ' AGAINST THIS RANGE : < 011250 - 009215 > '<CR><LF>
	015426	052123	052040	044510	
	015434	020123	040522	043516	
	015442	020105	004472	036040	
	015450	030040	030461	032462	
	015456	020060	020055	030060	
	015464	031071	032461	037040	
	015472	006440	012		
3499	015475	040	043111	044440	.ASCIIZ ' IF IT FALLS INTO THIS RANGE THEN YOU HAVE NO PROBLEM.'<CR><LF><LF>
	015502	020124	040506	046114	
	015510	020123	047111	047524	
	015516	052040	044510	020123	
	015524	040522	043516	020105	
	015532	044124	047105	054440	
	015540	052517	044040	053101	
	015546	020105	047516	050040	
	015554	047522	046102	046505	
	015562	006456	005012	000	
3500	015567	040	043111	052040	E.77T16: .ASCII ' IF THE REV IS LOWER THAN M2 THEN CHECK ACTUAL VALUE '<CR><LF>
	015574	042510	051040	053105	
	015602	044440	020123	047514	
	015610	042527	020122	044124	
	015616	047101	046440	020062	
	015624	044124	047105	041440	
	015632	042510	045503	040440	
	015640	052103	040525	020114	
	015646	040526	052514	020105	
	015654	005015			
3501	015656	040440	040507	047111	.ASCII ' AGAINST THIS RANGE : < 004500 - 003570 > '<CR><LF>
	015664	052123	052040	044510	
	015672	020123	040522	043516	
	015700	020105	004472	036040	
	015706	030040	032060	030065	
	015714	020060	020055	030060	
	015722	032463	030067	037040	
	015730	006440	012		
3502	015733	040	043111	044440	.ASCIIZ ' IF IT FALLS INTO THIS RANGE THEN YOU HAVE NO PROBLEM.'<CR><LF><LF>
	015740	020124	040506	046114	
	015746	020123	047111	047524	
	015754	052040	044510	020123	
	015762	040522	043516	020105	
	015770	044124	047105	054440	

	015776	052517	044040	053101		
	016004	020105	047516	050040		
	016012	047522	046102	046505		
	016020	006456	005012	000		
3503	016025	015	052012	046511	E.TIMOV:	.ASCIZ <CR><LF>'TIMER OVERFLOWED'<CR><LF>
	016032	051105	047440	042526		
	016040	043122	047514	042527		
	015046	006504	000012			
3504	016052	005015	044524	042515	E.TIMEX:	.ASCIZ <CR><LF>'TIME EXPIRED WAITING FOR RDY'<CR><LF>
	016060	042440	050130	051111		
	016066	042105	053440	044501		
	016074	044524	043516	043040		
	016102	051117	051040	054504		
	016110	005015	000			
3505	016113	040	040507	020120	E.GAP:	.ASCIZ ' GAP # '
	016120	020043	000			
3506						
3507						
3508	016123	015	025012	025052		:TIME DOCUMENT LINES
	016130	025052	025052	025052	L.HDR1:	.ASCIZ <CR><LF>'*****'
	016136	025052	025052	025052		
	016144	025052	025052	025052		
	016152	025052	025052	025052		
	016160	025052	025052	025052		
	016166	025052	025052	025052		
	016174	025052	025052	025052		
	016202	025052	025052	025052		
	016210	025052	025052	025052		
	016216	025052	025052	025052		
	016224	025052	025052	025052		
	016232	025052	006452	000012		
3509	016240	020052	046524	031460	L.HDR2:	.ASCII '* TM03 DRIVE FUNCTION TIMES- DRIVE # '
	016246	042040	044522	042526		
	016254	043040	047125	052103		
	016262	047511	020116	044524		
	016270	042515	026523	042040		
	016276	044522	042526	021440		
	016304	040				
3510	016305	060	051440	040514	L.DRV:	.ASCII '0 SLAVE # '
	016312	042526	021440	040		
3511	016317	060	020040	000	L.SLV:	.ASCIZ '0 '
3512	016323	124	030505	020066	L.TE16:	.ASCIZ 'TE16 '
	016330	000				
3513	016331	124	033525	020067	L.TU77:	.ASCIZ 'TU77 '
	016336	000				
3514	016337	123	051105	040511	L.SER:	.ASCIZ 'SERIAL # '
	016344	020114	020043	000		
3515	016351	040	005015	006452	L.HDR3:	.ASCII ' '<CR><LF>'*<CR><LF>
	016356	012				
3516	016357	052	043040	047125		.ASCIZ '* FUNCTION'<HT><HT>'TIME (SPECIFICATION)'<HT>'TIME (ACTUAL)'<CR><LF>
	016364	052103	047511	004516		
	016372	052011	046511	024105		
	016400	050123	041505	043111		
	016406	041511	052101	047511		
	016414	024516	052011	046511		
	016422	024105	041501	052524		

3517	016430	046101	006451	000012	
3518	016436	040522	043516	036505	L.RNG: .ASCIZ 'RANGE=<'
	016444	000074			
3519	016446	041501	052524	046101	L.ACT: .ASCIZ 'ACTUAL='
	016454	000075			
3520					
3521					:TEST DESCRIPTOR HEADERS
3522	016456	025052	044440	044516	A.T000: .ASCIZ '** INITIALIZATION ERROR'
	016464	044524	046101	055111	
	016472	052101	047511	020116	
	016500	051105	047522	000122	
3523	016506	020052	051127	052111	A.T001: .ASCIZ '* WRITE FROM BOT'<HT>
	016514	020105	051106	046517	
	016522	041040	052117	000011	
3524	016530	020052	051127	052111	A.T002: .ASCIZ '* WRITE START'<HT><HT>
	016536	020105	052123	051101	
	016544	004524	000011		
3525	016550	020052	051127	052111	A.T003: .ASCIZ '* WRITE SHUTDOWN'<HT>
	016556	020105	044123	052125	
	016564	047504	047127	000011	
3526	016572	020052	051127	052111	A.T004: .ASCIZ '* WRITE SETTLEDOWN'<HT>
	016600	020105	042523	052124	
	016606	042514	047504	047127	
	016614	000011			
3527	016616	020052	042522	042101	A.T005: .ASCIZ '* READ FROM BOT'<HT><HT>
	016624	043040	047522	020115	
	016632	047502	004524	000011	
3528	016640	020052	042522	042101	A.T006: .ASCIZ '* READ START'<HT><HT>
	016646	051440	040524	052122	
	016654	004411	000		
3529	016657	052	051040	040505	A.T007: .ASCIZ '* READ SHUTDOWN'<HT><HT>
	016664	020104	044123	052125	
	016672	047504	047127	004411	
	016700	000			
3530	016701	052	051040	040505	A.T010: .ASCIZ '* READ SETTLEDOWN'<HT>
	016706	020104	042523	052124	
	016714	042514	047504	047127	
	016722	000011			
3531	016724	020052	042522	042101	A.T011: .ASCIZ '* READ REV START'<HT>
	016732	051040	053105	051440	
	016740	040524	052122	000011	
3532	016746	020052	042522	042101	A.T012: .ASCIZ '* READ REV SHUTDOWN'<HT>
	016754	051040	053105	051440	
	016762	052510	042124	053517	
	016770	004516	000		
3533	016773	052	051040	040505	A.T013: .ASCIZ '* READ REV SETTLEDOWN'<HT>
	017000	020104	042522	020126	
	017006	042523	052124	042514	
	017014	047504	047127	000011	
3534	017022	020052	052524	047122	A.T014: .ASCIZ '* TURN AROUND DELAY F-R'<HT>
	017030	040440	047522	047125	
	017036	020104	042504	046514	
	017044	020131	026506	004522	
	017052	000			
3535	017053	052	052040	051125	A.T015: .ASCIZ '* TURN AROUND DELAY R-F'<HT>

	017060	020116	051101	052517		
	017066	042116	042040	046105		
	017074	054501	051040	043055		
	017102	000011				
3536	017104	020052	040507	020120	A.T016: .ASCIZ	'* GAP SIZE-STOP HALF'<HT>
	017112	044523	042532	051455		
	017120	047524	020120	040510		
	017126	043114	000011			
3537	017132	020052	040507	020120	A.T017: .ASCIZ	'* GAP SIZE-START HALF'<HT>
	017140	044523	042532	051455		
	017146	040524	052122	044040		
	017154	046107	004506	000		
3538	017161	052	043440	050101	A.T020: .ASCIZ	'* GAP SIZE-INTERRECORD'<HT>
	017166	051440	055111	026505		
	017174	047111	042524	051122		
	017202	041505	051117	004504		
	017210	000				
3539	017211	052	043440	050101	A.T021: .ASCIZ	'* GAP CONSISTANCY'<HT>
	017216	041440	047117	044523		
	017224	052123	047101	054503		
	017232	000011				
3540	017234	020052	040504	040524	A.T022: .ASCIZ	'* DATA TIME-800BPI'<HT>
	017242	052040	046511	026505		
	017250	030070	041060	044520		
	017256	000011				
3541	017260	020052	040504	040524	A.T023: .ASCIZ	'* DATA TIME-1600BPI'<HT>
	017266	052040	046511	026505		
	017274	033061	030060	050102		
	017302	004511	000			
3542	017305	052	042440	040522	A.T024: .ASCIZ	'* ERASE GAP TIME'<HT>
	017312	042523	043440	050101		
	017320	052040	046511	004505		
	017326	000				
3543	017327	052	053440	044522	A.T025: .ASCIZ	'* WRITE FILE MARK'<HT>
	017334	042524	043040	046111		
	017342	020105	040515	045522		
	017350	000011				
3544	017352	020052	040524	042520	A.T026: .ASCIZ	'* TAPE SPEED-FWD'<HT>
	017360	051440	042520	042105		
	017366	043055	042127	000011		
3545	017374	020052	040524	042520	A.T027: .ASCIZ	'* TAPE SPEED-REV'<HT>
	017402	051440	042520	042105		
	017410	051055	053105	000011		
3546						
3547	017416	005015	043536	000	L.CNTG: .ASCIZ	<CR><LF>'^G'
3548	017423	015	051412	051127	L.SWR: .ASCIZ	<CR><LF>'SWR='
	017430	000075				
3549	017432	020040	042516	036527	L.NEW: .ASCIZ	' NEW= '
	017440	000040				
3550	017442	005015	006477	000012	L.QUEST: .ASCIZ	<CR><LF>'?'<CR><LF>
3551						
3552		017450				.EVEN
3553		017450				RDBUF=.
3554	017450	000200				WTBUF=.
3555		000001				.BLKW 128.
						.END











CZTEEDO TMO3-TE16/TU77 DFT		MACY11 30(1046) 06-JUL-83 21:05		L 7 PAGE 26-5										SEQ 0089
CZTEED.P11 06-JUL-83 14:42		CROSS REFERENCE TABLE -- USER SYMBOLS												
		2831	2865	2891	2921	2957	2989	3019	3043	3071	3103	3160	3222	3259
		3292	3320	3399	3451									
TIMERR	005100	2005	2007#											
TIMERO	005114	2013#												
TIMER1	005044	1998#	2002											
TIMOK	005126	2025#	2688	2707	2733	2764	2785	2807	2835	2869	2895	2925	2961	2993
		3023	3047	3075	3107	3192	3230	3265	3296	3324	3406	3458		
TIMON	005000	1975#	2094	2108	2679	2698	2725	2756	2776	2798	2827	2862	2886	2918
		2954	2984	3014	3038	3066	3098	3155	3219	3256	3287	3315	3396	3448
TKB	= 177562	1086#	1784	1811										
TKISR	004126	1240	1811#											
TKS	= 177560	1086#	1781	2619*										
TKVEC	= 000060	1087#	1239											
TMBASE	001010	1262#	1925	2358	2365*	2366	2585	2612						
TMCMD	006174	2217	2238	2248	2258	2290#	3206	3243	3282	3310	3384	3421	3436	
TMCS1	= 172440	1090#	1262											
TMK	= 000004	1157#												
TPB	= 177566	1086#												
TPS	= 177564	1086#												
TPVEC	= 000064	1087#												
TRAPVE	= 000034	1087#												
TRE	= 040000	1130#												
TRTVEC	= 000014	1087#												
TSTNUM	001126	1273#	1764	1769	1851	1858	1863	1931	2041	2611*	2675*	2695*	2715*	2740*
		2772*	2793*	2815*	2843*	2878*	2903*	2933*	2976*	3001*	3030*	3054*	3083*	3125*
		3202*	3238*	3273*	3304*	3379*	3416*							
TSTTBL	002444	1502#	2642	2643										
TST000	010050	1502	1933	2611#	2664									
TST001	010344	1503	2665	2675#										
TST002	010430	1504	2695#											
TST003	010506	1505	2715#											
TST004	010576	1506	2740#											
TST005	010704	1507	2772#											
TST006	010770	1508	2793#											
TST007	011054	1509	2815#											
TST010	011154	1510	2843#											
TST011	011274	1511	2878#											
TST012	011372	1512	2903#											
TST013	011502	1513	2933#											
TST014	011642	1514	2976#											
TST015	011734	1515	3001#											
TST016	012042	1516	3030#											
TST017	012136	1517	3054#											
TST020	012246	1518	3083#											
TST021	012366	1519	3125#											
TST022	012672	1520	3202#											
TST023	013022	1521	3238#											
TST024	013152	1522	3273#											
TST025	013272	1523	3304#											
TST026	013626	1524	3374	3379#										
TST027	013764	1525	3416#											
TU77GT	002024	1401#												
TU77TT	001664	1368#	2600											
TYPDEC	003242	1621#	1667	1670	1674	1695	1698	1702						
TYPE	= 000004	1209#	1533	1536	1548	1605	1638	1665	1668	1671	1672	1675	1693	1696
		1699	1700	1703	1706	1767	1789	1794	1804	1806	1829	1850	1857	1862



CZTEEDO TMO3-TE16/TU77 DFT  
CZTEED.P11 06-JUL-83 14:42

MACY11 30(1046) 06-JUL-83 21:05 PAGE 27  
CROSS REFERENCE TABLE -- MACRO NAMES

N 7

SEQ 0091

INPUT	1068#	1537	2361	2386	2455	2572									
RESTOR	1065#	1606	1639	1730	1771	1905	2049	2089							
REWIND	1071#	2200	2625	2677	2773	2968	3127	3204	3240	3275	3328	3381	3418	3462	
SAVE	1062#	1585	1622	1714	1739	2025	2061								
SETGO	1081#	2202	2222	2273	2629	2799	2819	2848	2881	2887	2906	2910	2936	2940	2979
	2985	3004	3008	3015	3033	3039	3057	3061	3067	3086	3090	3094	3099	3131	3141
	3150	3156	3211	3248	3278	3288	3307	3316	3389	3426	3441				
TIMCHK	1078#	2097	2111	2684	2703	2729	2760	2781	2803	2831	2865	2891	2921	2957	2989
	3019	3043	3071	3103	3160	3222	3259	3292	3320	3399	3451				
TIMEON	1075#	2108	2679	2725	2756	2776	2798	2827	2862	2886	2918	2954	2984	3014	3038
	3066	3098	3155	3219	3256	3287	3315	3396	3448						
SCATCH	1018#	1224													
SCHAIN	1018#	2348													
SCPREG	1018#	1086													
SCPVEC	1018#	1087													
STYPE	1018#	1548													
.SACT1	1018#	1238													
.SEOP	1018#	3362													

. ABS. 020050 000

ERRORS DETECTED: 0

CZTEED,CZTEED/CRF=CZTEAD.SML/ML,CZTEED.P11  
RUN-TIME: 4 7 .9 SECONDS  
RUN-TIME RATIO: 20/13=1.4  
CORE USED: 11K (22 PAGES)