

TM03/TE16

DRIVE FUNCTION TIMER
CZTEEB0

AH-A804B-MC

COPYRIGHT 77-78

FICHE 1 OF 1

JAN 1979

digital

MADE IN USA

The table contains 16 small diagrams or tables, arranged in a 4x4 grid. Each cell contains technical information, possibly related to drive function timer settings or configurations. The content is too faint to read accurately but appears to include various parameters and diagrams.

.REM %

IDENTIFICATION

PRODUCT CODE: AC-A803B-MC
PRODUCT NAME: CZTEEB0 TM03-TE16/TU77 DRIVE FUNCTION TIMER
DATE CREATED: 15 NOV 78
MAINTAINER: DIAGNOSTIC GROUP
AUTHOR: .. G. ADAMS

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) 1977,1978 BY DIGITAL EQUIPMENT CORPORATION

TMO3 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 CAP
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 ERRORS.

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

TMO3 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, TMO3 DRIVES TO BE TESTED, TE16/TU77 SLAVES TO BE TESTED,
AND IF SPEED TESTS ARE TO BE RUN. IN ADDITION 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 TMO3 DRIVE #'S TO BE TESTED:ALL
FOR TMO3 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 'RUBOUT' TO DELETE LAST
CHARACTER(S).
PROGRAM WILL REPORT ERRORS, AND END OF PASS.

2.1 RESTART PROCEDURE

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

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

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

2.2 AUTOMATIC MODE OPERATION

IF THE PROGRAM IS LOADED AND RUN IN AUTOMATIC (CHAIN) MODE
DEFAULT RESPONSES TO OPERATOR REQUESTS ARE USED, AND ALL AVAIL-
ABLE TMO3-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 TMO3 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=" HAS BEEN TYPED THEN THE OPERATOR CAN DO ONE
OF THE FOLLOWING AT THE TTY:
A) TYPE A NUMBER TO BE LOADED INTO THE SOFTWARE SWR.
B) IF A <CR> IS THE FIRST KEY DEPRESSED THE SOFTWARE SWITCH
REGISTER CONTENTS WILL NOT BE CHANGED.
- 2) CONTROL A <^A>:
ALTERNATES USAGE OF THE SWR BETWEEN HARDWARE & SOFTWARE.
- 3) CONTROL C <^C>:
RESTARTS PROGRAM AT 200
- 4) CONTROL U <^U>:
DELETES ALL CHARACTERS TYPED IN RESPONSE TO A REQUEST.

SW15 (100000) HALT ON ERROR THIS SWITCH WHEN SET WILL HALT THE PROCESSOR WHEN AN ERROR IS DETECTED.

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

SW14 (040000) LOOP SUBTEST THIS SWITCH WHEN SET LOOPS THE CURRENT SUBTEST REGARDLESS OF ERROR CONDITION.

SW13 (020000) INHIBIT ERROR TYPEOUT THIS SWITCH WHEN SET INHIBITS ERROR TYPEOUT.

SW11 (004000) INHIBIT SUB-TEST ITERATION THIS SWITCH WHEN SET CAUSES EACH SUBTEST TO BE EXECUTED ONLY ONCE.

SW10 (002000) INHIBIT FUNCTION TIME PUBLICATION THIS SWITCH WHEN SET WILL INHIBIT THE PRINTING OF THE FUNCTION TIMES. (SEE CHAPTER 8.)

SW09 (001000) RING BELL ON ERROR THIS SWITCH WHEN SET WILL RING THE BELL ON THE TTY WHEN AN ERROR IS DETECTED.

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 SFELECTED

NOTE: A TEST CAN ONLY BE SELECTED DURING STARTUP (OR RESTART).
DO NOT INHIBIT SUBTEST ITERATIONS WHEN PROGRAM IS RUNNING.

TMO3 DRIVE FUNCTION TIMER
ERRORS

PAGE 7

CHAPTER 4
ERRORS

TWO TYPES OF ERRORS ARE DETECTED BY THIS PROGRAM, HARDWARE ERRORS AND
INCORRECT 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 BBBBBS CCCCCC DDDDDD EEEEE EFFFFF GGGGGG

WHERE:

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

4.2 ERROR TYPEOUT FORMAT (FUNCTION TIME OUT OF RANGE)

TEST # XXXXXX OUT OF RANGE ERROR

RANGE <AAAAAA-BBBBBB> ACTUAL = CCCCCC

TMO3 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.

TMO3 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 INITIALLY 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.

TMO3 DRIVE FUNCTION TIMER
PROGRAM DESCRIPTION

PAGE 11

CHAPTER 7
PROGRAM DESCRIPTION

7.1 SAMPLE TIME DOCUMENT

TYPE FIRST ADDRESS OF CONTROLLER:172440
TYPE TMO3 DRIVE #'S TO BE TESTED:ALL 0
FOR TMO3 DRIVE 0- TYPE SLAVE #'S TO BE TESTED:ALL 0
TAPE SPEED TESTS ONLY? (YES/NO):NO

* FUNCTION	TIME(SPECIFICATION)	TIME(ACTUAL)
* TMO3 DRIVE FUNCTION TIMES- DRIVE # 0 SLAVE # 7 TE16 SER. # 5009		
* 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=002710
* 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

TMO3 DRIVE FUNCTION TIMER

7.1.1 SAMPLE TIME DOCUMENT FOR TAPE SPEED TESTS

TYPE FIRST ADDRESS OF CONTROLLER 172440:
TYPE TMO3 DRIVE #'S TO BE TESTED:ALL 0
FOR TMO3 DRIVE 0- TYPE SLAVE #'S TO BE TESTED:ALL 7
SPEED TESTS ONLY? (YES/NO):NO Y

*TMO3 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

TMO3 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 SERVO LOOP |

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

TMO3 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 TMO3 (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 INTERRECORD GAP.

1. LEAVE TAPE AT ITS PRESENT POSITION. ASSURE THAT IT IS STOPPED
2. ISSUE A WRITE COMMAND.
3. MONITOR FRAME COUNTER AND BIT 4 OF 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 F(=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

TMO3 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

TMO3 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 5 OF TC)
7. TIME FROM GO FORWARD TO ACCL - 1 IS THE TURN AROUND TIME.
8. STOP.

TMO3 DRIVE FUNCTION TIMER

PAGE 19

GAP MEASUREMENTS:

THE PREVIOUS THIRTEEN (13) TESTS WERE MEASUREMENTS OF LOGIC DELAYS PERFORMED BY THE TMO3 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

TMO3 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.

TMO3 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

TMO3 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.

%

978
979
980
981
982
983
984
985
986
987
988
989
990
991
992
993
994
995
996
997
998
999
1000
1001
1002
1003
1004
1005
1006
1007
1008
1009
1010
1011
1012
1013
1014
1015
1016
1017
1018
1019
1020
1021
1022
1023
1024
1025
1026
1027
1028
1029
1030
1031
1032
1033

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,$CATCH,$TYPE,..$ACT11,..$EOP,$CHAIN
.TITLE CZTEEB0 1M03-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= 000200 ;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
```

```
1034 JSR PC, INPUT ;GET USER INPUT
1035 .ENDM INPUT
1036 .MACRO REWIND
1037 JSR PC, REWIND ;REWIND SLAVE
1038 BVS 99$ ;BRANCH IF ERROR ON REWIND
1039 .ENDM REWIND
1040 .MACRO TIMEON
1041 JSR PC, TIMON ;TURN TIMER ON
1042 .ENDM
1043 .MACRO TIMCHK
1044 JMP TIMER(R3) ;GO TO TIMER & RETURN VIA R2
1045 .ENDM
1046 .MACRO SETGO
1047 INC (R5) ;SET 'GO' BIT
1048 .ENDM
1049
1050 .SBTTL REGISTER ASSIGNMENTS
1051 ;;DEFINITIONS AND REGISTER ASSIGNMENTS
1052 ;;GENERAL REGISTER ASSIGNMENTS
1053 000000 R0=%0
1054 000001 R1=%1
1055 000002 R2=%2
1056 000003 R3=%3
1057 000004 R4=%4
1058 000005 R5=%5
1059 000006 SP=%6
1060 000007 PC=%7
1061 000000 *R10=%0
1062 000001 R11=%1
1063 000002 R12=%2
1064 000003 R13=%3
1065 000004 R14=%4
1066 000005 R15=%5
1067
1068 ;;REGISTER ADDRESSES
1069 177776 PSW= 177776 ;;PROCESSER STATUS WORD
1070 177774 SLR= 177774 ;;STACK LIMIT REGISTER (11/40,11/45)
1071 177772 PIRQ= 177772 ;;PROGRAM INTERRUPT REQ. (11/45)
1072 177770 UBREAK= 177770 ;;MICRO-BREAK REGISTER (11/45)
1073 177560 TKS= 177560 ;;KEYBOARD CSR
1074 177562 TKB= 177562 ;;KEYBOARD DATA BUFFER REGISTER
1075 177564 TPS= 177564 ;;TELEPRINTER CSR
1076 177566 TPB= 177566 ;;TELEPRINTER DATA BUFFER REGISTER
1077
1078 ;;VECTOR ADDRESSES
1079 000004 ERRVEC=4 ;;ADDRESS OF ERROR VECTOR
1080 000010 RESVEC=10 ;;ADDRESS OF RESERVED INST. TRAP VECTOR
1081 000014 TBITVEC=14 ;;ADDRESS OF 'T' BIT TRAP VECTOR
1082 000014 TRTVEC=14 ;;ADDRESS OF 'TRACE' TRAP VECTOR
1083 000014 BPTVEC=14 ;;ADDRESS OF 'BREAK' 'NT' TRAP VECTOR
1084 000020 IOTVEC=20 ;;ADDRESS OF IOT TRAP VECTOR
1085 000024 PFVEC=24 ;;ADDRESS OF POWER TRAP VECTOR
1086 000030 EMTVEC=30 ;;ADDRESS OF EMT VECTOR
1087 000034 TRAPVEC=34 ;;ADDRESS OF TRAP VECTOR
1088 000060 TKVEC= 60 ;;ADDRESS OF TTY KEYBOARD INT. VECTOR
1089 000064 TPVEC=64 ;;ADDRESS OF TTY PRINTER INTERRUPT VECTOR
```

1090	000114	PARVEC= 114	:::ADDRESS OF MA/MF PARITY ERROR VECTOR
1091	000240	PIRVEC=240	:::ADDRESS OF PIRQ VECTOR
1092	000744	FPEVEC=244	:::ADDRESS OF FLOATING POINT INT. VECTOR
1093	000250	MMVEC=250	:::ADDRESS OF MEM MGMT ERROR TRAP VECTOR
1094			

1095
1096 172440
1097
1098
1099 000000
1100 000002
1101 000004
1102 000006
1103 000010
1104 000012
1105 000014
1106 000016
1107 000022
1108 000024
1109 000026
1110 000030
1111 000032
1112
1113
1114
1115 000001
1116 000000
1117 000002
1118 000006
1119 000010
1120 000026
1121 000024
1122 000030
1123 000032
1124 000050
1125 000056
1126 000060
1127 000070
1128 000076
1129 000100
1130 000200
1131 000400
1132 001000
1133 002000
1134 004000
1135 020000
1136 040000
1137 100000
1138
1139 000000
1140 000001
1141 000002
1142 000003
1143 000004
1144 000005
1145 000006
1146 000007
1147 000010
1148 000020
1149 000040
1150 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

:PHCS2-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

1151	000200	OR=	200	
1152	000400	MDPE=	400	
1153	001000	MXF=	1000	
1154	002000	PGE=	2000	
1155	004000	NEM=	4000	
1156	010000	NED=	10000	
1157	020000	UPE=	20000	
1158	040000	WCE=	40000	
1159	100000	DLT=	100000	
1160		;RHDS-DS(R5)		
1161	000001	SLA=	1	
1162	000002	BOT=	2	
1163	000004	TMK=	4	
1164	000010	IDB=	10	
1165	000020	SDWN=	20	
1166	000040	PES=	40	
1167	000100	SSC=	100	
1168	000200	DRY=	200	
1169	000400	DPR=	400	
1170	002000	EOT=	2000	
1171	004000	WRL=	4000	
1172	010000	MOL=	10000	
1173	020000	PIP=	20000	
1174	040000	ERR=	40000	
1175	100000	ATA=	100000	
1176		;RHER-ER(R5)		
1177	000001	ILF=	1	
1178	000002	ILR=	2	
1179	000004	RMR=	4	
1180				
1181	000020	FMT=	20	
1182	000100	INCVAE=	100	
1183	000200	PEFLRC=	200	
1184	000400	NSG=	400	
1185	001000	FCE=	1000	
1186	002000	CSITM=	2000	
1187	004000	NEF=	4000	
1188	010000	DTE=	10000	
1189	020000	OPI=	20000	
1190	040000	UNS=	40000	
1191	075027	HRDERR=	UNS!OPI.DTE!NEF.FCE.FMT.RMR.ILR.ILF	;HARDERROR BITS
1192				
1193		;RHMR-MR(R5)		
1194	000100	OSC=	100	
1195				
1196		;RHDT-DT(R5)		
1197	002000	SPR=	2000	
1198	010000	CH7=	10000	
1199	040000	TAP=	40000	
1200				
1201		;RHTC-TC(R5)		
1202	001700	NORM11=	1700	
1203	000320	CDM11	320	
1204	000000	BPI200=	0	
1205	000400	BPI556=	000400	
1206	001000	BPI800=	001000	

1207	002300	PE1600=	002300	
1208	100000	ACCL=	100000	
1209				
1210				
1211				
1212		; INSTRUCTION EQUATES		
1213	104400	HLT=	TRAP	
1214	104000	SCOPE=	EMT	
1215	000004	TYPE=	IOT	
1216				
1217		; MISCELLANEOUS EQUATES		
1218	006274	OUTBUF=	INIT	; OUTPUT BUFFER STARTS AT BEG OF PROGRAM
1219	177400	FRMCNT=	-256.	; FRAME COUNT
1220	177600	WRDCNT=	-128.	; WORD COUNT
1221		; ASCII EQUATES		
1222	000001	CNTRLA=	1	; ASCII CODE FOR CONTROL A (^A)
1223	000003	CNTRLC=	3	; ASCII CODE FOR CONTROL C (^C)
1224	000007	CNTRLG=	7	; ASCII CODE FOR CONTROL G (^G)
1225	000011	HT=	11	; ASCII CODE FOR HORIZONTAL TAB
1226	000012	LF=	12	; ASCII CODE FOR LINE FEED
1227	000015	CR=	15	; ASCII CODE FOR CARRIAGE RETURN
1228	000017	CNTRLO=	17	; ASCII CODE FOR CONTROL O (^O)
1229	000025	CNTRLU=	25	; ASCII CODE FOR CONTROL U (^U)


```
1230 ;SETUP TRAP VECTORS
1231 000014 000014 .=TBITVEC
1232 000014 000016 .WORD .+2 ;SET 'T' TRAP TO TIMER ROUTINE
1233 000016 000000 .WORD HALT ;PRIORITY LEVEL 7
1234 000020 002636 .WORD .TYPE ;SET IOT TRAP TO .TYPE ROUTINE
1235 000022 000000 .WORD 0 ;PRIORITY LEVEL 0
1236 000024 000026 .WORD PFVEC+2 ;POWER FAIL TRAP TO HALT
1237 000026 000000 .WORD HALT ;AT PFVEC+2
1238 000030 004516 .WORD .SCOPE ;SET EMT TRAP TO .SCOPE ROUTINE
1239 000032 000340 .WORD 340 ;PRIORITY LEVEL 7
1240 000034 004252 .WORD .HLT ;SET TRAP TRAP TO .HLT ROUTINE
1241 000036 000340 .WORD 340 ;PRIORITY LEVEL 7
1242
1243 ;ACT11 HOOK *****
1244 . 000040 $SVPC= ;SAVE CURRENT LOCATION CTR
1245 000046 000046 .=46
1246 000046 013462 .WORD $ENDAD ;SET LOCATION 46
1247 000052 000052 .=52
1248 000052 000000 .WORD 0 ;SET LOCATION 52 = 0
1249 000040 000040 .=$SVPC ;RESTORE LOCATION CTR
1250
1251 .=TKVEC
1252 000060 004126 .WORD TKISR
1253 000062 000200 .WORD 200
1254
1255 ;SOFTWARE SWITCH REGISTER LOC. 176
1256 000176 000176 .=176
1257 000176 000000 SWREG: .WORD 0 ;SOFTWARE SWITCH REGISTER
1258
1259 .=200
1260 000200 000137 006274 JMP @#INIT ;GO TO START OF PROGRAM
1261 000210 000210 .=210
1262 000210 000137 007372 JMP @#RSTRT ;RESTART ADDRESS
1263
1264 .=500
1265 000600 000600 STKPTR= 600 ;STACK
1266
1267 .=1000
1268 ;PROGRAM TAGS
1269 001000 177570 SWR: 177570 ;SWITCH REGISTER
1270 001002 000000 SCPADR: .WORD 0
1271 001004 000000 DRVNUM: .BYTE 0 ;TM03 DRIVE UNDER TEST
1272 001005 000000 SLVNUM: .BYTE 0 ;TE16/TU77 SLAVE UNDER TEST
1273 001006 000000 SLVPTR: .WORD 0 ;POINTER TO SLAVE TABLE (SLVTBL) BELOW
1274 001010 172440 TMBASE: .WORD TMCS1 ;BASE ADDRESS OF TM03-TE16/TU77 REGISTERS
1275 001012 000000 OSCTIM: .WORD 0 ;US/TICK (56/80 FOR TE16/TU77)
1276 001014 000000 GAPDEL: .WORD 0 ;TICKS/MS (18/6 FOR TE16/TU77)
1277 001016 000000 ATIME: .WORD 0 ;CONTAINS 'TICK' COUNT
1278 001020 000020 ATIMTBL: .BLKW 16. ;EACH ENTRY CONTAINS TIME FOR FUNCTION
1279 ;ENTRIES ARE MADE BY 'SCOPE' ROUTINE
1280 001060 000020 GAP TBL: .BLKW 16. ;TIMES RECORDED BY 'GAP CONSISTANCY' TEST
1281 001120 000000 DELTIM: .WORD 0 ;VARIABLE DELAY
1282 001122 000000 OCTALO: .WORD 0
1283 001124 000000 GAP: .BYTE 0 ;CONTAINS GAP # (USED FOR TST 021)
1284 001125 000000 ITCNT: .BYTE 0 ;ITERATION COUNT
1285 001126 000000 TSTNUM: .BYTE 0 ;TEST #
```

1286	001127	000	ERFLG:	.BYTE	0		:ERROR FLAG
1287	001130	000	SKEWFLG:	.BYTE	0		:0/1 = DO NOT/DO SKEW (SPEED) TESTS
1288	001131	000	PRGFLG:	.BYTE	0		:PROGRAM FLAG
1289	001132	000	UNTFND:	.BYTE	0		:UNIT FOUND INDICATOR
1290	001133	000	TYPFLG:	.BYTE	0		
1291	001134	000	PSCNT:	.BYTE	0		:CONTAINS PASS COUNT
1292	001135	000	ASFLG:	.BYTE	0		:1/0 = YES/NO.
1293	001136	000	TE16:	.BYTE	0		:0/1 = TE16/TU77
1294		001140		.EVEN			
1295	001140	030460	DIGTAB:	'01			
1296	001142	031462		'23			
1297	001144	032464		'45			
1298	001146	033466		'67			
1299	001150	034470		'89			
1300	001152	000006	ODIGITS:	.BLKB	6		:RESERVE SPACE FOR CONVERTED DIGITS
1301	001160	000		.BYTE	0		:TERMINATOR
1302		001162		.EVEN			
1303	001162	000010	DRVTBL:	.BLKB	8.		:A 0/-1 = DRIVE NOT TO BE/TO BE TESTED
1304	001172	000100	SLVTBL:	.BLKB	64.		:A 0/-1 = SLAVE NOT TO BE/TO BE TESTED
1305	001272	000110	INBUF:	.BLKB	72.		:TELETYPE INPUT BUFFER
1306	001402	005015	CRLF:	.ASCIZ	<CR><LF>		:MISCELLANEOUS ASCII CHARACTERS
1307	001405	134	BKSLSH:	.ASCIZ	'\'		
1308	001407	060	ECHO:	.ASCIZ	'0'		
1309	001411	007	BELL:	.ASCIZ	<7>		
1310	001413	055	DASH:	.ASCIZ	'-'		
1311	001415	040	SPACE2:	.ASCIZ	' '		
1312	001416	000040	SPACE:	.ASCIZ	' '		
1313	001420	004476	ANGTAB:	.ASCIZ	'>'<HT>		
1314		001424		.EVEN			

1315
 1316
 1317
 1318
 1319
 1320
 1321
 1322 001424 000000 000000
 1323 001430 036050 035230
 1324 001434 001666 001546
 1325 001440 001522 001356
 1326 001444 002506 001332
 1327 001450 007164 006344
 1328 001454 000500 000360
 1329 001460 000632 000454
 1330 001464 002506 001332
 1331 001470 000500 000360
 1332 001474 000562 000512
 1333 001500 002506 001332
 1334 001504 003206 002056
 1335 001510 003206 002056
 1336 001514 002412 001666
 1337 001520 002234 001522
 1338 001524 002626 002354
 1339 001530 002570 002234
 1340 001534 004540 004230
 1341 001540 004716 004552
 1342 001544 023564 023110
 1343 001550 024240 023730
 1344 001554 004336 004172
 1345 001560 004336 004172
 1346
 1347
 1348
 1349
 1350
 1351
 1352
 1353
 1354
 1355 001564 002602 002412
 1356 001570 002652 002506
 1357 001574 002734 002532
 1358 001600 003016 002424
 1359 001604 003016 002304
 1360 001610 002734 002176
 1361 001614 002652 002176
 1362 001620 002652 002176
 1363 001624 002570 002176
 1364 001630 002570 002176
 1365 001634 002570 002176
 1366 001640 002570 002176
 1367 001644 002570 002176
 1368 001650 002570 002176
 1369 001654 002570 002176
 1370 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	_AT 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

1371
1372
1373
1374
1375
1376
1377
1378 001664 000000 000000
1379 001670 012606 011571
1380 001674 000520 000460
1381 001700 000346 000313
1382 001704 003410 001717
1383 001710 001315 001112
1384 001714 000111 000067
1385 001720 000111 000067
1386 001724 003410 001717
1387 001730 000111 000067
1388 001734 000101 000057
1389 001740 003410 001717
1390 001744 003500 002006
1391 001750 003510 002017
1392 001754 000725 000561
1393 001760 000620 000464
1394 001764 001012 000646
1395 001770 001034 000641
1396 001774 001515 001434
1397 002000 001536 001434
1398 002004 007020 006476
1399 002010 007176 006642
1400 002014 001560 001320
1401 002020 001560 001320
1402
1403
1404
1405
1406
1407
1408
1409
1410
1411 002024 000770 000676
1412 002030 000113 000717
1413 002034 001033 000732
1414 002040 001050 000742
1415 002044 001062 000746
1416 002050 001067 000742
1417 002054 001072 000736
1418 002060 001072 000724
1419 002064 001066 000704
1420 002070 001065 000660
1421 002074 001046 000627
1422 002100 001027 000572
1423 002104 000776 000632
1424 002110 000776 000632
1425 002114 000776 000632
1426 002120 000776 000632

.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	0469.,00369.	:4.69-3.69	GAP SIZE STOP	TST016
.WORD	0400.,00308.	:4.00-3.08	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.

: .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

1427
 1428
 1429
 1430
 1431
 1432
 1433 002124
 1434 002124 000000 000000
 1435 002130 036050 035230
 1436 002134 001666 001546
 1437 002140 001522 001356
 1438 002144 002506 001332
 1439 002150 007164 006344
 1440 002154 000500 000360
 1441 002160 000632 000454
 1442 002164 002506 001332
 1443 002170 000500 000360
 1444 002174 000562 000512
 1445 002200 002506 001332
 1446 002204 003206 002056
 1447 002210 003206 002056
 1448 002214 002412 001666
 1449 002220 002234 001522
 1450 002224 002626 002354
 1451 002230 002506 002234
 1452 002234 004540 004230
 1453 002240 004716 004552
 1454 002244 023564 023110
 1455 002250 024240 023730
 1456 002254 004336 004172
 1457 002260 004336 004172
 1458
 1459
 1460
 1461
 1462
 1463
 1464
 1465 002264 002544 002354
 1466 002270 002652 002506
 1467 002274 002722 002506
 1468 002300 002710 002474
 1469 002304 002570 002260
 1470 002310 002556 002114
 1471 002314 002532 002114
 1472 002320 002532 002114
 1473 002324 002506 002176
 1474 002330 002474 002176
 1475 002334 002474 002176
 1476 002340 002474 002176
 1477 002344 002474 002176
 1478 002350 002474 002176
 1479 002354 002474 002176
 1480 002360 002474 002176
 1481 002364

.SBTTL 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
 :
 :.WORD MAX,MIN :TIME IN MS FUNCTION TEST #
 :STTBL:
 :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.

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

1482
1483
1484 002364 015656
1485 002366 015706
1486 002370 015730
1487 002372 015750
1488 002374 015772
1489 002376 016016
1490 002400 016040
1491 002402 016057
1492 002404 016101
1493 002406 016124
1494 002410 016146
1495 002412 016173
1496 002414 016222
1497 002416 016253
1498 002420 016304
1499 002422 016332
1500 002424 016361
1501 002426 016411
1502 002430 016434
1503 002432 016460
1504 002434 016505
1505 002436 016527
1506 002440 016552
1507 002442 016574
1508
1509
1510 002444 007756
1511 002446 010252
1512 002450 010336
1513 002452 010414
1514 002454 010504
1515 002456 010612
1516 002460 010676
1517 002462 010762
1518 002464 011062
1519 002466 011202
1520 002470 011300
1521 002472 011410
1522 002474 011550
1523 002476 011642
1524 002500 011750
1525 002502 012044
1526 002504 012154
1527 002506 012274
1528 002510 012600
1529 002512 012730
1530 002514 013060
1531 002516 013200
1532 002520 013534
1533 002522 013672

.SBTTL 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

TSTTBL: .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


```

1590 002712 012600          TYPE3: MOV      (SP)+,R0          ;;RESTORE R0
1591 002714 000002          RTI                          ;;RETURN TO CALLER
1592
1593 002716 122716 000011  TYPE4: CMPB     #5HT,(SP)          ;;BRANCH IF HORIZONTAL TAB <HT>
1594 002722 001445          BEQ      9$
1595 002724 004737 002756  JSR      PC,5$          ;;TYPE CHARACTER
1596 002730 122726 000012  3$:  CMPB     #12,(SP)+          ;;CHECK IF CHARACTER WAS A LINE FEED
1597 002734 001350          BNE      TYPE1          ;;BRANCH IF NOT LINE FEED
1598 002736 013746 002616  MOV      $NULL,-(SP)      ;;GET # OF FILLERS REQUIRED AND FILLER
1599                                ;;CHARACTER.
1600
1601 002742 105366 000001  4$:  DECB     1(SP)          ;;DECREMENT FILLERS REQ. COUNT
1602 002746 002770          BLT      3$            ;;BRANCH IF NO MORE FILLERS ARE REQUIRED
1603 002750 004737 002756  JSR      PC,5$          ;;TYPE FILLER CHARACTER
1604 002754 000772          BR       4$
1605
1606 002756 105777 177640  5$:  TSTB     @5TPS          ;;WAIT FOR OUTPUT DEVICE
1607 002762 100375          BPL     .-4
1608 002764 122737 000017 002627  CMPB     #17,@5CNTRLO      ;;CHECK IF CONTROL O WAS TYPED
1609 002772 001403          BEQ     6$            ;;STOP TYPING MESSAGE IF ^O WAS TYPED
1610 002774 116677 000002 177622  MOVB     2(SP),@5TPB      ;;OUTPUT CHARACTER
1611 003002 122766 000015 000002  6$:  CMPB     #15,2(SP)          ;;BRANCH IF NOT <CR>
1612 003010 001003          BNE     7$
1613 003012 105037 002626  CLRB     $CHARCNT          ;;CLEAR CHARACTERS TYPED COUNT
1614 003016 000406          BR      8$
1615 003020 122766 000012 000002  7$:  CMPB     #12,2(SP)          ;;BRANCH IF <LF> OR 'NULL'
1616 003026 002002          BGE     8$
1617 003030 105237 002626  INCB     $CHARCNT          ;;INCREMENT CHARACTER TYPED COUNT
1618 003034 000207          RTS     PC
1619
1620                                ;;HORIZONTAL TAB <HT> PROCESSER
1621 003036 112716 000040  9$:  MOVB     #40,(SP)          ;;LOAD 'SPACE'
1622 003042 004737 002756 10$:  JSR      PC,5$          ;;TYPE 'SPACE'
1623 003046 132737 000007 002626  BITB     #7,$CHARCNT      ;;TYPE SPACES UNTIL A MULTIPLE
1624 003054 001372          BNE     10$           ;;OF 8 CHARACTERS HAVE BEEN TYPED
1625 003056 105726          TSTB     (SP)+          ;;POP SPACE
1626 003060 000676          BR      TYPE1          ;;GET NEXT CHARACTER
1627
1628                                ;SUBROUTINE TO SAVE GENERAL REGISTERS ON THE STACK
1629                                ;CALL:  SAVE
1630 003062 010546          .SAVE:  MOV      R5,-(SP)          ;SAVE REGISTERS ON THE STACK
1631 003064 010446          MOV      R4,-(SP)
1632 003066 010346          MOV      R3,-(SP)
1633 003070 010246          MOV      R2,-(SP)
1634 003072 010146          MOV      R1,-(SP)
1635 003074 010046          MOV      R0,-(SP)
1636 003076 016646 000014          MOV      14(SP),-(SP)          ;GET RETURN PC
1637 003102 000207          RTS     PC              ;RETURN
1638
1639                                ;SUBROUTINE TO RESTORE GENERAL REGISTERS FROM THE STACK
1640                                ;CALL:  RESTORE
1641 003104 012666 000014          .RESTORE:MOV (SP)+,14(SP)          ;MOVE RETURN PC
1642 003110 012600          MOV      (SP)+,R0          ;RESTORE REGISTERS
1643 003112 012601          MOV      (SP)+,R1
1644 003114 012602          MOV      (SP)+,R2
1645 003116 012603          MOV      (SP)+,R3
  
```



```
1646 003120 012604      MOV    (SP)+,R4
1647 003122 012605      MOV    (SP)+,R5
1648 003124 000207      RTS    PC                ;RETURN
1649
1650      ;SUBROUTINE TO CONVERT OCTAL DATA TO ASCII
1651      ;CALL: MOV    NUMBER,R2      ;MOVE NUMBER TO R2
1652      ;      JSR    PC,CNVTOCT
1653
1654 003126 110637 001133  CNVTOCT: MOV    SP,TYPFLG      ;SET DO NOT TYPE FLAG
1655 003132 000402      BR    CNVTO
1656
1657      .SBTTL      OCTAL TO ASCII & TYPE ROUTINE
1658      ;SUBROUTINE TO CONVERT OCTAL NUMBER TO ASCII AND TYPE IT OUT
1659      ;CALL: MOV    NUMBER,R2      ;PUT # IN R2
1660      ;      JSR    PC,TYPOCT      ;CALL ROUTINE
1661
1662 003134 105037 001133  TYPOCT: CLR    @#TYPFLG      ;SET TYPE FLAG
1663 003140  CNVTO:
1664 003140 004737 003062      JSR    PC,SAVE            ;SAVE REGISTERS ON THE STACK
1665 003144 012704 001152      MOV    #ODIGITS,R4      ;SET PTR TO OUTPUT
1666 003150 005003      CLR    R3              ;R3 WILL CONTAIN OCTAL DIGIT
1667 003152 010201      MOV    R2,R1            ;GET # TO BE TYPED
1668 003154 006302 1$: ASL    R2              ;SHIFT #
1669 003156 006103      ROL    R3
1670 003160 012700 000006      MOV    #6,R0            ;SET DIGIT COUNTER
1671 003164 000404      BR    3$
1672
1673 003166 006302 2$: ASL    R2              ;SHIFT # 3 PLACES LEFT
1674 003170 006103      ROL    R3
1675 003172 005301      DEC    R1
1676 003174 001374      BNE    2$
1677 003176 012701 000003 3$: MOV    #3,R1            ;SET SHIFT COUNTER
1678 003202 116324 001140      MOV    DIGTAB(R3),(R4)+ ;MOVE ASCII EQUIV TO OUTPUT
1679 003206 005003      CLR    R3
1680 003210 005300      DEC    R0            ;DECREMENT DIGIT COUNT
1681 003212 001365      BNE    2$              ;GET NEXT DIGIT
1682 003214 105737 001133      TST    @#TYPFLG        ;BRANCH IF ASCII IS
1683 003220 001002      BNE    4$              ;NOT TO BE TYPED
1684 003222 000004 001152      TYPE,ODIGITS
1685 003226 4$:
1686 003226 004737 003104      JSR    PC,RESTORE      ;RESTORE REGISTERS FROM THE STACK
1687 003232 000207      RTS    PC
1688
1689
1690      ;SUBROUTINE TO CONVERT OCTAL DATA TO DECIMAL ASCII
1691      ;CALL: MOV    NUMBER,R2      ;MOVE NUMBER TO R2
1692      ;      JSR    PC,CNVDEC
1693
1694 003234 110637 001133  CNVDEC: MOV    SP,@#TYPFLG    ;SET DO NOT TYPE FLAG
1695 003240 000402      BR    CNVTD
1696
1697      .SBTTL      OCTAL TO DECIMAL & TYPE ROUTINE
1698      ;THIS ROUTINE CONVERTS AN OCTAL # TO DECIMAL ASCII AND TYPES IT OUT
1699      ;CALL: MOV    NUMBER,R2      ;PUT # IN R2
1700      ;      JSR    PC,TYPDEC      ;CALL ROUTINE
1701 003242 105037 001133  TYPDEC: CLR    @#TYPFLG      ;SET TYPE FLAG
```

1702 003246
 1703 003246 004737 003062
 1704 003252 005000
 1705 003254 012704 001152
 1706 003260 005003
 1707 003262 166002 003342
 1708 003266 103402
 1709 003270 005203
 1710 003272 000773
 1711 003274 066002 003342
 1712 003300 116324 001140
 1713 003304 062700 000002
 1714 003310 005760 003342
 1715 003314 001361
 1716 003316 112724 000060
 1717 003322 105737 001133
 1718 003326 001002
 1719 003330 000004 001152
 1720 003334
 1721 003334 004737 003104
 1722 003340 000207
 1723
 1724 003342 023420
 1725 003344 001750
 1726 003346 000144
 1727 003350 000012
 1728 003352 000001
 1729 003354 000000
 1730
 1731
 1732
 1733
 1734
 1735
 1736
 1737
 1738
 1739
 1740
 1741
 1742 003356 010246
 1743 003360 010346
 1744 003362 006302
 1745 003364 006302
 1746 003366 010203
 1747 003370 000004 015636
 1748 003374 016302 002124
 1749 003400 004737 003242
 1750 003404 000004 001413
 1751 003410 016302 002126
 1752 003414 004737 003242
 1753 003420 000004 001420
 1754 003424 000004 015646
 1755 003430 013702 001016
 1756 003434 004737 003242
 1757 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$: TYPE, ODIGITS

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)).
; CCCCC 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
  
```

1758 003444 012603
 1759 003446 012602
 1760 003450 000207
 1761
 1762
 1763
 1764
 1765
 1766
 1767
 1768
 1769
 1770 003452 010246
 1771 003454 010346
 1772 003456 113703 001124
 1773 003462 006303
 1774 003464 006303
 1775 003466 000004 015636
 1776 003472 016302 002264
 1777 003476 004737 003242
 1778 003502 000004 001413
 1779 003506 016302 002266
 1780 003512 004737 003242
 1781 003516 000004 001420
 1782 003522 000004 015646
 1783 003526 013702 001016
 1784 003532 004737 003242
 1785 003536 000004 015313
 1786 003542 113702 001124
 1787 003546 004737 003134
 1788 003552 000004 001402
 1789 003556 012603
 1790 003560 012602
 1791 003562 000207
 1792
 1793
 1794
 1795
 1796 003564
 1797 003564 004737 003062
 1798 003570 012700 001272
 1799 003574 012701 001122
 1800 003600 005011
 1801 003602 005061 000002
 1802 003606 122710 000015
 1803 003612 001414
 1804 003614 112002
 1805 003616 042702 177770
 1806 003622 012702 000003
 1807 003626 006311
 1808 003630 006161 000002
 1809 003634 005303
 1810 003636 001373
 1811 003640 050211
 1812 003642 000761
 1813 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: MOV #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)
MOV #GAP,R3 ;GET GAP #
ASL R3
ASL R3
TYPE,L.RNG
MOV GTIMTBL(R3),R2 ;GET MAX TIME
JSR PC,TYPDEC ;CONVERT TO DECIMAL & TYPE
TYPE,DASH
MOV GTIMTBL+2(R3),R2 ;GET MIN TIME
JSR PC,TYPDEC ;CONVERT TO DECIMAL & TYPE
TYPE,ANGTAB ;TYPE <
TYPE,L.ACT
MOV @#ATIME,R2 ;GET ACTUAL TIME
JSR PC,TYPDEC ;CONVERT TO DECIMAL & TYPE
TYPE,E.GAP
MOV #GAP,R2 ;GET C-P #
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$: CMPB #CR,(R0) ;<CR> TERMINATES INPUT
BEQ 3$
MOV (R0)+,R2 ;GET 'OCTAL' DATA
BIC #177770,R2 ;STRIP UNUSED BITS
MOV #3,R3 ;SET SHIFT COUNT
2$: ASL (R1) ;SHIFT LAST
ROL 2(R1) ;OCTAL DIGIT
DEC R3
BNE 2$
BIS R2,(R1) ;AND INSERT THIS DIGIT
BR 1$ ;GO GET NEXT DIGIT
3$:

```

1814 003644 004737 003104
 1815 003650 000207

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

1816
 1817
 1818
 1819
 1820
 1821
 1822

.SBTTL PUBLISH SUBROUTINE
 ;THE PUBLISH SUBROUTINE AVERAGES THE RECORDED TIMES FOR EACH 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 .

1823 003652
 1824 003652 004737 003062
 1825 003656 012700 001020
 1826 003662 113701 001125
 1827 003666 122701 000001
 1828 003672 001423
 1829 003674 005002
 1830 003676 005003
 1831 003700 122701 000020
 1832 003704 001402
 1833 003706 000000
 1834 003710 000777

PUBLISH:
 JSR PC,SAVE ;SAVE REGISTERS ON THE STACK
 MOV #ATIMTBL,R0 ;GET TABLE ADDRESS CONTAINING TIMES
 MOV #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.

1835
 1836
 1837 003712 062002
 1838 003714 005503
 1839 003716 005301
 1840 003720 001374

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

1841
 1842 003722 012700 000004
 1843 003726 006203
 1844 003730 006002
 1845 003732 005300
 1846 003734 001374
 1847 003736 010237 001016

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

1848
 1849 003742 113700 001126
 1850 003746 006300
 1851 003750 016037 002364 003760
 1852 003756 000004
 1853 003760 000000
 1854 003762 113702 001126
 1855 003766 004737 003356
 1856 003772 004737 003104
 1857 003776 000207

4\$: MOV #ATSTNUM,R0 ;GET TEST #
 ASL R0
 MOV #NAMPTR(R0),5\$;GET TEST NAME STRING ADDRESS
 TYPE
 5\$: .WORD 0
 MOV #ATSTNUM,R2 ;GET TEST #
 JSR PC,OUTSPC ;OUTPUT TIMES
 JSR PC,RESTORE ;RESTORE REGISTERS FROM THE STACK
 RTS PC

1858
 1859
 1860
 1861
 1862
 1863

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

1864 004000 010046
 1865 004002 012700 001272
 1866 004006 105737 177560
 1867 004012 100375
 1868
 1869 004014 113746 177562

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

```

1870 004020 042716 000200      BIC      #200,(SP)
1871 004024 122716 000177      CMPB     #177,(SP)      ;CHECK RUBOUT
1872 004030 001004              BNE      3$
1873 004032 124026              CMPB     -(RO),(SP)+   ;REMOVE CHARACTER FROM INPUT
1874 004034 000004 001405      TYPE,BKSLSH
1875 004040 000762              BR       2$            ;WAIT FOR NEXT CHARACTER
1876 004042 122716 000025      3$:    CMPB     #CNTRLU,(SP) ;CHECK CONTROL U (^U)
1877 004046 001004              BNE      4$
1878 004050 005726              TST     (SP)+
1879 004052 000004 001402      TYPE,CRLF
1880 004056 000751              BR       1$
1881 004060 122716 000003      4$:    CMPB     #CNTRLC,(SP) ;BRANCH IF NOT CONTROL C
1882 004064 001003              BNE      40$
1883 004066 000005              RESET
1884 004070 000137 006274      JMP      @#INIT       ;RESTART PROGRAM
1885 004074 111637 001407      40$:   MOVB     (SP),@#ECHO
1886 004100 111620              MOVB     (SP),(RO)+
1887 004102 122726 000015      CMPB     #CR,(SP)+
1888 004106 001403              BEQ      5$
1889 004110 000004 001407      TYPE,ECHO
1890 004114 000734              BR       2$
1891 004116 000004 001402      5$:    TYPE,CRLF
1892 004122 012600              MOV      (SP)+,RO
1893 004124 000207              RTS      PC

;KEYBOARD INTERRUPT SERVICE ROUTINE
1896 004126 113746 177562      TKISR:  MOVB     @#TKB,-(SP) ;GET TYPED CHARACTER
1897 004132 042716 000200      BIC      #200,(SP)      ;STRIP PARITY BIT
1898 004136 122716 000017      CMPB     #CNTRLO,(SP)   ;BRANCH IF NOT CONTROL O (^O)
1899 004142 001002              BNE      1$
1900 004144 111637 002627      MOVB     (SP),#CNTRLO  ;SET CONTROL O INDICATOR IN TYPE ROUTINE
1901
1902 004150 122716 000003      1$:    CMPB     #3,(SP)      ;BRANCH IF NOT CONTROL C (^C)
1903 004154 001007              BNE      2$
1904 004156 023727 000042 013462      CMP      @#42,#$ENDAD  ;INHIBIT ^C IF ACT11 QV OR AA
1905 004164 001403              BEQ      2$
1906 004166 000005              RESET
1907 004170 000137 006274      JMP      @#INIT       ;RESTART PROGRAM
1908
1909 004174 122716 000001      2$:    CMPB     #CNTRLA,(SP)   ;BRANCH IF NOT ^A
1910 004200 001011              BNE      3$
1911 004202 022737 000176 001000      CMP      #SWREG,SWR    ;BRANCH IF HARDWARE SWR IS INVOKED
1912 004210 001010              BNE      4$
1913 004212 012737 177570 001000      MOV      #177570,SWR   ;INVOKE HARDWARE SWR
1914 004220 000004 014272      TYPE,M.HSWR
1915 004224 122716 000007      3$:    CMPB     #CNTRLG,(SP)   ;BRANCH IF NOT ^G
1916 004230 001005              BNE      5$
1917 004232 012737 000176 001000      4$:    MOV      #SWREG,SWR   ;INVOKE SOFTWARE SWR
1918 004240 004737 002526      JSR      PC,GTSWR     ;GET NEW SWITCH REGISTER
1919 004244 005726              5$:    TST     (SP)+        ;POP CHARACTER OFF THE STACK
1920 004246 000002              RTI                    ;RETURN TO CALLER
  
```

```

1921          .SBTTL          ERROR SERVICE ROUTINES
1922          ;ROUTINE TO PROCESS ERROR TRAPS (TRAPS TO 4)
1923 004250 000000  ERRTRP: HALT
1924
1925          ;ERROR SERVICE ROUTINE
1926          ;THIS ROUTINE PROCESSES TWO TYPES OF ERRORS (OUT OF RANGE AND HARDWARE)
1927          ;THE CALLS FOR AN OUT OF RANGE ERROR ARE <HLT+1>,<HLT+2> AND, FOR A
1928          ;HARDWARE ERROR THE CALL IS <HLT>.
1929
1930 004252 004737 003062          .HLT:  JSR      PC,,SAVE          ;SAVE REGISTERS ON THE STACK
1931 004256 110637 001127 1$:    MOVB    SP,@#ERFLG          ;SET ERROR FLAG
1932 004262 032777 020000 174510  BIT     #SW13,@SWR          ;BRANCH IF NO TYP0UT
1933 004270 001075          BNE     4$
1934 004272 000004 015114          TYPE,E.HDR
1935 004276 113702 001126          MOVB    @#TSTNUM,R2          ;GET TEST #
1936 004302 004737 003134          JSR     PC,TYP0CT          ;AND TYPE IT
1937 004306 016600 000016          MOV     16(SP),R0          ;GET RETURN PC
1938 004312 162700 000002          SUB     #2,R0              ;NOW PC OF HLT CALL
1939 004316 111000          MOVB    (R0),R0            ;NOW HLT CALL ITSELF
1940 004320 001417          BEQ    2$                  ;BRANCH IF HLT
1941 004322 000004 015177          TYPE,E.HDR2
1942 004326 122700 000002          CMPB   #2,R0              ;BRANCH IF NOT HLT+2
1943 004332 001005          BNE    10$
1944 004334 004737 003452          JSR     PC,OUTGAP          ;TYPE GAP SPECIFIED TIMES
1945 004340 000004 001402          TYPE,CRLF
1946 004344 000447          BR     4$
1947 004346 004737 003356 10$:   JSR     PC,OUTSPC          ;TYPE SPECIFIED TIMES
1948 004352 000004 001402          TYPE,CRLF
1949 004356 000442          BR     4$
1950 004360 016500 000014 2$:    MOV     ER(R5),R0
1951 004364 032765 002300 000032  BIT     #PE1600,TC(R5)
1952 004372 001403          BEQ    20$
1953 004374 042700 102100          BIC    #102100,R0
1954 004400 000402          BR     21$
1955 004402 042700 102300 20$:   BIC    #102300,R0
1956 004406 005700 21$:   TST    R0
1957 004410 001003          BNE    22$
1958 004412 000004 015070          TYPE,E.SFT                ;TYPE SOFT ERROR MESSAGE
1959 004416 000434          BR     6$
1960
1961 004420 000004 015124 22$:   TYPE,E.HDR1
1962 004424 010500          MOV     R5,R0              ;GET FIRST ADDRESS OF REGS.
1963 004426 012701 000007          MOV     #7,R1              ;TYPE FIRST 7 REGS.
1964 004432 012002 3$:    MOV     (R0)+,R2            ;GET REG CONTENTS
1965 004434 004737 003134          JSR     PC,TYP0CT          ;AND TYPE IT
1966 004440 000004 001415          TYPE,SPACE2
1967 004444 005301          DEC     R1
1968 004446 001371          BNE    3$
1969 004450 016502 000032          MOV     TC(R5),R2          ;GET CONTENTS OF TC REGISTER
1970 004454 004737 003134          JSR     PC,TYP0CT
1971 004460 000004 001402          TYPE,CRLF
1972
1973 004464 032777 001000 174306 4$:   BIT     #SW09,@SWR          ;BRANCH IF NO RING THE BELL
1974 004472 001402          BEQ    5$
1975 004474 000004 001411          TYPE,BELL
1976 004500 005777 174274 5$:   TST    @SWR                ;HALT ON ERROR?

```

CZTEEB0 TMO3-TE16/TU77 DFT
CZTEEB.P11 17-NOV-78 10:11

MACY11 30A(1052) 21-DEC-78 13:20 PAGE 46
ERROR SERVICE ROUTINES

H 4

SEQ 0046

1977 004504 100001
1978 004506 000000
1979 004510
1980 004510 004737 003104
1981 004514 000002
1982
1983

6\$: BPL 6\$
HALT
JSR PC,RESTORE
RTI

:RESTORE REGISTERS FROM THE STACK
:RETURN

```
1984 .SBTTL SCOPE SUBROUTINE
1985 :SCOPE ROUTINE
1986 :THIS ROUTINE IS ENTERED UPON COMPLETION OF EACH SUBTEST
1987 :THE SCOPE ROUTINE:
1988 : REPEATS TEST IF SW14 IS SET
1989 : STORES ACTUAL TIME FOR FUNCTION IN TIME TABLE (ATIMTBL)
1990 : PUBLISHES TIME IF SW10=0
1991 : UPDATES ITERATION COUNT AND IF ITERATIONS COMPLETE CONTINUES
1992 : TO NEXT TEST, OTHERWISE REPEATS TEST.
1993 : DELAYS BEFORE CONTINUING OR REPEATING TEST.
1994 : INITIALIZES DRIVE
1995 :RETURNS: R5=BASE ADDRESS OF TMO3 REGISTERS (ADDRESS OF CS1)
1996 : R1='DS' REG ADDRESS
1997 : R0='FC' REG ADDRESS
1998
1999 004516 013705 001010 .SCOPE: MOV @TMBASE,R5 ;SET R5 TO FIRST TM REG
2000 004522 032777 040000 174250 BIT #SW14,@SWR ;BRANCH IF CONTINUOUS LOOP
2001 004530 001432 BEQ 2$ ;NOT DESIRED
2002 004532 017701 174242 1$: MOV @SWR,R1 ;GET SWITCHES
2003 004536 042701 177740 BIC #177740,R1 ;CLEAR ALL BUT TEST #
2004 004542 001406 BEQ 11$ ;BRANCH IF ALL SELECTED
2005 004544 120137 001126 CMPB R1,@TSTNUM ;BRANCH IF RUNNING SELECTED TEST
2006 004550 001403 BEQ 11$
2007 004552 012737 007756 001002 MOV #TST000,SCPADR ;RESTART AT TST000
2008 004560 004737 005310 11$: JSR PC,DELAY ;DELAY 350 MS
2009 004564 004737 005544 JSR PC,RHINIT ;INIT
2010 004570 105037 001127 CLR B @WERFLG ;CLEAR ERROR FLAG
2011 004574 013716 001002 MOV SCPADR,(SP)
2012 004600 010501 MOV R5,R1
2013 004602 062701 000012 ADD #DS,R1 ;ADDRESS OF 'DS' REG IS IN R1
2014 004606 010500 MOV R5,R0
2015 004610 062700 000006 ADD #FC,R0 ;ADDRESS OF 'FC' REG IS IN R0
2016 004614 000002 RTI
2017
2018 004616 105737 001127 2$: TSTB @WERFLG ;BRANCH IF ERROR FLAG IS SET
2019 004622 001006 BNE 3$
2020 004624 113700 001125 MOV B @ITCNT,RO ;GET ITERATION COUNT
2021 004630 006300 ASL RO ;STORE TIME IN TABLE
2022 004632 013760 001016 001020 MOV @ATIME,ATIMTBL(RO)
2023 004640 105237 001125 3$: INCB @ITCNT ;INCREMENT ITERATION COUNT
2024 004644 105737 001134 TSTB @PSCNT ;INHIBIT ITERATIONS ON
2025 004650 001410 BEQ 4$ ;ON FIRST PASS
2026 004652 032777 004000 174120 BIT #SW11,@SWR ;BRANCH IF SINGLE ITERATION DESIRED
2027 004660 001004 BNE 4$
2028 004662 122737 000020 001125 CMPB #16,@ITCNT ;BRANCH IF ITERATIONS INCOMPLETE
2029 004670 001320 BNE 1$
2030 004672 032777 000037 174100 4$: BIT #37,@SWR ;IF TEST SELECTED IS TEST 0
2031 004700 001002 BNE 42$ ;TREAT AS ALL TESTS
2032 004702 011637 001002 40$: MOV (SP),@SCPADR ;SET SCOPE ADDRESS TO NEXT TEST
2033 004706 032777 002000 174064 42$: BIT #SW10,@SWR ;BRANCH IF NO PUBLICATION DESIRED
2034 004714 001002 BNE 5$
2035 004716 004737 003652 JSR PC,PUBLISH ;GO PUBLISH TEST DATA
2036 004722 105037 001125 5$: CLR B @ITCNT ;RESET ITERATION COUNT
2037 004726 000701 BR 1$
2038
2039 .SBTTL TIMER SUBROUTINES
```



```

2040
2041 ;SUBROUTINE TO SYNCHRONIZE THE TIMER AND TURN IT ON.
2042 ;REGISTER 4 IS CLEARED, AND THE OSCILLATOR POLARITY IS MONITORED
2043 ;THE ROUTINE IS EXITED WHEN THE OSCILLATOR POLARITY CHANGES WITH R3
2044 ;SET TO INDICATE THE POLARITY OF THE OSCILLATOR.
2045 ;CALL: JSR PC,TIMON
2046 ;RETURNS: R3 SET TO INDICATE LAST POLARITY (+24/-24=0/1)
2047 ; R4 = 0
2048
2049 004730 005004 TIMON: CLR R4 ;CLEAR TIME COUNT
2050 004732 012703 000024 MOV #24,R3 ;SET POLARITY TO '0' STATE
2051 004736 032765 000100 000024 BIT #OSC,MR(R5) ;BRANCH IF POLARITY IS '0'
2052 004744 001405 BEQ 2$
2053 004746 032765 000100 000024 1$: BIT #OSC,MR(R5) ;WAIT FOR OSCILLATOR TO RETURN
2054 004754 001374 BNE 1$
2055 004756 000405 BR 4$
2056
2057 004760 005403 2$: NEG R3 ;NEGATE PREV POLARITY INDICATOR
2058 004762 032765 000100 000024 3$: BIT #OSC,MR(R5) ;WAIT FOR OSCILLATOR TO RETURN
2059 004770 001774 BEQ 3$ ;TO '1' STATE
2060 004772 000207 4$: RTS PC
2061
2062 ;SUBROUTINE TO COUNT TIME
2063 ;EACH TIME THE OSCILLATOR TOGGLES (BIT <06> IN MR REG) REGISTER
2064 ;R4 IS INCREMENTED, AND THE REGISTER R3 IS NEGATED TO INDICATE
2065 ;THE LAST STATE OF THE OSCILLATOR.
2066 ;CALL JMP TIMER(R3) ;R3 IS SET BY TIMON ROUTINE
2067 ; R2=RETURN ADDRESS TO CALLER
2068 ;NOTE: THE TIME TO EXECUTE THIS ROUTINE IS VERY CRITICAL. IT MUST BE
2069 ;LESS THAN 40 US.
2070
2071 ;ENTER HERE VIA JMP TIMER(R3) WHEN R3--24 (PREV STATE-1)
2072 004774 032765 000100 000024 TIMER1: BIT #OSC,MR(R5) ;BRANCH IF CURRENT STATE IS '0'
2073 005002 001406 BEQ TIMFR ;GO INCREMENT TIME
2074 005004 000112 JMP (R2) ;RETURN TO TEST
2075
2076 .TIMER1+24
2077 005020 005403 TIMER: NEG R3 ;NEGATE PREV STATE INDICATOR
2078 005022 005204 INC R4 ;INCREMENT 'TICK' COUNT
2079 005024 100401 BMI TIMERR ;BRANCH ON OVERFLOW
2080 005026 000112 JMP (R2) ;RETURN TO TEST
2081 005030 000004 015225 TIMERR: TYPE,E,TIMOV ;TYPE 'TIMER OVERFLOWED'
2082 005034 104400 HLT ;REPORT HARDWARE ERROR
2083 005036 000177 173740 JMP @SCPADR ;RETURN TO BEGINNING OF TEST
2084
2085 .TIMER+24
2086 ;ENTER HERE VIA JMP TIMER(R3) WHEN R3+24 (PREV STATE=0)
2087 005044 032765 000100 000024 TIMERO: BIT #OSC,MR(R5) ;BRANCH IF CURRENT STATE = '1'
2088 005052 001362 BNE TIMER
2089 005054 000112 JMP (R2)
2090
2091 ;SUBROUTINE TO CHECK TIME RECORDED BY SUBTEST.
2092 ;THIS SUBROUTINE COMPUTES THE ACTUAL TIME (IN MICROSECONDS) AND CHECKS
2093 ;THAT THE TIME RECORDED BY THE SUBTEST IS CORRECT BY COMPARING THE TIME
2094 ;WITH THE HIGH LIMIT (STIMTBL(R0)) AND THE LOW LIMIT (STIMTBL+2(R0)).
2095 ;IF THE TIME IS OUT OF RANGE AN OUT OF RANGE ERROR TYPEOUT RESULTS.

```

```

2096 ;THE SUBROUTINE IS ENTERED WITH:
2097 ;   R4=TICK COUNT
2098
2099 TIMOK:
2100 005056 004737 003062 JSR PC,SAVE ;SAVE REGISTERS ON THE STACK
2101 005062 013700 001012 MOV @OSCTIM,R0 ;GET TIME PER TICK
2102 005066 010401 MOV R4,R1 ;GET TICKS COUNT
2103 005070 005002 CLR R2 ;CLEAR SUMMING REGISTERS
2104 005072 005003 CLR R3
2105 005074 060002 1$: ADD R0,R2 ;MULTIPLY TIME PER TICK
2106 005076 005503 ADC R3 ;BY TICK COUNT
2107 005100 005301 DEC R1
2108 005102 001374 BNE 1$
2109 005104 010246 MOV R2,-(SP) ;DIVIDE COUNT BY 10.
2110
2111 005106 010346 MOV R3,-(SP)
2112 005110 012746 000012 MOV #10,-(SP)
2113 005114 004737 005376 JSR PC,DIVIDE
2114 005120 005726 TST (SP)+ ;DISCARD REMAINDER
2115 005122 012637 001016 MOV (SP)+,@ATIME ;STORE QUOTIENT
2116 005126 113700 001126 MOVB @TSTNUM,R0 ;GET TEST #
2117 005132 006300 ASL R0
2118 005134 006300 ASL R0
2119 005136 023760 001016 002124 CMP @ATIME,STIMTBL(R0) ;CHECK THAT TIME IS WITHIN
2120 005144 101004 BHI 2$ ;LIMITS SPECIFIED
2121 005146 023760 001016 002126 CMP @ATIME,STIMTBL+2(R0)
2122 005154 101001 BHI 3$
2123 005156 104401 2$: HLT+1 ;CALL ERROR ROUTINE
2124 005160 3$:
2125 005160 004737 003104 JSR PC,.RESTORE ;RESTORE REGISTERS FROM THE STACK
2126 005164 000207 RTS PC ;RETURN
2127
2128 ;SUBROUTINE TO CHECK INDIVIDUAL GAP TIMES (PRODUCED BY TST021)
2129 ;SUBROUTINE COMPUTES THE ACTUAL TIME (IN MICROSECONDS) AND CHECKS
2130 ;THAT THE GAP TIME RECORDED BY THE SUBTEST (TST021) BY COMPARING THE
2131 ;TIME WITH THE MAX LIMIT (GTIMTBL-GAPTBL(R1)) AND THE MIN LIMIT
2132 ;(GTIMTBL+2-GAPTBL(R1)).
2133 ;CALL: MOV #TICK COUNT,R4 ;R4 CONTAINS TICK COUNT
2134 ; MOVB #GAP,@GAP ;LOCATION GAP CONTAINS GAP #
2135 ; JSR PC,GAPOK
2136
2137 GAPOK:
2138 005166 004737 003062 JSR PC,SAVE ;SAVE REGISTERS ON THE STACK
2139 005172 013700 001012 MOV @OSCTIM,R0 ;GET TIME PER TICK
2140 005176 010401 MOV R4,R1 ;GET TICK COUNT
2141 005200 005002 CLR R2 ;CLEAR SUMMING REGISTERS
2142 005202 005003 CLR R3
2143 005204 060002 1$: ADD R0,R2 ;MULTIPLY TICK COUNT
2144 005206 005503 ADC R3 ;BY TIME PER TICK
2145 005210 005301 DEC R1
2146 005212 001374 BNE 1$
2147
2148 005214 010246 MOV R2,-(SP) ;DIVIDE TIME BY 10.
2149 005216 010346 MOV R3,-(SP)
2150 005220 012746 000012 MOV #1,-(SP)
2151 005224 004737 005376 JSR PC,DIVIDE

```

```

2152 005230 005726          TST      (SP)+          ;DISCARD REMAINDER
2153 005232 012637 001016    MOV      (SP)+,@#ATIME ;STORE QUOTIENT
2154 005236 113703 001124    MOVB    @#GAP,R3      ;GET GAP #
2155 005242 006303          ASL      R3            ;MULTPLY BY 4
2156 005244 006303          ASL      R3            ;TO GET AT TABLE ENTRY
2157 005246 023763 001016 002264  CMP     @#ATIME,GTIMTBL(R3) ;CHECK TIME (MAX)
2158 005254 101004          BHI      2$           ;
2159 005256 023763 001016 002266  CMP     @#ATIME,GTIMTBL+2(R3) ;CHECK TIME (MIN)
2160 005264 101001          BHI      3$           ;
2161 005266 104402          HLT+2          ;REPORT OUT OF RANGE ERROR
2162 005270 032777 002000 173502 3$:    BIT     #SW10,@SWR    ;BRANCH IF TIMES NOT WANTED
2163 005276 001001          BNE      100$        ;
2164 005300 000240          NOP              ;
2165
2166 005302          100$:
2167 005302 004737 003104    JSR     PC,.RESTORE  ;RESTORE REGISTERS FROM THE STACK
2168 005306 000207          RTS     PC          ;RETURN TO TEST
2169
2170          .SBTTL      DELAY SUBROUTINES
2171          ;THIS SUBROUTINE CAUSES A DELAY OF 115 MS.
2172 005310 004737 004730    DELAY: JSR     PC,TIMON
2173 005314 010246          MOV     R2,-(SP)    ;SAVE R2 ON THE STACK
2174 005316 012702 005326    MOV     #2$,R2      ;SET RETURN ADDRESS FOR TIMER
2175 005322          1$:
2176 005322 000163 005020    JMP     TIMER(R3)   ;GO TO TIMER & RETURN VIA R2
2177 005326 032704 004000    2$:    BIT     #4000,R4
2178 005332 001773          BEQ     1$          ;
2179 005334 012602          MOV     (SP)+,R2    ;RESTORE R2
2180 005336 000207          RTS     PC
2181
2182          ;THIS SUBROUTINE ALLOWS A CALLER SPECIFIED DELAY.
2183          ;CALL: MOV     DELAY TIME,DELTIM ;LOAD DELAY TIME (# OF TICKS)
2184          ;      JSR     PC,DELAYV
2185 005340 005737 001120    DELAYV: TST     DELTIM ;BRANCH IF 0 DELAY
2186 005344 001413          BEQ     3$         ;
2187 005346 004737 004730    JSR     PC,TIMON   ;TURN TIMER ON
2188 005352 010246          MOV     R2,-(SP)   ;SAVE R2 ON THE STACK
2189 005354 012702 005364    MOV     #2$,R2     ;SET RETURN ADDRESS FROM TIMER
2190 005360          1$:
2191 005360 000163 005020    JMP     TIMER(R3)   ;GO TO TIMER & RETURN VIA R2
2192 005364 023704 001120    2$:    CMP     @#DELTIM,R4
2193 005370 101373          BHI     1$         ;
2194 005372 012602          MOV     (SP)+,R2   ;RESTORE R2
2195 005374 000207          3$:    RTS     PC
2196
2197          .SBTTL      DIVIDE SUBROUTINE
2198          ;THIS SUBROUTINE DIVIDES A DOUBLE PRECISION # AND RETURNS THE RESULT
2199          ;TO THE CALLER ON THE STACK. BOTH DIVIDEND & DIVISOR MUST BE POSITIVE.
2200          ;CALL: MOV     LEAST SIGNIFICANT HALF DIVIDEND,-(SP)
2201          ;      MOV     #MOST SIGNIFICANT HALF DIVIDEND,-(SP)
2202          ;      MOV     #DIVISOR,-(SP)
2203          ;      JSR     PC,DIVIDE
2204          ;RETURN
2205          ;      (SP) REMAINDER ON STACK
2206          ;      2(SP)-QUOTIENT
2207

```

```

2208 ;NOTE: THIS SUBROUTINE DESTROYS PREVIOUS CONTENTS OF R0,R1,R2 & R3.
2209
2210 005376 005046 DIVIDE: CLR -(SP) ;SAVE LOC FOR SIGNS
2211 005400 012746 000021 MOV #17,-(SP) ;SET ITERATION COUNT
2212 005404 016601 000012 MOV 12(SP),R1 ;GET LSH DIVIDEND
2213 005410 016600 000010 MOV 10(SP),R0 ;GET MSH DIVIDEND
2214 005414 016602 000006 MOV 6(SP),R2 ;GET DIVISOR
2215 005420 005402 NEG R2 ;NEGATE DIVISOR
2216 005422 000241 CLC ;CLEAR 'C' BIT IN PSW
2217 005424 000405 BR 2$
2218 005426 006100 1$: ROL R0 ;ROTATE MSH DIVIDEND
2219 005430 010003 MOV R0,R3 ;SAVE IN R3
2220 005432 060203 ADD R2,R3 ;SUBTRACT DIVISOR FROM MSH DIVIDEND
2221 005434 103001 BCC 2$ ;BRANCH IF DIVIDEND > DIVISOR
2222 005436 010300 MOV R3,R0 ;SAVE REMAINDER IN R0
2223 005440 006101 2$: ROL R1 ;ROTATE LSH DIVIDEND
2224 005442 005316 DEC (SP) ;DECREMENT ITERATION COUNT
2225 005444 001370 BNE 1$
2226 005446 005726 TST (SP)+ ;POP ITERATION COUNTER
2227 005450 005726 TST (SP)+ ;POP SIGN CORRECTION
2228 005452 010166 000006 MOV R1,6(SP) ;PUSH REMAINDER ON STACK
2229 005456 010066 000004 MOV R0,4(SP) ;PUSH QUOTIENT ONTO STACK
2230 005462 012616 MOV (SP)+,(SP)
2231 005464 000207 RTS PC
2232
2233 ;SMTL DRIVE SUBROUTINES
2234 ;SUBROUTINE TO CHECK IF DRIVE IS AVAILABLE
2235 ;CALL: MOV B DRIVE#,DRVNUM
2236 ; JSR PC,DRVAVA
2237 ;RETURN: 'C' BIT SET IF NOT AVAILABLE
2238 005466 113765 001004 000010 DRVAVA: MOV B @DRVNUM,CS2(R5) ;LOAD DRIVE #
2239 005474 032765 040000 000026 BIT #TAP,DT(R5) ;CHECK IF TAPE UNIT
2240 005502 001003 BNE 1$
2241 005504 004737 005544 JSR PC,RHINIT
2242 005510 000262
2243 005512 000207 1$: RTS PC ;SET 'V' TO IND NOT AVAIL
2244 ;RETURN
2245
2246 ;SUBROUTINE TO CHECK IF TE16/TU77 SLAVE IS AVAILABLE FOR TEST
2247 ;CALL: MOV B DRIVE #,@DRVNUM ;PASS DRIVE # VIA DRVNUM
2248 ; MOV B SLAVE #,@SLVNUM ;PASS SLAVE # VIA SLVNUM
2249 ; JSR PC,SLVAVA ;CALL SUBROUTINE
2250 005514 113765 001004 000010 SLVAVA: MOV B @DRVNUM,CS2(R5) ;LOAD DRIVE #
2251 005522 113765 001005 000032 MOV B @SLVNUM,TC(R5) ;AND SLAVE #
2252 005530 032765 002000 000026 BIT #SPR,DT(R5) ;BRANCH IF SLAVE PRESENT
2253 005536 001001 BNE 1$
2254 005540 000262 SEV ;SET 'V' TO INDICATE NO SLAVE
2255 005542 000207 1$: RTS PC
2256
2257 ;SUBROUTINE TO INITIALIZE RH CONTROLLER
2258 ;CALL: JSR PC,RHINIT
2259 005544 012765 000040 000010 RHINIT: MOV #40,CS2(R5)
2260 005552 113765 001004 000010 MOV B @DRVNUM,CS2(R5)
2261 005560 005046 CLR -(SP)
2262 005562 113716 001005 MOV B @SLVNUM,(SP)
2263 005566 012665 000032 MOV (SP)+,TC(R5) ;LOAD SLAVE # INTO TC REG

```

```

2264 005572 052765 001700 000032      BIS      #NORM11,TC(R5)
2265 005600 000207                RTS      PC
2266
2267      ;SUBROUTINE TO WAIT FOR DRIVE READY (DRY)
2268 005602 005027      WAITRDY:CLR      (PC)+      ;CLEAR WAIT TIMER
2269 005604 000000      WAITTIM:.WORD    0
2270 005606 105765 000012      1$:      TSTB      DS(R5)      ;WAIT FOR READY TO SET
2271 005612 100406                BMI      2$
2272 005614 005237 005604                INC      WAITTIM      ;INCREMENT WAIT TIMER
2273 005620 001372                BNE      1$           ;BRANCH IF TIME HAS NOT EXPIRED
2274 005622 000004 015252                TYPE,E.TIMEXP      ;TYPE 'TIME EXPIRED WAITING FOR RDY'
2275 005626 000425                BR       99$         ;TAKE ERROR EXIT
2276 005630 032765 002000 000012 2$:      BIT      #EOT,DS(R5)    ;CHECK FOR END OF TAPE
2277 005636 001415                BEQ      3$           ;BRANCH IF NO EOT
2278 005640 000004 014112                TYPE,M.NAM
2279 005644 000004 014630                TYPE,M.EOT          ;TYPE 'END OF TAPE'
2280 005650 004737 005706                JSR     PC,.REWIND  ;REWIND SLAVE
2281 005654 102412                BVS     99$         ;BRANCH IF ERROR ON REWIND
2282 005656 004737 005770                JSR     PC,WRITE   ;WRITE A RECORD
2283 005662 005215                INC     (R5)       ;SET 'GO' BIT
2284 005664 004737 005602                JSR     PC,WAITRDY ;WAIT FOR READY
2285 005670 000404                BR      99$         ;TAKE ERROR EXIT
2286 005672 032765 040000 000012 3$:      BIT      #ERR,DS(R5)  ;CHECK ERROR EXIT
2287 005700 001401                BEQ     100$
2288 005702 000262      99$:      SEV
2289 005704 000207      100$:     RTS      PC
2290      ;SUBROUTINE TO REWIND A UNIT (DRIVE/SLAVE COMBINATION)
2291      ;CALL      MOVB   DRIVE #,@#DRVNUM
2292      ;          MOVB   SLAVE #,@#SLVNUM
2293      ;          JSR    PC,.REWIND
2294      ;SUBROUTINE RETURNS TO CALLER WITH SELECTED SLAVE AT 'BOT', & 'V' SET IF
2295      ;AN ERROR OCCURS.
2296
2297 005706 004737 005544      .REWIND:JSR     PC,RHINIT      ;INITIALIZE CONTROLLER
2298 005712 004337 006124                JSR     R3,TCMD           ;GO TO TM COMMAND SUBROUTINE
2299 005716 000000                .WORD  0                 ;BUS ADDRESS (NOT USED)
2300 005720 000000                .WORD  0                 ;WORD COUNT (NOT USED)
2301 005722 000000                .WORD  0                 ;FRAME COUNT (NOT USED)
2302 005724 000006                .WORD  RWD              ;REWIND COMMAND
2303 005726 005215                INC     (R5)            ;SET 'GO' BIT
2304 005730 032765 000002 000012 1$:      BIT      #BOT,DS(R5)    ;BRANCH IF 'BOT' SET
2305 005736 001005                BNE     2$
2306 005740 032765 040000 000012                BIT      #ERR,DS(R5)    ;CHECK ERROR BIT
2307 005746 001006                BNE     99$           ;BRANCH IF ERROR BIT SET
2308 005750 000767                BR      1$
2309
2310 005752 032765 020000 000012 2$:      BIT      #PIP,DS(R5)    ;WAIT FOR TAPE MOTION TO STOP
2311 005760 001374                BNE     2$
2312 005762 000401                BR      100$
2313 005764 000262      99$:      SEV
2314 005766 000207      100$:     RTS      PC
2315
2316      ;SUBROUTINE TO WRITE 256. WORD RECORD
2317      ;CALL:      JSR    PC,WRITE
2318
2319 005770 004337 006124      WRITE:  JSR     R3,TCMD           ;GO TO TM COMMAND SUBROUTINE
  
```

```

2320 005774 016650          .WORD  WTBUF          ;BUS ADDRESS
2321 005776 177600          .WORD  WRDCNT         ;WORD COUNT
2322 006000 177400          .WORD  FRMCNT         ;FRAME COUNT
2323 006002 000060          .WORD  WFWD           ;WRITE FORWARD COMMAND
2324 006004 000207          RTS      PC
2325
2326          ;SUBROUTINE TO READ A 256. WORD RECORD.
2327          ;CALL: JSR      PC,READ
2328
2329 006006 004337 006124    READ: JSR      R3,@TMCMD
2330 006012 016650          .WORD  RDBUF          ;ADDRESS OF READ BUFFER
2331 006014 177600          .WORD  WRDCNT         ;2'S COMPLEMENT OF WORD COUNT
2332 006016 177400          .WORD  FRMCNT         ;2'S COMPLEMENT OF FRAME COUNT
2333 006020 000070          .WORD  RDFWD         ;READ FORWARD COMMAND
2334 006022 000207          RTS      PC
2335
2336          ;SUBROUTINE TO INITIATE READ REVERSE COMMAND
2337          ;CALL: JSR      PC,REVRD
2338
2339 006024 004337 006124    REVRD: JSR      R3,TMCMD
2340 006030 017250          .WORD  RDBUF+256.    ;ADDRESS OF READ REVERSE BUFFER
2341 006032 177600          .WORD  WRDCNT         ;2'S COMPLEMENT OF WORD COUNT
2342 006034 177400          .WORD  FRMCNT         ;2'S COMPLEMENT OF FRAME COUNT
2343 006036 000076          .WORD  RDREV         ;READ REVERSE COMMAND
2344 006040 000207          RTS      PC
2345
2346          ;SUBROUTINE TO SPACE FORWARD 1 RECORD
2347 006042 012765 177777 000006    FWDSPC: MOV     #-1,FC(R5) ;LOAD RECORD COUNT
2348 006050 012715 000031          MOV     #SPCFWD+1,(R5) ;LOAD COMMAND
2349 006054 004737 005602          JSR     PC,WAITRDY    ;WAIT FOR READY
2350 006060 000207          RTS      PC          ;RETURN
2351
2352          ;SUBROUTINE TO WRITE A RECORD AND BACK SPACE OVER THE RECORD.
2353 006062 004737 005770    WRT.BK: JSR     PC,WRITE ;WRITE THE RECORD
2354 006066 005215          INC     (R5)          ;SET 'GO' BIT
2355 006070 004737 005602          JSR     PC,WAITRDY
2356 006074 102412          BVS    2$
2357 006076 012765 177777 000006    MOV     #-1,FC(R5)    ;LOAD RECORD COUNT
2358 006104 012715 000033          MOV     #SPCREV+1,(R5) ;LOAD COMMAND
2359 006110 004737 005602          JSR     PC,WAITRDY
2360 006114 102402          BVS    2$
2361 006116 004737 005310    1$: JSR     PC,DELAY    ;WAIT FOR TAPE MOTION TO STOP
2362 006122 000207          2$: RTS      PC
2363
2364          ;SUBROUTINE TO LOAD A COMMAND
2365          ;CALL: JSR      R3,TMCMD
2366          :          .WORD  BUS ADDRESS
2367          :          .WORD  WORD COUNT (2'S COMPLEMENT)
2368          :          .WORD  FRAME COUNT (2'S COMPLEMENT)
2369          :          .WORD  COMMAND
2370
2371 006124 012365 000004    TMCMD: MOV     (R3)+,BA(R5) ;LOAD BUS ADDRESS
2372 006130 012365 000002          MOV     (R3)+,WC(R5) ;LOAD WORD COUNT
2373 006134 012365 000006          MOV     (R3)+,FC(R5) ;LOAD FRAME COUNT
2374 006140 012315          MOV     (R3)+,(R5)   ;LOAD COMMAND
2375 006142 000203          RTS      R3         ;RETURN
  
```

```
2376  
2377  
2378 ;SUBROUTINE TO PRINT SERIAL NUMBER  
2379 ;JSR PC,SNPT  
2380 SNPT: MOV SN(R5),R3  
2381 MOV #ODIGITS,R1  
2382 SWAB R3  
2383 ROR R3  
2384 ROR R3  
2385 ROR R3  
2386 ROR R3 ;GET FIRST DIGIT  
2387 BIC #177760,R3  
2388 BIS #260,R3  
2389 MOV R3,(R1)+ ;FILL FIRST DIGIT  
2390 MOV SN(R5),R3  
2391 SWAB R3  
2392 BIC #177760,R3  
2393 BIS #260,R3  
2394 MOV R3,(R1)+ ;GET SECOND DIGIT  
2395 MOV SN(R5),R3  
2396 ROR R3  
2397 ROR R3  
2398 ROR R3  
2399 ROR R3  
2400 BIC #177760,R3  
2401 BIS #260,R3  
2402 MOV R3,(R1)+ ;GET THIRD DIGIT  
2403 MOV SN(R5),R3  
2404 BIC #177760,R3  
2405 BIS #260,R3  
2406 MCVB R3,(R1)+ ;GET FOURTH DIGIT  
2407 CLRB (R1)  
2408 TYPE,ODIGITS ;TYPE SERIAL NUMBER  
2409 RTS PC ;RETURN  
2410
```

```

2411          .SBTTL  PROGRAM INITIALIZATION
2412 006274 012706 000600      INIT:  MOV      #STKPTR,SP      ;SET STACK PTR
2413 006300 005037 001272      CLR      @#INBUF
2414
2415 006304 013746 000006      MOV      @#6,-(SP)      ;SAVE VECTORS
2416 006310 013746 000004      MOV      @#4,-(SP)
2417 006314 012737 006334 000004  MOV      #61$,@#4      ;SET UP FOR TIMEOUT
2418 006322 022777 177777 172450  CMP      #-1,@SWR      ;REFERENCE HARDWARE SWITCH REGISTER
2419 006330 001402
2420 006332 000404
2421 006334 022626
2422 006336 012737 000176 001000 61$:  CMP      (SP)+,(SP)+      ;ADJUST STACK
2423 006344 012637 000004      60$:  MOV      #SWREG,SWR      ;POINT TO SOFTWARE SWITCH REG
2424 006350 012637 000006      62$:  MOV      (SP)+,@#4      ;RESTORE VECTORS
2425 006354 105037 001131      MOV      (SP)+,@#6
2426 006360 105037 001135      CLRB     @#PRGFLG      ;CLEAR PROGRAM FLAG
2427 006364 105037 001134      CLRB     @#ASFLG      ;CLEAR ASK FLAG
2428 006370 005027
2429 006372 000000      CLRB     @#PSCNT      ;SET PASS COUNT = 0
2430          CHNFLG: .WORD 0      ;;CLEAR CHAIN INDICATOR
2431 006374 005737 000042      TST      @#42          ;;CHAIN MODE INDICATOR
2432 006400 001407
2433 006402 012737 000176 001000  BEQ      50$           ;;1/0 = CHAIN/NOT CHAIN MODE
2434 006410 005237 006372      MOV      #SWREG,SWR      ;;BRANCH IF IN DUMP MODE
2435 006414 000137 006420      INC      CHNFLG          ;;INVOKE SOFTWARE SWR
2436 006420
2437 006420 122737 000006 000041 50$:  JMP      1$           ;;SET CHNFLG = CHAIN MODE
2438 006426 001003
2439 006430 000004 014322      JMP      1$           ;;GO TO CHAIN ADDRESS
2440 006434 000000
2441 006436 000004 014112      CMPB     #6,@#41      ;BRANCH IF NOT LOADED VIA TMDP
2442 006442 005737 006372      BNE      2$           ;ADVISE USER TO REMOVE TMDP
2443 006446 001025
2444 006450 105037 014112      TYPE,I.REM
2445 006454 000004 014374      HALT
2446 006460 013702 001010      TYPE,M.NAM          ;TYPE TITLE
2447 006464 004737 003134      TST      CHNFLG      ;SEE IF CHAIN MODE
2448 006470 000004 001416      BNE      5$           ;IF SO: BR
2449 006474 004737 004000      CLRB     M.NAM      ;DO NOT TYPE TITLE ON RESTART
2450 006500 122737 000015 001272  TYPE,I.REG          ;ASK USER TO TYPE CONT BASE ADRS
2451 006506 001405
2452 006510 004737 003564      MOV      @#TMBASE,R2  ;GET CURRENT CONT BASE ADDRESS
2453 006514 013737 001122 001010  JSR      PC,TYPOCT   ;AND TYPE IT
2454 006522 013705 001010      JSR      PC,SPACE    ;GET USER INPUT
2455
2456          ;ROUTINE TO CHECK IF CONTROLLER (RH11) IS AVAILAABLE
2457 006526 000261
2458 006530 005715      JSR      PC,INPUT    ;DO NOT CHANGE CURRENT VALUE
2459 006532 103003
2460 006534 000004 014667      CMPB     #CR,@#INBUF ;IF USER TYPES <CR>
2461 006540 000655
2462 006542 012737 004250 000004 4$:  BEQ      5$           ;CONVERT ASCII TO OCTAL
2462 006542 012737 004250 000004 5$:  JSR      PC,CNVTAO   ;SET NEW ADDRESS
2462 006542 012737 004250 000004 6$:  MOV      @#OCTALO,@#TMBASE
2462 006542 012737 004250 000004 6$:  MOV      @#TMBASE,R5
  
```



```

2463 ;ROUTINE TO GET TMO3 DRIVES USER DESIRES TO TEST
2464 006550 105037 001127 DRIVES: CLR B @#ERFLG ;CLEAR ERROR FLAG
2465 006554 012701 001162 MOV #DRVTBL,R1 ;MARK ALL DRIVES AS NOT TO
2466 006560 012700 000004 MOV #4,R0 ;BE TESTED. A '0' INDICATES
2467 006564 005021 1$: CLR (R1)+ ;THAT A DRIVE IS NOT TO BE
2468 006566 005300 DEC R0 ;TESTED
2469 006570 001375 BNE 1$
2470 006572 005737 006372 TST CHNFLG ;BRANCH IF IN CHAIN MODE
2471 006576 001014 BNE 2$
2472 006600 000004 014441 TYPE,I.DRVS
2473 006604 004737 004000 JSR PC,INPUT ;GET USER INPUT
2474 006610 012700 001272 MOV #INBUF,R0
2475 006614 122710 000101 CMPB #'A,(R0) ;IF USER RESPONDS WITH 'A' OR
2476 006620 001403 BEQ 2$ ;<CR> THEN ALL AVAILABLE DRIVES
2477 006622 122710 000015 CMPB #CR,(R0) ;ARE TO BE TESTED
2478 006626 001013 BNE 4$
2479 006630 110637 001131 2$: MOV SP,PRGFLG ;SET FLAG TO IND ALL DRIVES
2480 006634 012701 001162 MOV #DRVTBL,R1 ;MARK ALL DRIVES TO BE TESTED
2481 006640 012700 000004 MOV #4,R0 ;A '-1' INDICATES THAT A DRIVE
2482 006644 012721 177777 3$: MOV #-1,(R1)+ ;IS TO BE TESTED
2483 006650 005300 DEC R0
2484 006652 001374 BNE 3$
2485 006654 000417 BR CHKDRV ;GO CHECK DRIVE AVAILABILITY
2486
2487 ;GET USER SELECTED DRIVES AND MARK EACH DRIVE SELECTED TO BE TESTED
2488 006656 122710 000015 4$: CMPB #CR,(R0)
2489 006662 001414 BEQ CHKDRV
2490 006664 121027 000054 CMPB (R0),'#', ;CHECK IF 'COMMA'
2491 006670 001001 BNE 5$
2492 006672 105720 TSTB (R0)+ ;STEP PTR PAST 'COMMA'
2493 006674 112001 5$: MOVB (R0)+,R1
2494 006676 042701 177770 BIC #177770,R1
2495 006702 112761 177777 001162 MOVB #-1,DRVTBL(R1)
2496 006710 000240 NOP
2497 006712 000761 BR 4$
2498
2499 ;ASCERTAIN THAT DRIVES (TMO3'S) SPECIFIED ARE AVAILABLE
2500 006714 005000 CHKDRV: CLR R0 ;A (0) IN DRVTBL(R0) INDICATES
2501 006716 105760 001162 1$: TSTB DRVTBL(R0) ;THE DRIVE IS NOT TO BE TESTED
2502 006722 001005 BNE 3$ ;A '1' INDICATES TO BE TESTED
2503 006724 005200 2$: INC R0
2504 006726 122700 000010 CMPB #8.,R0
2505 006732 001371 BNE 1$
2506 006734 000424 BR 5$
2507 006736 110037 001004 3$: MOVB R0,@#DRVNUM ;GET DRIVE #
2508 006742 004737 005466 JSR PC,@#DRVAVA ;AND CHECK IF AVAILABLE
2509 006746 102366 BVC 2$ ;'V' BIT SET INDICATES NOT AVAIL
2510 006750 105737 001131 TSTB @#PRGFLG ;DO NOT TYPE NOT AVAILABLE
2511 006754 001011 BNE 4$ ;MESSAGE IF ALL SELECTED
2512 006756 000004 014734 TYPE,E.NDRV
2513 006762 116037 001140 014766 MOVB DIGTAB(R0),@#E.NAVA ;SET DRIVE # IN MESSAGE
2514 006770 000004 014766 TYPE,E.NAVA
2515 006774 110637 001127 MOVB SP,@#ERFLG ;SET 'ERROR' FLAG
2516 007000 105060 001162 4$: CLR B DRVTBL(R0) ;MARK DRIVE UNAVAILABLE
2517 007004 000747 BR 2$ ;CHECK NEXT DRIVE
2518 007006 105737 001127 5$: TSTB @#ERFLG ;GO GET SLAVES IF NO ERROR

```

```

2519 007012 001256          BNE      DRIVES          ;ELSE ASK USER TO RETYPE DRIVES
2520
2521          ;ROUTINE TO GET SLAVES (TE16/TU77'S) USER DESIRES TO TEST
2522 007014 105037 001127  SLAVES: CLR      @#ERFLG          ;CLEAR ERROR INDICATOR
2523 007020 012701 001172      MOV      #SLVTBL,R1          ;MARK ALL SLAVES (64.) AS NOT
2524 007024 012700 000040      MOV      #32.,R0           ;TO BE TESTED.A 0 INDICATES THAT
2525 007030 005021          1$: CLR      (R1)+           ;A DRIVE'S SLAVE IS NOT TO BE
2526 007032 005300          DEC      R0                ;TESTED
2527 007034 001375          BNE      1$
2528 007036 012701 001172      MOV      #SLVTBL,R1          ;R1 POINTS TO DRIVE'S SLAVE
2529 007042 105760 001162      2$: TSTB   DRVTBL(R0)        ;BRANCH IF DRIVE IS TO BE TESTED
2530 007046 001007          BNE      4$                ;& IS AVAILABLE
2531 007050 062701 000010      3$: ADD      #8.,R1           ;STEP SLAVE PTR TO NEXT DRIVE'S
2532 007054 005200          INC      R0                ;SLAVES AND INCREMENT DRIVE #
2533 007056 122700 000010      CMPB   #8.,R0             ;CHECK ALL DRIVES
2534 007062 001367          BNE      2$                ;AND WHEN ALL DRIVES CHECKED
2535 007064 000457          BR       CHKSLV           ;GO CHECK SLAVE AVAILABILITY
2536
2537 007066 105737 001131      4$: TSTB   @#PRGFLG          ;BRANCH IF USER SELECTED ALL
2538 007072 001021          BNE      5$                ;DRIVES
2539 007074 110037 001004      MOVB   R0,DRVNUM          ;GET DRIVE #
2540 007100 116037 001140 014530  MOVB   DIGTAB(R0),@#I.DRV  ;PREPARE USER ACTION MESSAGE
2541 007106 000004 014511      TYPE,I.SLVS
2542 007112 004737 004000      JSR    PC,INPUT           ;GET USER INPUT
2543 007116 012703 001272      MOV    #INBUF,R3          ;SET PTR TO USER INPUT
2544 007122 122713 000101      CMPB  #'A,(R3)           ;AN 'A' OR <CR> AS FIRST CHAR
2545 007126 001403          BEQ    5$                 ;INDICATES TEST ALL SLAVES
2546 007130 122713 000015      CMPB  #CR,(R3)
2547 007134 001015          BNE    7$
2548 007136 110637 001131      5$: MOVB   SP,@#PRGFLG          ;SET 'ALL' INDICATOR
2549 007142 012701 001172      MOV    #SLVTBL,R1          ;MARK ALL SLAVES FOR ALL
2550 007146 012700 000040      MOV    #32.,R0            ;DRIVES AS TO BE TESTED
2551 007152 012721 177777      6$: MOV    #-1,(R1)+
2552 007156 005300          DEC    R0
2553 007160 001374          BNE    6$
2554 007162 105737 001131      TSTB  @#PRGFLG           ;BRANCH IF ALL WAS SELECTED
2555 007166 001016          BNE    CHKSLV
2556
2557 007170 122713 000015      7$: CMPB  #CR,(R3)           ;GET USER SELECTED SLAVES FOR
2558 007174 001725          BEQ    3$                 ;DRIVE
2559 007176 121327 000054      CMPB  (R3),#',           ;STEP PTR PAST 'COMMA'
2560 007202 001001          BNE    8$
2561 007204 105723          TSTB  (R3)+
2562 007206 112304          8$: MOVB  (R3)+,R4           ;AND MARK SELECED SLAVE
2563 007210 042704 177770      BIC   #177770,R4          ;AS TO BE TESTED
2564 007214 060104          ADD   R1,R4
2565 007216 112714 177777      MOVB  #-1,(R4)
2566 007222 000762          BR    7$
2567
2568          ;ASCERTAIN THAT SLAVES (TE16/TU77'S) SELECTED ARE AVAILABLE
2569 007224 005000 001172  CHKSLV: CLR    R0           ;R0 WILL CONTAIN THE DRIVE #
2570 007226 005001 001162      CLR    R1                ;AND R1 THE SLAVE #
2571 007230 012702 001172      MOV    #SLVTBL,R2          ;SET PTR TO SLAVE TABLE
2572 007234 105760 001162      1$: TSTB  DRVTBL(R0)        ;BRANCH IF DRIVE SELECTED
2573 007240 001007          BNE    3$                 ;& AVAILABLE FOR TEST
2574 007242 005200          2$: INC    R0              ;INCREMENT DRIVE #

```

```

2575 007244 062702 000010          ADD    #8.,R2          ;STEP SLAVE PTR TO NEXT DRIVE'S
2576 007250 022700 000010          CMP    #8.,R0          ;SLAVES. BRANCH TO 1$ IF NOT ALL
2577 007254 001367                    BNE    1$              ;DRIVES CHECKED OTHERWISE EXIT
2578 007256 000437                    BR     8$
2579
2580 007260 005001          3$:   CLR    R1          ;SET SLAVE # 0
2581 007262 105712          4$:   TSTB   (R2)        ;BRANCH IF DRIVE'S SLAVE IS SEL-
2582 007264 001006                    BNE    6$              ;ECTED FOR TEST
2583 007266 005201          5$:   INC    R1          ;INCREMENT SLAVE #
2584 007270 005202                    INC    R2              ;STEP PTR TO NEXT SLAVE
2585 007272 022701 000010          CMP    #8.,R1          ;GO TO 4$ IF ALL SLAVES NOT
2586 007276 001371                    BNE    4$              ;CHECKED
2587 007300 000760                    BR     2$              ;OTHERWISE GO TO 2$ ABOVE
2588
2589 007302 110037 001004          6$:   MOVB   R0,@#DRVNUM ;PASS DRIVE & SLAVE #
2590 007306 110137 001005          MOVB   R1,@#SLVNUM
2591 007312 004737 005514          JSR    PC,@#SLVAVA
2592 007316 102363                    BVC    5$              ;AND CHECK IF AVAILABLE
2593 007320 105737 001131          TSTB   @#PRGFLG        ;'V' SET INDICATES ERROR
2594 007324 001012                    BNE    7$              ;DO NOT TYPE ERROR MSG IF ALL
2595 007326 116037 001140 014756          MOVB   DIGTAB(R0),@#E.DRV ;SLAVES SELECTED
2596 007334 116137 001140 014766          MOVB   DIGTAB(R1),@#E.NAVA ;ICATES ERROR. PREPARE ERROR
2597 007342 000004 014750          TYPE,E.NSLV           ;MESSAGE
2598 007346 110637 001127          MOVB   SP,@#ERFLG
2599 007352 105012          7$:   CLRB   (R2)        ;SET ERROR INDICATOR
2600 007354 000744          BR     5$              ;CLEAR SLAVE TABLE ENTRY
2601
2602 007356 105737 001127          8$:   TSTB   @#ERFLG        ;BRANCH IF ERROR
2603 007362 001214                    BNE    SLAVES          ;ASK USER TO RETYPE SLAVES
2604 007364 012737 004250 000004 100$:  MOV    #ERRTRP,@#ERRVEC
2605
2606          ;SCAN DRIVE AND SLAVE TABLE FOR DRIVE/SLAVE COMBINATION TO TEST.
2607          ;RESTART ADDRESS--PROGRAM STARTS HERE WHEN START ADDRESS = 210 AND
2608          ;AFTER ALL SELECTED DRIVE/SLAVE COMBINATIONS HAVE BEEN TESTED.
2609 007372 012706 000600          RSTRT: MOV    #600,SP    ;SET STACK PTR
2610 007376 105037 001004          CLRB   @#DRVNUM        ;SET DRIVE AND SLAVE # 0
2611 007402 105037 001005          CLRB   @#SLVNUM
2612 007406 012737 001172 001006          MOV    #SLVTBL,@#SLVPTR ;SET PTR TO SLAVE TABLE
2613 007414 105037 001132          CLRB   @#UNTFND        ;CLEAR 'UNIT FOUND' IND.
2614
2615          ;PROGRAM RESTARTS HERE AFTER A DRIVE/SLAVE HAS BEEN TESTED.
2616 007420 113700 001004          BEGIN: MOVB   @#DRVNUM,R0 ;GET DRIVE #
2617 007424 113701 001005          MOVB   @#SLVNUM,R1     ;AND SLAVE #
2618 007430 013702 001006          MOV    @#SLVPTR,R2     ;GET SLAVE PTR
2619 007434 122737 000006 000041          CMPB   #6,@#41        ;BRANCH IF LOADED VIA TMDP
2620 007442 001001                    BNE    1$
2621 007444 105012          CLRB   (R2)            ;SET DRIVE #0,SLAVE #0 NOT TO
2622          ;BE TESTED.
2623 007446 105760 001162          1$:   TSTB   DRV TBL(R0) ;BRANCH IF DRIVE AVAIL TO TEST
2624 007452 001011                    BNE    3$
2625 007454 005001                    CLR    R1              ;CLEAR SLAVE #
2626 007456 062702 000010          ADD    #8.,R2          ;AND STEP PTR TO NEXT DRIVE'S
2627 007462 005200          2$:   INC    R0          ;SLAVES AND INCREMENT DRIVE #
2628 007464 022700 000010          CMP    #8.,R0          ;EXIT TEST IF ALL DRIVES
2629 007470 001366                    BNE    1$              ;CHECKED OTHERWISE CONTINUE
2630 007472 000137 013410          JMP    @#END           ;SCAN FOR NEXT 'UNIT'

```

```

2631
2632 007476 105712          3$:   TSTB   (R2)           ;BRANCH IF SLAVE ON DRIVE IS
2633 007500 001007          BNE    4$           ;AVAILABLE THERWISE STEP
2634 007502 005202          INC    R2           ;PTR TO NEXT SLAVE
2635 007504 005201          INC    R1           ;INCREMENT SLAVE #
2636 007506 122701 000010  CMPB   #8.,R1      ;UNTIL ALL SLAVES CHECKED
2637 007512 001371          BNE    3$           ;WHEN ALL SLAVES CHECKED
2638 007514 005001          CLR    R1           ;SET SLAVE # 0
2639 007516 000761          BR     2$           ;AND CONTINUE SCAN
2640
2641 007520 110637 001132  4$:   MOVB   SP,@#UNTFND  ;INDICATE THAT A 'UNIT' IS FOUND
2642 007524 110037 001004  MOVB   R0,@#DRVNUM  ;SET DRIVE #
2643 007530 110137 001005  MOVB   R1,@#SLVNUM  ;SET SLAVE #
2644 007534 010237 001006  MOV    R2,@#SLVPTR  ;SAVE SLAVE PTR
2645
2646 007540 105737 001135  5$:   TSTB   @#ASFLG
2647 007544 001034          BNE    7$
2648 007546 112737 000001 001135  MOVB   #1,ASFLG
2649 007554 005737 006372  TST    CHNFLG      ;BRANCH IF IN CHAIN MODE
2650 007560 001026          BNE    7$
2651 007562 105037 001130  CLRB   @#SKEWFLG   ;++B CLEAR SKEW (SPEED) TESTS SELECTED FLAG
2652 007566 000004 014575  TYPE,I,SPD        ;ASK USER IF HE WANTS TO RUN SPEED TESTS
2653 007572 004737 004000  JSR    PC,INPUT    ;GET USER INPUT
2654 007576 012703 001272  MOV    #INBUF,R3   ;GET REPLY
2655 007602 122713 000015  CMPB   #CR,(R3)    ;DO NOT DO SKEW TESTS IF <CR> IS FIRST
2656 007606 001405          BEQ    6$
2657 007610 132713 000001  BITB   #1,(R3)     ;BRANCH IF 'N'
2658 007614 001402          BEQ    6$
2659 007616 111337 001130  MOVB   (R3),@#SKEWFLG ;SET INDICATOR
2660 007622 022737 000176 001000 6$:   CMP    #SWREG,SWR  ;BRANCH IF SOFTWARE SWR
2661 007630 001002          BNE    7$           ;NOT INVOKED
2662 007632 004737 002526  JSR    PC,GTSWR    ;GET SWITCH REGISTER
2663
2664          7$:
2665          ;ROUTINE TO ASCERTAIN SLAVE TYPE AND LOAD APPROPRIATE SPECIFICATION
2666          ;TABLES (TE16 OR TU77) INTO STIMTBL AND GTIMTBL.
2666 007636 013705 001010  MOV    @#TMBASE,R5 ;GET BASE ADDRESS OF REGISTERS
2667 007642 004737 005544  JSR    PC,RHINIT   ;INIT DRIVE/SLAVE
2668 007646 012704 000240  MOV    #ENDTBL-STTBL,R4 ;GET TABLE LENGTH
2669 007652 012703 002124  MOV    #STIMTBL,R3 ;AND STARTING ADDRESS OF TABLE
2670 007656 012702 001424  MOV    #TE16TTBL,R2 ;GET ADDRESS OF TE16 TIME TABLE
2671 007662 105037 001136  CLRB   @#TE16      ;SET FLAG = TE16
2672 007666 032765 000004 000026  BIT    #4,DT(R5)   ;BRANCH IF TU77
2673 007674 001012          BNE    9$
2674 007676 012737 000070 001012  MOV    #56.,OSCTIM ;SET US/TICK = 56
2675 007704 012737 000022 001014  MOV    #18.,GAPDEL ;SET 18 TICKS/MS
2676 007712 112223          8$:   MOVB   (R2)+,(R3)+ ;MOVE TE16 TIME AND GAP TABLES
2677 007714 005304          DEC    R4           ;INTO STIMTBL & GTIMTBL
2678 007716 001375          BNE    8$
2679 007720 000416          BR     11$         ;EXIT ROUTINE
2680
2681 007722 012702 001664          9$:   MOV    #TU77TTBL,R2 ;GET ADDRESS OF TU77 TIME TABLE
2682 007726 112737 000001 001136  MOVB   #1,@#TE16   ;SET FLAG - TU77
2683 007734 012737 000120 001012  MOV    #80.,OSCTIM ;SET US/TICK = 80
2684 007742 012737 000003 001014  MOV    #3,GAPDEL   ;SET 3 TICKS PER .25MS
2685 007750 112223          10$:  MOVB   (R2)+,(R3)+ ;MOVE TU77 TIME AND GAP TABLES
2686 007752 005304          DEC    R4           ;INTO STIMTBL & GTIMTBL

```

```
2687 007754 001375          BNE      10$
2688 007756          11$:
2689
2690          ;NOTE THIS IS NOT A TEST
2691          ;INITIALIZE PROGRAM FLAGS
2692 007756 105037 001126  TST000: CLRB   @#TSTNUM          ;SET TEST # 0
2693 007762 013705 001010      MOV   @#TMBASE,R5          ;SET ADDRESS OF FIRST TM03 REG
2694 007766 010500          MOV   R5,R0
2695 007770 062700 000006      ADD   #FC,R0              ;R0 CONTAINS ADDRESS OF FC REG
2696 007774 010501          MOV   R5,R1
2697 007776 062701 000012      ADD   #DS,R1              ;R1 CONTAINS ADDRESS OF DS REG
2698 010002 012703 005020      MOV   #TIMER,R3          ;SET JUMP ADDRESS TO TIMER
2699 010006 105037 001125      CLRB   @#ITCNT           ;CLEAR SUBTEST ITERATION COUNT
2700 010012 052737 000100 177560  BIS   #100,@#TKS          ;SET KEYBOARD IE BIT
2701
2702          ;GET USER RUN PROCEDURE
2703          ;IF SWR <05::00> IS NOT 0 THEN RUN TEST IN SWR<05::00>
2704          ;OTHERWISE RUN ALL TESTS
2705
2706 010020 004737 005706          JSR   PC,REWIND          ;REWIND SLAVE
2707 010024 102504          BVS   99$                ;BRANCH IF ERROR ON REWIND
2708 010026 105737 001130          TSTB  @#SKEWFLG         ;++B BRANCH IF SWEW (SPEED) TEST SELECTED
2709 010032 001006          BNE   10$
2710 010034 004737 005770          JSR   PC,WRITE          ;WRITE A RECORD
2711 010040 005215          INC   (R5)              ;SET 'GO' BIT
2712 010042 004737 005602          JSR   PC,WAITRDY        ;WAIT FOR READY
2713 010046 102473          BVS   99$
2714 010050 117702 170724 10$:  MOVB  @SWR,R2              ;GET SWITCHES
2715 010054 042702 177740          BIC   #177740,R2        ;CLEAR ALL BUT TEST #
2716 010060 001421          BEQ   2$                ;BRANCH IF TEST 0 WAS SELECTED
2717 010062 000004 015114          TYPE ,E.HDR            ;TYPE TEST #
2718 010066 004737 003134          JSR   PC,TYPOCT
2719 010072 006302          ASL   R2
2720 010074 016237 002364 010104  MOV   NAMPTR(R2),1$     ;FORM INDEX VALUE
2721 010102 000004          TYPE
2722 010104 000000          .WORD 0                ;GET ADDRESS OF TEST'S NAME
2723 010106 000004 001402 1$:  TYPE ,CRLF              ;AND TYPE IT
2724 010112 016237 002444 001002  MOV   TSTTBL(R2),@#SCPADR ;SET SCOPE ADDRESS FOR TEST
2725 010120 000172 002444          JMP   @TSTTBL(R2)       ;GO TO TEST
2726 010124 032777 002000 170646 2$:  BIT   #SW10,@SWR        ;BRANCH IF TIMES NOT TO BE TYPED
2727 010132 001034          BNE   5$
2728 010134 000004 015323          TYPE ,L.HDR1
2729 010140 113702 001004          MOVB  DRVNUM,R2          ;GET DRIVE #
2730 010144 113704 001005          MOVB  SLVNUM,R4          ;AND SLAVE #
2731 010150 116237 001140 015505  MOVB  DIGTAB(R2),@#L.DRV ;SET DRIVE AND SLAVE #'S
2732 010156 116437 001140 015517  MOVB  DIGTAB(R4),@#L.SLV ;INTO L.HDR2 MESSAGE
2733 010164 000004 015440          TYPE ,L.HDR2
2734 010170 105737 001136          TSTB  @#TE16            ;BRANCH IF NOT TE16
2735 010174 001003          BNE   3$
2736 010176 000004 015523          TYPE ,L.TE16           ;TYPE 'TE16'
2737 010202 000402          BR    4$
2738 010204 000004 015531 3$:  TYPE ,L.TU77           ;TYPE 'TU77'
2739 010210 000004 015537 4$:  TYPE ,L.SER            ;TYPE 'SERIAL #'
2740 010214 004737 006144          JSR   PC,SNPT           ;PRINT SLAVE SERIAL #
2741 010220 000004 015551          TYPE ,L.HDR3
2742 010224 105737 001130 5$:  TSTB  @#SKEWFLG         ;++B BRANCH IF SPEED TESTS NOT
```



```
2749 .SBTTL START OF TESTS
2750 ;TEST 001 - WRITE FROM BOT
2751 ;THIS TEST WILL MEASURE ACCELERATION DELAY REQUIRED TO
2752 ;MOVE THE TAPE APPROXIMATELY SEVEN (7) INCHES FORWARD
2753 ;FROM DEAD STOP BEFORE STARTING TO TRANSFER DATA.
2754
2755 ;THIS TEST MEASURES TIME FROM 'GO'=1 TO 'ACCL'=0.
2756 010252 112737 000001 001126 TST001: MOVB #1, @TSTNUM ;SET TEST #
2757 010260 012702 010304 MOV #1$, R2 ;SET RETURN PC FROM TIMER
2758 010264 004737 005706 JSR PC, .REWIND ;REWIND SLAVE
2759 010270 102420 BVS 99$ ;BRANCH IF ERROR ON REWIND
2760 010272 004737 005770 JSR PC, WRITE ;GO SETUP WRITE COMMAND
2761 010276 004737 004730 JSR PC, TIMON ;TURN TIMER ON
2762 010302 005215 INC (R5) ;SET 'GO' BIT
2763
2764 010304 005765 000032 1$: TST TC(R5) ;BRANCH WHEN 'ACCL'=0
2765 010310 100002 BPL 2$
2766 010312 000163 005020 JMP TIMER(R3) ;GO TO TIMER & RETURN VIA R2
2767
2768 010316 004737 005602 2$: JSR PC, WAITRDY ;WAIT FOR COMMAND TO FINISH
2769 010322 102403 BVS 99$ ;BRANCH IF ERROR
2770 010324 004737 005056 JSR PC, TIMOK ;GC CHECK TIME
2771 010330 000401 BR 100$
2772 010332 104400 99$: HLT
2773 010334 104000 100$: SCOPE
2774
2775 ;TEST 002 - WRITE START
2776 ;THIS TST MEASURES TIME FROM 'GO'=1 TO 'ACCL'=0.
2777 010336 112737 000002 001126 TST002: MOVB #2, @TSTNUM ;SET TEST # 2
2778 010344 004737 005770 JSR PC, WRITE ;INITIATE WRITE COMMAND
2779 010350 012702 010362 MOV #1$, R2 ;SET RETURN PC FROM TIMER
2780 010354 004737 004730 JSR PC, TIMON
2781 010360 005215 INC (R5) ;SET 'GO' BIT
2782
2783 010362 005765 000032 1$: TST TC(R5) ;BRANCH WHEN 'ACCL'=0
2784 010366 100002 BPL 2$
2785 010370 000163 005020 JMP TIMER(R3) ;GO TO TIMER & RETURN VIA R2
2786
2787 010374 004737 005602 2$: JSR PC, WAITRDY ;WAIT FOR READY
2788 010400 102403 BVS 99$ ;BRANCH IF ERROR
2789 010402 004737 005056 JSR PC, TIMOK ;GO CHECK TIME RECORDED
2790 010406 000401 BR 100$ ;EXIT VIA SCOPE
2791
2792 010410 104400 99$: HLT ;REPORT ERROR
2793 010412 104000 100$: SCOPE
2794
2795 ;TEST 003- WRITE SHUTDOWN
2796 ;THIS TEST MEASURES TIME FROM 'FC REG'=-0 TO 'SWDN'=1.
2797 010414 112737 000003 001126 TST003: MOVB #3, @TSTNUM ;SET TEST#3
2798 010422 004737 005770 JSR PC, WRITE ;INITIATE WRITE COMMAND
2799 010426 005215 INC (R5) ;SET 'GO' BIT
2800
2801 010430 005710 1$: TST (R0) ;BRANCH WHEN WRITING FINISHED
2802 010432 001404 BEQ 2$
2803 010434 032711 040000 BIT #1, (R1) ;MONITOR ERROR BIT
2804 010440 001017 BNF 99$
```

```
2805 010442 000772          BR      1$
2806
2807 010444          2$:
2808 010444 004737 004730      JSR      PC,TIMON      ;TURN TIMER ON
2809 010450 010702          MOV      PC,R2        ;LOAD RETURN PC FROM TIMER
2810 010452 032711 000020      3$:      BIT      #SDWN,(R1) ;BRANCH WHEN DS <SDWN> SETS
2811 010456 001002          BNE     4$
2812 010460 000163 005020      JMP     TIMER(R3)    ;GO TO TIMER & RETURN VIA R2
2813
2814 010464 004737 005602      4$:      JSR      PC,WAITRDY   ;WAIT FOR READY
2815 010470 102403          BVS     99$
2816 010472 004737 005056      JSR      PC,TIMOK     ;GO CHECK TIME RECORDFD
2817 010476 000401          BR      100$
2818 010500          99$:      HLT
2819 010502 104000          100$:    SCOPE      ;REPORT ERROR
2820
2821          ;TEST 004 - WRITE SETTLEDOWN
2822          ;THIS TEST MEASURES TIME FROM 'SWDN'=1 TO 'SWDN' 0.
2823 010504 112737 000004 001126 TST004: MOVB     #4,#TSTNUM
2824 010512 004737 005770          JSR      PC,WRITE
2825 010516 005215          INC     (R5)         ;SET 'GO' BIT
2826
2827 010520 005710          1$:      TST     (R0)         ;BRANCH WHEN WRITING FINISHED
2828 010522 001404          BEQ     2$
2829 010524 032711 040000      BIT     #ERR,(R1)   ;CHECK ERROR BIT
2830 010530 001026          BNE     99$
2831 010532 000772          BR      1$
2832
2833 010534 032711 000020      2$:      BIT     #SDWN,(R1) ;WAIT FOR ASSERTION OF 'SDWN'
2834 010540 001004          BNE     3$
2835 010542 032711 040000      BIT     #ERR,(R1)   ;MONITOR ERROP BIT
2836 010546 001017          BNE     99$
2837 010550 000771          BR      2$
2838
2839          3$:
2840 010552 004737 004730      JSR      PC,TIMON   ;TURN TIMER ON
2841 010556 010702          MOV      PC,R2     ;SET RETURN PC FROM TIMER
2842 010560 032711 000020      BIT     #SDWN,(R1) ;BRANCH WHEN SWDN CLEARS
2843 010564 001402          BEQ     5$
2844 010566 000163 005020      JMP     TIMER(R3)   ;GO TO TIMER & RETURN VIA R2
2845
2846 010572 004737 005602      5$:      JSR      PC,WAITRDY   ;WAIT FOR READY
2847 010576 102403          BVS     99$
2848 010600 004737 005056      JSR      PC,TIMOK
2849 010604 000401          BR      100$
2850
2851          99$:      HLT
2852 010610 104000          100$:    SCOPE
2853
2854          ;TEST 005 - READ FROM BOT
2855          ;THIS TEST MEASURES TIME FROM 'GO'-1 TO 'ACCL' 0.
2856 010612 112737 000005 001126 TST005: MOVB     #5,#TSTNUM
2857 010620 004737 005706          JSR      PC,.REWIND ;SET TEST #5
2858 010624 102422          BVS     99$         ;REWIND SLAVE
2859 010626 004737 006006          JSR      PC,READ    ;BRANCH IF ERROR ON REWIND
2860 010632 012702 010644          MOV     #1$,R2     ;SET RETURN PC FROM TIMER
```



```

2861 010636 004737 004730      JSR    PC,TIMON      ;TURN TIMER ON
2862 010642 005215              INC    (R5)          ;SET 'GO' BIT
2863
2864 010644 005765 000032      1$:    TST    TC(R5)      ;BRANCH WHEN 'ACCL' RESETS
2865 010650 100002              BPL    2$            ;
2866 010652 000163 005020      JMP    TIMER(R3)     ;GO TO TIMER & RETURN VIA R2
2867
2868 010656 004737 005602      2$:    JSR    PC,WAITRDY  ;WAIT FOR READY
2869 010662 102403              BVS    99$          ;BRANCH IF ERROR
2870 010664 004737 005056      JSR    PC,TIMOK      ;CHECK RECORDED TIME
2871 010670 000401              BR     100$         ;
2872
2873 010672 104400              99$:   HLT
2874 010674 104000              100$:  SCOPE
2875
2876 ;TEST 006 - READ START
2877 ;THIS TEST MEASURES TIME FROM 'GO'-1 TO 'ACCL'=0.
2878 010676 112737 000006 001126 1ST006: MOVB    #6,#TSTNUM ;SET TEST #6
2879 010704 004737 006062      JSR    PC,WRT.BK     ;WRITE A RECORD & BACK SPACE
2880 010710 102422              BVS    99$          ;
2881 010712 004737 006006      JSR    PC,READ
2882 010716 012702 010730      MOV    #1$,R2       ;SET RETURN PC FROM TIMER
2883 010722 004737 004730      JSR    PC,TIMON      ;TURN TIMER ON
2884 010726 005215              INC    (R5)          ;SET 'GO' BIT
2885
2886 010730 005765 000032      1$:    TST    TC(R5)      ;BRANCH WHEN 'ACCL' RESETS
2887 010734 100002              BPL    2$            ;
2888 010736 000163 005020      JMP    TIMER(R3)     ;GO TO TIMER & RETURN VIA R2
2889
2890 010742 004737 005602      2$:    JSR    PC,WAITRDY  ;WAIT FOR READY
2891 010746 102403              BVS    99$          ;BRANCH IF ERROR
2892 010750 004737 005056      JSR    PC,TIMOK      ;CHECK RECORDED TIME
2893 010754 000401              BR     100$         ;
2894
2895 010756 104400              99$:   HLT
2896 010760 104000              100$:  SCOPE
2897
2898 ;TEST 007 - READ SHUTDOWN
2899 ;THIS TEST MEASURES TIME FROM 'FC REG'-FRAME COUNT TO 'SWDN' 1.
2900 010762 112737 000007 001126 1ST007: MOVB    #7,#TSTNUM ;SET TEST #7
2901 010770 004737 006062      JSR    PC,WRT.BK     ;WRITE A RECORD & BACK SPACE
2902 010774 102430              BVS    99$          ;BRANCH IF ERROR
2903 010776 004737 006006      JSR    PC,READ
2904 011002 005215              INC    (R5)          ;SET 'GO' BIT
2905
2906 011004 022710 000400      1$:    CMP    #-FRMCNT,(R0) ;WAIT FOR FRAME COUNT TO
2907 011010 001404              BEQ    2$            ; # OF FRAMES WRITTEN
2908 011012 032711 040000      BIT    #ERR,(R1)     ;MONITOR ERROR BIT
2909 011016 001017              BNE    99$          ;
2910 011020 000771              BR     1$            ;
2911
2912 011022              2$:
2913 011022 004737 004730      JSR    PC,TIMON      ;TURN TIMER ON
2914 011026 010702              MOV    PC,R2         ;SET RETURN PC FROM TIMER
2915 011030 032711 000020      BIT    #SDWN,(R1)    ;BRANCH WHEN SDWN SETS
2916 011034 001002              BNF    3$            ;

```

```

2917 011036 000163 005020          JMP      TIMER(R3)          ;GO TO TIMER & RETURN VIA R2
2918
2919 011042 004737 005602          3$:     JSR      PC, WAITRDY
2920 011046 102403                    BVS     99$
2921 011050 004737 005056          JSR     PC, TIMOK
2922 011054 000401                    BR      100$
2923
2924 011056 104400          99$:    HLT
2925 011060 104000          100$:   SCOPE          ;REPORT ERROR
2926
2927
2928          ;TEST 010 - READ SETTLEDOWN
2929          ;THIS TEST MEASURES TIME FROM 'SWDN'-1 TO 'SWDN'=0.
2929 011062 112737 000010 001126  TST010: MOVB   #10, @TSTNUM          ;SET TEST #10
2930 011070 012702 011146          MOV     #4$, R2          ;SET RETURN PC FROM TIMER
2931 011074 004737 006062          JSR     PC, WRT.BK       ;WRITE A RECORD & BACK SPACE
2932 011100 102436                    BVS     99$
2933 011102 004737 006006          JSR     PC, READ
2934 011106 005215                    INC     (R5)          ;SET 'GO' BIT
2935
2936 011110 105711          1$:     TSTB   (R1)          ;WAIT FOR READY
2937 011112 100404                    BMI     2$
2938 011114 032711 040000          BIT     #ERR, (R1)       ;BRANCH WHEN SET
2939 011120 001026                    BNE     99$            ;CHECK ERROR BIT
2940 011122 000772                    BR      1$
2941
2942 011124 032711 000020          2$:     BIT     #SDWN, (R1)  ;WAIT FOR ASSERTION OF 'SDWN'
2943 011130 001004                    BNE     3$
2944 011132 032711 040000          BIT     #ERR, (R1)       ;MONITOR ERROR BIT
2945 011136 001017                    BNE     99$
2946 011140 000771                    BR      2$
2947
2948
2949 011142 004737 004730          3$:     JSR     PC, TIMON
2950 011146 032765 000020 000012  4$:     BIT     #SDWN, DS(R5)  ;TURN TIMER ON
2951 011154 001402                    BEQ     5$            ;WAIT FOR NEGATION OF SDWN
2952 011156 000163 005020          JMP     TIMER(R3)       ;GO TO TIMER & RETURN VIA R2
2953
2954 011162 004737 005602          5$:     JSR     PC, WAITRDY
2955 011166 102403                    BVS     99$
2956 011170 004737 005056          JSR     PC, TIMOK
2957 011174 000401                    BR      100$
2958
2959 011176 104400          99$:    HLT
2960 011200 104000          100$:   SCOPE
2961
2962
2963          ;TEST 011-READ REVERSE START
2964          ;THIS TEST MEASURES TIME FROM 'GO'-1 TO 'ACCL'=0.
2965 011202 112737 000011 001126  TST011: MOVB   #11, @TSTNUM
2966 011210 012702 011246          MOV     #1$, R2          ;SET RETURN PC FROM TIMER
2967 011214 004737 005770          JSR     PC, WRITE       ;WRITE A RECORD
2968 011220 005215                    INC     (R5)            ;SET 'GO' BIT
2969 011222 004737 005602          JSR     PC, WAITRDY
2970 011226 102422                    BVS     99$
2971 011230 004737 005310          JSR     PC, DELAY
2972 011234 004737 006024          JSR     PC, REVRD       ;WAIT FOR TAPE MOTION TO STOP
  
```

```
2973 011240 004737 004730      JSR    PC,TIMON      ;TURN TIMER ON
2974 011244 005215              INC    (R5)          ;SET 'GO' BIT
2975
2976 011246 005765 000032      1$:   TST    TC(R5)      ;BRANCH WHEN 'ACCL' = 0
2977 011252 100002              BPL    2$
2978 011254 000163 005020      JMP    TIMER(R3)     ;GO TO TIMER & RETURN VIA R2
2979
2980 011260 004737 005602      2$:   JSR    PC,WAITRDY
2981 011264 102403              BVS    99$           ;BRANCH IF ERROR
2982 011266 004737 005056      JSR    PC,TIMOK
2983 011272 000401              BR     100$
2984
2985 011274 104400      99$:   HLT
2986 011276 104000      100$:  SCOPE
2987
2988      ;TEST 012-READ REVERSE SHUTDOWN
2989      ;THIS TEST MEASURES TIME FROM 'FC REG' = FRAME COUNT TO 'SDWN'=1.
2990 011300 112737 000012 001126  TST012: MOVB   #12,@TSTNUM
2991 011306 012702 011356      MOV    #3$,R2        ;SET RETURN PC FROM TIMER
2992 011312 004737 005770      JSR    PC,WRITE      ;WRITE A RECORD
2993 011316 005215              INC    (R5)          ;SET 'GO' BIT
2994 011320 004737 005602      JSR    PC,WAITRDY
2995 011324 102427              BVS    99$
2996 011326 004737 006024      JSR    PC,REVRD
2997 011332 005215              INC    (R5)          ;SET 'GO' BIT
2998
2999 011334 022710 000400      1$:   CMP    #-FRMCNT,(R0) ;BRANCH WHEN FRAME COUNT
3000 011340 001404              BEQ    2$            ;- # OF RECORD WRITTEN
3001 011342 032711 040000      BIT    #ERR,(R1)    ;MONITOR ERROR BIT IN 'DS' REG
3002 011346 001016              BNE    99$
3003 011350 000771              BR     1$
3004
3005      2$:
3006 011352 004737 004730      JSR    PC,TIMON      ;TURN TIMER ON
3007 011356 032711 000020      3$:   BIT    #SDWN,(R1)  ;BRANCH WHEN SDWN SETS
3008 011362 001002              BNE    4$
3009 011364 000163 005020      JMP    TIMER(R3)     ;GO TO TIMER & RETURN VIA R2
3010
3011 011370 004737 005602      4$:   JSR    PC,WAITRDY  ;WAIT FOR READY
3012 011374 102403              BVS    99$
3013 011376 004737 005056      JSR    PC,TIMOK
3014 011402 000401              BR     100$
3015
3016 011404 104400      99$:   HLT
3017 011406 104000      100$:  SCOPE
3018
3019      ;TEST 013-READ REVERSE SETTLEDOWN
3020      ;THIS TEST MEASURES TIME FROM 'SDWN'=1 TO 'SDWN'=0.
3021 011410 112737 000013 001126  TST013: MOVB   #13,@TSTNUM
3022 011416 012702 011502      MOV    #4$,R2        ;SET RETURN PC FROM TIMER
3023 011422 004737 005770      JSR    PC,WRITE      ;WRITE A RECORD
3024 011426 005215              INC    (R5)          ;SET 'GO' BIT
3025 011430 004737 005602      JSR    PC,WAITRDY
3026 011434 102435              BVS    99$
3027 011436 004737 006024      JSR    PC,REVRD
3028 011442 005215              INC    (R5)          ;SET 'GO' BIT
```

```

3029
3030 011444 105711          1$:   TSTB   (R1)           ;BRANCH WHEN
3031 011446 100404          BMI    2$           ;READY SETS
3032 011450 032711 040000  BIT    #ERR,(R1)
3033 011454 001025          BNE    99$
3034 011456 000772          BR     1$
3035
3036 011460 032711 000020  2$:   BIT    #SDWN,(R1)
3037 011464 001004          BNE    3$
3038 011466 032711 040000  BIT    #ERR,(R1)
3039 011472 001016          BNE    99$
3040 011474 000771          BR     2$
3041
3042 011476
3043 011476 004737 004730  3$:   JSR    PC,TIMON      ;TURN TIMER ON
3044 011502 032711 000020  4$:   BIT    #SDWN,(R1)  ;BRANCH WHEN SWDN = 0
3045 011506 001402          BEQ    5$
3046 011510 000163 005020  JMP    TIMER(R3)     ;GO TO TIMER & RETURN VIA R2
3047
3048 011514 004737 005602  5$:   JSR    PC,WAITRDY   ;WAIT FOR READY
3049 011520 102403          BVS    99$
3050 011522 004737 005056  JSR    PC,TIMOK
3051 011526 000401          BR     100$
3052
3053 011530 104400          99$:   HLT
3054 011532 104000          100$:  SCOPE
3055
3056          ;REWIND DRIVE
3057          A:
3058 011534 004737 005706  JSR    PC,.REWIND    ;REWIND SLAVE
3059 011540 102401          BVS    99$           ;BRANCH IF ERROR ON REWIND
3060 011542 102002          BVC    100$
3061 011544 104400          99$:   HLT
3062 011546 000772          BR     A
3063 011550          100$:
3064
3065          ;TEST 014-TURN AROUND DELAY (FORWARD-REVERSE)
3066          ;THIS TEST MEASURES TIME FROM 'GO'-1 (READ REVERSE) TO 'ACCL'=0
3067 011550 112737 000014 001126 TST014: MOVB   #14,@#TSTNUM
3068 011556 012702 011610  MOV    #2$,R2        ;SET RETURN PC FROM TIMER
3069 011562 004737 005770  JSR    PC,WRITE      ;WRITE A RECORD
3070 011566 005215          INC    (R5)          ;SET 'GO' BIT
3071 011570 004737 005602  JSR    PC,WAITRDY
3072 011574 102420          BVS    99$
3073
3074 011576 004737 006024  1$:   JSR    PC,REVRD    ;READ THE RECORD (REVERSE)
3075 011602 004737 004730  JSR    PC,TIMON      ;TURN TIMER ON
3076 011606 005215          INC    (R5)          ;SET 'GO' BIT
3077
3078 011610 005765 000032  2$:   TST    TC(R5)     ;WAIT FOR 'ACCL' - 0
3079 011614 100002          BPL    3$
3080 011616 000163 005020  JMP    TIMER(R3)     ;GO TO TIMER & RETURN VIA R2
3081
3082 011622 004737 005602  3$:   JSR    PC,WAITRDY
3083 011626 102403          BVS    99$
3084 011630 004737 005056  JSR    PC,TIMOK

```

```
3085 011634 000401 BR 100$
3086
3087 011636 104400 99$: HLT
3088 011640 104000 100$: SCOPE
3089
3090 ;TEST 015- TURN AROUND DELAY (REVERSE-FORWARD)
3091 ;THIS TEST MEASURES TIME FROM 'GO'=1 (READ) TO 'ACCL'=0.
3092 011642 112737 000015 001126 TST015: MOVB #15,@TSTNUM
3093 011650 012702 011716 MOV #2$,R2 ;SET RETURN PC FROM TIMER
3094 011654 004737 005770 JSR PC,WRITE ;WRITE A RECORD
3095 011660 005215 INC (R5) ;SET 'GO' BIT
3096 011662 004737 005602 JSR PC,WAITRDY ;WAIT FOR READY
3097 011666 102426 BVS 99$
3098 011670 004737 006024 JSR PC,REVRD ;READ A RECORD IN THE
3099 011674 005215 INC (R5) ;SET 'GO' BIT
3100
3101 011676 004737 005602 JSR PC,WAITRDY
3102 011702 102420 BVS 99$
3103
3104 011704 004737 006006 1$: JSR PC,READ ;READ RECORD FORWARD
3105 011710 004737 004730 JSR PC,TIMON ;TURN TIMER ON
3106 011714 005215 INC (R5) ;SET 'GO' BIT
3107
3108 011716 005765 000032 2$: TST TC(R5) ;WAIT FOR 'ACCL' - 0
3109 011722 100002 BPL 3$
3110 011724 000163 005020 JMP TIMER(R3) ;GO TO TIMER & RETURN VIA R2
3111
3112 011730 004737 005602 3$: JSR PC,WAITRDY
3113 011734 102403 BVS 99$
3114 011736 004737 005056 JSR PC,TIMOK
3115 011742 000401 BR 100$
3116
3117 011744 104400 99$: HLT
3118 011746 104000 100$: SCOPE
3119
3120 ;TEST 016-GAP SIZE (STOP HALF)
3121 011750 112737 000016 001126 TST016: MOVB #16,@TSTNUM
3122 011756 012702 012014 MOV #1$,R2 ;SET RETURN PC FROM TIMER
3123 011762 004737 005770 JSR PC,WRITE ;WRITE A RECORD
3124 011766 005215 INC (R5) ;SET 'GO' BIT
3125 011770 004737 005602 JSR PC,WAITRDY
3126 011774 102421 BVS 99$
3127 011776 004737 005310 JSR PC,DELAY ;DELAY 350 MS
3128 012002 004737 006024 JSR PC,REVRD ;READ REVERS RECORD
3129 012006 004737 004730 JSR PC,TIMON ;TURN TIMER ON
3130 012012 005215 INC (R5) ;SET 'GO' BIT
3131
3132 012014 005710 1$: TST (R0) ;WAIT FOR FRAME COUNT > 0
3133 012016 001002 BNE 2$
3134 012020 000163 005020 JMP TIMER(R3) ;GO TO TIMER & RETURN VIA R2
3135
3136 012024 004737 005602 2$: JSR PC,WAITRDY ;WAIT FOR READY BIT TO SET
3137 012030 102403 BVS 99$
3138 012032 004737 005056 JSR PC,TIMOK ;CHECK TIME
3139 012036 000401 BR 100$
3140
```

```
3141 012040 104400          99$:   HLT
3142 012042 104000          100$:  SCOPE
3143
3144          ;TEST 017-GAP SIZE (START HALF)
3145 012044 112737 000017 001126 TST017: MOVB  #17,@TSTNUM
3146 012052 012702 012124          MOV  #1$,R2          ;SET RETURN PC FROM TIMER
3147 012056 004737 005770          JSR  PC,WRITE        ;WRITE A RECORD
3148 012062 005215          INC  (R5)           ;SET 'GO' BIT
3149 012064 004737 005602          JSR  PC,WAITRDY      ;WAIT FOR READY
3150 012070 102427          BVS  99$
3151 012072 004737 006024          JSR  PC,REVRD        ;READ REVERSE THE RECORD
3152 012076 005215          INC  (R5)           ;SET 'GO' BIT
3153 012100 004737 005602          JSR  PC,WAITRDY      ;WAIT FOR READY
3154 012104 102421          BVS  99$           ;BRANCH ON ERROR
3155 012106 004737 005310          JSR  PC,DELAY        ;WAIT FOR TAPE MOTION TO STOP
3156 012112 004737 006006          JSR  PC,READ         ;READ RECORD
3157 012116 004737 004730          JSR  PC,TIMON        ;TURN TIMER ON
3158 012122 005215          INC  (R5)           ;SET 'GO' BIT
3159
3160 012124 005710          1$:   TST  (R0)          ;WAIT FOR FRAME COUNT > 0
3161 012126 001002          BNE  2$
3162 012130 000163 005020          JMP  TIMER(R3)      ;GO TO TIMER & RETURN VIA R2
3163
3164 012134 004737 005602          2$:   JSR  PC,WAITRDY      ;WAIT FOR READY
3165 012140 102403          BVS  99$
3166 012142 004737 005056          JSR  PC,TIMOK        ;CHECK TIME
3167 012146 000401          BR   100$
3168
3169 012150 104400          99$:   HLT
3170 012152 104000          100$:  SCOPE
3171
3172          ;TEST 020- GAP SIZE (INTERRECORD)
3173          ;THIS TEST MEASURES TIME FROM 'GO'-1 TO 'FC REG' >0.
3174 012154 112737 000020 001126 TST020: MOVB  #20,@TSTNUM
3175 012162 012702 012244          MOV  #1$,R2          ;SET RETURN PC FROM TIMER
3176 012166 004737 005770          JSR  PC,WRITE        ;WRITE A RECORD
3177 012172 005215          INC  (R5)           ;SET 'GO' BIT
3178 012174 004737 005602          JSR  PC,WAITRDY      ;WAIT FOR READY
3179 012200 102433          BVS  99$
3180 012202 004737 005770          JSR  PC,WRITE        ;WRITE SECOND RECORD
3181 012206 005215          INC  (R5)           ;SET 'GO' BIT
3182 012210 004737 005602          JSR  PC,WAITRDY      ;WAIT FOR READY
3183 012214 102425          BVS  99$
3184 012216 004737 006024          JSR  PC,REVRD        ;READ REVERSE SECOND RECORD
3185 012222 005215          INC  (R5)           ;SET 'GO' BIT
3186 012224 004737 005602          JSR  PC,WAITRDY      ;WAIT FOR READY
3187 012230 102417          BVS  99$
3188 012232 004737 006024          JSR  PC,REVRD        ;READ REVERSE FIRST RECORD
3189 012236 004737 004730          JSR  PC,TIMON        ;TURN TIMER ON
3190 012242 005215          INC  (R5)           ;SET 'GO' BIT
3191
3192 012244 005710          1$:   TST  (R0)          ;WAIT FOR FRAME COUNT > 0
3193 012246 001002          BNE  2$
3194 012250 000163 005020          JMP  TIMER(R3)      ;GO TO TIMER & RETURN VIA R2
3195
3196 012254 004737 005602          2$:   JSR  PC,WAITRDY      ;WAIT FOR READY
```

3197 012260 102403
 3198 012262 004737 005056
 3199 012266 000401

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

3200
 3201 012270 104400
 3202 012272 104000

99\$: HLT
 100\$: SCOPE

3203
 3204
 3205
 3206
 3207
 3208
 3209
 3210
 3211
 3212
 3213
 3214
 3215

```

: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.
  
```

3216 012274 112737 000021 001126
 3217 012302 012702 012440
 3218 012306 004737 005706
 3219 012312 102530
 3220 012314 005037 001120
 3221 012320 012700 000021
 3222 012324 004737 005770
 3223 012330 005215
 3224 012332 004737 005602
 3225 012336 102516
 3226 012340 004737 005340
 3227 012344 063737 001014 001120
 3228 012352 005300
 3229 012354 001363

```

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.,R0 ;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 R0 ;DECREMENT RECORDS WRITTEN COUNT
BNE 1$
  
```

3230
 3231 012356 012700 000021
 3232 012362 004737 006024
 3233 012366 005215
 3234 012370 004737 005602
 3235 012374 102477
 3236 012376 005300
 3237 012400 001370

```

MOV #17.,R0 ;SET # OF RECS. TO REVERSE READ
2$: JSR PC,REVRD ;REVERSE READ 17. RECORDS
INC (R5) ;SET 'GO' BIT
JSR PC,WAITRDY ;WAIT FOR READY
BVS 99$
DEC R0 ;DECREMENT RECORD COUNT
BNE 2$
  
```

3238
 3239 012402 012700 000020
 3240 012406 012701 001060
 3241 012412 004737 006006
 3242 012416 005215

```

MOV #16.,R0 ;SET # OF RECORDS TO READ
MOV #GAPTBL,R1 ;SET PTR TO GAP TABLE FOR TEST
3241: JSR PC,READ ;READ A RECORD
INC (R5) ;SET 'GO' BIT
  
```

3243
 3244 012420 004737 005602
 3245 012424 102463
 3246 012426 004737 006006
 3247 012432 004737 004730
 3248 012436 005215

```

33$: JSR PC,WAITRDY ;WAIT FOR READY
BVS 99$
JSR PC,READ ;READ NEXT RECORD
JSR PC,TIMON ;TURN TIMER ON
INC (R5) ;SET 'GO' BIT
  
```

3249
 3250 012440 005765 000006
 3251 012444 001002
 3252 012446 000163 005020

```

4$: TST FC(R5) ;WAIT FOR FRAME COUNT > 0
BNE 5$
JMP TIMER(R3) ;GO TO TIMER & RETURN VIA R2
  
```

```

3253
3254 012452 004737 005602      5$: JSR PC, WAITRDY      ;WAIT FOR READY
3255 012456 102446              BVS 99$
3256 012460 010421              MOV R4, (R1)+      ;STORE TIME IN GAPTBL
3257 012462 005300              DEC R0              ;DECREMENT # OF RECORDS READ
3258 012464 001355              BNE 3$
3259
3260 012466 105037 001124      CLRB @#GAP          ;SET GAP # 0
3261 012472 012700 000020      MOV #16., R0
3262 012476 012701 001060      MOV #GAPTBL, R1
3263
3264 012502 012104      6$: MOV (R1)+, R4      ;GET GAP TICK COUNT
3265 012504 004737 005166      JSR PC, GAPOK      ;CHECK TIME
3266 012510 105237 001124      INCB @#GAP         ;INCREMENT GAP #
3267 012514 122737 000020 001124 CMPB #16., @#GAP   ;BRANCH IF ALL GAPS NOT CHECKED
3268 012522 001367      BNE 6$
3269
3270 012524 012700 000020      MOV #16., R0      ;SETUP TO AVERAGE GAP SIZES
3271 012530 012701 001060      MOV #GAPTBL, R1   ;SET PTR TO TABLE
3272 012534 005002      CLR R2            ;CLEAR 'SUM' REGISTERS
3273 012536 005003      CLR R3
3274 012540 062102      7$: ADD (R1)+, R2    ;ADD ALL GAP SIZES TOGETHER
3275 012542 005503      ADC R3
3276 012544 005300      DEC R0
3277 012546 001374      BNE 7$
3278 012550 012700 000004      MOV #4., R0       ;NOW DIVIDE BY 16.
3279 012554 006203      8$: ASR R3           ;BY SHIFTING 4 PLACES RIGHT
3280 012556 006002      ROR R2
3281 012560 005300      DEC R0
3282 012562 001374      BNE 8$
3283 012564 010204      MOV R2, R4        ;MOVE AVERAGED TIMES TO R4
3284 012566 004737 005056      JSR PC, TIMOK     ;CHECK AVERAGED TIMES
3285 012572 000401      BR 100$
3286
3287 012574 104400      99$: HLT
3288 012576 104000      100$: SCOPE
3289
3290
3291
3292
3293      ;TEST 022-DATA TIME (800BPI)
3294      ;THIS TEST MEASURES THE TIME FROM FC REG >-6400 TO 'RDY' 1.
3294 012600 112737 000022 001126 TST022: MOVB #022, @#TSTNUM
3295 012606 012702 012666      MOV #3$, R2      ;SET RETURN PC FROM TIMER
3296 012612 004737 005706      JSR PC, .REWIND   ;REWIND SLAVE
3297 012616 102442      BVS 99$          ;BRANCH IF ERROR ON REWIND
3298 012620 052765 001700 000032 BIS #NORM11, TC(R5) ;SET 800 BPI
3299 012626 004337 006124      JSR R3, TMCMD    ;WRITE 3200. WORD RECORD
3300 012632 016650      .WORD WTBUF
3301 012634 171600      .WORD -3200.
3302 012636 163400      .WORD -6400.
3303 012640 000060      .WORD WFW
3304 012642 005215      INC (R5)         ;SET 'GO' BIT
3305
3306 012644 022710 163400      1$: CMP #-6400., (R0) ;WAIT FOR WRITING TO START
3307 012650 001004      BNE 2$
3308 012652 032711 040000      BIT #ERR, (R1)   ;MONITOR ERROR BIT
  
```


3309	012656	001022			BNE	99\$	
3310	012660	000771			BR	1\$	
3311							
3312	012662			2\$:			
3313	012662	004737	004730		JSR	PC,TIMON	:TURN TIMER ON
3314	012666	105711		3\$:	TSTB	(R1)	:BRANCH WHEN READY SETS
3315	012670	100402			BMI	4\$	
3316	012672	000163	005020		JMP	TIMER(R3)	:GO TO TIMER & RETURN VIA R2
3317							
3318	012676	012700	000003	4\$:	MOV	#3,R0	:SET SHIFT COUNT
3319	012702	006204		5\$:	ASR	R4	
3320	012704	005300			DEC	R0	
3321	012706	001375			BNE	5\$	
3322	012710	004737	005602		JSR	PC,WAITRDY	
3323	012714	102403			BVS	99\$	
3324	012716	004737	005056		JSR	PC,TIMOK	:CHECK TIME
3325	012722	000401			BR	100\$	
3326							
3327	012724	104400		99\$:	HLT		
3328	012726	104000		100\$:	SCOPE		
3329							
3330							
3331							:TEST 023-DATA TIME (1600BPI)
3332	012730	112737	000023	001126			:THIS TEST MEASURES THE TIME FROM FC REG >-6400 TO 'RDY' - 1.
3333	012736	012702	013024		TST023: MOV	#023,#TSTNUM	
3334	012742	004737	005706		MOV	#3\$,R2	:SET RETURN PC FROM TIMER
3335	012746	102442			JSR	PC,.REWIND	:REWIND SLAVE
3336	012750	042765	003700	000032	BVS	99\$:BRANCH IF ERROR ON REWIND
3337	012756	052765	002300	000032	BIC	#3700,TC(R5)	:CLEAR CURRENT DENSITY
3338	012764	004337	006124		BIS	#PE1600,TC(R5)	:SET 1600 BPI
3339	012770	016650			JSR	R3,TCMD	:WRITE 3200. WORD RECORD
3340	012772	171600			.WORD	WTBUF	
3341	012774	163400			.WORD	-3200.	
3342	012776	000060			.WORD	-6400.	
3343	013000	005215			.WORD	WFD	
3344					INC	(R5)	:SET 'GO' BIT
3345	013002	022710	163400		1\$:	CMP	#-6400.,(R0)
3346	013006	001004			BNE	2\$:BRANCH WHEN WRITING STARTS
3347	013010	032711	040000		BIT	#ERR,(R1)	:MONITOR ERROR BIT
3348	013014	001017			BNE	99\$	
3349	013016	000771			BR	1\$	
3350							
3351	013020			2\$:			
3352	013020	004737	004730		JSR	PC,TIMON	:TURN TIMER ON
3353	013024	105711		3\$:	TSTB	(R1)	:BRANCH WHEN READY SETS
3354	013026	100402			BMI	4\$	
3355	013030	000163	005020		JMP	TIMER(R3)	:GO TO TIMER & RETURN VIA R2
3356							
3357	013034	006204		4\$:	ASR	R4	:DIVIDE TIME BY 4
3358	013036	006204			ASR	R4	
3359	013040	004737	005602		JSR	PC,WAITRDY	
3360	013044	102403			BVS	99\$	
3361	013046	004737	005056		JSR	PC,TIMOK	:CHECK TIME
3362	013052	000401			BR	100\$	
3363							
3364	013054	104400		99\$:	HLT		

```
3365 013056 104000 100$: SCOPE
3366
3367 :TEST 024-ERASE
3368 :THIS TEST MEASURES TIME FROM 'GO'=1 TO 'RDY'=1.
3369 013060 112737 000024 001126 TST024: MOVB #24,@TSTNUM
3370 013066 012702 013150 MOV #2$,R2 ;SET RETURN PC FROM TIMER
3371 013072 004737 005706 JSR PC,REWIND ;REWIND SLAVE
3372 013076 102436 BVS 99$ ;BRANCH IF ERROR ON REWIND
3373 013100 004737 005544 JSR PC,RHINIT ;SET NRZ
3374 013104 004737 005770 JSR PC,WRITE ;WRITE A RECORD
3375 013110 005215 INC (R5) ;SET 'GO' BIT
3376 013112 004737 005602 JSR PC,WAITRDY
3377 013116 102426 BVS 99$
3378 013120 012737 013126 001002 MOV #1$,@SCPADR
3379 013126 004337 006124 1$: JSR R3,@TMCMD
3380 013132 000000 .WORD 0
3381 013134 000000 .WORD 0
3382 013136 000000 .WORD 0
3383 013140 000024 .WORD ERASE
3384 013142 004737 004730 JSR PC,TIMON ;TURN TIMER ON
3385 013146 005215 INC (R5) ;SET 'GO' BIT
3386
3387 013150 105711 2$: TSTB (R1) ;BRANCH WHEN READY SETS
3388 013152 100402 BMI 3$
3389 013154 000163 005020 JMP TIMER(R3) ;GO TO TIMER & RETURN VIA R2
3390
3391 013160 004737 005602 3$: JSR PC,WAITRDY
3392 013164 102403 BVS 99$
3393 013166 004737 005056 JSR PC,TIMOK
3394 013172 000401 BR 100$
3395
3396 013174 104400 99$: HLT
3397 013176 104000 100$: SCOPE
3398
3399 :TEST 025 TAPE MARK
3400 :THIS TEST MEASURES TIME FROM 'GO'-1 TO 'RDY'-1.
3401 013200 112737 000025 001126 TST025: MOVB #25,@TSTNUM
3402 013206 012702 013250 MOV #1$,R2 ;SET RETURN PC FROM TIMER
3403 013212 004737 005770 JSR PC,WRITE ;WRITE A RECORD
3404 013216 005215 INC (R5) ;SET 'GO' BIT
3405 013220 004737 005602 JSR PC,WAITRDY
3406 013224 102423 BVS 99$
3407 013226 004337 006124 JSR R3,@TMCMD
3408 013232 000000 .WORD 0
3409 013234 000000 .WORD 0
3410 013236 000000 .WORD 0
3411 013240 000026 .WORD WFMK
3412 013242 004737 004730 JSR PC,TIMON ;TURN TIMER ON
3413 013246 005215 INC (R5) ;SET 'GO' BIT
3414
3415 013250 105711 1$: TSTB (R1) ;BRANCH WHEN READY SETS
3416 013252 100402 BMI 2$
3417 013254 000163 005020 JMP TIMER(R3) ;GO TO TIMER & RETURN VIA R2
3418
3419 013260 004737 005602 2$: JSR PC,WAITRDY
3420 013264 102403 BVS 99$
```

CZTEEB0 TM03-TE16/TU77 DFT
CZTEEB.P11 17-NOV-78 10:11

MACY11 30A(1052) 21-DEC-78 13:20 J 6
START OF TESTS PAGE 74

SEQ 0074

3421 013266 004737 005056
3422 013272 000401
3423
3424 013274 104400
3425 013276
3426 013276 004737 005706
3427 013302 102774
3428 013304 104000
3429

JSR PC TIMOK
BR 100\$
99\$: HLT
100\$:
JSR PC .REWIND :REWIND SLAVE
BVS 99\$:BRANCH IF ERROR ON REWIND
SCOPE

```

3430 013306 032777 002000 165464 FINISH: BIT #SW10,@SWR ;DO NOT SPACE PAPER
3431 013314 001011 BNE 2$ ;IF USER SELECTED NO OUTPUT
3432 013316 005737 006372 TST CHNFLG ;OR IF IN CHAIN MODE
3433 013322 001006 BNE 2$
3434 013324 012700 000012 MOV #10.,R0 ;SET LINE FEED COUNT
3435 013330 000004 001402 1$: TYPE,CRLF
3436 013334 005300 DEC R0
3437 013336 001374 BNE 1$
3438
3439
3440 013340 105237 001005 2$: INCB @#SLVNUM ;SET NEXT SLAVE #
3441 013344 005237 001006 INC @#SLVPTR ;AND ITS POINTER
3442 013350 122737 000010 001005 CMPB #8.,@#SLVNUM ;BRANCH IF LAST SLAVE (7)
3443 013356 001402 BEQ 3$
3444 013360 000137 007420 JMP @#BEGIN ;BEGIN TEST ON NEXT SLAVE
3445 013364 105037 001005 3$: CLRB @#SLVNUM ;SET SLAVE #0
3446 013370 105237 001004 INCB @#DRVNUM ;AND INCREMENT DRIVE #
3447 013374 122737 000010 001004 CMPB #8.,@#DRVNUM ;AND CHECK IF LAST DRIVE
3448 013402 001402 BEQ END
3449 013404 000137 007420 JMP @#BEGIN
3450
3451 013410 105737 001132 END: TSTB @#UNTFND ;BRANCH IF A UNIT WAS FOUND
3452 013414 001004 BNE 1$
3453 013416 000004 015021 TYPE,E.UNIT
3454 013422 000137 006274 JMP @#INIT
3455 013426 105237 001134 1$: INCB @#PSCNT ;INCREMENT PASS COUNT
3456 013432 000004 014253 TYPE,M.EOP
3457 013436 113702 001134 MOVB @#PSCNT,R2 ;GET PASSCOUNT
3458 013442 004737 003134 JSR PC,TYPCT ;AND TYPE IT
3459 013446 000004 001402 TYPE,CRLF
3460 013452 013700 000042 MOV @#42,R0 ;GET ACT11 RETURN ADDRESS
3461 013456 001405 BEQ HERE ;BRANCH IF NOT ACT11
3462 013460 000005 RESET
3463 013462 004710 $ENDAD: JSR PC,(R0)
3464 013464 000240 NOP
3465 013466 000240 NOP
3466 013470 000240 NOP
3467 013472 000240 HERE: NOP
3468 013474 005737 006372 TST CHNFLG ;BRANCH IF CHAIN MODE
3469 013500 001004 BNE 1$
3470 013502 032777 000100 165270 BIT #SW06,@SWR ;BRANCH IF NOT CONTINOUS LOOP
3471 013510 001402 BEQ 2$
3472 013512 000137 007372 1$: JMP @#RSTRT ;RESTART
3473 013516 000000 2$: HALT
3474 013520 000005 RESET
3475 013522 000137 006274 JMP @#INIT ;RESTART
  
```

```
3476 ;SKEW TAPE TIMING TESTS
3477 ;THE FOLLOWING TESTS REQUIRE A SPECIALLY WRITTEN 800 BPI SKEW TAPE
3478 013526 012737 013534 001002 SKEWTST:MOV #TST026,@#SCPADR ;SET SCOPE POINTER
3479
3480 ;TEST 026- SKEW TAPE SPEED TEST-FORWARD
3481 ;THIS TEST READS 32'' OF TAPE (26400.-800. = 25600. FRAMES), THEN
3482 ;DIVIDES TIME BY 32. TO GET TIME TO READ 1'' (800. FRAMES) OF TAPE.
3483 013534 112737 000026 001126 TST026: MOV #26,@#TSTNUM
3484 013542 012702 013630 MOV #2$,R2 ;SET RETURN PC FROM TIMER
3485 013546 004737 005706 JSR PC,REWIND ;REWIND SLAVE
3486 013552 102445 BVS 99$ ;BRANCH IF ERROR ON REWIND
3487 013554 052765 001700 000032 BIS #NORM11,TC(R5) ;SET 800 BPI
3488 013562 052765 000010 000010 BIS #BAI,CS2(R5) ;INHIBIT BUS ADDRESS INCREMENT
3489 013570 004337 006124 JSR R3,@#TMCMD ;READ 32'' OF TAPE-FORWARD
3490 013574 016650 .WORD RDBUF
3491 013576 177777 .WORD -1.
3492 013600 063440 0$: .WORD 26400. ;FRAME COUNT
3493 013602 000070 .WORD RDFWD
3494 013604 005215 INC (R5) ;SET 'GO' BIT
3495
3496 013606 032765 075027 000014 1$: BIT #HRDERR,ER(R5) ;BRANCH IF ANY HARD ERROR BIT SETS
3497 013614 001024 BNE 99$
3498 013616 022710 001440 CMP #800.,(R0) ;WAIT FOR FIRST 800 FRAMES
3499 013622 101371 BHI 1$ ;TO BE READ
3500
3501 013624 004737 004730 JSR PC,TIMON ;TURN TIMER ON
3502 013630 023710 013600 2$: CMP @#10$, (R0) ;WAIT FOR READING TO FINISH
3503 013634 103402 BLO 3$
3504 013636 000163 005020 JMP TIMER(R3) ;GO TO TIMER & RETURN VIA R2
3505
3506 013642 012700 000005 3$: MOV #5,R0 ;DIVIDE TIME BY 32.
3507 013646 006204 4$: ASR R4
3508 013650 005300 DEC R0
3509 013652 001375 BNE 4$
3510 013654 004737 005544 JSR PC,RHINI' ;INIT DRIVE
3511 013660 004737 005056 JSR PC,TIMOK ;CHECK TIME
3512 013664 000401 BR 100$
3513
3514 013666 104400 99$: HLT
3515 013670 104000 100$: SCOPE
3516
3517 ;TEST 027-SKEW TAPE SPEED TEST-REVERSE
3518 ;THIS TEST READS FORWARD 40'' (32000. FRAMES) OF TAPE, THEN READS REVERSE
3519 ;32'' (26400.-800. = 25600. FRAMES) OF TAPE. THE TIME IS THEN DIVIDED BY
3520 ;32. TO GET TIME TO READ 1'' (800. FRAMES) OF TAPE.
3521 013672 112737 000027 001126 TST027: MOV #27,@#TSTNUM
3522 013700 012702 014036 MOV #3$,R2 ;SET RETURN PC FROM TIMER
3523 013704 004737 005706 JSR PC,REWIND ;REWIND SLAVE
3524 013710 102471 BVS 99$ ;BRANCH IF ERROR ON REWIND
3525 013712 052765 001700 000032 BIS #NORM11,TC(R5)
3526 013720 052765 000010 000010 BIS #BAI,CS2(R5)
3527 013726 004337 006124 JSR R3,@#TMCMD ;READ FORWARD 32000. FRAMES
3528 013732 016650 .WORD RDBUF
3529 013734 177777 .WORD -1. ;WORD COUNT
3530 013736 076400 10$: .WORD 32000. ;FRAME COUNT
3531 013740 000070 .WORD RDFWD ;READ FORWARD
```

```

3532 013742 005215          INC      (R5)          ;SET 'GO' BIT
3533
3534 013744 032711 040000    1$:     BIT      #ERR,(R1)      ;BRANCH IF ERROR BITS SETS
3535 013750 001051          BNE     99$
3536 013752 023710 013736    CMP     @#10$, (R0)
3537 013756 101372          BHI     1$
3538
3539 013760 004737 005544          JSR     PC,RHINIT      ;INIT DRIVE
3540 013764 004737 005310          JSR     PC,DELAY      ;WAIT FOR TAPE MOTION TO STOP
3541 013770 052765 000010 000010    BIS     #BAI,CS2(R5)  ;INHIBIT BUS ADDRESS INCREMENT
3542 013776 004337 006124          JSR     R3,@#TMCMD    ;READ REVERSE 32' OF TAPE
3543 014002 016650          .WORD  RDBUF         ;READ BUFFER
3544 014004 177777          .WORD  -1            ;WORD COUNT
3545 014006 063440          11$:   .WORD  26400.    ;FRAME COUNT
3546 014010 000076          .WORD  RDREV        ;READ REVERSE
3547 014012 005215          INC     (R5)         ;SET 'GO' BIT
3548
3549 014014 032765 075027 000014  2$:     BIT      #HRDERR,ER(R5) ;EXIT TEST IF ERROR BIT SETS
3550 014022 001024          BNE     99$
3551 014024 022710 001440          CMP     #800., (R0)  ;WAIT FOR FIRST 800 FRAMES
3552 014030 101371          BHI     2$          ;TO BE READ
3553
3554 014032 004737 004730          JSR     PC,TIMON      ;TURN TIMER ON
3555 014036 023710 014006          3$:     CMP     @#11$, (R0) ;WAIT FOR ALL FRAMES TO BE READ
3556 014042 103402          BLO     4$
3557 014044 000163 005020          JMP     TIMER(R3)    ;GO TO TIMER & RETURN VIA R2
3558
3559 014050 012700 000005          4$:     MOV     #5,R0    ;DIVIDE TIME BY 32.
3560 014054 006204          5$:     ASR     R4
3561 014056 005300          DEC     R0
3562 014060 001375          BNE     5$
3563 014062 004737 005544          JSR     PC,RHINIT
3564 014066 004737 005056          JSR     PC,TIMOK
3565 014072 000401          BR     100$
3566
3567 014074 104400          99$:    HLT
3568 014076          100$:
3569 014076 004737 005706          JSR     PC,.REWIND   ;REWIND SLAVE
3570 014102 102774          BVS     99$         ;BRANCH IF ERROR ON REWIND
3571 014104 104000          SCOPE
3572
3573 014106 000137 013306          JMP     @#FINISH
3574
3575
3576

```

3577
3578
3579 014112 005015 046524 031460
3580 014120 052055 030505 027466
3581 014126 052524 033467 042040
3582 014134 044522 042526 043040
3583 014142 047125 052103 047511
3584 014150 020116 044524 042515
3585 014156 020122 041450 052132
3586 014164 042505 030102 051
3587 014171 015 052012 050131
3588 014176 020105 041474 037122
3589 014204 052040 020117 042524
3590 014212 046522 047111 052101
3591 014220 020105 042522 050123
3592 014226 047117 042523 023040
3593 014234 057040 020103 047524
3594 014242 051040 051505 040524
3595 014250 052122 000
3596 014253 015 042412 042116
3597 014260 047440 020106 040520
3598 014266 051523 000040
3599 014272 005015 040510 042122
3600 014300 040527 042522 051440
3601 014306 051127 044440 020116
3602 014314 051525 006505 000012
3603 014322 005015 042522 047515
3604 014330 042526 052040 042115
3605 014336 020120 051106 046517
3606 014344 052040 030505 027466
3607 014352 052524 033467 052040
3608 014360 020117 042502 052040
3609 014366 051505 042524 000104
3610 014374 005015 054524 042520
3611 014402 043040 051111 052123
3612 014410 040440 042104 042522
3613 014416 051523 047440 020106
3614 014424 047503 052116 047522
3615 014432 046114 051105 020040
3616 014440 000
3617 014441 124 050131 020105
3618 014446 046524 031460 042040
3619 014454 044522 042526 021440
3620 014462 051447 052040 020117
3621 014470 042502 052040 051505
3622 014476 042524 035104 020040
3623 014504 046101 020114 000
3624 014511 106 051117 052040
3625 014516 030115 020063 051104
3626 014524 053111 020105
3627 014530 026460 052040 050131
3628 014536 020105 046123 053101
3629 014544 020105 023443 020123
3630 014552 047524 041040 020105
3631 014560 042524 052123 042105
3632 014566 020072 046101 020114

.SBTTL PROGRAM MESSAGES
:OPERATOR INSTRUCTIONS
M.NAM: .ASCII <CR><LF>'TMO3-TE16/TU77 DRIVE FUNCTION TIMER (CZTEEB0)';++B

.ASCIZ <CR><LF>'TYPE <CR> TO TERMINATE RESPONSE & ^C TO RESTART'

M.EOP: .ASCIZ <CR><LF>'END OF PASS '

M.HSWR: .ASCIZ <CR><LF>'HARDWARE SWR IN USE'<CR><LF>

I.REM: .ASCIZ <CR><LF>'REMOVE TMDP FROM TE16/TU77 TO BE TESTED'

I.REG: .ASCIZ <CR><LF>'TYPE FIRST ADDRESS OF CONTROLLER '

I.DRVS: .ASCIZ %TYPE TMO3 DRIVE #'S TO BE TESTED: ALL %

I.SLVS: .ASCII 'FOR TMO3 DRIVE '

I.DRV: .ASCIZ %0- TYPE SLAVE #'S TO BE TESTED: ALL %

3633	014574	000				
3634	014575	123	042520	042105	I.SPD: .ASCIZ	'SPEED TESTS? (YES/NO): NO '
3635	014602	052040	051505	051524		
3636	014610	020077	054450	051505		
3637	014616	047057	024517	020072		
3638	014624	047516	000040			
3639	014630	005015	047105	020104	M.EOT: .ASCIZ	<CR><LF>'END OF TAPE'<CR><LF>
3640	014636	043117	052040	050101		
3641	014644	006505	000012			
3642						
3643					:ERROR MESSAGES	
3644	014650	005015	051124	050101	E.TRP4: .ASCIZ	<CR><LF>'TRAPPED TO 4'
3645	014656	042520	020104	047524		
3646	014664	032040	000			
3647	014667	116	020117	047503	E.NCON: .ASCIZ	'NO CONTROLLER AT ADDRESS SPECIFIED'<CR><LF>
3648	014674	052116	047522	046114		
3649	014702	051105	040440	020124		
3650	014710	042101	051104	051505		
3651	014716	020123	050123	041505		
3652	014724	043111	042511	006504		
3653	014732	000012				
3654	014734	046524	031460	042040	E.NDRV: .ASCIZ	'TMO3 DRIVE '
3655	014742	044522	042526	000040		
3656	014750	051104	053111	020105	E.NSLV: .ASCII	'DRIVE '
3657	014756	020060	046123	053101	E.DRV: .ASCII	'0 SLAVE '
3658	014764	020105				
3659	014766	020060	047516	020124	E.NAVA: .ASCIZ	'0 NOT AVAILABLE FOR TEST'<CR><LF>
3660	014774	053101	044501	040514		
3661	015002	046102	020105	047506		
3662	015010	020122	042524	052123		
3663	015016	005015	000			
3664	015021	116	020117	046524	E.UNIT: .ASCIZ	'NO TMO3-TE16/TU77 UNIT FOUND TO TEST'<CR><LF>
3665	015026	031460	052055	030505		
3666	015034	027466	052524	033467		
3667	015042	052440	044516	020124		
3668	015050	047506	047125	020104		
3669	015056	047524	052040	051505		
3670	015064	006524	000012			
3671	015070	047523	052106	042440	E.SFT: .ASCIZ	'SOFT ERROR (DATA)'<CR><LF>
3672	015076	051122	051117	024040		
3673	015104	040504	040524	006451		
3674	015112	000012				
3675	015114	042524	052123	021440	E.HDR: .ASCIZ	'TEST # '
3676	015122	000040				
3677	015124	042040	053105	041511	E.HDR1: .ASCII	' DEVICE ERROR'<CR><LF>
3678	015132	020105	051105	047522		
3679	015140	006522	012			
3680	015143	103	030523	053411	.ASCIZ	'CS1'<HT>'WC'<HT>'BA'<HT>'FC'<HT>'CS2'<HT>'DS'<HT>'ER'<HT>'TC'<CR><LF>
3681	015150	004503	040502	043011		
3682	015156	004503	051503	004462		
3683	015164	051504	042411	004522		
3684	015172	041524	005015	000		
3685	015177	040	052517	020124	E.HDR2: .ASCIZ	' OUT OF RANGE ERROR'<CR><LF>
3686	015204	043117	051040	047101		
3687	015212	042507	042440	051122		
3688	015220	051117	005015	000		


```
3689 015225 015 052012 046511 E.TIMOV: .ASCIZ <CR><LF>'TIMER OVERFLOWED'<CR><LF>
3690 015232 051105 047440 042526
3691 015240 043122 047514 042527
3692 015246 006504 000012
3693 015252 005015 044524 042515 E.TIMEX: .ASCIZ <CR><LF>'TIME EXPIRED WAITING FOR RDY'<CR><LF>
3694 015260 042440 050130 051111
3695 015266 042105 053440 044501
3696 015274 044524 043516 043040
3697 015302 051117 051040 054504
3698 015310 005015 000
3699 015313 040 040507 020120 E.GAP: .ASCIZ ' GAP # '
3700 015320 020043 000
3701
3702 ;TIME DOCUMENT LINES
3703 015323 015 025012 025052 L.HDR1: .ASCIZ <CR><LF>'*****
3704 015330 025052 025052 025052
3705 015336 025052 025052 025052
3706 015344 025052 025052 025052
3707 015352 025052 025052 025052
3708 015360 025052 025052 025052
3709 015366 025052 025052 025052
3710 015374 025052 025052 025052
3711 015402 025052 025052 025052
3712 015410 025052 025052 025052
3713 015416 025052 025052 025052
3714 015424 025052 025052 025052
3715 015432 025052 006452 000012
3716 015440 020052 046524 031460 L.HDR2: .ASCII '* TMO3 DRIVE FUNCTION TIMES- DRIVE # '
3717 015446 042040 044522 042526
3718 015454 043040 047125 052103
3719 015462 047511 020116 044524
3720 015470 042515 026523 042040
3721 015476 044522 042526 021440
3722 015504 040
3723 015505 060 051440 040514 L.DRV: .ASCII '0 SLAVE # '
3724 015512 042526 021440 040
3725 015517 060 020040 000 L.SLV: .ASCIZ '0 '
3726 015523 124 030505 020066 L.TE16: .ASCIZ 'TE16 '
3727 015530 000
3728 015531 124 033525 020067 L.TU77: .ASCIZ 'TU77 '
3729 015536 000
3730 015537 123 051105 040511 L.SER: .ASCIZ 'SERIAL # '
3731 015544 020114 020043 000
3732 015551 040 005015 006452 L.HDR3: .ASCII ' '<CR><LF>'*%<CR><LF>
3733 015556 012
3734 015557 052 043040 047125 .ASCIZ '* FJUNCTION'<HT><HT>'TIME (SPECIFICATION)'<H^->'TIME (ACTUAL)'<<CR><LF>
3735 015564 052103 047511 004516
3736 015572 052011 046511 024105
3737 015600 050123 041505 043111
3738 015606 041511 052101 047511
3739 015614 024516 052011 046511
3740 015622 024105 041501 052524
3741 015630 046101 006451 000012
3742
3743 015636 040522 043516 036505 L.RNG: .ASCIZ 'RANGE <'
3744 015644 000074
```

```

3745 015646 041501 052524 046101 L.ACT: .ASCIZ 'ACTUAL='
3746 015654 000075
3747
3748 ;TEST DESCRIPTOR HEADERS
3749 015656 025052 044440 044516 A.T000: .ASCIZ '** INITIALIZATION ERROR'
3750 015664 044524 046101 055111
3751 015672 052101 047511 020116
3752 015700 051105 047522 000122
3753 015706 020052 051127 052111 A.T001: .ASCIZ '* WRITE FROM BOT'<HT>
3754 015714 020105 051106 046517
3755 015722 041040 052117 000011
3756 015730 020052 051127 052111 A.T002: .ASCIZ '* WRITE START'<HT><HT>
3757 015736 020105 052123 051101
3758 015744 004524 000011
3759 015750 020052 051127 052111 A.T003: .ASCIZ '* WRITE SHUTDOWN'<HT>
3760 015756 020105 044123 052125
3761 015764 047504 047127 000011
3762 015772 020052 051127 052111 A.T004: .ASCIZ '* WRITE SETTLEDOWN'<HT>
3763 016000 020105 042523 052124
3764 016006 042514 047504 047127
3765 016014 000011
3766 016016 020052 042522 042101 A.T005: .ASCIZ '* READ FROM BOT'<HT><HT>
3767 016024 043040 047522 020115
3768 016032 047502 004524 000011
3769 016040 020052 042522 042101 A.T006: .ASCIZ '* READ START'<HT><HT>
3770 016046 051440 040524 052122
3771 016054 004411 000
3772 016057 052 051040 040505 A.T007: .ASCIZ '* READ SHUTDOWN'<HT><HT>
3773 016064 020104 044123 052125
3774 016072 047504 047127 004411
3775 016100 000
3776 016101 052 051040 040505 A.T010: .ASCIZ '* READ SETTLEDOWN'<HT>
3777 016106 020104 042523 052124
3778 016114 042514 047504 047127
3779 016122 000011
3780 016124 020052 042522 042101 A.T011: .ASCIZ '* READ REV START'<HT>
3781 016132 051040 053105 051440
3782 016140 040524 052122 000011
3783 016146 020052 042522 042101 A.T012: .ASCIZ '* READ REV SHUTDOWN'<HT>
3784 016154 051040 053105 051440
3785 016162 052510 042124 053517
3786 016170 004516 000
3787 016173 052 051040 040505 A.T013: .ASCIZ '* READ REV SETTLEDOWN'<HT>
3788 016200 020104 042522 020126
3789 016206 042523 052124 042514
3790 016214 047504 047127 000011
3791 016222 020052 052524 047122 A.T014: .ASCIZ '* TURN AROUND DELAY F-R'<HT>
3792 016230 040440 047522 047125
3793 016236 020104 042504 040514
3794 016244 020131 026506 004522
3795 016252 000
3796 016253 052 052040 051125 A.T015: .ASCIZ '* TURN AROUND DELAY R-F'<HT>
3797 016260 020116 051101 052517
3798 016266 042116 042040 046105
3799 016274 054501 051040 043055
3800 016302 000011
  
```

3801	016304	020052	040507	020120	A.T016: .ASCIZ	'* GAP SIZE-STOP HALF'<HT>
3802	016312	044523	042532	051455		
3803	016320	047524	020120	040510		
3804	016326	043114	000011			
3805	016332	020052	040507	020120	A.T017: .ASCIZ	'* GAP SIZE-START HALF'<HT>
3806	016340	044523	042532	051455		
3807	016346	040524	052122	044040		
3808	016354	046101	004506	000		
3809	016361	052	043440	050101	A.T020: .ASCIZ	'* GAP SIZE-INTERRECORD'<HT>
3810	016366	051440	055111	026505		
3811	016374	047111	042524	051122		
3812	016402	041505	051117	004504		
3813	016410	000				
3814	016411	052	043440	050101	A.T021: .ASCIZ	'* GAP CONSISTANCY'<HT>
3815	016416	041440	047117	044523		
3816	016424	052123	047101	054503		
3817	016432	000011				
3818	016434	020052	040504	040524	A.T022: .ASCIZ	'* DATA TIME-800BPI'<HT>
3819	016442	052040	046511	026505		
3820	016450	030070	041060	044520		
3821	016456	000011				
3822	016460	020052	040504	040524	A.T023: .ASCIZ	'* DATA TIME-1600BPI'<HT>
3823	016466	052040	046511	026505		
3824	016474	033061	030060	050102		
3825	016502	004511	000			
3826	016505	052	042440	040522	A.T024: .ASCIZ	'* ERASE GAP TIME'<HT>
3827	016512	042523	043440	050101		
3828	016520	052040	046511	004505		
3829	016526	000				
3830	016527	052	053440	044522	A.T025: .ASCIZ	'* WRITE FILE MARK'<HT>
3831	016534	042524	043040	046111		
3832	016542	020105	040515	045522		
3833	016550	000011				
3834	016552	020052	040524	042520	A.T026: .ASCIZ	'* TAPE SPEED-FWD'<HT>
3835	016560	051440	042520	042105		
3836	016566	043055	042127	000011		
3837	016574	020052	040524	042520	A.T027: .ASCIZ	'* TAPE SPEED-REV'<HT>
3838	016602	051440	042520	042105		
3839	016610	051055	053105	000011		
3840						
3841	016616	005015	043536	000	L.CNTG: .ASCIZ	<CR><LF>'^G'
3842	016623	015	051412	051127	L.SWR: .ASCIZ	<CR><LF>'SWR='
3843	016630	000075				
3844	016632	020040	042516	036527	L.NEW: .ASCIZ	' NEW= '
3845	016640	000040				
3846	016642	005015	006477	000012	L.QUEST: .ASCIZ	<CR><LF>'?'<CR><LF>
3847						
3848		016650			.EVFN	
3849		016650			RDBUF=.	
3850	016650	000200			WTBUF=.	
3851		000001			.BLKW 128.	
					.END	

INPUT	1033#	1544	2449	2473	2542	2653										
RESTOR	1030#	1685	1720	1813	1856	1979	2124	2166								
REWIND	1036#	2280	2706	2758	2857	3057	3218	3296	3334	3371	3425	3485	3523	3568		
SAVE	1027#	1663	1702	1796	1823	2099	2137									
SETGO	1046#	2283	2303	2354	2711	2884	2904	2934	2968	2974	2993	2997	3024	3028	3070	
	3076	3095	3099	3106	3124	3130	3148	3152	3158	3177	3181	3185	3190	3223	3233	
	3242	3248	3304	3343	3375	3385	3404	3413	3494	3532	3547					
TIMCHK	1043#	2175	2190	2766	2785	2812	2844	2866	2888	2917	2952	2978	3009	3046	3080	
	3110	3134	3162	3194	3252	3316	3355	3389	3417	3504	3557					
TIMEON	1040#	2187	2761	2807	2839	2861	2883	2912	2948	2973	3005	3042	3075	3105	3129	
	3157	3189	3247	3312	3351	3384	3412	3501	3554							
\$CATCH	983#	1230														
\$CHAIN	983#	2428														
\$CPREG	983#	1051														
\$CPVEC	983#	1078														
\$TYPE	983#	1555														
.\$ACT1	983#	1242														
.\$EOP	983#	3460														

. ABS. 017250 000

ERRORS DETECTED: 0

DSKZ:CZTEEB,DSKZ:CZTEEB,SEQ/CRF/SOL-CZTEAB.SML/ML,CZTEEB.P11

RUN-TIME: 59.9 SECONDS

RUN-TIME RATIO: 98/15 6.2

CORE USED: 8K (15 PAGES)