

TM03 TM03/TE16, TU77 DFT
TE16 TU77 CZTEEE0

COPYRIGHT (c) 1977-84
AH-A804E-MC
FICHE 01 OF 01

JUL 1984
digital
Made In USA

(Faint, illegible data table with multiple columns and rows)



.REM *

IDENTIFICATION

PRODUCT CODE: AC-A803E MC
PRODUCT NAME: CZTEEO TM03-TE16/TU77 DRIVE FUNCTION TIMER
PRODUCT DATE: 15 MARCH 1984
MAINTAINER: TAPE DIAGNOSTIC GROUP
AUTHOR: J. MITT

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

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

REVISED TO REV E J. MITT
15 MARCH 1984
ADD XON/XOFF FUNCTIONALITY

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) 1977,1984 BY DIGITAL EQUIPMENT CORPORATION

IM03 DRIVE FUNCTION TIMER
TABLE OF CONTENTS

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 TIMEOUT FORMAT (HARDWARE)
- 4.2 ERROR TIMEOUT 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

TM03 DRIVE FUNCTION TIMER
ABSTRACT

ABSTRACT

PROGRAM DZTEE MEASURES THE TIME REQUIRED AND GAP SIZES PRODUCED BY THE TM03 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

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

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 RMXX
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 TM03-TE16/TU77 COMBINATIONS ARE TESTED. ADDITIONALLY THE SOFTWARE
SWR IS INVOKED WITH A SWITCH SETTING OF 000000 IF LOADED VIA ACT11.
NO OPERATOR INTERVENTION IS REQUIRED

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

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

TM03 DRIVE FUNCTION TIMER
SWITCH SETTINGS

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 SELECTED

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

IM03 DRIVE FUNCTION TIMER
ERRORS

CHAPTER 4
ERRORS

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

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

TEST # XXXXXX DEVICE ERROR

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

WHERE:

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

4.2 ERROR TIMEOUT FORMAT (FUNCTION TIME OUT OF RANGE)

TEST # XXXXXX OUT OF RANGE ERROR

RANGE <AAAAAA BBBBBB> ACTUAL = CCCCCCC

IM03 DRIVE FUNCTION TIMER
SUBROUTINE ABSTRACTS

CHAPTER 5
SUBROUTINE ABSTRACTS

5.1 .SCOPE

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

1. LOADS R5 WITH BASE ADDRESS
2. PROVIDES CONTINUOUS LOOP <SW14>
3. MOVES FUNCTION TIME INTO TABLE
4. OUTPUTS LINE ITEM <SW10>=1
5. DELAYS 350MS BEFORE STARTING TEST
6. INIT'S DRIVE/SLAVE
7. CLEARS THE ERROR FLAG (ERFLG)
8. CHECK FOR CONTROL G (↑G)

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

5.2 PUBLISH

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

TM03 DRIVE FUNCTION TIMER
SUBROUTINE ABSTRACTS

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.

TM03 DRIVE FUNCTION TIMER
MISCELLANEOUS

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.

TM03 DRIVE FUNCTION TIMER
PROGRAM DESCRIPTION

CHAPTER 7

PROGRAM DESCRIPTION

7.1 SAMPLE TIME DOCUMENT

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

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

* FUNCTION	TIME(SPECIFICATION)	TIME(ACTUAL)
* WRITE FROM BOT	RANGE=<176000-172000>	ACTUAL=174740
* WRITE START	RANGE=<009500-008700>	ACTUAL=009120
* WRITE SHUTDOWN	RANGE=<008500-007500>	ACTUAL=008840
* WRITE SETTLEDOWN	RANGE=<013500-007300>	ACTUAL=010970
* READ FROM BOT	RANGE=<045000-041000>	ACTUAL=043580
* READ START	RANGE=<003200-002400>	ACTUAL=002740
* READ SHUTDOWN	RANGE=<004100-003050>	ACTUAL=004360
* READ SETTLEDOWN	RANGE=<013500-007300>	ACTUAL=010970
* READ REV START	RANGE=<003200-002400>	ACTUAL=002740
* READ REV SHUTDOWN	RANGE=<003700-003300>	ACTUAL=003520
* READ REV SETTLEDOWN	RANGE=<013500-007300>	ACTUAL=010970
* TURN AROUND DELAY F-R	RANGE=<016700-010700>	ACTUAL=013600
* TURN AROUND DELAY R-F	RANGE=<016700-010700>	ACTUAL=013660
* GAP SIZE-STOP HALF	RANGE=<012900-009500>	ACTUAL=012200
* GAP SIZE-START HALF	RANGE=<011800-008500>	ACTUAL=010520
* GAP SIZE-INTERRECORD	RANGE=<014300-012600>	ACTUAL=014500
* GAP CONSIANCY	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

B.P

TM03 DRIVE FUNCTION TIMER

7.1.1 SAMPLE TIME DOCUMENT FOR TAPE SPEED TESTS

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

•TM03 DRIVE FUNCTION TIMES DRIVE # 0 SLAVE # 7 TE16 SERIAL # 5009
•
•FUNCTION TIME(SPECIFICATION) TIME(ACTUAL)
•TAPE SPEED FWD RANGE=<022700-021700> ACTUAL=022500
•TAPE SPEED REV RANGE=<022700 021700> ACTUAL=022500

C.

IM03 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 C
5. READ FROM BOT	*"	*M8931/M8940 ROM *M8933 ACCL CNTR
6. READ START	*'	* ' *
7. READ SHUTDOWN	*'	* ' *
10. READ SETTLEDOWN	*"	*M8916/M8940 (TU77) SETTLEDOWN C
11. READ REVERSE START	*"	*M8931/M8940 ROM *M8933 ACCL CNTR
12. READ REVERSE SHUTDOWN	*"	* * "
13. READ REVERSE SETTLEDOWN	*"	*M8916/M8940 (TU77) SETTLEDOWN C
14. TURN AROUND F-R	*"	*M8931/M8940 ROM *M8933 ACCL CNTR
15. TURN AROUND R-F	*"	* *
16. GAP SIZE STOP HALF	*FWD/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	*

D.P

TMO3 DRIVE FUNCTION TIMER

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 ACCI 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

TM03 DRIVE FUNCTION TIMER

7.3 SUBTEST DESCRIPTIONS:

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

T1. WRITE FROM BOT:

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

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

T2. WRITE START:

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

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

T3. WRITE SHUTDOWN:

THIS TEST WILL MEASURE THE TIME FROM EOR (LAST CHARACTER WRITTEN ON TAPE) TO THE START OF SETTLEDOWN TIME. THIS ASSURES, IN PART, A PROPER 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 FC=RECORD SIZE (LAST FRAME READ) TO SDWN-1 IS THE SHUTDOWN TIME.
5. STOP

T10. READ SETTLEDOWN:

THIS TEST MEASURES THE SAME DELAY AS IN T4.

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

TM03 DRIVE FUNCTION TIMER

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

T15. TURN AROUND DELAY REVERSE TO FORWARD

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

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

TM03 DRIVE FUNCTION TIMER

GAP MEASUREMENTS:

THE PREVIOUS THIRTEEN (13) TESTS WERE MEASUREMENTS OF LOGIC DELAYS PERFORMED BY THE TM03 OR TE16/TU77 IN ORDER TO ALLOW FOR PROPER ACCELERATION AND DECELERATION OF TAPE ACCORDING TO THE DESIRED INTERCORD GAP (.6 INCHES). THIS TEST, HOWEVER, WILL MEASURE THE PHYSICAL SIZE OF THE INTERCORD GAP THAT EXISTS ON TAPE AS A RESULT OF THE START/STOP TIMES OF THE CAPSTAN ITSELF. BECAUSE THE INTERCORD GAP IS CREATED BY TWO ACTIONS, THE START OF MOTION AND THE STOP OF MOTION IT IS NECESSARY TO MAKE TWO SEPERATE MEASUREMENTS. A THIRD MEASUREMENT, MADE ON THE FLY, OF THE ENTIRE LENGTH OF THE GAP WILL ALSO BE MADE.

T16. GAP SIZE (STOP HALF)

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

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

T17. GAP SIZE (START HALF)

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

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

T20. GAP SIZE (INTERRECORD)

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

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

TM03 DRIVE FUNCTION TIMER

T21. GAP CONSISTENCY:

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

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

T24. ERASE:

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

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

T25. TAPE MARK:

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

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

TM03 DRIVE FUNCTION TIMER

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.

```

1015 .LIST BIN,LOC,SEQ
1016 .NLIST MC
1017 .NLIST TOC
1018 .LIST ME
1019 .ENABLE ABS,AMA
1020 .MCALL $CPVEC,$CPREG,$CATCH,$TYPE,..$ACT11,..$OP,$CHAIN
1021 .TITLE CZTEEE0 TM03-TE16/TU77 DFT
1022 ;DRIVE FUNCTION TIMER
1023 .SBTTL STARTING INSTRUCTIONS
1024 ;LOADING AND STARTING PROCEDURE
1025 ; LOAD PROGRAM USING ABS LOADER
1026 ; LOAD ADDRESS 200
1027 ; SET SWITCH OPTIONS
1028 ; PRESS START
1029
1030 ;RESTART PROCEDURE
1031 ; LOAD ADDRESS 210
1032 ; SET SWITCH OPTIONS
1033 ; PRESS START
1034
1035 ;SWITCH REGISTER SWITCH ASSIGNMENTS
1036 100000 SW15= 100000 ;HALT ON ERROR
1037 040000 SW14= 040000 ;LOOP SUBTEST
1038 020000 SW13= 020000 ;INHIBIT ERROR TYPE OUT
1039 004000 SW11= 004000 ;INHIBIT SUBTEST ITERATION
1040 002000 SW10= 002000 ;INHIBIT PUBLISHING TIME SPECIFICATION
1041 001000 SW09= 001000 ;RING BELL ON ERROR
1042 000400 SW08= 000400 ;
1043 000200 SW07= 000200 ;NOT USED
1044 000100 SW06= 000100 ;CONTINUOUS CYCLE
1045 ; SW05-SW00 ;RUN TEST SELECTED
1046 ; **NOTE: IF <SW15-SW00> = 177777 AT STARTUP USE SOFTWARE
1047 ; SWITCH REGISTER.
1048
1049 ;CONSOLE COMMANDS
1050 ; CONTROL C ;RESTART PROGRAM (SAME AS START @ 200)
1051 ; CONTROL G ;SET NEW SOFTWARE SWITCH REGISTER
1052 ; CONTROL U ;DELETE LINE TYPED
1053 ; RUBOUT (DELETE) ;DELETE LAST CHAR TYPED
1054
1055 ;GENERAL REGISTER USAGE:
1056 ; R0=ADDRESS OF 'FC' REGISTER (SET BY SCOPE)
1057 ; R1=ADDRESS OF 'DS' REGISTER (SET BY SCOPE)
1058 ; R2=RETURN PC FROM TIMER (SET BY EACH TEST)
1059 ; R3=INDEX INDICATING PREVIOUS OSCILLATOR POLARITY (SET BY TIMER)
1060 ; R4=CONTAINS 'TICK' COUNT WHEN TIMER IS RUNNING (SET BY TIMER)
1061 ; R5=ADDRESS OF CS1 (SET BY SCOPE)
1062
1063 .SBTTL MACRO DEFINITIONS
1064 .MACRO SAVE
1065 JSR PC,SAVE ;SAVE REGISTERS ON THE STACK
1066 .ENDM
1067 .MACRO RESTORE
1068 JSR PC,RESTORE ;RESTORE REGISTERS FROM THE STACK
1069 .ENDM
1070 .MACRO INPUT

```


(1)
(1)
(1)
(1)
(1)
(1)

000114
000240
000244
000250

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

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

1091
 1092 172440
 1093
 1094
 1095 000000
 1096 000002
 1097 000004
 1098 000006
 1099 000010
 1100 000012
 1101 000014
 1102 000016
 1103 000022
 1104 000024
 1105 000026
 1106 000030
 1107 000032
 1108
 1109
 1110
 1111 000001
 1112 000000
 1113 000002
 1114 000006
 1115 000010
 1116 000026
 1117 000024
 1118 000030
 1119 000032
 1120 000050
 1121 000056
 1122 000060
 1123 000070
 1124 000076
 1125 000100
 1126 000200
 1127 000400
 1128 001000
 1129 002000
 1130 004000
 1131 020000
 1132 040000
 1133 100000
 1134
 1135 000000
 1136 000001
 1137 000002
 1138 000003
 1139 000004
 1140 000005
 1141 000006
 1142 000007
 1143 000010
 1144 000020
 1145 000040
 1146 000100

;RH.TM03 TE16/TU77 REGISTERS
 TMCS1- 172440

;TM03 TE16/TU77 INDEX VALUES
 CS1- 00
 WC- 02
 BA- 04
 FC- 06
 CS2- 10
 DS- 12
 ER- 14
 AS- 16
 DB- 22
 MR- 24
 DT- 26
 SN- 30
 TC- 32

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

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

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

;RHCS2-CS2(R5)

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

1147	000200	OR=	200	
1148	000400	IDPE=	400	
1149	001000	MXF=	1000	
1150	002000	PGE=	2000	
1151	004000	NEM=	4000	
1152	010000	NED=	10000	
1153	020000	UPE=	20000	
1154	040000	WCE=	40000	
1155	100000	DLT=	100000	
1156		;RHDS-DS(R5)		
1157	000001	SLA=	1	
1158	000002	BOT=	2	
1159	000004	TMK=	4	
1160	000010	IDB=	10	
1161	000020	SDWN=	20	
1162	000040	PES=	40	
1163	000100	SSC=	100	
1164	000200	DRY=	200	
1165	000400	DPR=	400	
1166	002000	EOT=	2000	
1167	004000	WRL=	4000	
1168	010000	MOL=	10000	
1169	020000	PIP=	20000	
1170	040000	ERR=	40000	
1171	100000	ATA=	100000	
1172		;RHER-ER(R5)		
1173	000001	ILF=	1	
1174	000002	ILR=	2	
1175	000004	RMR=	4	
1176				
1177	000020	FMT=	20	
1178	000100	INCVAE=	100	
1179	000200	PEFLRC=	200	
1180	000400	NSG=	400	
1181	001000	FCE=	1000	
1182	002000	CSITH=	2000	
1183	004000	NEF=	4000	
1184	010000	DTE=	10000	
1185	020000	OPI=	20000	
1186	040000	UNS=	40000	
1187	075027	HRDERR=	UNS!OPI!DTE!NEF!FCE!FMT!RMR!ILR!ILF	;HARDERROR BITS
1188				
1189		;RHMR-MR(R5)		
1190	000100	OSC=	100	
1191				
1192		;RHDT-DT(R5)		
1193	002000	SPR=	2000	
1194	010000	CH7=	10000	
1195	040000	TAP=	40000	
1196				
1197		;RHIC TC(R5)		
1198	001700	NORM11=	1700	
1199	000320	CDM11=	320	
1200	000000	BPI200=	0	
1201	000400	BPI556=	000400	
1202	001000	BPI800=	001000	

1203	002300
1204	100000
1205	
1206	
1207	
1208	
1209	104400
1210	104000
1211	000004
1212	
1213	
1214	006466
1215	177400
1216	177600
1217	
1218	000001
1219	000003
1220	000007
1221	000011
1222	000012
1223	000015
1224	000017
1225	000025

PE1600= 002300
ACCL= 100000

; INSTRUCTION EQUATES

HLT= TRAP
SCOPE EMT
TYPE= IOT

; MISCELLANEOUS EQUATES

OUTBUF= INIT
FRMCNT= -256.
WRDCNT= -128.

; ASCII EQUATES

CNTRLA= 1
CNTRLC= 3
CNTRLG= 7
HT= 11
LF= 12
CR= 15
CNTRLO= 17
CNTRLU= 25

; OUTPUT BUFFER STARTS AT BEG OF PROGRAM
; FRAME COUNT
; WORD COUNT

; ASCII CODE FOR CONTROL A (^A)
; ASCII CODE FOR CONTROL C (^C)
; ASCII CODE FOR CONTROL G (^G)
; ASCII CODE FOR HORIZONTAL TAB
; ASCII CODE FOR LINE FEED
; ASCII CODE FOR CARRIAGE RETURN
; ASCII CODE FOR CONTROL O (^O)
; ASCII CODE FOR CONTROL U (^U)

```

1228 ;SETUP TRAP VECTORS
1229 . = TBITVEC
1230 000014 000016 .WORD .+2 ;SET 'T' TRAP TO TIMER ROUTINE
1231 000016 000000 .WORD HALT ;PRIORITY LEVEL 7
1232 000020 002640 .WORD .TYPE ;SET IOT TRAP TO .TYPE ROUTINE
1233 000022 000000 .WORD 0 ;PRIORITY LEVEL 0
1234 000024 000026 .WORD PFVEC+2 ;POWER FAIL TRAP TO HALT
1235 000026 000000 .WORD HALT ;AT PFVEC+2
1236 000030 004710 .WORD .SCOPE ;SET EMT TRAP TO .SCOPE ROUTINE
1237 000032 000340 .WORD 340 ;PRIORITY LEVEL 7
1238 000034 004374 .WORD .HLT ;SET TRAP TRAP TO .HLT ROUTINE
1239 000036 000340 .WORD 340 ;PRIORITY LEVEL 7
1240
(1) ;ACT11 HOOK *****
(1) 000040 $SVPC= . ;SAVE CURRENT LOCATION CTR
(1) 000042 . = 42
(1) 000042 000000 .WORD 0
(1) 000046 . = 46
(1) 000046 013676 .WORD $ENDAD ;SET LOCATION 46
(1) 000052 . = 52
(1) 000052 000000 .WORD 0 ;SET LOCATION 52 = 0
(1) 000040 C0C040 . = $SVPC ;RESTORE LOCATION CTR
(1)
1241 . = TKVEC
1242 000060 004216 .WORD TKISR
1243 000062 000200 .WORD 200
1244
1245 ;SOFTWARE SWITCH REGISTER LOC. 176
1246 . = 176
1247 000176 000000 SWREG: .WORD 0 ;SOFTWARE SWITCH REGISTER
1248
1249 . = 200
1250 000200 000137 006466 JMP @INIT ;GO TO START OF PROGRAM
1251 . = 210
1252 000210 000137 007606 JMP @RSTRT ;RESTART ADDRESS
1253
1254 . = 500
1255 000600 STKPTR= 600 ;STACK
1256
1257 . = 1000
1258 ;PROGRAM TAGS
1259 001000 177570 SWR: 177570 ;SWITCH REGISTER
1260 001002 000000 SCPADR: .WORD 0
1261 001004 000 DRVNUM: .BYTE 0 ;TM03 DRIVE UNDER TEST
1262 001005 000 SLVNUM: .BYTE 0 ;TE16/TU77 SLAVE UNDER TEST
1263 001006 000000 SLVPTR: .WORD 0 ;POINTER TO SLAVE TABLE (SLVTBL) BELOW
1264 001010 172440 TMBASE: .WORD TMC51 ;BASE ADDRESS OF TM03-TE16/TU77 REGISTERS
1265 001012 000000 OSCTIM: .WORD 0 ;US/TICK (56/80 FOR TE16/TU77)
1266 001014 000000 GAPDEL: .WORD 0 ;TICKS/MS (18/6 FOR TE16/TU77)
1267 001016 000000 ATIME: .WORD 0 ;CONTAINS 'TICK' COUNT
1268 001020 000020 ATIMTBL: .BLKW 16. ;EACH ENTRY CONTAINS TIME FOR FUNCTION
1269 ;ENTRIES ARE MADE BY 'SCOPE' ROUTINE
1270 001060 000020 GAP_TBL: .BLKW 16. ;TIMES RECORDED BY 'GAP CONSISTANCY' TEST
1271 001120 000000 DELTIM: .WORD 0 ;VARIABLE DELAY
1272 001122 000000 OCTALO: .WORD 0
1273 001124 000 GAP: .BYTE 0 ;CONTAINS GAP # (USED FOR TST 021)

```

1274	001125	000	
1275	001126	000	
1276	001127	000	
1277	001130	000	
1278	001131	000	
1279	001132	000	
1280	001133	000	
1281	001134	000	
1282	001135	000	
1283	001136	000	
1284		001140	
1285	001140	030460	
1286	001142	031462	
1287	001144	032464	
1288	001146	033466	
1289	001150	034470	
1290	001152	000006	
1291	001160	000	
1292		001162	
1293	001162	000010	
1294	001172	000100	
1295	001272	000110	
1296	001402	005015	000
1297	001405	134	000
1298	001407	060	000
1299	001411	007	000
1300	001413	055	000
1301	001415	040	
1302	001416	000040	
1303	001420	004476	000
1304		001424	

ITCNT:	.BYTE	0
TSTNUM:	.BYTE	0
ERFLG:	.BYTE	0
SKEWFLG:	.BYTE	0
PRGFLG:	.BYTE	0
UNTFND:	.BYTE	0
TYPFLG:	.BYTE	0
PSCNT:	.BYTE	0
ASFLG:	.BYTE	0
TE16:	.BYTE	0
	.EVEN	
DIGTAB:	"01	
	"23	
	"45	
	"67	
	"89	
ODIGITS:	.BLKB	6
	.BYTE	0
	.EVEN	
DRVTBL:	.BLKB	8.
SLVTBL:	.BLKB	64.
INBUF:	.BLKB	72.
CRLF:	.ASCIZ	<CR><LF>
BKSLSH:	.ASCIZ	'\'
ECHO:	.ASCIZ	'0'
BELL:	.ASCIZ	<7>
DASH:	.ASCIZ	'-'
SPACE2:	.ASCIZ	' '
SPACE:	.ASCIZ	' '
ANGTAB:	.ASCIZ	'>'<HT>
	.EVEN	

; ITERATION COUNT
; TEST #
; ERROR FLAG
; 0/1 = DO NOT/DO SKEW (SPEED) TESTS
; PROGRAM FLAG
; UNIT FOUND INDICATOR
; CONTAINS PASS COUNT
; 1/0 = YES/NO.
; 0/1 = TE16/TU77
; RESERVE SPACE FOR CONVERTED DIGITS
; TERMINATOR
; A 0/-1 = DRIVE NOT TO BE/TO BE TESTED
; A 0/-1 = SLAVE NOT TO BE/TO BE TESTED
; TELETYPE INPUT BUFFER
; MISCELLANEOUS ASCII CHARACTERS

1306
 1307
 1308
 1309
 1310
 1311
 1312
 1313 001424 000000 000000
 1314 001430 036050 035230
 1315 001434 001666 001546
 1316 001440 001522 001356
 1317 001444 002506 001332
 1318 001450 007164 006344
 1319 001454 000500 000360
 1320 001460 000632 000454
 1321 001464 002506 001332
 1322 001470 000500 000360
 1323 001474 000562 000512
 1324 001500 002506 001332
 1325 001504 003206 002056
 1326 001510 003206 002056
 1327 001514 002412 001666
 1328 001520 002234 001522
 1329 001524 002626 002354
 1330 001530 002570 002234
 1331 001534 004540 004230
 1332 001540 004716 004552
 1333 001544 023564 023110
 1334 001550 024240 023730
 1335 001554 004336 004172
 1336 001560 004336 004172
 1337
 1338
 1339
 1340
 1341
 1342
 1343
 1344
 1345
 1346 001564 002602 002412
 1347 001570 002652 002506
 1348 001574 002734 002532
 1349 001600 003016 002424
 1350 001604 003016 002304
 1351 001610 002734 002176
 1352 001614 002652 002176
 1353 001620 002652 002176
 1354 001624 002570 002176
 1355 001630 002570 002176
 1356 001634 002570 002176
 1357 001640 002570 002176
 1358 001644 002570 002176
 1359 001650 002570 002176
 1360 001654 002570 002176
 1361 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 SHUTDOWN TST012
.WORD	01350.,00730.	;	13.5-7.3	RD REV STLDWN TST013
.WORD	01670.,01070.	;	16.7-10.7	TRN RND DLY F-R TST014
.WORD	01670.,01070.	;	16.7-10.7	TRN RND DLY R F TST015
.WORD	01290.,00950.	;	12.9-9.5	GAP SIZE STOP TST016
.WORD	01180.,00850.	;	11.8-8.5	GAP SIZE STRT TST017
.WORD	01430.,01260.	;	14.3-12.6	GAP SIZE INTER TST020
.WORD	01400.,01180.	;	14.0-11.8	GAP CONSISANCY TST021
.WORD	02400.,02200.	;	24.0-22.0	DAT TIME 800BPI TST022
.WORD	02510.,02410.	;	25.1-24.1	DAT TIME 1600PE TST023
.WORD	10100.,09800.	;	101.0-98.0	ERASE TST024
.WORD	10400.,10200.	;	104.0-102.0	WRT FILE MARK TST025
.WORD	02270.,02170.	;	22.7-21.7	TAPE SPEED FWD TST026
.WORD	02270.,02170.	;	22.7-21.7	TAPE SPEED REV TST027

;NOTE: TEST 26 AND 27 REQUIRE PRERECORDED 800BPI SKEW TAPE.
 .SBTTL TE16 GAP TIME SPECIFICATION TABLE
 ;THIS TABLE CONTAINS THE TE16 GAP SIZES (IN TENS OF MICROSECONDS) FOR EACH
 ;OF THE 16. GAPS RECORDED BY THE GAP CONSISTANCY TEST (TST021).
 ;NOTE: GAP #'S ARE IN OCTAL.

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

1363
 1364
 1365
 1366
 1367
 1368
 1369
 1370 001664 000000 000000
 1371 001670 012606 011571
 1372 001674 000520 000460
 1373 001700 000346 000313
 1374 001704 003410 001717
 1375 001710 001315 001112
 1376 001714 000111 000067
 1377 001720 000111 000067
 1378 001724 003410 001717
 1379 001730 000111 000067
 1380 001734 000101 000057
 1381 001740 003410 001717
 1382 001744 003500 002006
 1383 001750 003510 002017
 1384 001754 001510 001231
 1385 001760 000642 000505
 1386 001764 001012 000646
 1387 001770 001034 000641
 1388 001774 001515 001434
 1389 002000 001536 001434
 1390 002004 007020 006476
 1391 002010 007176 006642
 1392 002014 001560 001320
 1393 002020 001560 001320
 1394
 1395
 1396
 1397
 1398
 1399
 1400
 1401
 1402
 1403 002024 000770 000676
 1404 002030 001013 000717
 1405 002034 001033 000732
 1406 002040 001050 000742
 1407 002044 001062 000746
 1408 002050 001067 000742
 1409 002054 001072 000736
 1410 002060 001072 000724
 1411 002064 001066 000704
 1412 002070 001065 000660
 1413 002074 001046 000627
 1414 002100 001027 000572
 1415 002104 000776 000632
 1416 002110 000776 000632
 1417 002114 000776 000632
 1418 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	0840.,00665.	;	8.4-6.65	GAP SIZE STOP	TST016
	.WORD	0418.,00325.	;	4.18-3.25	GAP SIZE STRT	TST017
	.WORD	0522.,0422.	;	5.22-4.22	GAP SIZE INTER	TST020
	.WORD	0540.,0417.	;	5.40-4.17	GAP CONSISANCY	TST021
	.WORD	0845.,0796.	;	8.45-7.96	DAT TIME 800BPI	TST022
	.WORD	0862.,0796.	;	8.62-7.96	DAT TIME 1600PE	TST023
	.WORD	3600.,03390.	;	36.00-33.90	ERASE	TST024
	.WORD	3710.,3490.	;	37.10-34.90	WRT FILE MARK	TST025
	.WORD	00880.,00720.	;	8.8-7.2	TAPE SPEED FWD	TST026
	.WORD	00880.,00720.	;	8.8-7.2	TAPE SPEED REV	TST027

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

.SBTTL TU77 GAP TIME SPECIFICATION TABLE
 ;THIS TABLE CONTAINS THE TU77 GAP SIZES (IN TENS OF MICROSECONDS) FOR EACH
 ;OF THE 16. GAPS RECORDED BY THE GAP CONSISTANCY TEST (TST021).
 ;NOTE: GAP #'S ARE IN OCTAL.

;	.WORD	MAX,MIN(10)	;	TIME IN MS(10)	GAP #	DELAY IN MS(:0)
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

103

1420
1421
1422
1423
1424
1425
1426 002124
1427 002124 000000 000000
1428 002130 036050 035230
1429 002134 001666 001546
1430 002140 001522 001356
1431 002144 002506 001332
1432 002150 007164 006344
1433 002154 000500 000560
1434 002160 000632 000454
1435 002164 002506 001332
1436 002170 000500 000360
1437 002174 000562 000512
1438 002200 002506 001332
1439 002204 003206 002056
1440 002210 003206 002056
1441 002214 002412 001666
1442 002220 002234 001522
1443 002224 002626 002354
1444 002230 002506 002234
1445 002234 004540 004230
1446 002240 004716 004552
1447 002244 023564 023110
1448 002250 024240 023730
1449 002254 004336 004172
1450 002260 004336 004172
1451
1452
1453
1454
1455
1456
1457
1458 002264 002544 002354
1459 002270 002652 002506
1460 002274 002722 002506
1461 002300 002710 002474
1462 002304 002570 002260
1463 002310 002556 002114
1464 002314 002532 002114
1465 002320 002532 002114
1466 002324 002506 002176
1467 002330 002474 002176
1468 002334 002474 002176
1469 002340 002474 002176
1470 002344 002474 002176
1471 002350 002474 002176
1472 002354 002474 002176
1473 002360 002474 002176
1474 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:	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 :

1476
1477
1478 002364 016600
1479 002366 016630
1480 002370 016652
1481 002372 016672
1482 002374 016714
1483 002376 016740
1484 002400 016762
1485 002402 017001
1486 002404 017023
1487 002406 017046
1488 002410 017070
1489 002412 017115
1490 002414 017144
1491 002416 017175
1492 002420 017226
1493 002422 017254
1494 002424 017303
1495 002426 017333
1496 002430 017356
1497 002432 017402
1498 002434 017427
1499 002436 017451
1500 002440 017474
1501 002442 017516
1502
1503
1504 002444 010172
1505 002446 010466
1506 002450 010552
1507 002452 010630
1508 002454 010720
1509 002456 011026
1510 002460 011112
1511 002462 011176
1512 002464 011276
1513 002466 011416
1514 002470 011514
1515 002472 011624
1516 002474 011764
1517 002476 012056
1518 002500 012164
1519 002502 012260
1520 002504 012370
1521 002506 012510
1522 002510 013014
1523 002512 013144
1524 002514 013274
1525 002516 013414
1526 002520 013750
1527 002522 014106

.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

```

1529 002524 000000 TIB: .WORD 0
1530 ;ROUTINE TO LOAD SOFTWARE SWR
1531
1532 002526 022737 000176 001000 GTSWR: CMP @SWREG,SWR ;BRANCH IF SOFTWARE SWR
1533 002534 001027 BNE 2$ ;NOT INVOKED
1534 002536 004737 003152 JSR PC,SAVE ;SAVE REGISTERS ON THE STACK
1535 002542 000004 017545 TYPE,L,SWR
1536 002546 017702 176226 MOV @SWR,R2
1537 002552 004737 003224 JSR PC,TYPEOCT
1538 002556 000004 017554 TYPE,L,NEW
1539 002562 004737 004070 JSR PC,INPUT ;GET USER INPUT
1540 002566 122737 000015 001272 CMPB @CR,@INBUF ;EXIT IF FIRST CHAR IS <CR>
1541 002574 001405 BEQ 1$
1542 002576 004737 003654 JSR PC,CNVTAO ;CONERT ASCII TO OCTAL
1543 002602 013777 001122 176170 MOV @OCTALO,@SWR ;SET NEW SWITCH REG CONTENTS
1544 002610 004737 003174 1$: JSR PC,RESTORE
1545 002614 000207 2$: RTS PC
1546
1547
1548 .SBTTL PROGRAM SUBROUTINES
1549 .SBTTL TYPE SUBROUTINE
1550 ;;ROUTINE TO TYPE ASCII MESSAGE. MESSAGE MUST TERMINATE WITH A 0 BYTE.
(1) ;;THE ROUTINE WILL INSERT A NUMBER OF NULL CHARACTERS AFTER A LINE FEED.
(1) ;;CALL: TYPE ;;A TRAP TYPE INSTRUCTION
(1) ;; MESADR ;;MESADR IS FIRST ADDRESS OF ASCIZ STRING
(1)
(1) ;;TAGS USED BY THE TYPE ROUTINE BELOW
(1) 000011 $HT=11 ;;HORIZONTAL TAB
(1) 002616 000 $NULL: .BYTE 0 ;;CONTAINS NULL CHARACTER
(1) 002617 002 $FILL: .BYTE 2 ;;CONTAINS # OF FILLER CHARACTERS
(1) 002620 000 $TPFLG: .BYTE 0 ;;CONTAINS TELEPRINTER AVAILABLE FLAG
(1) ;;0/377 = AVAIL/NOT AVAIL
(1) 002621 000 $TKFLG: .BYTE 0 ;;CONTAINS KEYBOARD AVAILABLE FLAG
(1) 002622 177564 $TPS: .WORD 177564 ;;ADDRESS OF TELEPRINTER STATUS REGISTER
(1) 002624 177566 $TPB: .WORD 177566 ;;ADDRESS OF TELEPRINTER DATA BUFFER
(1) 002626 000000 $CTRLS: .WORD 0 ;;FLAG FOR XON/XOFF PROCESSING
(1) 002630 000 $CHARCNT: .BYTE 0 ;;CONTAINS # OF CHARS TYPED
(1) 002631 000 $CNTRLO: .BYTE 0 ;;CONTAINS CONTROL 0 CHAR (IF TYPED)
(1) 002632 005015 000 $CRLF: .ASCIZ <15><12>
(1) 002636 .EVEN
(1) 002636 000000 RDSW: .WORD 0
(1)
(1) 002640 010046 .TYPE: MOV R0,-(SP) ;;SAVE R0
(1) 002642 017600 000002 MOV @2(SP),R0 ;;GET MESSAGE ADDRESS
(1) 002646 062766 000002 000002 ADD @2,2(SP) ;;ADJUST RETURN PC
(1) 002654 105037 002631 CLR B $CNTRLO
(1)
(1) 002660 105737 002631 TYPE1: TSTB $CNTRLO ;;BRANCH IF CONTROL 0(↑0) WASN'T TYPED
(1) 002664 001410 BEQ TYPE2
(1) 002666 000004 002632 TCRLF: TYPE,$CRLF ;;TYPE <CR><LF>
(1) 002672 105737 002636 TSTB RDSW
(1) 002676 100006 BPL TYPE3
(1) 002700 005037 002636 CLR RDSW
(1) 002704 000207 RTS PC
(1) 002706 112046 TYPE2: MOV B (R0),-(SP) ;;PUSH CHARACTER TO BE TYPED ONTO STACK
(1) 002710 001003 BNE TYPE4 ;;BRANCH IF NOT THE TERMINATOR

```

```

(1) 002712 005726          TST      (SP)+      ;;POP TERMINATOR CHAR OFF THE STACK
(1) 002714 012600          TYPE3: MOV      (SP)+,R0    ;;RESTORE R0
(1) 002716 000002          RTI                      ;;RETURN TO CALLER
(1)
(1) 002720 122716 000011          TYPE4: CMPB     #HT,(SP)    ;;BRANCH IF HORIZONTAL TAB <HT>
(1) 002724 001500          BEQ      11$
(1) 002726 004737 002760          JSR      PC,5$          ;;TYPE CHARACTER
(1) 002732 122726 000012          3$:  CMPB     #12,(SP)+    ;;CHECK IF CHARACTER WAS A LINE FEED
(1) 002736 001350          BNE      TYPE1          ;;BRANCH IF NOT LINE FEED
(1) 002740 013746 002616          MOV      $NULL,-(SP)    ;;GET # OF FILLERS REQUIRED AND FILLER
(1)                                ;;CHARACTER.
(1)
(1) 002744 105366 000001          4$:  DECB     1(SP)        ;;DECREMENT FILLERS REQ. COUNT
(1) 002750 002770          BLT      3$            ;;BRANCH IF NO MORE FILLERS ARE REQUIRED
(1) 002752 004737 002760          JSR      PC,5$
(1) 002756 000772          BR       4$
(1)
(1) 002760 105737 177560          5$:  TSTB     TKS           ;;SEE IF INPUT AVAILABLE
(1) 002764 100433          BMI     9$            ;;BRANCH IF SO
(1) 002766 105777 177630          TSTB     #TPS          ;;WAIT FOR OUTPUT DEVICE
(1) 002772 100372          BPL     5$
(1) 002774 105737 002626          TSTB     $CTRLS        ;;SEE IF WE'RE IN XOFF MODE
(1) 003000 100767          BMI     5$            ;;IF SO, WAIT FOR XON
(1) 003002 122737 000017 002631          CMPB     #17,#$CNTRL0  ;;CHECK IF CONTROL O WAS TYPED
(1) 003010 001403          BEQ     6$            ;;STOP TYPING MESSAGE IF O WAS TYPED
(1) 003012 116677 000002 177604          MOVB     2(SP),#$TPB   ;;OUTPUT CHARACTER
(1) 003020 122766 000015 000002          6$:  CMPB     #15,2(SP)    ;;BRANCH IF NOT <CR>
(1) 003026 001003          BNE     7$
(1) 003030 105037 002630          CLRB     $CHARCNT      ;;CLEAR CHARACTERS TYPED COUNT
(1) 003034 000406          BR      8$
(1) 003036 122766 000012 000002          7$:  CMPB     #12,2(SP)    ;;BRANCH IF <LF> OR 'NULL'
(1) 003044 002002          BGE     8$
(1) 003046 105237 002630          INCB     $CHARCNT      ;;INCREMENT CHARACTER TYPED COUNT
(1) 003052 000207          8$:  RTS      PC
(1)
(1) 003054 113737 177562 002627          9$:  MOVB     TKB,$CTRLS+1 ;;MOVE CHARACTER INTO BUFFER
(1) 003062 142737 000200 002627          BICB     #200,$CTRLS+1 ;;CLEAR PARITY BIT
(1) 003070 122737 000023 002627          CMPB     #23,$CTRLS+1 ;;SEE IF XOFF TYPED AT KEYBOARD
(1) 003076 001004          BNE     10$           ;;IF SO THEN
(1) 003100 112737 000377 002626          MOVB     #377,$CTRLS  ;;SET XOFF FLAG
(1) 003106 000724          BR      5$
(1) 003110 122737 000021 002627          10$: CMPB     #21,$CTRLS+1 ;;SEE IF XON TYPED
(1) 003116 001320          BNE     5$            ;;IF SO THEN
(1) 003120 105037 002626          CLRB     $CTRLS        ;;CLEAR XOFF FLAG
(1) 003124 000715          BR      5$
(1)
(1)                                ;;HORIZONTAL TAB <HT> PROCESSER
(1) 003126 112716 000040          11$: MOVB     #40,(SP)    ;;LOAD 'SPACE'
(1) 003132 004737 002760          12$: JSR      PC,5$          ;;TYPE 'SPACE'
(1) 003136 132737 000007 002630          BITB     #7,$CHARCNT   ;;TYPE SPACES UNTIL A MULTIPLE
(1) 003144 001372          BNE     12$           ;;OF 8 CHARACTERS HAVE BEEN TYPED
(1) 003146 105726          TSTB     (SP)+        ;;POP SPACE
(1) 003150 000643          BR      TYPE1          ;;GET NEXT CHARACTER

```

1551
1552
1553

```

;SUBROUTINE TO SAVE GENERAL REGISTERS ON THE STACK
;CALL:  SAVE

```

134

```

1554 003152 010546      .SAVE:  MOV     R5, (SP)      ;SAVE REGISTERS ON THE STACK
1555 003154 010446      MOV     R4, (SP)
1556 003156 010346      MOV     R3, (SP)
1557 003160 010246      MOV     R2, (SP)
1558 003162 010146      MOV     R1, (SP)
1559 003164 010046      MOV     R0, (SP)
1560 003166 016646 000014  MOV     14(SP), (SP)      ;GET RETURN PC
1561 003172 000207      RTS     PC                  ;RETURN
1562
1563      ;SUBROUTINE TO RESTORE GENERAL REGISTERS FROM THE STACK
1564      ;CALL:  RESTORE
1565 003174 012666 000014  .RESTORE:MOV    (SP),14(SP)      ;MOVE RETURN PC
1566 003200 012600      MOV    (SP),R0             ;RESTORE REGISTERS
1567 003202 012601      MOV    (SP),R1
1568 003204 012602      MOV    (SP),R2
1569 003206 012603      MOV    (SP),R3
1570 003210 012604      MOV    (SP),R4
1571 003212 012605      MOV    (SP),R5
1572 003214 000207      RTS     PC                  ;RETURN
1573
1574      ;SUBROUTINE TO CONVERT OCTAL DATA TO ASCII
1575      ;CALL:  MOV     NUMBER,R2      ;MOVE NUMBER TO R2
1576      ;      JSR     PC,CNVOCT
1577
1578 003216 110637 001133  CNVOCT: MOV    SP,TYPFLG      ;SET DO NOT TYPE FLAG
1579 003222 000402      BR     CNVTO
1580
1581      .SBTTL      OCTAL TO ASCII & TYPE ROUTINE
1582      ;SUBROUTINE TO CONVERT OCTAL NUMBER TO ASCII AND TYPE IT OUT
1583      ;CALL:  MOV     NUMBER,R2      ;PUT # IN R2
1584      ;      JSR     PC,TYPOCT      ;CALL ROUTINE
1585
1586 003224 105037 001133  TYPOCT: CLRB   @TYPFLG      ;SET TYPE FLAG
1587 003230  CNVTO:
1588 (1) 003230 004737 003152      JSR    PC,.SAVE            ;SAVE REGISTERS ON THE STACK
1589 003234 012704 001152      MOV    @ODIGITS,R4        ;SET PTR TO OUTPUT
1590 003240 005003      CLR    R3                ;R3 WILL CONTAIN OCTAL DIGIT
1591 003242 010201      MOV    R2,R1             ;GET # TO BE TYPED
1592 003244 006302      18:   ASL    R2            ;SHIFT #
1593 003246 006103      ROL    R3
1594 003250 012700 000006      MOV    @6,R0             ;SET DIGIT COUNTER
1595 003254 000404      BR     38
1596 003256 005302      28:   ASL    R2            ;SHIFT # 3 PLACES LEFT
1597 003260 006103      ROL    R3
1598 003262 005301      DEC    R1
1599 003264 001374      BNE    28
1600 003266 012701 000003      38:   MOV    @3,R1            ;SET SHIFT COUNTER
1601 003272 116324 001140      MOV    DIGTAB(R3),(R4)    ;MOVE ASCII EQUIV TO OUTPUT
1602 003276 005003      CLR    R3
1603 003300 005300      DEC    R0                ;DECREMENT DIGIT COUNT
1604 003302 001365      BNE    28                ;GET NEXT DIGIT
1605 003304 105737 001133      TSTB   @TYPFLG           ;BRANCH IF ASCII IS
1606 003310 001002      BNE    48                ;NOT TO BE TYPED
1607 003312 000004 001152      TYPE ,ODIGITS
1608 003316      48:

```

C4

```

(1) 003316 004737 003174 JSR PC,RESTORE ;RESTORE REGISTERS FROM THE STACK
1609 003322 000207 RTS PC
1610
1611
1612 ;SUBROUTINE TO CONVERT OCTAL DATA TO DECIMAL ASCII
1613 ;CALL: MOV NUMBER,R2 ;MOVE NUMBER TO R2
1614 ; JSR PC,CNVDEC
1615
1616 003324 110637 001133 CNVDEC: MOVB SP,@TYPFLG ;SET DO NOT TYPE FLAG
1617 003330 000402 BR CNVTD
1618 .SBTTL OCTAL TO DECIMAL & TYPE ROUTINE
1619 ;THIS ROUTINE CONVERTS AN OCTAL # TO DECIMAL ASCII AND TYPES IT OUT
1620 ;CALL: MOV NUMBER,R2 ;PUT # IN R2
1621 ; JSR PC,TYPDEC ;CALL ROUTINE
1622
1623 003332 105037 001133 TYPDEC: CLRB @TYPFLG ;SET TYPE FLAG
1624 003336 CNVTD:
(1) 003336 004737 003152 JSR PC,SAVE ;SAVE REGISTERS ON THE STACK
1625 003342 005000 CLR R0 ;R0 IS INDEX TO DECIMAL CONSTANT
1626 003344 012704 001152 MOV #ODIGITS,R4 ;SET OUTPUT PTR
1627 003350 005003 1#: CLR R3 ;R3 CONTAINS DECIMAL DIGIT
1628 003352 166002 003432 2#: SUB DCONST(R0),R2 ;SUBTRACT DECIMAL CONSTANT UNTIL
1629 003356 103402 BLO 3# ;INPUT # GOES NEGATIVE
1630 003360 005203 INC R3 ;KEEPING TRACK OF SUBTRACTIONS
1631 003362 000773 BR 2#
1632 003364 066002 003432 3#: ADD DCONST(R0),R2 ;ADD BACK CONSTANT WHEN NEGATIVE
1633 003370 116324 001140 MOVB DIGTAB(R3),(R4) ;MOVE ASCII EQUIVALENT
1634 003374 062700 000002 ADD #2,R0 ;NEXT CONSTANT
1635 003400 005760 003432 TST DCONST(R0) ;UNTIL ALL CONSTANTS DONE
1636 003404 001361 BNE 1#
1637 003406 112724 000060 MOVB #'0,(R4) ;LAST DIGIT IS 0
1638 003412 105737 001133 TSTB @TYPFLG ;BRANCH IF ASCII IS
1639 003416 001002 BNE 4# ;NOT TO BE TYPED
1640 003420 000004 001152 TYPE,ODIGITS
1641 003424
(1) 003424 004737 003174 JSR PC,RESTORE ;RESTORE REGISTERS FROM THE STACK
1642 003430 000207 RTS PC
1643
1644 003432 023420 DCONST: .WORD 10000.
1645 003434 001750 .WORD 1000.
1646 003436 000144 .WORD 100.
1647 003440 000012 .WORD 10.
1648 003442 000001 .WORD 1.
1649 003444 000000 .WORD 0 ;TERMINATOR
1650
1651 .SBTTL TYPE SPECIFIED TIMES ROUTINE
1652 ;THIS SUBROUTINE OUTPUTS THE TIME SPECIFICATIONS FOR THE TEST
1653 ;AND ALSO THE ACTUAL TIME RECORDED (ATIME)
1654 ;FORMAT OF LINE TYPED
1655 ;RANGE-<AAAAAA-BBBBBB> ACTUAL=CCCCC
1656 ;WHERE: AAAAAA IS MAXIMUM TIME FOR TEST (STIMTBL(TSTNUMX4)).
1657 ; BBBBBB IS MINIMUM TIME FOR TEST (STIMTBL(TSTNUMX4*2)).
1658 ; CCCCCC IS ACTUAL TIME RECORDED BY TEST (ATIME).
1659 ;CALL: MOVB TEST NUMBER,R2 ;LOAD TEST NUMBER
1660 ; MOV #TIME,@ATIME ;MOVE TIME TO ATIME
1661 ; JSR PC,OUTSPC

```

1662 003446 010246
 1663 003450 010346
 1664 003452 006302
 1665 003454 006302
 1666 003456 010203
 1667 003460 000004 016560
 1668 003464 016302 002124
 1669 003470 004737 003332
 1670 003474 000004 001413
 1671 003500 016302 002126
 1672 003504 004737 003332
 1673 003510 000004 001420
 1674 003514 000004 016570
 1675 003520 013702 001016
 1676 003524 004737 003332
 1677 003530 000004 001402
 1678 003534 012603
 1679 003536 012602
 1680 003540 000207

```

OUTSPC: MOV     R2, (SP)           ;SAVE R2 & R3 ON THE STACK
        MOV     R3, (SP)
        ASL     R2               ;MULTIPLY TEST # TIMES 4
        ASL     R2               ;TO FORM INDEX INTO STIMTBL
        MOV     R2,R3           ;R3 CONTAINS INDEX INTO TABLE
        TYPE,L.RNG
        MOV     STIMTBL(R3),R2   ;GET MAXIMUM SPEC TIME
        JSR     PC,TYPDEC       ;CONVERT TO DECIMAL & TYPE
        TYPE,DASH
        MOV     STIMTBL+2(R3),R2 ;GET MINIMUM TIME
        JSR     PC,TYPDEC       ;CONVERT TO DECIMAL & TYPE
        TYPE,ANGTAB
        TYPE,L.ACT
        MOV     @#ATIME,R2      ;GET ACTUAL TIME
        JSR     PC,TYPDEC       ;CONVERT TO DECIMAL & TYPE
        TYPE,CRLF
        MOV     (SP)+,R3
        MOV     (SP)+,R2
        RTS     PC              ;RETURN
  
```

1681
 1682
 1683
 1684
 1685
 1686
 1687
 1688
 1689

```

.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 MLT ROUTINE (MLT+2).
;CALL:  MOV#   @GAP,GAP          ;LOAD GAP # INTO GAP
;       MOV    @TIME,ATIME       ;LOAD ACTUAL TIME INTO ATIME
;       JSR    PC,OUTGAP
  
```

1690 003542 010246
 1691 003544 010346
 1692 003546 113703 001124
 1693 003552 006303
 1694 003554 006303
 1695 003556 000004 016560
 1696 003562 016302 002264
 1697 003566 004737 003332
 1698 003572 000004 001413
 1699 003576 016302 002266
 1700 003602 004737 003332
 1701 003606 000004 001420
 1702 003612 000004 016570
 1703 003616 013702 001016
 1704 003622 004737 003332
 1705 003626 000004 016235
 1706 003632 113702 001124
 1707 003636 004737 003224
 1708 003642 000004 001402
 1709 003646 012603
 1710 003650 012602
 1711 003652 000207

```

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,L.ACT
        MOV     @#ATIME,R2      ;GET ACTUAL TIME
        JSR     PC,TYPDEC       ;CONVERT TO DECIMAL & TYPE
        TYPE,E.GAP
        MOV#    @#GAP,R2        ;GET GAP #
        JSR     PC,TYPDEC       ;TYPE GAP #
        TYPE,CRLF
        MOV     (SP)+,R3
        MOV     (SP)+,R2
        RTS     PC              ;RESTORE R3 AND R2
  
```

1712
 1713
 1714
 1715
 1716 003654
 (1) 003654 004737 003152

```

.SBTTL      ASCII TO OCTAL CONVERT SUBROUTINE
;SUBROUTINE TO CONVERT ASCII DATA TO OCTAL. CONVERTED OCTAL DATA
;IS LEFT IN OCTALO <15 00>.
CNVTA0: JSR     PC, .SAVE        ;SAVE REGISTERS ON THE STACK
  
```

```

1717 003660 012700 001272      MOV      #INBUF,R0      ;SET PTR TO ASCII DATA
1718 003664 012701 001122      MOV      #OCTALO,R1    ;GET ADDRESS OF OCTAL DATA
1719 003670 005011              CLR      (R1)          ;CLEAR OUT OLD OCTAL DATA
1720 003672 005061 000002      CLR      2(R1)
1721 003676 122710 000015      1$:     CMPB     #CR,(R0) ;<CR> TERMINATES INPUT
1722 003702 001414              BEQ      3$
1723 003704 112002              MOVB     (R0),R2       ;GET 'OCTAL' DATA
1724 003706 042702 177770      BIC      #177770,R2    ;STRIP UNUSED BITS
1725 003712 012703 000003      MOV      #3,R3        ;SET SHIFT COUNT
1726 003716 006311              2$:     ASL      (R1)      ;SHIFT LAST
1727 003720 006161 000002      ROL      2(R1)        ;OCTAL DIGIT
1728 003724 005303              DEC      R3
1729 003726 001373              BNE      2$
1730 003730 050211              BIS      R2,(R1)      ;AND INSERT THIS DIGIT
1731 003732 000761              BR       1$          ;GO GET NEXT DIGIT
1732 003734              3$:
( ) 003734 004737 003174      JSR      PC,.RESTORE  ;RESTORE REGISTERS FROM THE STACK
1733 003740 000207              RTS       PC          ;RETURN

```

```

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

```

```

PUBLISH:
      JSR      PC,.SAVE      ;SAVE REGISTERS ON THE STACK
      MOV      #ATIMTBL,R0  ;GET TABLE ADDRESS CONTAINING TIMES
      MOVB     #ITCNT,R1    ;GET # OF ENTRIES (GIVEN BY ITERATION COUNT)
      CMPB     #1,R1        ;BRANCH IF SINGLE ITERATION
      BEQ      4$
      CLR      R2           ;CLEAR 'SUM' REGISTERS
      CLR      R3
      CMPEB   #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.
1$:     ADD      (R0),R2    ;SUM INDIVIDUAL TIMES
      ADC      R3
      DEC      R1
      BNE      1$
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
4$:     MOVB     #TSTNUM,R0 ;GET TEST #
      ASL      R0
      MOV      #NAMPTR(R0),5$ ;GET TEST NAME STRING ADDRESS
5$:     .WORD    0

```

```

1741 003742 (1) 003742 004737 003152
1742 003746 012700 001020
1743 003752 113701 001125
1744 003756 122701 000001
1745 003762 001423
1746 003764 005002
1747 003766 005003
1748 003770 122701 000020
1749 003774 001402
1750 003776 000000
1751 004000 000777
1752
1753
1754 004002 062002 1$:
1755 004004 005503
1756 004006 005301
1757 004010 001374
1758
1759 004012 012700 000004 2$:
1760 004016 006203 3$:
1761 004020 006002
1762 004022 005300
1763 004024 001374
1764 004026 010237 001016
1765
1766 004032 113700 001126 4$:
1767 004036 006300
1768 004040 016037 002364 004050
1769 004046 000004
1770 004050 000000 5$:

```

14

1771	004052	113702	001126	MOVB	@TSTNUM,R2	;GET TEST #
1772	004056	004737	003446	JSR	PC,OUTSPC	;OUTPUT TIMES
1773	004062	004737	003174	JSR	PC,.RESTORE	;RESTORE REGISTERS FROM THE STACK
1774	004066	000207		RTS	PC	
1775						
1776						
1777						
1778						
1779						
1780						
1781	004070	010046				
1782	004072	012700	001272	.INPUT: MOV	RO,-(SP)	;SAVE RO ON THE STACK
1783	004076	105737	177560	1\$: MOV	@INBUF,RO	
1784	004102	100375		2\$: TSTB	@TKS	
1785				BPL	2\$	
1786	004104	113746	177562	MOVB	@TKB,-(SP)	;GET CHARACTER
1787	004110	042716	000200	BIC	@200,(SP)	
1788	004114	122716	000177	CMPB	@177,(SP)	;CHECK RUBOUT
1789	004120	001004		BNE	3\$	
1790	004122	124026		CMPB	-(RO),(SP).	;REMOVE CHARACTER FROM INPUT
1791	004124	000004	001405	TYPE,BKSLSH		
1792	004130	000762		BR	2\$;WAIT FOR NEXT CHARACTER
1793	004132	122716	000025	3\$: CMPB	@CNTRLU,(SP)	;CHECK CONTROL U (+U)
1794	004136	001004		BNE	4\$	
1795	004140	005726		TST	(SP).	
1796	004142	000004	001402	TYPE,CRLF		
1797	004146	000751		BR	1\$	
1798	004150	122716	000003	4\$: CMPB	@CNTRLC,(SP)	;BRANCH IF NOT CONTROL C
1799	004154	001003		BNE	40\$	
1800	004156	000005		RESET		;RESET I/O
1801	004160	000137	006466	JMP	@INIT	;RESTART PROGRAM
1802	004164	111637	001407	40\$: MOVB	(SP),@ECHO	
1803	004170	111620		MOVB	(SP),(RO).	
1804	004172	122726	000015	CMPB	@CR,(SP).	
1805	004176	001403		BEQ	5\$	
1806	004200	000004	001407	TYPE,ECHO		
1807	004204	000734		BR	2\$	
1808	004206	000004	001402	5\$: TYPE,CRLF		
1809	004212	012600		MOV	(SP),RO	
1810	004214	000207		RTS	PC	

```

1812
1813 ;KEYBOARD INTERRUPT SERVICE ROUTINE
1814
1815 004216 113746 177562 TKISR: MOVB @TKB,(SP) ;GET TYPED CHARACTER
1816 004222 042716 000200 BIC @200,(SP) ;STRIP PARITY BIT
1817 004226 122716 000017 CMPB @CNTRLO,(SP) ;BRANCH IF NOT CONTROL 0 (^O)
1818 004232 001002 BNE 1$
1819 004234 111637 002631 MOVB (SP),%CNTRLO ;SET CONTROL 0 INDICATOR IN TYPE ROUTINE
1820
1821 004240 122716 000003 1$: CMPB @3,(SP) ;BRANCH IF NOT CONTROL C (^C)
1822 004244 001007 BNE 2$
1823 004246 023727 000042 013676 CMP @@42,@$ENDAD ;INHIBIT ^C IF ACT11 QV OR AA
1824 004254 001403 BEQ 2$
1825 004256 000005 RESET
1826 004260 000137 006466 JMP @INIT ;RESTART PROGRAM
1827
1828 004264 122716 000001 2$: CMPB @CNTR^A,(SP) ;BRANCH IF NOT ^A
1829 004270 001011 BNE 3$
1830 004272 022737 000176 001000 CMP @SWREG,SWR ;BRANCH IF HARDWARE SWR IS INVOKED
1831 004300 001010 BNE 4$
1832 004302 012737 177570 001000 MOV @177570,SWR ;INVOKE HARDWARE SWR
1833 004310 000004 014506 TYPE,M.HSWR
1834 004314 122716 000007 3$: CMPB @CNTRLG,(SP) ;BRANCH IF NOT ^G
1835 004320 001006 BNE 5$
1836 004322 012737 000176 001000 4$: MOV @SWREG,SWR ;INVOKE SOFTWARE SWR
1837 004330 004737 002526 JSR PC,GTSWR ;GET NEW SWITCH REGISTER
1838 004334 000414 BR 7$
1839 004336 122716 000023 5$: CMPB @23,(SP) ;SEE IF ^S
1840 004342 001004 BNE 6$ ;BRANCH IF NOT
1841 004344 112737 000377 002626 MOVB @377,%CTRLS ;SET XOFF FLAG
1842 004352 000405 BR 7$
1843 004354 122716 000021 6$: CMPB @21,(SP) ;SEE IF ^Q
1844 004360 001002 BNE 7$ ;BRANCH IF NOT
1845 004362 105037 002626 CLRB %CTRLS
1846 004366 005726 7$: TST (SP)+ ;POP CHARACTER OFF STACK
1847 004370 000002 RTI ;RETURN
1848

```

```

1850          .SBTTL          ERROR SERVICE ROUTINES
1851          ;ROUTINE TO PROCESS ERROR TRAPS (TRAPS TO 4)
1852 004372 000000          ERRTRP: HALT
1853
1854          ;ERROR SERVICE ROUTINE
1855          ;THIS ROUTINE PROCESSES TWO TYPES OF ERRORS (OUT OF RANGE AND HARDWARE)
1856          ;THE CALLS FOR AN OUT OF RANGE ERROR ARE <HLT+1>,<HLT+2> AND, FOR A
1857          ;HARDWARE ERROR THE CALL IS <HLT>.
1858
1859 004374 004737 003152          .HLT: JSR PC,SAVE          ;SAVE REGISTERS ON THE STACK
1860 004400 110637 001127          1$: MOVB SP,@ERFLG          ;SET ERROR FLAG
1861 004404 032777 020000 174366          BIT @SW13,@SWR          ;BRANCH IF NO TYP0UT
1862 004412 001121          BNE 4$
1863 004414 000004 015331          TYPE,E.HDR
1864 004420 113702 001126          MOVB @@TSTNUM,R2          ;GET TEST #
1865 004424 004737 003224          JSR PC,TYPOCT          ;AND TYPE IT
1866 004430 016600 000016          MOV 16(SP),R0          ;GET RETURN PC
1867 004434 162700 000002          SUB @2,R0          ;NOW PC OF HLT CALL
1868 004440 111000          MOVB (R0),R0          ;NOW HLT CALL ITSELF
1869 004442 001443          BEQ 2$          ;BRANCH IF HLT
1870 004444 000004 015414          TYPE,E.HDR2
1871 004450 122737 000005 001126          CMPB @5,@@TSTNUM          ;SEE IF IT IS TEST 5
1872 004456 001006          BNE 99$          ;CONTINUE IF NOT TEST 5
1873 004460 122737 000001 001136          CMPB @1,@@TE16          ;CHECK DRIVE TYPE
1874 004466 001002          BNE 99$          ;BRANCH IF NOT TU77
1875 004470 000004 015442          TYPE,E.77T5          ;TYP TU77 SPECIFIC MESSAGE
1876 004474 122737 000016 001126 99$: CMPB @16,@@TSTNUM          ;SEE IF IT IS TEST 16
1877 004502 001006          BNE 98$          ;CONTINUE IF NOT TEST 16
1878 004504 122737 000001 001136          CMPB @1,@@TE16          ;CHECK DRIVE TYPE
1879 004512 001002          BNE 98$          ;BRANCH IF NOT TU77
1880 004514 000004 015711          TYPE,E.7T16          ;TYP TU77 SPECIFIC MESSAGE
1881 004520 122700 000002 98$: CMPB @2,R0          ;BRANCH IF NOT HLT+2
1882 004524 001005          BNE 10$
1883 004526 004737 003542          JSR PC,OUTGAP          ;TYPE GAP SPECIFIED TIMES
1884 004532 000004 001402          TYPE,CRLF
1885 004536 000447          BR 4$
1886 004540 004737 003446 10$: JSR PC,OUTSPC          ;TYPE SPECIFIED TIMES
1887 004544 000004 001402          TYPE,CRLF
1888 004550 000442          BR 4$
1889 004552 016500 000014 2$: MOV ER(R5),R0
1890 004556 032765 002300 000032          BIT @PE1600,TC(R5)
1891 004564 001403          BEQ 20$
1892 004566 042700 102100          BIC @102100,R0
1893 004572 000402          BR 21$
1894 004574 042700 102300 20$: BIC @102300,R0
1895 004600 005700 21$: TST R0
1896 004602 001003          BNE 22$
1897 004604 000004 015305          TYPE,E.SFT          ;TYPE SOFT ERROR MESSAGE
1898 004610 000434          BR 6$
1899
1900 004612 000004 015341 22$: TYPE,E.HDR1
1901 004616 010500          MOV R5,R0          ;GET FIRST ADDRESS OF REGS.
1902 004620 012701 000007          MOV @7.,R1          ;TYPE FIRST 7 REGS.
1903 004624 012002 3$: MOV (R0),R2          ;GET REG CONTENTS
1904 004626 004737 003224          JSR PC,TYPOCT          ;AND TYPE IT
1905 004632 000004 001415          TYPE,SPACE2

```

1906	004636	005301				DEC	R1	
1907	004640	001371				BNE	3*	
1908	004642	016502	000032			MOV	TC(R5),R2	;GET CONTENTS OF TC REGISTER
1909	004646	004737	003224			JSR	PC,TYPECT	
1910	004652	000004	001402			TYPE,CRLF		
1911								
1912	004656	032777	001000	174114	4*:	BIT	@SW09,@SWR	;BRANCH IF NO RING THE BELL
1913	004664	001402				BEQ	5*	
1914	004666	000004	001411			TYPE,BELL		
1915	004672	005777	174102		5*:	TST	@SWR	;HALT ON ERROR?
1916	004676	100001				BPL	6*	
1917	004700	000000				HALT		
1918	004702				6*:			
(1)	004702	004737	003174			JSR	PC,.RESTORE	;RESTORE REGISTERS FROM THE STACK
1919	004706	000002				RTI		;RETURN
1920								
1921								

14

```

1923          .SBTTL          SCOPE SUBROUTINE
1924          ;SCOPE ROUTINE
1925          ;THIS ROUTINE IS ENTERED UPON COMPLETION OF EACH SUBTEST
1926          ;THE SCOPE ROUTINE:
1927          ;
1928          ;   REPEATS TEST IF SW14 IS SET
1929          ;   STORES ACTUAL TIME FOR FUNCTION IN TIME TABLE (ATIMTBL)
1930          ;   PUBLISHES TIME IF SW10=0
1931          ;   UPDATES ITERATION COUNT AND IF ITERATIONS COMPLETE CONTINUES
1932          ;   TO NEXT TEST, OTHERWISE REPEATS TEST.
1933          ;   DELAYS BEFORE CONTINUING OR REPEATING TEST.
1934          ;   INITIALIZES DRIVE
1935          ;RETURNS:      R5=BASE ADDRESS OF TM03 REGISTERS (ADDRESS OF CS1)
1936          ;              R1='DS' REG ADDRESS
1937          ;              R0='FC' REG ADDRESS
1938          004710 013705 001010          .SCOPE: MOV      @@TMBASE,R5          ;SET R5 TO FIRST TM REG
1939          004714 032777 040000 174056  BIT      @SW14,@SWR          ;BRANCH IF CONTINUOUS LOOP
1940          004722 001432                   BEQ      2$          ;NOT DESIRED
1941          004724 017701 174050          1$:  MOV      @SWR,R1          ;GET SWITCHES
1942          004730 042701 177740          BIC      @177740,R1          ;CLEAR ALL BUT TEST #
1943          004734 001406                   BEQ      11$          ;BRANCH IF ALL SELECTED
1944          004736 120137 001126          CMPB    R1,@@TSTNUM          ;BRANCH IF RUNNING SELECTED TEST
1945          004742 001403                   BEQ      11$
1946          004744 012737 010172 001002  MOV      @@TST000,SCPADR          ;RESTART AT TST000
1947          004752 004737 005502          11$:  JSR      PC,DELAY          ;DELAY 350 MS
1948          004756 004737 005736          JSR      PC,RHINIT          ;INIT
1949          004762 105037 001127          CLRB    @@ERFLG          ;CLEAR ERROR FLAG
1950          004766 013716 001002          MOV      SCPADR,(SP)
1951          004772 010501          MOV      R5,R1
1952          004774 062701 000012          ADD     @DS,R1          ;ADDRESS OF 'DS' REG IS IN R1
1953          005000 010500          MOV      R5,R0
1954          005002 062700 000006          ADD     @FC,R0          ;ADDRESS OF 'FC' REG IS IN R0
1955          005006 000002          RTI
1956
1957          005010 105737 001127          2$:  TSTB    @@ERFLG          ;BRANCH IF ERROR FLAG IS SET
1958          005014 001006                   BNE     3$
1959          005016 113700 001125          MOVB    @@ITCNT,R0          ;GET ITERATION COUNT
1960          005022 006300                   ASL     R0          ;STORE TIME IN TABLE
1961          005024 013760 001016 001020  MOV      @@ATIME,ATIMTBL(R0)
1962          005032 105237 001125          3$:  INCB    @@ITCNT          ;INCREMENT ITERATION COUNT
1963          005036 105737 001134          TSTB    @@PSCNT          ;INHIBIT ITERATIONS ON
1964          005042 001410                   BEQ     4$          ;ON FIRST PASS
1965          005044 032777 004000 173726  BIT      @SW11,@SWR          ;BRANCH IF SINGLE ITERATION DESIRED
1966          005052 001004                   BNE     4$
1967          005054 122737 000020 001125  CMPB    @16.,@@ITCNT          ;BRANCH IF ITERATIONS INCOMPLETE
1968          005062 001320                   BNE     1$
1969          005064 032777 000037 173706  4$:  BIT      @37,@SWR          ;IF TEST SELECTED IS TEST 0
1970          005072 001002                   BNE     42$          ;TREAT AS ALL TESTS
1971          005074 011637 001002          40$:  MOV      (SP),@@SCPADR          ;SET SCOPE ADDRESS TO NEXT TEST
1972          005100 032777 002000 173672  42$:  BIT      @SW10,@SWR          ;BRANCH IF NO PUBLICATION DESIRED
1973          005106 001002                   BNE     5$
1974          005110 004737 003742          JSR     PC,PUBLISH          ;GO PUBLISH TEST DATA
1975          005114 105037 001125          5$:  CLRB    @@ITCNT          ;RESET ITERATION COUNT
1976          005120 000701                   BR      1$
1977
1978          .SBTTL          TIMER SUBROUTINES

```

```

1979
1980 ;SUBROUTINE TO SYNCHRONIZE THE TIMER AND TURN IT ON.
1981 ;REGISTER 4 IS CLEARED, AND THE OSCILLATOR POLARITY IS MONITORED
1982 ;THE ROUTINE IS EXITED WHEN THE OSCILLATOR POLARITY CHANGES WITH R3
1983 ;SET TO INDICATE THE POLARITY OF THE OSCILLATOR.
1984 ;CALL: JSR PC,TIMON
1985 ;RETURNS: R3 SET TO INDICATE LAST POLARITY (.24/ 24=0/1)
1986 ;
1987 ;
1988 005122 005004 TIMON: CLR R4 ;CLEAR TIME COUNT
1989 005124 012703 000024 MOV #24,R3 ;SET POLARITY TO '0' STATE
1990 005130 032765 000100 000024 BIT #OSC,MR(R5) ;BRANCH IF POLARITY IS '0'
1991 005136 001405 BEQ 2$
1992 005140 032765 000100 000024 1$: BIT #OSC,MR(R5) ;WAIT FOR OSCILLATOR TO RETURN
1993 005146 001374 BNE 1$
1994 005150 000405 BR 4$
1995
1996 005152 005403 2$: NEG R3 ;NEGATE PREV POLARITY INDICATOR
1997 005154 032765 000100 000024 3$: BIT #OSC,MR(R5) ;WAIT FOR OSCILLATOR TO RETURN
1998 005162 001774 BEQ 3$ ;TO '1' STATE
1999 005164 000207 4$: RTS PC
2000
2001 ;SUBROUTINE TO COUNT TIME
2002 ;EACH TIME THE OSCILLATOR TOGGLES (BIT <06> IN MR REG) REGISTER
2003 ;R4 IS INCREMENTED, AND THE REGISTER R3 IS NEGATED TO INDICATE
2004 ;THE LAST STATE OF THE OSCILLATOR.
2005 ;CALL JMP TIMER(R3) ;R3 IS SET BY TIMON ROUTINE
2006 ; R2=RETURN ADDRESS TO CALLER
2007 ;NOTE: THE TIME TO EXECUTE THIS ROUTINE IS VERY CRITICAL. IT MUST BE
2008 ;LESS THAN 40 US.
2009
2010 ;ENTER HERE VIA JMP TIMER(R3) WHEN R3=-24 (PREV STATE=1)
2011 005166 032765 000100 000024 TIMER1: BIT #OSC,MR(R5) ;BRANCH IF CURRENT STATE IS '0'
2012 005174 001406 BEQ TIMER ;GO INCREMENT TIME
2013 005176 000112 JMP (R2) ;RETURN TO TEST
2014
2015 ;
2016 005212 005212 TIMER: . =TIMER1+24
2017 005214 005204 NEG R3 ;NEGATE PREV STATE INDICATOR
2018 005216 100401 INC R4 ;INCREMENT 'TICK' COUNT
2019 005220 000112 BMI TIMERR ;BRANCH ON OVERFLOW
2020 005222 000004 016147 JMP (R2) ;RETURN TO TEST
2021 005226 104400 TIMERR: TYPE,E.TIMOV ;TYPE 'TIMER OVERFLOWED'
2022 005230 000177 173546 HLT ;REPORT HARDWARE ERROR
2023 ;
2024 ;
2025 ;ENTER HERE VIA JMP TIMER(R3) WHEN R3=.24 (PREV STATE=0)
2026 005236 032765 000100 000024 TIMERO: BIT #OSC,MR(R5) ;BRANCH IF CURRENT STATE = 1
2027 005244 001362 BNE TIMER
2028 005246 000112 JMP (R2)
2029
2030 ;SUBROUTINE TO CHECK TIME RECORDED BY SUBTEST.
2031 ;THIS SUBROUTINE COMPUTES THE ACTUAL TIME (IN MICROSECONDS) AND CHECKS
2032 ;THAT THE TIME RECORDED BY THE SUBTEST IS CORRECT BY COMPARING THE TIME
2033 ;WITH THE HIGH LIMIT (STIMTBL(R0)) AND THE LOW LIMIT (STIMTBL+2(R0)).
2034 ;IF THE TIME IS OUT OF RANGE AN OUT OF RANGE ERROR TIMEOUT RESULTS.

```

```

2035 ;THE SUBROUTINE IS ENTERED WITH:
2036 ; R4=TICK COUNT
2037
2038 TIMOK:
(1) 005250 004737 003152 JSR PC,SAVE ;SAVE REGISTERS ON THE STACK
2039 005254 013700 001012 MOV @OSCTIM,R0 ;GET TIME PER TICK
2040 005260 010401 MOV R4,R1 ;GET TICKS COUNT
2041 005262 005002 CLR R2 ;CLEAR SUMMING REGISTERS
2042 005264 005003 CLR R3
2043 005266 060002 1$: ADD R0,R2 ;MULTIPLY TIME PER TICK
2044 005270 005503 ADC R3 ;BY TICK COUNT
2045 005272 005301 DEC R1
2046 005274 001374 BNE 1$
2047 005276 010246 MOV R2,(SP) ;DIVIDE COUNT BY 10.
2048
2049 005300 010346 MOV R3,-(SP)
2050 005302 012746 000012 MOV #10,-(SP)
2051 005306 004737 005570 JSR PC,DIVIDE
2052 005312 005726 TST (SP)+ ;DISCARD REMAINDER
2053 005314 012637 001016 MOV (SP)+,@#ATIME ;STORE QUOTIENT
2054 005320 113700 001126 MOVB @#TSTNUM,R0 ;GET TEST #
2055 005324 006300 ASL R0
2056 005326 006300 ASL R0
2057 005330 023760 001016 002124 CMP @#ATIME,STIMTBL(R0) ;CHECK THAT TIME IS WITHIN
2058 005336 101004 BHI 2$ ;LIMITS SPECIFIED
2059 005340 023760 001016 002126 CMP @#ATIME,STIMTBL+2(R0)
2060 005346 101001 BHI 3$
2061 005350 104401 2$: HLT,1 ;CALL ERROR ROUTINE
2062 005352 3$:
(1) 005352 004737 003174 JSR PC,.RESTORE ;RESTORE REGISTERS FROM THE STACK
2063 005356 000207 RTS PC ;RETURN
2064
2065 ;SUBROUTINE TO CHECK INDIVIDUAL GAP TIMES (PRODUCED BY TST021)
2066 ;SUBROUTINE COMPUTES THE ACTUAL TIME (IN MICROSECONDS) AND CHECKS
2067 ;THAT THE GAP TIME RECORDED BY THE SUBTEST (TST021) BY COMPARING THE
2068 ;TIME WITH THE MAX LIMIT (GTIMTBL-GAPTBL(R1)) AND THE MIN LIMIT
2069 ;(GTIMTBL+2-GAPTBL(R1)).
2070 ;CALL: MOV #TICK COUNT,R4 ;R4 CONTAINS TICK COUNT
2071 ; MOVB #GAP,@#GAP ;LOCATION GAP CONTAINS GAP #
2072 ; JSR PC,GAPOK
2073
2074 GAPOK:
(1) 005360 004737 003152 JSR PC,SAVE ;SAVE REGISTERS ON THE STACK
2075 005364 013700 001012 MOV @#OSCTIM,R0 ;GET TIME PER TICK
2076 005370 010401 MOV R4,R1 ;GET TICK COUNT
2077 005372 005002 CLR R2 ;CLEAR SUMMING REGISTERS
2078 005374 005003 CLR R3
2079 005376 060002 1$: ADD R0,R2 ;MULTIPLY TICK COUNT
2080 005400 005503 ADC R3 ;BY TIME PER TICK
2081 005402 005301 DEC R1
2082 005404 001374 BNE 1$
2083
2084 005406 010246 MOV R2,-(SP) ;DIVIDE TIME BY 10.
2085 005410 010346 MOV R3,-(SP)
2086 005412 012746 000012 MOV #10,-(SP)
2087 005416 004737 005570 JSR PC,DIVIDE

```

```

2088 005422 005726          TST      (SP)+          ;DISCARD REMAINDER
2089 005424 012637 001016    MOV      (SP)+,@@ATIME    ;STORE QUOTIENT
2090 005430 113703 001124    MOVB    @@GAP,R3        ;GET GAP #
2091 005434 006303          ASL      R3              ;MULTIPLY BY 4
2092 005436 006303          ASL      R3              ;TO GET AT TABLE ENTRY
2093 005440 023763 001016 002264    CMP      @@ATIME,GTIMTBL(R?) ;CHECK TIME (MAX)
2094 005446 101004          BHI      2$              ;
2095 005450 023763 001016 002266    CMP      @@ATIME,GTIMTBL+2(R3) ;CHECK TIME (MIN)
2096 005456 101001          BHI      3$              ;
2097 005460 104402          HLT+2                ;REPORT OUT OF RANGE ERROR
2098 005462 032777 002000 173310 3$:    BIT      @SW10,@SWR      ;BRANCH IF TIMES NOT WANTED
2099 005470 001001          BNE      100$          ;
2100 005472 000240          NOP                    ;
2101
2102          005474          100$:
(1) 005474 004737 003174    JSR      PC,.RESTORE    ;RESTORE REGISTERS FROM THE STACK
2103 005500 000207          RTS      PC            ;RETURN TO TEST
2104
2105          .SBTTL          DELAY SUBROUTINES
2106          ;THIS SUBROUTINE CAUSES A DELAY OF 115 MS.
2107 005502 004737 005122    DELAY: JSR      PC,TIMON
2108 005506 010246          MOV      R2,-(SP)      ;SAVE R2 ON THE STACK
2109 005510 012702 005520    MOV      @2$,R2        ;SET RETURN ADDRESS FOR TIMER
2110 005514          1$:
(1) 005514 000163 005212    JMP      TIMER(R3)     ;GO TO TIMER & RETURN VIA R2
2111 005520 032704 004000    2$:    BIT      @4000,R4
2112 005524 001773          BEQ      1$
2113 005526 012602          MOV      (SP)+,R2     ;RESTORE R2
2114 005530 000207          RTS      PC
2115
2116          ;THIS SUBROUTINE ALLOWS A CALLER SPECIFIED DELAY.
2117          ;CALL: MOV      DELAY TIME,DELTIM    ;LOAD DELAY TIME (# OF TICKS)
2118          ;      JSR      PC,DELAYV
2119 005532 005737 001120    DELAYV: TST      DELTIM    ;BRANCH IF 0 DELAY
2120 005536 001413          BEQ      3$
2121 005540 004737 005122    JSR      PC,TIMON     ;TURN TIMER ON
2122 005544 010246          MOV      R2,-(SP)     ;SAVE R2 ON THE STACK
2123 005546 012702 005556    MOV      @2$,R2       ;SET RETURN ADDRESS FROM TIMER
2124 005552          1$:
(1) 005552 000163 005212    JMP      TIMER(R3)     ;GO TO TIMER & RETURN VIA R2
2125 005556 023704 001120    2$:    CMP      @DELTIM,R4
2126 005562 101373          BHI      1$
2127 005564 012602          MOV      (SP)+,R2     ;RESTORE R2
2128 005566 000207          3$:    RTS      PC
2129
2130          .SBTTL          DIVIDE SUBROUTINE
2131          ;THIS SUBROUTINE DIVIDES A DOUBLE PRECISION # AND RETURNS THE RESULT
2132          ;TO THE CALLER ON THE STACK. BOTH DIVIDEND & DIVISOR MUST BE POSITIVE.
2133          ;CALL: MOV      LEAST SIGNIFICANT HALF DIVIDEND,-(SP)
2134          ;      MOV      @MOST SIGNIFICANT HALF DIVIDEND,-(SP)
2135          ;      MOV      @DIVISOR,-(SP)
2136          ;      JSR      PC,DIVIDE
2137          ;RETURN
2138          ;      (SP)=REMAINDER ON STACK
2139          ;      2(SP)=QUOTIENT
2140

```

```

2141          ;NOTE: THIS SUBROUTINE DESTROYS PREVIOUS CONTENTS OF R0,R1,R2 & R3.
2142
2143 005570 005046          DIVIDE: CLR      -(SP)          ;SAVE LOC FOR SIGNS
2144 005572 012746 000021      MOV      #17,-(SP)      ;SET ITERATION COUNT
2145 005576 016601 000012      MOV      12(SP),R1      ;GET LSH DIVIDEND
2146 005602 016600 000010      MOV      10(SP),R0      ;GET MSH DIVIDEND
2147 005606 016602 000006      MOV      6(SP),R2       ;GET DIVISOR
2148 005612 005402          NEG      R2              ;NEGATE DIVISOR
2149 005614 000241          CLC              ;CLEAR 'C' BIT IN PSW
2150 005616 000405          BR      2$
2151 005620 006100          1$:  ROL      R0              ;ROTATE MSH DIVIDEND
2152 005622 010003          MOV      R0,R3          ;SAVE IN R3
2153 005624 060203          ADD      R2,R3          ;SUBTRACT DIVISOR FROM MSH DIVIDEND
2154 005626 103001          BCC     2$              ;BRANCH IF DIVIDEND > DIVISOR
2155 005630 010300          MOV      R3,R0          ;SAVE REMAINDER IN R0
2156 005632 006101          2$:  ROL      R1              ;ROTATE LSH DIVIDEND
2157 005634 005316          DEC     (SP)           ;DECREMENT ITERATION COUNT
2158 005636 001370          BNE     1$
2159 005640 005726          TST     (SP)+          ;POP ITERATION COUNTER
2160 005642 005726          TST     (SP)+          ;POP SIGN CORRECTION
2161 005644 010166 000006      MOV      R1,6(SP)       ;PUSH REMAINDER ON STACK
2162 005650 010066 000004      MOV      R0,4(SP)       ;PUSH QUOTIENT ONTO STACK
2163 005654 012616          MOV     (SP)+,(SP)
2164 005656 000207          RTS      PC
2165
2166          .SBTTL  DRIVE SUBROUTINES
2167          ;SUBROUTINE TO CHECK IF DRIVE IS AVAILABLE
2168          ;CALL:  MOVB   DRIVE#,DRVNUM
2169          ;      JSR    PC,DRVAVA
2170          ;RETURN: 'C' BIT SET IF NOT AVAILABLE
2171 005660 113765 001004 000010  DRVAVA: MOVB   @DRVNUM,CS2(R5)  ;LOAD DRIVE #
2172 005666 032765 040000 000026      BIT    #TAP,DT(R5)      ;CHECK IF TAPE UNIT
2173 005674 001003          BNE     1$
2174 005676 004737 005736          JSR    PC,RHINIT
2175 005702 000262          SEV              ;SET 'V' TO IND NOT AVAIL
2176 005704 000207          1$:  RTS      PC      ;RETURN
2177
2178          ;SUBROUTINE TO CHECK IF TE16/TU77 SLAVE IS AVAILABLE FOR TEST
2179          ;CALL:  MOVB   DRIVE #,@DRVNUM  ;PASS DRIVE # VIA DRVNUM
2180          ;      MOVB   SLAVE #,@SLVNUM   ;PASS SLAVE # VIA SLVNUM
2181          ;      JSR    PC,SLVAVA        ;CALL SUBROUTINE
2182 005706 113765 001004 000010  SLVAVA: MOVB   @DRVNUM,CS2(R5)  ;LOAD DRIVE #
2183 005714 113765 001005 000032      MOVB   @SLVNUM,TC(R5)    ;AND SLAVE #
2184 005722 032765 002000 000026      BIT    #SPR,DT(R5)      ;BRANCH IF SLAVE PRESENT
2185 005730 001001          BNE     1$
2186 005732 000262          SEV              ;SET 'V' TO INDICATE NO SLAVE
2187 005734 000207          1$:  RTS      PC
2188
2189          ;SUBROUTINE TO INITIALIZE RH CONTROLLER
2190          ;CALL:  JSR    PC,RHINIT
2191
2192 005736 012765 000040 000010  RHINIT: MOV    #40,CS2(R5)
2193 005744 113765 001004 000010      MOVB   @DRVNUM,CS2(R5)
2194 005752 005046          CLR    -(SP)
2195 005754 113716 001005          MOVB   @SLVNUM,(SP)
2196 005760 012665 000032          MOV    (SP)+,TC(R5)    ;LOAD SLAVE # INTO TC REG

```

```

197 005764 052765 001700 000032 BIS @NORM11,TC(R5)
198 005772 000207 RTS PC
2199
2200 ;SUBROUTINE TO WAIT FOR DRIVE READY (DRY)
2201 005774 005027 WAITRDY:CLR (PC); CLEAR WAIT TIMER
2202 005776 000000 WAITTIM: .WORD 0
2203 006000 105765 000012 1$: TSTB DS(R5); WAIT FOR READY TO SET
2204 006004 100406 BMI 2$
2205 006006 005237 005776 INC WAITTIM; INCREMENT WAIT TIMER
2206 006012 001372 BNE 1$; BRANCH IF TIME HAS NOT EXPIRED
2207 006014 000004 016174 TYPE,E,TIMEXP; TYPE 'TIME EXPIRED WAITING FOR RDY'
2208 006020 000425 BR 99$; TAKE ERROR EXIT
2209 006022 032765 002000 000012 2$: BIT @EOT,DS(R5); CHECK FOR END OF TAPE
2210 006030 001415 BEQ 3$; BRANCH IF NO EOT
2211 006032 000004 014326 TYPE,M,NAM; TYPE 'END OF TAPE'
2212 006036 000004 015045 TYPE,M,EOT; REWIND SLAVE
2213 006042 004737 006100 JSR PC,.REWIND; BRANCH IF ERROR ON REWIND
(1) 006046 102412 BVS 99$; WRITE A RECORD
2214 006050 004737 006162 JSR PC,WRITE; SET 'GO' BIT
2215 006054 005215 INC (R5); WAIT FOR READY
2216 006056 004737 005774 JSR PC,WAITRDY; TAKE ERROR EXIT
2217 006062 000404 BR 99$; CHECK ERROR EXIT
2218 006064 032765 040000 000012 3$: BIT @ERR,DS(R5)
2219 006072 001401 BEQ 100$;
2220 006074 000262 99$: SEV
2221 006076 000207 100$: RTS PC
2222 ;SUBROUTINE TO REWIND A UNIT (DRIVE/SLAVE COMBINATION)
2223 ;CALL MOVB DRIVE @, @DRVNUM
2224 ; MOVB SLAVE @, @SLVNUM
2225 ; JSR PC,.REWIND
2226 ;SUBROUTINE RETURNS TO CALLER WITH SELECTED SLAVE AT 'BOT' & 'V' SET IF
2227 ;AN ERROR OCCURS.
2228
2229 006100 004737 005736 .REWIND:JSR PC,RHINIT; INITIALIZE CONTROLLER
2230 006104 004337 006316 JSR R3,IMCMD; GO TO IM COMMAND SUBROUTINE
2231 006110 000000 .WORD 0; BUS ADDRESS (NOT USED)
2232 006112 000000 .WORD 0; WORD COUNT (NOT USED)
2233 006114 000000 .WORD 0; FRAME COUNT (NOT USED)
2234 006116 000006 .WORD RWD; REWIND COMMAND
2235 006120 005215 INC (R5); SET 'GO' BIT
2236 006122 032765 000002 000012 1$: BIT @BOT,DS(R5); BRANCH IF 'BOT' SET
2237 006130 001005 BNE 2$
2238 006132 032765 040000 000012 BIT @ERR,DS(R5); CHECK ERROR BIT
2239 006140 001006 BNE 99$; BRANCH IF ERROR BIT SET
2240 006142 000767 BR 1$
2241
2242 006144 032765 020000 000012 2$: BIT @PIP,DS(R5); WAIT FOR TAPE MOTION TO STOP
2243 006152 001374 BNE 2$
2244 006154 000401 BR 100$
2245 006156 000262 99$: SEV
2246 006160 000207 100$: RTS PC
2247
2248 ;SUBROUTINE TO WRITE 256 WORD RECORD
2249 ;CALL: JSR PC,WRITE
2250
2251 006162 004337 006316 WRITE: JSR R3,IMCMD; GO TO IM COMMAND SUBROUTINE

```

```

2252 006166 017572          .WORD  WTBUF          ;BUS ADDRESS
2253 006170 177600          .WORD  WRDCNT         ;WORD COUNT
2254 006172 177400          .WORD  FRMCNT         ;FRAME COUNT
2255 006174 000060          .WORD  WFWD           ;WRITE FORWARD COMMAND
2256 006176 000207          RTS      PC
2257
2258          ;SUBROUTINE TO READ A 256. WORD RECORD.
2259          ;CALL: JSR      PC,READ
2260
2261 006200 004337 006316  READ: JSR      R3,@TMCMD
2262 006204 017572          .WORD  RDBUF          ;ADDRESS OF READ BUFFER
2263 006206 177600          .WORD  WRDCNT         ;2'S COMPLEMENT OF WORD COUNT
2264 006210 177400          .WORD  FRMCNT         ;2'S COMPLEMENT OF FRAME COUNT
2265 006212 000070          .WORD  RDFWD          ;READ FORWARD COMMAND
2266 006214 000207          RTS      PC
2267
2268          ;SUBROUTINE TO INITIATE READ REVERSE COMMAND
2269          ;CALL: JSR      PC,REVRD
2270
2271 006216 004337 006316  ;REVRD: JSR      R3,TMCMD
2272 006222 020172          .WORD  RDBUF+256.    ;ADDRESS OF READ REVERSE BUFFER
2273 006224 177600          .WORD  WRDCNT         ;2'S COMPLEMENT OF WORD COUNT
2274 006226 177400          .WORD  FRMCNT         ;2'S COMPLEMENT OF FRAME COUNT
2275 006230 000076          .WORD  RDREV          ;READ REVERSE COMMAND
2276 006232 000207          RTS      PC
2277
2278          ;SUBROUTINE TO SPACE FORWARD 1 RECORD
2279 006234 012765 177777 000006  FWDSPC: MOV     @-1,FC(R5) ;LOAD RECORD COUNT
2280 006242 012715 000031          MOV     @SPCFWD+1,(R5) ;LOAD COMMAND
2281 006246 004737 005774          JSR     PC,WAITRDY    ;WAIT FOR READY
2282 006252 000207          RTS      PC          ;RETURN
2283
2284          ;SUBROUTINE TO WRITE A RECORD AND BACK SPACE OVER THE RECORD.
2285 006254 004737 006162  WRT.BK: JSR     PC,WRITE ;WRITE THE RECORD
2286 006260 005215          INC     (R5)          ;SET GO BIT
2287 006262 004737 005774          JSR     PC,WAITRDY
2288 006266 102412          BVS    2$
2289 006270 012765 177777 000006  MOV     @-1,FC(R5) ;LOAD RECORD COUNT
2290 006276 012715 000033          MOV     @SPCREV+1,(R5) ;LOAD COMMAND
2291 006302 004737 005774          JSR     PC,WAITRDY
2292 006306 102402          BVS    2$
2293 006310 004737 005502  1$: JSR     PC,DELAY    ;WAIT FOR TAPE MOTION TO STOP
2294 006314 000207          2$: RTS      PC
2295
2296          ;SUBROUTINE TO LOAD A COMMAND
2297          ;CALL: JSR      R3,TMCMD
2298          ;          .WORD  BUS ADDRESS
2299          ;          .WORD  WORD COUNT (2'S COMPLEMENT)
2300          ;          .WORD  FRAME COUNT (2'S COMPLEMENT)
2301          ;          .WORD  COMMAND
2302
2303 006316 012365 000004  TMCMD: MOV     (R3)+,BA(R5) ;LOAD BUS ADDRESS
2304 006322 012365 000002          MOV     (R3)+,WC(R5) ;LOAD WORD COUNT
2305 006326 012365 000006          MOV     (R3)+,FC(R5) ;LOAD FRAME COUNT
2306 006332 012315          MOV     (R3)+,(R5) ;LOAD COMMAND
2307 006334 000203          RTS      R3          ;RETURN

```

D5

2308					
2309					
2310					;SUBROUTINE TO PRINT SERIAL NUMBER
2311					;JSR PC,SNPT
2312	006336	016503	000030	SNPT:	MOV SN(R5),R3
2313	006342	012701	001152		MOV #0DIGITS,R1
2314	006346	000303			SWAB R3
2315	006350	006003			ROR R3
2316	006352	006003			ROR R3
2317	006354	006003			ROR R3
2318	006356	006003			ROR R3
2319	006360	042703	177760		BIC #177760,R3 ;GET FIRST DIGIT
2320	006364	052703	000260		BIS #260,R3
2321	006370	110321			MOVB R3,(R1); ;FILL FIRST DIGIT
2322	006372	016503	000030		MOV SN(R5),R3
2323	006376	000303			SWAB R3
2324	006400	042703	177760		BIC #177760,R3
2325	006404	052703	000260		BIS #260,R3
2326	006410	110321			MOVB R3,(R1); ;GET SECOND DIGIT
2327	006412	016503	000030		MOV SN(R5),R3
2328	006416	006003			ROR R3
2329	006420	006003			ROR R3
2330	006422	006003			ROR R3
2331	006424	006003			ROR R3
2332	006426	042703	177760		BIC #177760,R3
2333	006432	052703	000260		BIS #260,R3
2334	006436	110321			MOVB R3,(R1); ;GET THIRD DIGIT
2335	006440	016503	000030		MOV SN(R5),R3
2336	006444	042703	177760		BIC #177760,R3
2337	006450	052703	000260		BIS #260,R3
2338	006454	110321			MOVB R3,(R1); ;GET FOURTH DIGIT
2339	006456	105011			CLAB (R1)
2340	006460	000004	001152		TYPE,0DIGITS ;TYPE SERIAL NUMBER
2341	006464	000207			RTS PC ;RETURN
2342					

F5

```
2344 .SBTTL PROGRAM INITIALIZATION
2345 006466 012706 000600 INIT: MOV @STKPTR,SP ;SET STACK PTR
2346 006472 005037 001272 CLR @INBUF
2347
2348 006476 013746 00000E MOV @6,-(SP) ;SAVE VECTORS
2349 006502 013746 000004 MOV @4,-(SP)
2350 006506 012737 006526 000004 MOV @61,@4 ;SET UP FOR TIMEOUT
2351 006514 022777 177777 172256 CMP @-1,@SWR ;REFERENCE HARDWARE SWITCH REGISTER
2352 006522 001402 BEQ 60
2353 006524 000404 BR 62
2354 006526 022626 61: CMP (SP),-(SP) ;ADJUST STACK
2355 006530 012737 000176 001000 60: MOV @SWREG,SWR ;POINT TO SOFTWARE SWITCH REG
2356 006536 012637 000004 62: MOV (SP),@4 ;RESTORE VECTORS
2357 006542 012637 000006 MOV (SP),@6
2358 006546 105037 001131 CLRB @PRGFLG ;CLEAR PROGRAM FLAG
2359 006552 105037 001135 CLRB @ASFLG ;CLEAR ASK FLAG
2360 006556 105037 001134 CLRB @PSCNT ;SET PASS COUNT = 0
2361 006562 005027 CLR (PC) ;CLEAR CHAIN INDICATOR
(1) 006564 000000 CHNFLG: .WORD 0 ;CHAIN MODE INDICATOR
(1) ;1/0 = CHAIN/NOT CHAIN MODE
(1) ;BRANCH IF IN DUMP MODE
(1) 006566 005737 000042 TST @42
(1) 006572 001407 BEQ 50
(1) 006574 012737 000176 001000 MOV @SWREG,SWR ;INVOKE SOFTWARE SWR
(1) 006602 005237 006564 INC CHNFLG ;SET CHNFLG = CHAIN MODE
(1) 006606 000137 006612 JMP 1 ;GO TO CHAIN ADDRESS
(1) 006612 50:
2362 006612 122737 000006 000041 1: CMPB @6,@41 ;BRANCH IF NOT LOADED VIA TMDP
2363 006620 001003 BNE 2
2364 006622 006004 014536 TYPE,I.REM ;ADVISE USER TO REMOVE TMDP
2365 006626 000000 HALT
2366 006630 000004 014326 2: TYPE,M.NAM ;TYPE TITLE
2367 006634 005737 006564 TST CHNFLG ;SEE IF CHAIN MODE
2368 006640 001025 BNE 5 ;IF SO: BR
2369 006642 105037 014326 CLRB M.NAM ;DO NOT TYPE TITLE ON RESTART
2370 006646 000004 014611 TYPE,I.REG ;ASK USER TO TYPE CONT BASE ADRS
2371 006652 013702 001010 MOV @TMBASE,R2 ;GET CURRENT CONT BASE ADDRESS
2372 006656 004737 003224 JSR PC,TYPECT ;AND TYPE IT
2373 006662 000004 001416 TYPE,SPACE
2374 006666 004737 004070 JSR PC,INPUT ;GET USER INPUT
2375 006672 122737 000015 001272 CMPB @CR,@INBUF ;DO NOT CHANGE CURRENT VALUE
2376 006700 001405 BEQ 5 ;IF USER TYPES <CR>
2377 006702 004737 003654 4: JSR PC,CNVTA0 ;CONVERT ASCII TO OCTAL
2378 006706 013737 001122 001010 MOV @OCTALO,@TMBASE ;SET NEW ADDRESS
2379 006714 013705 001010 5: MOV @TMBASE,R5
2380
2381 ;ROUTINE TO CHECK IF CONTROLLER (RM11) IS AVAILAABLE
2382 006720 000261 SEC ;SET 'C' IN PSW
2383 006722 005715 TST (R5) ;BRANCH IF CONTROLLER AVAIL
2384 006724 103003 BCC 6
2385 006726 000004 015104 TYPE,E.NCON
2386 006732 000655 BR INIT
2387 006734 012737 004372 000004 6: MOV @ERRTRP,@ERRVEC ;SET ERROR TRAP VECTOR
```

```

2389 ;ROUTINE TO GET TM03 DRIVES USER DESIRES TO TEST
2390 006742 105037 001127 DRIVES: CLR B @ERFLG ;CLEAR ERROR FLAG
2391 006746 012701 001162 MOV @DRVTBL,R1 ;MARK ALL DRIVES AS NOT TO
2392 006752 012700 000004 MOV @4,R0 ;BE TESTED. A '0' INDICATES
2393 006756 005021 1$: CLR (R1); ;THAT A DRIVE IS NOT TO BE
2394 006760 005300 DEC R0 ;TESTED
2395 006762 001375 BNE 1$
2396 006764 005737 006564 TST CHNFLG ;BRANCH IF IN CHAIN MODE
2397 006770 001014 BNE 2$
2398 006772 000004 014656 TYPE,I,DRVS
2399 006776 004737 004070 JSR PC,,INPUT ;GET USER INPUT
2400 007002 012700 001272 MOV @INBUF,R0
2401 007006 122710 000101 CMPB @'A',(R0) ;IF USER RESPONDS WITH 'A' OR
2402 007012 001403 BEQ 2$ ;<CR> THEN ALL AVAILABLE DRIVES
2403 007014 122710 000015 CMPB @CR,(R0) ;ARE TO BE TESTED
2404 007020 001013 BNE 4$
2405 007022 110637 001131 2$: MOV B SP,PRGFLG ;SET FLAG TO IND ALL DRIVES
2406 007026 012701 001162 MOV @DRVTBL,R1 ;MARK ALL DRIVES TO BE TESTED
2407 007032 012700 000004 MOV @4,R0 ;A '-1' INDICATES THAT A DRIVE
2408 007036 012721 177777 3$: MOV @-1,(R1); ;IS TO BE TESTED
2409 007042 005300 DEC R0
2410 007044 001374 BNE 3$
2411 007046 000417 BR CHKDRV ;GO CHECK DRIVE AVAILABILITY
2412
2413 ;GET USER SELECTED DRIVES AND MARK EACH DRIVE SELECTED TO BE TESTED
2414 007050 122710 000015 4$: CMPB @CR,(R0)
2415 007054 001414 BEQ CHKDRV
2416 007056 121027 000054 CMPB (R0),@' , ;CHECK IF 'COMMA'
2417 007062 001001 BNE 5$
2418 007064 105720 TSTB (R0); ;STEP PTR PAST 'COMMA'
2419 007066 112001 5$: MOV B (R0),R1
2420 007070 042701 177770 BIC @177770,R1
2421 007074 112761 177777 001162 MOV B @-1,DRVTBL(R1)
2422 007102 000240 NOP
2423 007104 000761 BR 4$
2424
2425 ;ASCERTAIN THAT DRIVES (TM03'S) SPECIFIED ARE AVAILABLE
2426 007106 005000 CHKDRV: CLR R0 ;A (0) IN DRVTBL(R0) INDICATES
2427 007110 105760 001162 1$: TSTB DRVTBL(R0) ;THE DRIVE IS NOT TO BE TESTED
2428 007114 001005 BNE 3$ ;A '1' INDICATES TO BE TESTED
2429 007116 005200 2$: INC R0
2430 007120 122700 000010 CMPB @8,,R0
2431 007124 001371 BNE 1$
2432 007126 000424 BR 5$
2433 007130 110037 001004 3$: MOV B R0,@DRVNUM ;GET DRIVE #
2434 007134 004737 005660 JSR PC,@DRVAVA ;AND CHECK IF AVAILABLE
2435 007140 102366 BVC 2$ ;'V' BIT SET INDICATES NOT AVAIL
2436 007142 105737 001131 TSTB @PRGFLG ;DO NOT TYPE NOT AVAILABLE
2437 007146 001011 BNE 4$ ;MESSAGE IF ALL SELECTED
2438 007150 000004 015151 TYPE,E,NDRV
2439 007154 116037 001140 015203 MOV B DIGTAB(R0),@E.NAVA ;SET DRIVE # IN MESSAGE
2440 007162 000004 015203 TYPE,E,NAVA
2441 007166 110637 001127 MOV B SP,@ERFLG ;SET 'ERROR' FLAG
2442 007172 105060 001162 4$: CLR B DRVTBL(R0) ;MARK DRIVE UNAVAILABLE
2443 007176 000747 BR 2$ ;CHECK NEXT DRIVE
2444 007200 105737 001127 5$: TSTB @ERFLG ;GO GET SLAVES IF NO ERROR

```

```

2445 007204 001256          BNE      DRIVES          ;ELSE ASK USER TO RETYPF DRIVES
2446
2447          ;ROUTINE TO GET SLAVES (TE16/TU77'S) USER DESIRES TO TEST
2448 007206 105037 001127  SLAVES: CLR      @PRGFLG          ;CLEAR ERROR INDICATOR
2449 007212 012701 001172  MOV      @SLVTBL,R1          ;MARK ALL SLAVES (64.) AS NOT
2450 007216 012700 000040  MOV      @32.,R0            ;TO BE TESTED.A 0 INDICATES THAT
2451 007222 005021          1$: CLR      (R1).           ;A DRIVE'S SLAVE IS NOT TO BE
2452 007224 005300          DEC      R0                  ;TESTED
2453 007226 001375          BNE      1$
2454 007230 012701 001172  MOV      @SLVTBL,R1          ;R1 POINTS TO DRIVE'S SLAVE
2455 007234 105760 001162  2$: TSTB  DRVTBL(R0)         ;BRANCH IF DRIVE IS TO BE TESTED
2456 007240 001007          BNE      4$                  ;E IS AVAILABLE
2457 007242 062701 000010  3$: ADD      @8.,R1           ;STEP SLAVE PTR TO NEXT DRIVE'S
2458 007246 005200          INC      R0                  ;SLAVES AND INCREMENT DRIVE #
2459 007250 122700 000010  CMPB    @8.,R0              ;CHECK ALL DRIVES
2460 007254 001367          BNE      2$                  ;AND WHEN ALL DRIVES CHECKED
2461 007256 000457          BR       CHKSLV             ;GO CHECK SLAVE AVAILABILITY
2462
2463 007260 105737 001131  4$: TSTB  @PRGFLG           ;BRANCH IF USER SELECTED ALL
2464 007264 001021          BNE      5$                  ;DRIVES
2465 007266 110037 001004  MOVB    R0,DRVNUM          ;GET DRIVE #
2466 007272 116037 001140  MOVB    DIGTAB(R0),@I.DRV  ;PREPARE USER ACTION MESSAGE
2467 007300 000004 014726  TYPE,I.SLVS
2468 007304 004737 004070  JSR     PC,,INPUT          ;GET USER INPUT
2469 007310 012703 001272  MOV      @INBUF,R3          ;SET PTR TO USER INPUT
2470 007314 122713 000101  CMPB    @'A',(R3)          ;AN 'A' OR <CR> AS FIRST CHAR
2471 007320 001403          BEQ     5$                  ;INDICATES TEST ALL SLAVES
2472 007322 122713 000015  CMPB    @CR,(R3)
2473 007326 001015          BNE      7$
2474 007330 110637 001131  5$: MOVB    SP,@PRGFLG       ;SET 'ALL' INDICATOR
2475 007334 012701 001172  MOV      @SLVTBL,R1          ;MARK ALL SLAVES FOR ALL
2476 007340 012700 000040  MOV      @32.,R0            ;DRIVES AS TO BE TESTED
2477 007344 012721 177777  6$: MOV      @-1,(R1).
2478 007350 005300          DEC      R0
2479 007352 001374          BNE      6$
2480 007354 105737 001131  TSTB    @PRGFLG           ;BRANCH IF ALL WAS SELECTED
2481 007360 001016          BNE      CHKSLV
2482
2483 007362 122713 000015  7$: CMPB    @CR,(R3)          ;GET USER SELECTED SLAVES FOR
2484 007366 001725          BEQ     3$                  ;DRIVE
2485 007370 121327 000054  CMPB    (R3),@'.           ;STEP PTR PAST 'COMMA
2486 007374 001001          BNE      8$
2487 007376 105723          TSTB    (R3).
2488 007400 112304          8$: MOVB    (R3),R4           ;AND MARK SELECED SLAVE
2489 007402 042704 177770  BIC     @177770,R4         ;AS TO BE TESTED
2490 007406 060104          ADD     R1,R4
2491 007410 112714 177777  MOVB    @-1,(R4)
2492 007414 000762          BR      7$
2493
2494          ;ASCERTAIN THAT SLAVES (TE16/TU77'S) SELETED ARE AVAILABLE
2495 007416 005000  CHKSLV: CLR      R0          ;R0 WILL CONTAIN THE DRIVE #
2496 007420 005001          CLR     R1                  ;AND R1 THE SLAVE #
2497 007422 012702 001172  MOV      @SLVTBL,R2          ;SET PTR TO SLAVE TABLE
2498 007426 105760 001162  1$: TSTB  DRVTBL(R0)         ;BRANCH IF DRIVE SELECTED
2499 007432 001020          BNE      3$                  ;E AVAILABLE FOR TEST
2500 007434 005200          2$: INC     R0                ;INCREMENT DRIVE #

```

```

2501 007436 105760 001161          TSTB  <DRVTBL-1>(R0)      ;++C WAS PREVIOUS DRIVE SELECTED
2502 007442 001003          BNE   9$                  ;++C BRANCH IF AVAIL.
2503 007444 062702 000010          ADD   #8.,R2              ;++C ADJUST SLAVE POINTER
2504 007450 000405          BR    10$                ;
2505 007452 105737 001131          9$:  TSTB  @PRGFLG          ;++C WAS ALL SELECTED
2506 007456 001002          BNE   10$                ;
2507 007460 062702 000010          ADD   #8.,R2              ;++C ADJUST SLAVE POINTER
2508 007464 022700 000010          10$: CMP   #8.,R0          ;SLAVES. BRANCH TO 1$ IF NOT ALL
2509 007470 001356          BNE   1$                  ;DRIVES CHECKED OTHERWISE EXIT
2510 007472 000437          BR    8$                  ;
2511                                ;
2512 007474 005001          3$:  CLR   R1              ;SET SLAVE # 0
2513 007476 105712          4$:  TSTB  (R2)            ;BRANCH IF DRIVE'S SLAVE IS SEL-
2514 007500 001006          BNE   6$                  ;ECTED FOR TEST
2515 007502 005201          5$:  INC   R1              ;INCREMENT SLAVE #
2516 007504 005202          INC   R2                  ;STEP PTR TO NEXT SLAVE
2517 007506 022701 000010          CMP   #8.,R1              ;GO TO 4$ IF ALL SLAVES NOT
2518 007512 001371          BNE   4$                  ;CHECKED
2519 007514 000747          BR    2$                  ;OTHERWISE GO TO 2$ ABOVE
2520                                ;
2521 007516 110037 001004          6$:  MOVB  R0,@DRVNUM       ;PASS DRIVE & SLAVE #
2522 007522 110137 001005          MOVB  R1,@SLVNUM         ;
2523 007526 004737 005706          JSR   PC,@SLVAVA         ;AND CHECK IF AVAILABLE
2524 007532 102363          BVC   5$                  ;'V' SET INDICATES ERROR
2525 007534 105737 001131          TSTB  @PRGFLG           ;DO NOT TYPE ERROR MSG IF ALL
2526 007540 001012          BNE   7$                  ;SLAVES SELECTED
2527 007542 116037 001140 015173  MOVB  DIGTAB(R0),@E.DRV   ;ICATES ERROR. PREPARE ERROR
2528 007550 116137 001140 015203  MOVB  DIGTAB(R1),@E.NAVA ;MESSAGE
2529 007556 000004 015165          TYPE,E.NSLV              ;
2530 007562 110637 001127          MOVB  SP,@ERFLG          ;SET ERROR INDICATOR
2531 007566 105012          7$:  CLRB  (R2)            ;CLEAR SLAVE TABLE ENTRY
2532 007570 000744          BR    5$                  ;GET NEXT SLAVE
2533                                ;
2534 007572 105737 001127          8$:  TSTB  @ERFLG          ;BRANCH IF ERROR
2535 007576 001203          BNE   SLAVES              ;ASK USER TO RETYPE SLAVES
2536 007600 012737 004372 000004  100$: MOV   @ERRTRP,@ERRVEC ;
2537                                ;
2538                                ;SCAN DRIVE AND SLAVE TABLE FOR DRIVE/SLAVE COMBINATION TO TEST.
2539                                ;RESTART ADDRESS--PROGRAM STARTS HERE WHEN START ADDRESS = 210 AND
2540                                ;AFTER ALL SELECTED DRIVE/SLAVE COMBINATIONS HAVE BEEN TESTED.
2541 007606 012706 000600          RSTRT: MOV   #600,SP      ;SET STACK PTR
2542 007612 105037 001004          CLRB  @DRVNUM            ;SET DRIVE AND SLAVE # 0
2543 007616 105037 001005          CLRB  @SLVNUM            ;
2544 007622 012737 001172 001006  MOV   @SLVTBL,@SLVPTR    ;SET PTR TO SLAVE TABLE
2545 007630 105037 001132          CLRB  @UNTFND           ;CLEAR 'UNIT FOUND' IND.
2546                                ;
2547                                ;PROGRAM RESTARTS HERE AFTER A DRIVE/SLAVE HAS BEEN TESTED.
2548 007634 113700 001004          BEGIN: MOVB  @DRVNUM,R0   ;GET DRIVE #
2549 007640 113701 001005          MOVB  @SLVNUM,R1        ;AND SLAVE #
2550 007644 013702 001006          MOV   @SLVPTR,R2        ;GET SLAVE PTR
2551 007650 122737 000006 000041  CMPB  #6,@#41           ;BRANCH IF LOADED VIA TMOP
2552 007656 001001          BNE   1$                  ;
2553 007660 105012          CLRB  (R2)              ;SET DRIVE #0,SLAVE #0 NOT TO
2554                                ;BE TESTED.
2555 007662 105760 001162          1$:  TSTB  DRVTBL(R0)     ;BRANCH IF DRIVE AVAIL TO TEST
2556 007666 001011          BNE   3$                  ;

```

CZTEEE0 TM03 TE16/TU77 DFT
CZTEEE.P11 06 APR 84 11:05

MACY11 30(1046) 06 APR-84 11:07 PAGE 25-15
PROGRAM INITIALIZATION

SEG 0060

```

2557 007670 005001          CLR      R1          ;CLEAR SLAVE #
2558 007672 062702 000010  ADD      #8.,R2      ;AND STEP PTR TO NEXT DRIVE S
2559 007676 005200          INC      R0          ;SLAVES AND INCREMENT DRIVE #
2560 007700 022700 000010  CMP      #8.,R0      ;EXIT TEST IF ALL DRIVES
2561 007704 001366          BNE     1$          ;CHECKED OTHERWISE CONTINUE
2562 007706 000137 013624  JMP      @#END       ;SCAN FOR NEXT 'UNIT'
2563
2564 007712 105712          3$:     TSTB     (R2)    ;BRANCH IF SLAVE ON DRIVE IS
2565 007714 001007          BNE     4$          ;AVAILABLE THERWISE STEP
2566 007716 005202          INC      R2          ;PTR TO NEXT SLAVE
2567 007720 005201          INC      R1          ;INCREMENT SLAVE #
2568 007722 122701 000010  CMPB    #8.,R1      ;UNTIL ALL SLAVES CHECKED
2569 007726 001371          BNE     3$          ;WHEN ALL SLAVES CHECKED
2570 007730 005001          CLR      R1          ;SET SLAVE # 0
2571 007732 000761          BR      2$          ;AND CONTINUE SCAN
2572
2573 007734 110637 001132          4$:     MOVB     SP,@#UNTFND ;INDICATE THAT A 'UNIT' IS FOUND
2574 007740 110037 001004          MOVB    R0,@#DRVNUM  ;SET DRIVE #
2575 007744 110137 001005          MOVB    R1,@#SLVNUM  ;SET SLAVE #
2576 007750 010237 001006          MOV     R2,@#SLVPTR  ;SAVE SLAVE PTR
2577
2578 007754 105737 001135          5$:     TSTB     @#ASFLG
2579 007760 001034          BNE     7$
2580 007762 112737 000001 001135  MOVB    #1,ASFLG
2581 007770 005737 006564          TST     @#CNFLG      ;BRANCH IF IN CHAIN MODE
2582 007774 001026          BNE     7$
2583 007776 105037 001130          CLRB    @#SKEWFLG    ;..B CLEAR SKEW (SPEED) TESTS SELECTED FLAG
2584 010002 000004 015012          TYPE,I.SPD          ;ASK USER IF HE WANTS TO RUN SPEED TESTS
2585 010006 004737 004070          JSR     PC, .INPUT   ;GET USER INPUT
2586 010012 012703 001272          MOV     @#INBUF,R3   ;GET REPLY
2587 010016 122713 000015          CMPB    @#CR,(R3)    ;DO NOT DO SKEW TESTS IF <CR> IS FIRST
2588 010022 001405          BEQ     6$
2589 010024 132713 000001          BITB    #1,(R3)     ;BRANCH IF 'N'
2590 010030 001402          BEQ     6$
2591 010032 111337 001130          MOVB    (R3),@#SKEWFLG ;SET INDICATOR
2592 010036 022737 000176 001000  6$:     CMP      @#SWREG,SWR ;BRANCH IF SOFTWARE SWR
2593 010044 001002          BNE     7$          ;NOT INVOKED
2594 010046 004737 002526          JSR     PC,GTSWR     ;GET SWITCH REGISTER
2595 010052
2596
2597          7$:     ;ROUTINE TO ASCERTAIN SLAVE TYPE AND LOAD APPROPRIATE SPECIFICATION
2598          ;TABLES (TE16 OR TU77) INTO STIMTBL AND GTIMTBL.
2598 010052 013705 001010          MOV     @#TMBASE,R5  ;GET BASE ADDRESS OF REGISTERS
2599 010056 004737 005736          JSR     PC,RHINIT    ;INIT DRIVE/SLAVE
2600 010062 012704 000240          MOV     @#ENDTBL-STTBL,R4 ;GET TABLE LENGTH
2601 010066 012703 002124          MOV     @#STIMTBL,R3 ;AND STARTING ADDRESS OF TABLE
2602 010072 012702 001424          MOV     @#TE16TBL,R2 ;GET ADDRESS OF TE16 TIME TABLE
2603 010076 105037 001136          CLRB    @#TE16      ;SET FLAG = TE16
2604 010102 032765 000004 000026  BIT     #4,DT(R5)    ;BRANCH IF TU77?
2605 010110 001012          BNE     9$
2606 010112 012737 000070 001012  MOV     #56.,OSCTIM  ;SET US/TICK = 56
2607 010120 012737 000022 001014  MOV     #18.,GAPDEL  ;SET 18 TICKS/MS
2608 010126 112223          8$:     MOVB    (R2),.(R3). ;MOVE TE16 TIME AND GAP TABLES
2609 010130 005304          DEC     R4          ;INTO STIMTBL & GTIMTBL
2610 010132 001375          BNE     8$
2611 010134 000416          BR      11$
2612

```

J5

CZTEEE0 TM03 TE16/TU77 DFI
CZTEEE.P11 06-APR 84 11:05

MACY11 30(1046) 06 APR 84 11:07 PAGE 25 16
PROGRAM INITIALIZATION

SEQ 0061

```

2613 010136 012702 001664          9$:  MOV    #TU77TTBL,R2    ;GET ADDRESS OF TU77 TIME TABLE
2614 010142 112737 000001 001136  MOVB   #1,@#TE16      ;SET FLAG = TU77
2615 010150 012737 000120 001012  MOV    #80.,OSCTIM    ;SET US/TICK = 80
2616 010156 012737 000003 001014  MOV    #3,GAPDEL     ;SET 3 TICKS PER .25MS
2617 010164 112223          10$: MOVB   (R2),.(R3),    ;MOVE TU77 TIME AND GAP TABLES
2618 010166 005304          DEC    R4              ;INTO STIMTBL & GTIMTBL
2619 010170 001375          BNE    10$
2620 010172          11$:
2621
2622          ;NOTE THIS IS NOT A TEST
2623          ;INITIALIZE PROGRAM FLAGS
2624 010172 105037 001126  TST000: CLRB   @#TSTNUM    ;SET TEST # 0
2625 010176 013705 001010  MOV    @#TMBASE,R5    ;SET ADDRESS OF FIRST TM03 REG
2626 010202 010500          MOV    R5,R0
2627 010204 062700 000006  ADD    #FC,R0        ;R0 CONTAINS ADDRESS OF FC REG
2628 010210 010501          MOV    R5,R1
2629 010212 062701 000012  ADD    #DS,R1        ;R1 CONTAINS ADDRESS OF DS REG
2630 010216 012703 005212  MOV    #TIMER,R3     ;SET JUMP ADDRESS TO TIMER
2631 010222 105037 001125  CLRB   @#ITCNT       ;CLEAR SUBTEST ITERATION COUNT
2632 010226 052737 000100 177560  BIS    #100,@#TKS    ;SET KEYBOARD IE BIT
2633
2634          ;GET USER RUN PROCEDURE
2635          ;IF SW? <05::00> IS NOT 0 THEN RUN TEST IN SWR<05::00>
2636          ;OTHERWISE RUN ALL TESTS
2637
2638 010234 004737 006100          JSR    PC,.REWIND     ;REWIND SLAVE
2639 010240 102504          BVS    99$           ;BRANCH IF ERROR ON REWIND
2640 010242 105737 001130          TSTB   @#SKEWFLG    ;...B BRANCH IF SWEW (SPEED) TEST SELECTED
2641 010246 001006          BNE    10$
2642 010250 004737 006162          JSR    PC,WRITE      ;WRITE A RECORD
2643 010254 005215          INC    (R5)         ;SET 'GO' BIT
2644 010256 004737 005774          JSR    PC,WAITRDY   ;WAIT FOR READY
2645 010262 102473          BVS    99$
2646 010264 117702 170510  10$: MOVB   @SWR,R2        ;GET SWITCHES
2647 010270 042702 177740  BIC    #177740,R2    ;CLEAR ALL BUT TEST #
2648 010274 001421          BEQ    2$           ;E BRANCH IF TEST 0 WAS SELECTED
2649 010276 000004 015331          TYPE,E.HDR         ;TYPE TEST #
2650 010302 004737 003224          JSR    PC,TYPOCT
2651 010306 006302          ASL    R2            ;FORM INDEX VALUE
2652 010310 016237 002364 010320  MOV    NAMPTR(R2),1$ ;GET ADDRESS OF TEST'S NAME
2653 010316 000004          TYPE           ;AND TYPE IT
2654 010320 000000          1$: .WORD 0
2655 010322 000004 001402          TYPE,CRLF
2656 010326 016237 002444 001002  MOV    TSTTBL(R2),@#SCPADR ;SET SCOPE ADDRESS FOR TEST
2657 010334 000172 002444          JMP    @TSTTBL(R2)  ;GO TO TEST
2658 010340 032777 002000 170432  2$: BIT    #SW10,@SWR   ;BRANCH IF TIMES NOT TO BE TYPED
2659 010346 001034          BNE    5$
2660 010350 000004 016245          TYPE,L.HDR1
2661 010354 113702 001004          MOVB   DRVNUM,R2    ;GET DRIVE #
2662 010360 113704 001005          MOVB   SLVNUM,R4    ;AND SLAVE #
2663 010364 116237 001140 016427  MOVB   DIGTAB(R2),@#L.DRV ;SET DRIVE AND SLAVE #'S
2664 010372 116437 001140 016441  MOVB   DIGTAB(R4),@#L.SLV ;INTO L.HDR2 MESSAGE
2665 010400 000004 016362          TYPE,L.HDR2
2666 010404 105737 001136          TSTB   @#TE16      ;BRANCH IF NOT TE16
2667 010410 001003          BNE    3$
2667 010412 000004 016445          TYPE,L.TE16       ;TYPE TE16

```

CZTEEE0 TM03 TE16/TU7 DFT
CZTEEE.P11 06-APR 84 11:05

MACY11 30(1046) 06-APR-84 11:07 PAGE 25 17
PROGRAM INITIALIZATION

SEQ 0062

2668	010416	000402			OR	4\$		
2669	010420	000004	016453	3\$:	TYPE,L.TU77		;TYPE 'TU77	
2670	010424	000004	016461	4\$:	TYPE,L.SER		;TYPE 'SERIAL #'	
2671	010430	004737	006336		JSR	PC,SNPT		;PRINT SLAVE SERIAL #
2672	010434	000004	016473		TYPE,L.HDR3			
2673	010440	105737	001130	5\$:	TSTB	@SKEWFLG		;B BRANCH IF SPEED TESTS NOT
2674	010444	001405			BEQ	100\$;SELECTED
2675	010446	000137	013742		JMP	@SKEWTST		;GO DO SPEED TESTS
2676	010452	104400		99\$:	HLT			
2677	010454	000137	010172		JMP	TST000		;LOOP TEST AS LONG AS ERROR PERSISTS
2678	010460	012737	010466	001002	100\$:	MOV	@TST001,@SCPADR	;SET SCOPE LOOP ADDRESS
2679								

```
2681 .SBTTL START OF TESTS
2682 ;TEST 001 - WRITE FROM BOT
2683 ;THIS TEST WILL MEASURE ACCELERATION DELAY REQUIRED TO
2684 ;MOVE THE TAPE APPROXIMATELY SEVEN (7) INCHES FORWARD
2685 ;FROM DEAD STOP BEFORE STARTING TO TRANSFER DATA.
2686
2687 ;THIS TEST MEASURES TIME FROM 'GO'=1 TO 'ACCL'=0.
2688 010466 112737 000001 001126 TST001: MOVB #1,@TSTNUM ;SET TEST #
2689 010474 012702 010520 MOV #1,R2 ;SET RETURN PC FROM TIMER
2690 010500 004737 006100 JSR PC,REWIND ;REWIND SLAVE
(1) 010504 102420 BVS 99$ ;BRANCH IF ERROR ON REWIND
2691 010506 004737 006162 JSR PC,WRITE ;GO SETUP WRITE COMMAND
2692 010512 004737 005122 JSR PC,TIMON ;TURN TIMER ON
2693 010516 005215 INC (R5) ;SET 'GO' BIT
2694
2695 010520 005765 000032 1$: TST TC(R5) ;BRANCH WHEN 'ACCL'=0
2696 010524 100002 BPL 2$
2697 010526 000163 005212 JMP TIMER(R3) ;GO TO TIMER & RETURN VIA R2
2698
2699 010532 004737 005774 2$: JSR PC,WAITRDY ;WAIT FOR COMMAND TO FINISH
2700 010536 102403 BVS 99$ ;BRANCH IF ERROR
2701 010540 004737 005250 JSR PC,TIMOK ;GO CHECK TIME
2702 010544 000401 BR 100$
2703 010546 104400 99$: HLT
2704 010550 104000 100$: SCOPE
2705
2706 ;TEST 002 - WRITE START
2707 ;THIS TST MEASURES TIME FROM 'GO'=1 TO 'ACCL'=0.
2708 010552 112737 000002 001126 TST002: MOVB #2,@TSTNUM ;SET TEST # 2
2709 010560 004737 006162 JSR PC,WRITE ;INITIATE WRITE COMMAND
2710 010564 012702 010576 MOV #1,R2 ;SET RETURN PC FROM TIMER
2711 010570 004737 005122 JSR PC,TIMON
2712 010574 005215 INC (R5) ;SET 'GO' BIT
2713
2714 010576 005765 000032 1$: TST TC(R5) ;BRANCH WHEN 'ACCL'=0
2715 010602 100002 BPL 2$
2716 010604 000163 005212 JMP TIMER(R3) ;GO TO TIMER & RETURN VIA R2
2717
2718 010610 004737 005774 2$: JSR PC,WAITRDY ;WAIT FOR READY
2719 010614 102403 BVS 99$ ;BRANCH IF ERROR
2720 010616 004737 005250 JSR PC,TIMOK ;GO CHECK TIME RECORDED
2721 010622 000401 BR 100$ ;EXIT VIA SCOPE
2722
2723 010624 104400 99$: HLT ;REPORT ERROR
2724 010626 104000 100$: SCOPE
2725
2726 ;TEST 003 - WRITE SHUTDOWN
2727 ;THIS TEST MEASURES TIME FROM 'FC REG'=0 TO 'SWDN'=1.
2728 010630 112737 000003 001126 TST003: MOVB #3,@TSTNUM ;SET TEST#3
2729 010636 004737 006162 JSR PC,WRITE ;INITIATE WRITE COMMAND
2730 010642 005215 INC (R5) ;SET 'GO' BIT
2731
2732 010644 005710 1$: TST (R0) ;BRANCH WHEN WRITING FINISHED
2733 010646 001404 BEQ 2$
2734 010650 032711 040000 BIT #ERR,(R1) ;MONITOR ERROR BIT
2735 010654 001017 BNE 99$
```

CZTEEE0 TM03 TE16/TU77 DFT
CZTEEE.P11 06-APR-84 11:05MACY11 30(1046) 06 APR-84 11:07 PAGE 25 19
START OF TESTS

SEQ 0064

```

2736 010656 000772          BR      1$
2737
2738 010660          2$:
(1) 010660 004737 005122      JSR      PC,TIMON      ;TURN TIMER ON
2739 010664 010702          MOV      PC,R2        ;LOAD RETURN PC FROM TIMER
2740 010666 032711 000020      3$:      BIT      @SDWN,(R1) ;BRANCH WHEN DS <SDWN> SETS
2741 010672 001002          BNE     4$
2742 010674 000163 005212      JMP      TIMER(R3)    ;GO TO TIMER & RETURN VIA R2
2743
2744 010700 004737 005774      4$:      JSR      PC,WAITRDY   ;WAIT FOR READY
2745 010704 102403          BVS     99$
2746 010706 004737 005250      JSR      PC,TIMOK     ;GO CHECK TIME RECORDED
2747 010712 000401          BR      100$
2748 010714 104400      99$:     HLT
2749 010716 104000      100$:   SCOPE        ;REPORT ERROR
2750
2751          ;TEST 004 - WRITE SETTLEDOWN
2752          ;THIS TEST MEASURES TIME FROM 'SWDN' =1 TO 'SWDN' =0.
2753 010720 112737 000004 001126 TST004: MOVB   @4,@TSTNUM
2754 010726 004737 006162          JSR      PC,WRITE
2755 010732 005215          INC      (R5)        ;SET 'GO' BIT
2756
2757 010734 005710          1$:      TST      (R0)        ;BRANCH WHEN WRITING FINISHED
2758 010736 001404          BEQ     2$
2759 010740 032711 040000      BIT      @ERR,(R1)   ;CHECK ERROR BIT
2760 010744 001026          BNE     99$
2761 010746 000772          BR      1$
2762
2763 010750 032711 000020      2$:      BIT      @SDWN,(R1) ;WAIT FOR ASSERTION OF 'SDWN'
2764 010754 001004          BNE     3$
2765 010756 032711 040000      BIT      @ERR,(R1)   ;MONITOR ERROR BIT
2766 010762 001017          BNE     99$
2767 010764 000771          BR      2$
2768
2769 010766          3$:
(1) 010766 004737 005122      JSR      PC,TIMON     ;TURN TIMER ON
2770 010772 010702          MOV      PC,R2        ;SET RETURN PC FROM TIMER
2771 010774 032711 000020      BIT      @SDWN,(R1)  ;BRANCH WHEN SWDN CLEARS
2772 011000 001402          BEQ     5$
2773 011002 000163 005212      JMP      TIMER(R3)    ;GO TO TIMER & RETURN VIA R2
2774
2775 011006 004737 005774      5$:      JSR      PC,WAITRDY   ;WAIT FOR READY
2776 011012 102403          BVS     99$
2777 011014 004737 005250      JSR      PC,TIMOK     ;GO CHECK TIME RECORDED
2778 011020 000401          BR      100$
2779
2780 011022 104400      99$:     HLT
2781 011024 104000      100$:   SCOPE        ;REPORT ERROR
2782
2783          ;TEST 005 - READ FROM BOT
2784          ;THIS TEST MEASURES TIME FROM 'GO' =1 TO 'ACCL' =0.
2785 011026 112737 000005 001126 TST005: MOVB   @5,@TSTNUM
2786 011034 004737 006100          JSR      PC,.REWIND  ;REWIND SLAVE
(1) 011040 102422          BVS     99$          ;BRANCH IF ERROR ON REWIND
2787 011042 004737 006200          JSR      PC,READ
2788 011046 012702 011060          MOV      @1$,R2      ;SET RETURN PC FROM TIMER

```

```

2789 011052 004737 005122      JSR    PC,TIMON      ;TURN TIMER ON
2790 011056 005215              INC    (R5)          ;SET 'GO' BIT
2791
2792 011060 005765 000032      1$:   TST    TC(R5)      ;BRANCH WHEN 'ACCL' RESETS
2793 011064 100002              BPL    2$            ;
2794 011066 000163 005212      JMP    TIMER(R3)    ;GO TO TIMER & RETURN VIA R2
2795
2796 011072 004737 005774      2$:   JSR    PC,WAITRDY  ;WAIT FOR READY
2797 011076 102403              BVS    99$          ;BRANCH IF ERROR
2798 011100 004737 005250      JSR    PC,TIMOK     ;CHECK RECORDED TIME
2799 011104 000401              BR     100$         ;
2800
2801 011106 104400              99$:   HLT
2802 011100 104000              100$:  SCOPE
2803
2804      ;TEST 006 - READ START
2805      ;THIS TEST MEASURES TIME FROM 'GO'=1 TO 'ACCL'=0.
2806 011112 112737 000006 001126 TST006: MOVB    #6,#TSTNUM ;SET TEST #6
2807 011120 004737 006254      JSR    PC,WRT.BK    ;WRITE A RECORD & BACK SPACE
2808 011124 102422              BVS    99$          ;
2809 011126 004737 006200      JSR    PC,READ
2810 011132 012702 011144      MOV    #1,R2        ;SET RETURN PC FROM TIMER
2811 011136 004737 005122      JSR    PC,TIMON     ;TURN TIMER ON
2812 011142 005215              INC    (R5)          ;SET 'GO' BIT
2813
2814 011144 005765 000032      1$:   TST    TC(R5)      ;BRANCH WHEN 'ACCL' RESETS
2815 011150 100002              BPL    2$            ;
2816 011152 000163 005212      JMP    TIMER(R3)    ;GO TO TIMER & RETURN VIA R2
2817
2818 011156 004737 005774      2$:   JSR    PC,WAITRDY  ;WAIT FOR READY
2819 011162 102403              BVS    99$          ;BRANCH IF ERROR
2820 011164 004737 005250      JSR    PC,TIMOK     ;CHECK RECORDED TIME
2821 011170 000401              BR     100$         ;
2822
2823 011172 104400              99$:   HLT
2824 011174 104000              100$:  SCOPE
2825
2826      ;TEST 007 - READ SHUTDOWN
2827      ;THIS TEST MEASURES TIME FROM 'FC REG'=FRAME COUNT TO 'SDWN'=1.
2828 011176 112737 000007 001126 TST007: MOVB    #7,#TSTNUM ;SET TEST #7
2829 011204 004737 006254      JSR    PC,WRT.BK    ;WRITE A RECORD & BACK SPACE
2830 011210 102430              BVS    99$          ;BRANCH IF ERROR
2831 011212 004737 006200      JSR    PC,READ
2832 011216 005215              INC    (R5)          ;SET 'GO' BIT
2833
2834 011220 022710 000400      1$:   CMP    #FRMCNT,(R0)  ;WAIT FOR FRAME COUNT TO
2835 011224 001404              BEQ    2$            ;= # OF FRAMES WRITTEN
2836 011226 032711 040000      BIT    #ERR,(R1)    ;MONITOR ERROR BIT
2837 011232 001017              BNE    99$          ;
2838 011234 000771              BR     1$            ;
2839
2840 011236              2$:   JSR    PC,TIMON     ;TURN TIMER ON
(1) 011236 004737 005122      MOV    PC,R2        ;SET RETURN PC FROM TIMER
2841 011242 010702              BIT    #SDWN,(R1)   ;BRANCH WHEN SDWN SETS
2842 011244 032711 000020      BNE    3$            ;
2843 011250 001002              3$:

```

```

2844 011252 000163 005212          JMP      TIMER(R3)          ;GO TO TIMER & RETURN VIA R2
2845
2846 011256 004737 005774          3$:     JSR      PC,WAITRDY
2847 011262 102403                    BVS     99$
2848 011264 004737 005250          JSR     PC,TIMOK
2849 011270 000401                    BR      100$
2850
2851 011272 104400                    99$:    HLT
2852 011274 104000                    100$:   SCOPE          ;REPORT ERROR
2853
2854
2855          ;TEST 010 - READ SETTLEDOWN
2856 011274 11273 000010 001126  TST010: MOVB   #10,#TSTNUM      ;SWDN' =1 TO 'SWDN' =0.
2857 011304 012702 011362          MOV     #4,R2              ;SET TEST #10
2858 011310 004737 006254          JSR     PC,WRT,BK         ;SET RETURN PC FROM TIMER
2859 011314 102436                    BVS     99$               ;WRITE A RECORD & BACK SPACE
2860 011316 004737 006200          JSR     PC,READ
2861 011322 005215                    INC     (R5)              ;SET 'GO' BIT
2862
2863 011324 105711                    1$:     TSTB   (R1)          ;WAIT FOR READY
2864 011326 100404                    BMI     2$               ;BRANCH WHEN SET
2865 011330 032711 040000          BIT     @ERR,(R1)        ;CHECK ERROR BIT
2866 011334 001026                    BNE     99$
2867 011336 000772                    BR      1$
2868
2869 011340 032711 000020          2$:     BIT     @SDWN,(R1)   ;WAIT FOR ASSERTION OF 'SDWN'
2870 011344 001004                    BNE     3$
2871 011346 032711 040000          BIT     @ERR,(R1)        ;MONITOR ERROR BIT
2872 011352 001017                    BNE     99$
2873 011354 000771                    BR      2$
2874
2875 011356                    3$:
2876 011356 004737 005122          (1)    JSR     PC,TIMON      ;TURN TIMER ON
2877 011362 032765 000020 000012  4$:     BIT     @SDWN,DS(R5)   ;WAIT FOR NEGATION OF SDWN
2878 011370 001402                    BEQ     5$
2879 011372 000163 005212          JMP     TIMER(R3)        ;GO TO TIMER & RETURN VIA R2
2880 011376 004737 005774          5$:     JSR     PC,WAITRDY
2881 011402 102403                    BVS     99$
2882 011404 004737 005250          JSR     PC,TIMOK
2883 011410 000401                    BR      100$
2884
2885 011412 104400                    99$:    HLT
2886 011414 104000                    100$:   SCOPE
2887
2888
2889          ;TEST 011-READ REVERSE START
2890          ;THIS TEST MEASURES TIME FROM 'GO' =1 TO 'ACCL' =0.
2891 011416 112737 000011 001126  TST011: MOVB   #11,#TSTNUM
2892 011424 012702 011462          MOV     #1,R2              ;SET RETURN PC FROM TIMER
2893 011430 004737 006162          JSR     PC,WRITE         ;WRITE A RECORD
2894 011434 005215                    INC     (R5)              ;SET 'GO' BIT
2895 011436 004737 005774          JSR     PC,WAITRDY
2896 011442 102422                    BVS     99$
2897 011444 004737 005502          JSR     PC,DELAY         ;WAIT FOR TAPE MOTION TO STOP
2898 011450 004737 006216          JSR     PC,REVRD

```

06

2899	011454	004737	005122		JSR	PC,TIMON		;TURN TIMER ON
2900	011460	005215			INC	(R5)		;SET 'GO' BIT
2901								
2902	011462	005765	000032	1#:	TST	TC(R5)		;BRANCH WHEN 'ACCL' = 0
2903	011466	100002			BPL	2#		
2904	011470	000163	005212		JMP	TIMER(R3)		;GO TO TIMER & RETURN VIA R2
2905								
2906	011474	004737	005774	2#:	JSR	PC,WAITRDY		
2907	011500	102403			BVS	99#		;BRANCH IF ERROR
2908	011502	004737	005250		JSR	PC,TIMOK		
2909	011506	000401			BR	100#		
2910								
2911	011510	104400		99#:	HLT			
2912	011512	104000		100#:	SCOPE			
2913								
2914								;TEST 012-READ REVERSE SHUTDOWN
2915								;THIS TEST MEASURES TIME FROM 'FC REG' = FRAME COUNT TO 'SDWN' =1.
2916	011514	112737	000012	001126	TST012:	MOV	#12,#01STNUM	
2917	011522	012702	011572			MOV	#31,R2	
2918	011526	004737	006162			JSR	PC,WRITE	;SET RETURN PC FROM TIMER
2919	011532	005215				INC	(R5)	;WRITE A RECORD
2920	011534	004737	005774			JSR	PC,WAITRDY	;SET 'GO' BIT
2921	011540	102427				BVS	99#	
2922	011542	004737	006216			JSR	PC,REVRD	
2923	011546	005215				INC	(R5)	;SET 'GO' BIT
2924								
2925	011550	022710	000400	1#:	CMP	#-FRMCNT,(R0)		;BRANCH WHEN FRAME COUNT
2926	011554	001404			BEQ	2#		;= # OF RECORD WRITTEN
2927	011556	032711	040000		BIT	#ERR,(R1)		;MONITOR ERROR BIT IN 'DS' REG
2928	011562	001016			BNE	99#		
2929	011564	000771			BR	1#		
2930								
2931	011566			2#:				
(1)	011566	004737	005122		JSR	PC,TIMON		;TURN TIMER ON
2932	011572	032711	000020	3#:	BIT	#SDWN,(R1)		;BRANCH WHEN SDWN SETS
2933	011576	001002			BNE	4#		
2934	011600	000163	005212		JMP	TIMER(R3)		;GO TO TIMER & RETURN VIA R2
2935								
2936	011604	004737	005774	4#:	JSR	PC,WAITRDY		;WAIT FOR READY
2937	011610	102403			BVS	99#		
2938	011612	004737	005250		JSR	PC,TIMOK		
2939	011616	000401			BR	100#		
2940								
2941	011620	104400		99#:	HLT			
2942	011622	104000		100#:	SCOPE			
2943								
2944								;TEST 013-READ REVERSE SETTLEDOWN
2945								;THIS TEST MEASURES TIME FROM 'SDWN' =1 TO 'SDWN' =0.
2946	011624	112737	000013	001126	TST013:	MOV	#13,#01STNUM	
2947	011632	012702	011716			MOV	#41,R2	
2948	011636	004737	006162			JSR	PC,WRITE	;SET RETURN PC FROM TIMER
2949	011642	005215				INC	(R5)	;WRITE A RECORD
2950	011644	004737	005774			JSR	PC,WAITRDY	;SET 'GO' BIT
2951	011650	102435				BVS	99#	
2952	011652	004737	006216			JSR	PC,REVRD	
2953	011656	005215				INC	(R5)	;SET 'GO' BIT

2954						
2955	011660	105711		1\$:	TSTB (R1)	;BRANCH WHEN
2956	011662	100404			BMI 2\$;READY SETS
2957	011664	032711	040000		BIT 0ERR,(R1)	
2958	011670	001025			BNE 99\$	
2959	011672	000772			BR 1\$	
2960						
2961	011674	032711	000020	2\$:	BIT 0SDWN,(R1)	
2962	011700	001004			BNE 3\$	
2963	011702	032711	040000		BIT 0ERR,(R1)	
2964	011706	001016			BNE 99\$	
2965	011710	000771			BR 2\$	
2966						
2967	011712			3\$:		
(1)	011712	004737	005122		JSR PC,TIMON	;TURN TIMER ON
2968	011716	032711	000020	4\$:	BIT 0SDWN,(R1)	;BRANCH WHEN SWDN = 0
2969	011722	001402			BEQ 5\$	
2970	011724	000163	005212		JMP TIMER(R3)	;GO TO TIMER & RETURN VIA R2
2971						
2972	011730	004737	005774	5\$:	JSR PC,WAITRDY	;WAIT FOR READY
2973	011734	102403			BVS 99\$	
2974	011736	004737	005250		JSR PC,TIMOK	
2975	011742	000401			BR 100\$	
2976						
2977	011744	104400		99\$:	HLT	
2978	011746	104000		100\$:	SCOPE	
2979						
2980					;REWIND DRIVE	
2981	011750			A:		
(1)	011750	004737	006100		JSR PC,.REWIND	;REWIND SLAVE
(1)	011754	102401			BVS 99\$;BRANCH IF ERROR ON REWIND
2982	011756	102002			BVC 100\$	
2983	011760	104400		99\$:	HLT	
2984	011762	000772			BR A	
2985	011764			100\$:		
2986						
2987					;TEST 014-TURN AROUND DELAY (FORWARD-REVERSE)	
2988					;THIS TEST MEASURES TIME FROM 'GO' = 1 (READ REVERSE) TO ACCL' = 0	
2989	011764	112737	000014	001126	TST014: MOVB #14,00TSTNUM	
2990	011772	012702	012024		MOV #2\$,R2	;SET RETURN PC FROM TIMER
2991	011776	004737	006162		JSR PC,WRITE	;WRITE A RECORD
2992	012002	005215			INC (R5)	;SET 'GO' BIT
2993	012004	004737	005774		JSR PC,WAITRDY	
2994	012010	102420			BVS 99\$	
2995						
2996	012012	004737	006216	1\$:	JSR PC,REVRD	;READ THE RECORD (REVERSE)
2997	012016	004737	005122		JSR PC,TIMON	;TURN TIMER ON
2998	012022	005215			INC (R5)	;SET 'GO' BIT
2999						
3000	012024	005765	000032	2\$:	TST TC(R5)	;WAIT FOR 'ACCL' = 0
3001	012030	100002			BPL 3\$	
3002	012032	000163	005212		JMP TIMER(R3)	;GO TO TIMER & RETURN VIA R2
3003						
3004	012036	004737	005774	3\$:	JSR PC,WAITRDY	
3005	012042	102403			BVS 99\$	
3006	012044	004737	005250		JSR PC,TIMOK	

```

3007 012050 000401 BR 100$
3008
3009 012052 104400 99$: HLT
3010 012054 104000 100$: SCOPE
3011
3012 ;TEST 015- TURN AROUND DELAY (REVERSE FORWARD)
3013 ;THIS TEST MEASURES TIME FROM 'GO'=1 (READ) TO 'ACCL'=0.
3014 012056 112737 000015 001126 TST015: MOVB #15,#TSTNUM
3015 012064 012702 012132 MOV #2#,R2 ;SET RETURN PC FROM TIMER
3016 012070 004737 006162 JSR PC,WRITE ;WRITE A RECORD
3017 012074 005215 INC (R5) ;SET 'GO' BIT
3018 012076 004737 005774 JSR PC,WAITRDY ;WAIT FOR READY
3019 012102 102426 BVS 99$
3020 012104 004737 006216 JSR PC,REVRD ;READ A RECORD IN THE
3021 012110 005215 INC (R5) ;SET 'GO' BIT
3022
3023 012112 004737 005774 JSR PC,WAITRDY
3024 012116 102420 BVS 99$
3025
3026 012120 004737 006200 1$: JSR PC,READ ;READ RECORD FORWARD
3027 012124 004737 005122 JSR PC,TIMON ;TURN TIMER ON
3028 012130 005215 INC (R5) ;SET 'GO' BIT
3029
3030 012132 005765 000032 2$: TST TC(R5) ;WAIT FOR 'ACCL' = 0
3031 012136 100002 BPL 3$
3032 012140 000163 005212 JMP TIMER(R3) ;GO TO TIMER & RETURN VIA R2
3033
3034 012144 004737 005774 3$: JSR PC,WAITRDY
3035 012150 102403 BVS 99$
3036 012152 004737 005250 JSR PC,TIMOK
3037 012156 000401 BR 100$
3038
3039 012160 104400 99$: HLT
3040 012162 104000 100$: SCOPE
3041
3042 ;TEST 016-GAP SIZE (STOP HALF)
3043 012164 112737 000016 001126 TST016: MOVB #16,#TSTNUM
3044 012172 012702 012230 MOV #1#,R2 ;SET RETURN PC FROM TIMER
3045 012176 004737 006162 JSR PC,WRITE ;WRITE A RECORD
3046 012202 005215 INC (R5) ;SET 'GO' BIT
3047 012204 004737 005774 JSR PC,WAITRDY
3048 012210 102421 BVS 99$
3049 012212 004737 005502 JSR PC,DELAY ;DELAY 350 MS
3050 012216 004737 006216 JSR PC,REVRD ;READ REVERSE RECORD
3051 012222 004737 005122 JSR PC,TIMON ;TURN TIMER ON
3052 012226 005215 INC (R5) ;SET 'GO' BIT
3053
3054 012230 005710 1$: TST (R0) ;WAIT FOR FRAME COUNT > 0
3055 012232 001002 BNE 2$
3056 012234 000163 005212 JMP TIMER(R3) ;GO TO TIMER & RETURN VIA R2
3057
3058 012240 004737 005774 2$: JSR PC,WAITRDY ;WAIT FOR READY BIT TO SET
3059 012244 102403 BVS 99$
3060 012246 004737 005250 JSR PC,TIMOK ;CHECK TIME
3061 012252 000401 BR 100$
3062

```

16

```

3063 012254 104400          99$: HLT
3064 012256 104000        100$: SCOPE
3065
3066          ;TEST 017-GAP SIZE (START HALF)
3067 012260 112737 000017 001126 TST017: MOVB  #17,0@TSTNUM
3068 012266 012702 012340      MOV  #1$,R2          ;SET RETURN PC FROM TIMER
3069 012272 004737 006162      JSR  PC,WRITE      ;WRITE A RECORD
3070 012276 005215          INC  (R5)          ;SET 'GO' BIT
3071 012300 004737 005774      JSR  PC,WAITRDY   ;WAIT FOR READY
3072 012304 102427          BVS  99$
3073 012306 004737 006216      JSR  PC,REVRD     ;READ REVERSE THE RECORD
3074 012312 005215          INC  (R5)          ;SET 'GO' BIT
3075 012314 004737 005774      JSR  PC,WAITRDY   ;WAIT FOR READY
3076 012320 102421          BVS  99$          ;BRANCH ON ERROR
3077 012322 004737 005502      JSR  PC,DELAY     ;WAIT FOR TAPF MOTION TO STOP
3078 012326 004737 006200      JSR  PC,READ      ;READ RECORD
3079 012332 004737 005122      JSR  PC,TIMON     ;TURN TIMER ON
3080 012336 005215          INC  (R5)          ;SET 'GO' BIT
3081
3082 012340 005710          1$: TST  (R0)      ;WAIT FOR FRAME COUNT > 0
3083 012342 001002          BNE  2$
3084 012344 000163 005212      JMP  TIMER(R3)   ;GO TO TIMER & RETURN VIA R2
3085
3086 012350 004737 005774          2$: JSR  PC,WAITRDY   ;WAIT FOR READY
3087 012354 102403          BVS  99$
3088 012356 004737 005250      JSR  PC,TIMOK     ;CHECK TIME
3089 012362 000401          BR   100$
3090
3091 012364 104400          99$: HLT
3092 012366 104000        100$: SCOPE
3093
3094          ;TEST 020- GAP SIZE (INTERRECORD)
3095          ;THIS TEST MEASURES TIME FROM 'GO' -1 TO 'FC REG' >0.
3096 012370 112737 000020 001126 TST020: MOVB  #20,0@TSTNUM
3097 012376 012702 012460      MOV  #1$,R2          ;SET RETURN PC FROM TIMER
3098 012402 004737 006162      JSR  PC,WRITE      ;WRITE A RECORD
3099 012406 005215          INC  (R5)          ;SET 'GO' BIT
3100 012410 004737 005774      JSR  PC,WAITRDY   ;WAIT FOR READY
3101 012414 102433          BVS  99$
3102 012416 004737 006162      JSR  PC,WRITE      ;WRITE SECOND RECORD
3103 012422 005215          INC  (R5)          ;SET 'GO' BIT
3104 012424 004737 005774      JSR  PC,WAITRDY   ;WAIT FOR READY
3105 012430 102425          BVS  99$
3106 012432 004737 006216      JSR  PC,REVRD     ;READ REVERSE SECOND RECORD
3107 012436 005215          INC  (R5)          ;SET 'GO' BIT
3108 012440 004737 005774      JSR  PC,WAITRDY   ;WAIT FOR READY
3109 012444 102417          BVS  99$
3110 012446 004737 006216      JSR  PC,REVRD     ;READ REVERSE FIRST RECORD
3111 012452 004737 005122      JSR  PC,TIMON     ;TURN TIMER ON
3112 012456 005215          INC  (R5)          ;SET 'GO' BIT
3113
3114 012460 005710          1$: TST  (R0)      ;WAIT FOR FRAME COUNT > 0
3115 012462 001002          BNE  2$
3116 012464 000163 005212      JMP  TIMER(R3)   ;GO TO TIMER & RETURN VIA R2
3117
3118 012470 004737 005774          2$: JSR  PC,WAITRDY   ;WAIT FOR READY

```

```

3119 012474 102403          BVS 99%
3120 012476 004737 005250 JSR PC,TIMOK
3121 012502 000401          BR 100%
3122
3123 012504 104400          99%: HLT
3124 012506 104000          100%: SCOPE
3125
3126 ;TEST 021 GAP CONSISTANCY
3127 ;THIS TEST MEASURES TIME FROM 'GO'=1 TO 'FC REG' > 0.
3128 ;THE TEST REWINDS THE TAPE,WRITES 17 RECORDS WITH A DELAY FROM 1-16 MS
3129 ;BETWEEN EACH WRITE COMMAND. AFTER THE 17. RECORDS ARE WRITTEN THE
3130 ;PROGRAM READ REVERSES 16 RECORDS. AT THIS POINT THE TAPE IS STOPPED BE
3131 ;TWEEN THE FIRST AND SECOND RECORD. A READ COMMAND IS EXECUTED TO READ
3132 ;THE 16 RECORDS WITH THE TIME BETEWEN GO=1 TO FC > 0 STORED IN 'GAPTBL'
3133 ;FOR EACH RECORD READ. AFTER 16 RECORDS HAVE BEEN READ THE TIME IS VER
3134 ;IFIED FOR EACH READ. AFTER ALL RECORD TIMES ARE VERIFIED THEY ARE AVER
3135 ;AGED AND PLACED IN THE 'ATIMTBL' (BY SCOPE). THE ABOVE PROCESS IS RE
3136 ;PEATED FOR EACH ITERATION.
3137
3138 012510 112737 000021 001126 TST021: MOVB #21,#R2STNUM
3139 012516 012702 012654          MOV #4,R2 ;SET RETURN PC FROM TIMER
3140 012522 004737 006100          JSR PC,.REWIND ;REWIND SLAVE
(1) 012526 102530          BVS 99% ;BRANCH IF ERROR ON REWIND
3141 012530 005037 001120          CLR DELTIM ;CLEAR VARIABLE DELAY TIME
3142 012534 012700 000021          MOV #17,R0 ;SET # OF RECORDS TO WRITE
3143 012540 004737 006162          1%: JSR PC,WRITE ;WRITE 17. RECORDS
3144 012544 005215          INC (R5) ;SET 'GO' BIT
3145 012546 004737 005774          JSR PC,WAITRDY ;WAIT FOR READY
3146 012552 102516          BVS 99%
3147 012554 004737 005532          JSR PC,DELAYV ;DELAY BEFORE WRITING NEXT REC.
3148 012560 063737 001014 001120 ADD GAPDEL,DELTIM ;ADD 1MS TO DELAY TIME
3149 012566 005300          DEC R0 ;DECREMENT RECORDS WRITTEN COUN.
3150 012570 001363          BNE 1%
3151
3152 012572 012700 000021          MOV #17,R0 ;SET # OF RECS. TO REVERSE READ
3153 012576 004737 006216          2%: JSR PC,REVRD ;REVERSE READ 17. RECORDS
3154 012602 005215          INC (R5) ;SET 'GO' BIT
3155 012604 004737 005774          JSR PC,WAITRDY ;WAIT FOR READY
3156 012610 102477          BVS 99%
3157 012612 005300          DEC R0 ;DECREMENT RECORD COUNT
3158 012614 001370          BNE 2%
3159
3160 012616 012700 000020          MOV #16,R0 ;SET # OF RECORDS TO READ
3161 012622 012701 001060          MOV #GAPTBL,R1 ;SET PTR TO GAP TABLE FOR TEST
3162 012626 004737 006200          JSR PC,READ ;READ A RECORD
3163 012632 005215          INC (R5) ;SET 'GO' BIT
3164
3165 012634 004737 005774          3%: JSR PC,WAITRDY ;WAIT FOR READY
3166 012640 102463          BVS 99%
3167 012642 004737 006200          JSR PC,READ ;READ NEXT RECORD
3168 012646 004737 005122          JSR PC,TIMON ;TURN TIMER ON
3169 012652 005215          INC (R5) ;SET 'GO' BIT
3170
3171 012654 005765 000006          4%: TST FC(R5) ;WAIT FOR FRAME COUNT > 0
3172 012660 001002          BNE 5%
3173 012662 000163 005212          JMP TIMER(R3) ;GO TO TIMER & RETURN VIA R3

```

```

3174
3175 012666 004737 005774      5$: JSR    PC, WAITRDY      ;WAIT FOR READY
3176 012672 102446             BVS    99$
3177 012674 010421             MOV    R4, (R1)+         ;STORE TIME IN GAP TBL
3178 012676 005300             DEC    R0                ;DECREMENT # OF RECORDS READ
3179 012700 001355             BNE    3$
3180
3181 012702 105037 001124      CLR    #GAP              ;SET GAP # 0
3182 012706 012700 000020      MOV    #16., R0
3183 012712 012701 001060      MOV    #GAPTBL, R1
3184
3185 012716 012104             6$: MOV    (R1)+, R4        ;GET GAP TICK COUNT
3186 012720 004737 005360      JSR    PC, GAPOK         ;CHECK TIME
3187 012724 105237 001124      INCB   #GAP              ;INCREMENT GAP #
3188 012730 122737 000020 001124 CMPB   #16., #GAP        ;BRANCH IF ALL GAPS NOT CHECKED
3189 012736 001367             BNE    6$
3190
3191 012740 012700 000020      MOV    #16., R0          ;SETUP TO AVERAGE GAP SIZES
3192 012744 012701 001060      MOV    #GAPTBL, R1      ;SET PTR TO TABLE
3193 012750 005002             CLR    R2                ;CLEAR 'SUM' REGISTERS
3194 012752 005003             CLR    R3
3195 012754 062102             7$: ADD    (R1)+, R2        ;ADD ALL GAP SIZES TOGETHER
3196 012756 005503             ADC    R3
3197 012760 005300             DEC    R0
3198 012762 001374             BNE    7$
3199 012764 012700 000004      MOV    #4, R0            ;NOW DIVIDE BY 16.
3200 012770 006203             8$: ASR    R3                ;BY SHIFTING 4 PLACES RIGHT
3201 012772 006002             ROR    R2
3202 012774 005300             DEC    R0
3203 012776 001374             BNE    8$
3204 013000 010204             MOV    R2, R4            ;MOVE AVERAGED TIMES TO R4
3205 013002 004737 005250      JSR    PC, TIMOK         ;CHECK AVERAGED TIMES
3206 013006 000401             BR     100$
3207
3208 013010 104400             99$: HLT
3209 013012 104000             100$: SCOPE
3210
3211
3212
3213
3214
3215 013014 112737 000022 001126 ;TEST 022-DATA TIME (800BPI)
3216 013022 012702 013102      ;THIS TEST MEASURES THE TIME FROM FC REG > -6400 TO 'RDY' = 1.
3217 013026 004737 006100      TST02: MOVB   #022, #TSTNUM
3218 (1) 013032 102442             MOV    #3$, R2          ;SET RETURN PC FROM TIMER
3219 013034 052765 001700 000032 JSR    PC, .REWIND       ;REWIND SLAVE
3220 013042 004337 006316      BVS    99$              ;BRANCH IF ERROR ON REWIND
3221 013046 017572             BIS    #NORM11, TC(R5)  ;SET 800 BPI
3222 013050 171600             JSR    R3, TMCMD        ;WRITE 3200. WORD RECORD
3223 013052 163400             .WORD  WTBUF
3224 013054 000060             .WORD  -3200.
3225 013056 005215             .WORD  -6400.
3226 013060 022710 163400      INC    (R5)              ;SET 'GO' BIT
3227 013064 001004             1$: CMP    #6400., (R0)  ;WAIT FOR WRITING TO START
3228 013066 032711 040000      BNE    2$
3229 013066 032711 040000      BIT    #ERR, (R1)       ;MONITOR ERROR BIT

```

```

3229 013072 001022          BNE 99$
3230 013074 000771          BR 1$
3231
3232 013076          2$:
(1) 013076 004737 005122      JSR PC,TIMON          ;TURN TIMER ON
3233 013102 105711          3$: TSTB (R1)          ;BRANCH WHEN READY SETS
3234 013104 100402          BMI 4$
3235 013106 000163 005212      JMP TIMER(R3)        ;GO TO TIMER & RETURN VIA R2
3236
3237 013112 012700 000003      4$: MOV #3,R0          ;SET SHIFT COUNT
3238 013116 006204          5$: ASR R4
3239 013120 005300          DEC R0
3240 013122 001375          BNE 5$
3241 013124 004737 005774      JSR PC,WAITRDY
3242 013130 102403          BVS 99$
3243 013132 004737 005250      JSR PC,TIMOK          ;CHECK TIME
3244 013136 000401          BR 100$
3245
3246 013140 104400          99$: HLT
3247 013142 104000          100$: SCOPE
3248
3249          ;TEST 023-DATA TIME (1600BPI)
3250          ;THIS TEST MEASURES THE TIME FROM FC REG >-6400 TO 'RDY' = 1.
3251 013144 112737 000023 001126  TST023: MOVB #023,#TSTNUM
3252 013152 012702 013240          MOV #3$,R2          ;SET RETURN PC FROM TIMER
3253 013156 004737 006100          JSR PC,.REWIND      ;REWIND SLAVE
(1) 013162 102442          BVS 99$             ;BRANCH IF ERROR ON REWIND
3254 013164 042765 003700 000032      BIC #3700,TC(R5)   ;CLEAR CURRENT DENSITY
3255 013172 052765 002300 000032      BIS #PE1600,TC(R5) ;SET 1600 BPI
3256 013200 004337 006316          JSR R3,TMCMO        ;WRITE 3200. WORD RECORD
3257 013204 017572          .WORD WTBUF
3258 013206 171600          .WORD -3200.
3259 013210 163400          .WORD -6400.
3260 013212 000060          .WORD WFWO
3261 013214 005215          INC (R5)           ;SET 'GO' BIT
3262
3263 013216 022710 163400          1$: CMP #-6400.,(R0) ;BRANCH WHEN WRITING STARTS
3264 013222 001004          BNE 2$
3265 013224 032711 040000          BIT #ERR,(R1)      ;MONITOR ERROR BIT
3266 013230 001017          BNE 99$
3267 013232 000771          BR 1$
3268
3269 013234          2$:
(1) 013234 004737 005122      JSR PC,TIMON          ;TURN TIMER ON
3270 013240 105711          3$: TSTB (R1)          ;BRANCH WHEN READY SETS
3271 013242 100402          BMI 4$
3272 013244 000163 005212      JMP TIMER(R3)        ;GO TO TIMER & RETURN VIA R2
3273
3274 013250 006204          4$: ASR R4          ;DIVIDE TIME BY 4
3275 013252 006204          ASR R4
3276 013254 004737 005774      JSR PC,WAITRDY
3277 013260 102403          BVS 99$
3278 013262 004737 005250      JSR PC,TIMOK          ;CHECK TIME
3279 013266 000401          BR 100$
3280
3281 013270 104400          99$: HLT

```

```
3282 013272 104000          100$: SCOPE
3283
3284          ;TEST 024-ERASE
3285          ;THIS TEST MEASURES TIME FROM GO =1 TO 'RDY' =1.
3286 013274 112737 000024 001126 TST024: MOVB  #24,@TSTNUM
3287 013302 012702 013364          MOV  #2,@R2          ;SET RETURN PC FROM TIMER
3288 013306 004737 006100          JSR  PC,REWIND      ;REWIND SLAVE
(1) 013312 102436          BVS  99$          ;BRANCH IF ERROR ON REWIND
3289 013314 004737 005736          JSR  PC,RHINIT     ;SET NRZ
3290 013320 004737 006162          JSR  PC,WRITE     ;WRITE A RECORD
3291 013324 005215          INC  (R5)         ;SET 'GO' BIT
3292 013326 004737 005774          JSR  PC,WAITRDY
3293 013332 102426          BVS  99$
3294 013334 012737 013342 001002 MOV  #1,@SCPADR
3295 013342 004337 006316          JSR  R3,@TMCMD
3296 013346 000000          .WORD 0
3297 013350 000000          .WORD 0
3298 013352 000000          .WORD 0
3299 013354 000024          .WORD ERASE
3300 013356 004737 005122          JSR  PC,TIMON     ;TURN TIMER ON
3301 013362 005215          INC  (R5)         ;SET 'GO' BIT
3302
3303 013364 105711          2$: TSTB (R1)     ;BRANCH WHEN READY SETS
3304 013366 100402          BMI  3$
3305 013370 000163 005212          JMP  TIMER(R3)   ;GO TO TIMER & RETURN VIA R2
3306
3307 013374 004737 005774          3$: JSR  PC,WAITRDY
3308 013400 102403          BVS  99$
3309 013402 004737 005250          JSR  PC,TIMOK
3310 013406 000401          BR   100$
3311
3312 013410 104400          99$: HLT
3313 013412 104000          100$: SCOPE
3314
3315          ;TEST 025 TAPE MARK
3316          ;THIS TEST MEASURES TIME FROM 'GO =1 TO 'RDY' =1.
3317 013414 112737 000025 001126 TST025: MOVB  #25,@TSTNUM
3318 013422 012702 013464          MOV  #1,@R2          ;SET RETURN PC FROM TIMER
3319 013426 004737 006162          JSR  PC,WRITE     ;WRITE A RECORD
3320 013432 005215          INC  (R5)         ;SET 'GO' BIT
3321 013434 004737 005774          JSR  PC,WAITRDY
3322 013440 102423          BVS  99$
3323 013442 004337 006316          JSR  R3,@TMCMD
3324 013446 000000          .WORD 0
3325 013450 000000          .WORD 0
3326 013452 000000          .WORD 0
3327 013454 000026          .WORD WFMK
3328 013456 004737 005122          JSR  PC,TIMON     ;TURN TIMER ON
3329 013462 005215          INC  (R5)         ;SET 'GO' BIT
3330
3331 013464 105711          1$: TSTB (R1)     ;BRANCH WHEN READY SETS
3332 013466 100402          BMI  2$
3333 013470 000163 005212          JMP  TIMER(R3)   ;GO TO TIMER & RETURN VIA R2
3334
3335 013474 004737 005774          2$: JSR  PC,WAITRDY
3336 013500 102403          BVS  99$
```

K6

CZTEEE0 1M03 TE16/TU?? DFT
CZTEEE.P11 06 APR 84 11:05

MAC11 30(1046) 06 APR-84 11:07 PAGE 25 30
START OF TESTS

SEQ 0075

3337 013502 004737 005250
3338 013506 000401
3339
3340 013510 104400
3341 013512
(1) 013512 004737 006100
(1) 013516 102774
3342 013520 104000
3343

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

;REWIND SLAVE
;BRANCH IF ERROR ON REWIND

```

3345 013522 032777 002000 165250 FINISH: BIT @SW10,@SWR ;DO NOT SPACE PAPER
3346 013530 001011 BNE 2$ ;IF USER SELECTED NO OUTPUT
3347 013532 005737 006564 TST CHNFLG ;OR IF IN CHAIN MODE
3348 013536 001006 BNE 2$
3349 013540 012700 000012 MOV @10.,R0 ;SET LINE FEED COUNT
3350 013544 000004 001402 1$: TYPE,CRLF
3351 013550 005300 DEC R0
3352 013552 001374 BNE 1$
3353
3354
3355 013554 105237 001005 2$: INCB @@SLVNUM ;SET NEXT SLAVE #
3356 013560 005237 001006 INC @@SLVPTR ;AND ITS POINTER
3357 013564 122737 000010 001005 CMPB @8.,@@SLVNUM ;BRANCH IF LAST SLAVE (7)
3358 013572 001402 BEQ 3$
3359 013574 000137 007634 JMP @@BEGIN ;BEGIN TEST ON NEXT SLAVE
3360 013600 105037 001005 3$: CLRB @@SLVNUM ;SET SLAVE #0
3361 013604 105237 001004 INCB @@DRVNUM ;AND INCREMENT DRIVE #
3362 013610 122737 000010 001004 CMPB @8.,@@DRVNUM ;AND CHECK IF LAST DRIVE
3363 013616 001402 BEQ END
3364 013620 000137 007634 JMP @@BEGIN
3365
3366 013624 105737 001132 END: TSTB @@UNTFND ;BRANCH IF A UNIT WAS FOUND
3367 013630 001004 BNE 1$
3368 013632 000004 015236 TYPE,E.UNIT
3369 013636 000137 006466 JMP @@INIT
3370 013642 105237 001134 1$: INCB @@PSCNT ;INCREMENT PASS COUNT
3371 013646 000004 014467 TYPE,M.EOP
3372 013652 113702 001134 MOV @@PSCNT,R2 ;GET PASSCOUNT
3373 013656 004737 003224 JSR PC,TYPECT ;AND TYPE IT
3374 013662 000004 001402 TYPE,CRLF
3375 013666 013700 000042 MOV @42,R0 ;GET ACT11 RETURN ADDRESS
(1) 013672 001405 BEQ HERE ;BRANCH IF NOT ACT11
(1) 013674 000005 RESET
(1) 013676 004710 $ENDAD: JSR PC,(R0)
(1) 013700 000240 NOP
(1) 013702 000240 NOP
(1) 013704 000240 NOP
(1) 013706 000240 HERE: NOP
3376 013710 005737 006564 TST CHNFLG ;BRANCH IF CHAIN MODE
3377 013714 001004 BNE 1$
3378 013716 032777 000100 165054 BIT @SW06,@SWR ;BRANCH IF NOT CONTINOUS LGDP
3379 013724 001402 BEQ 2$
3380 013726 000137 007606 1$: JMP @@RSTRT ;RESTART
3381 013732 000000 2$: HALT
3382 013734 000005 RESET
3383 013736 000137 006466 JMP @@INIT ;RESTART

```

```

3385 ;SKEW TAPE TIMING TESTS
3386 ;THE FOLLOWING TESTS REQUIRE A SPECIALLY WRITTEN 800 BPI SKEW TAPE
3387 013742 012737 013750 001002 SKEWTST:MOV #TST026,@#SCPADR ;SET SCOPE POINTER
3388
3389 ;TEST 026- SKEW TAPE SPEED TEST-FORWARD
3390 ;THIS TEST READS 32" OF TAPE (26400.-800. * 25600. FRAMES), THEN
3391 ;DIVIDES TIME BY 32. TO GET TIME TO READ 1" (800. FRAMES) OF TAPE.
3392 013750 112737 000026 001126 TST026: MOV# #26,@#TSTNUM
3393 013756 012702 014044 MOV #2$,R2 ;SET RETURN PC FROM TIMER
3394 013762 004737 006100 JSR PC,.REWIND ;REWIND SLAVE
(1) 013766 102445 BVS 99$ ;BRANCH IF ERROR ON REWIND
3395 013770 052765 001700 000032 BIS #NORM11,TC(R5) ;SET 800 BPI
3396 013776 052765 000010 000010 BIS #BAI,CS2(R5) ;INHIBIT BUS ADDRESS INCREMENT
3397 014004 004337 006316 JSR R3,@#TMCMD ;READ 32" OF TAPE-FORWARD
3398 014010 017572 .WORD RDBUF
3399 014012 177777 .WORD -1.
3400 014014 063440 10$: .WORD 26400. ;FRAME COUNT
3401 014016 000070 .WORD RDFWD
3402 014020 005215 INC (R5) ;SET 'GO' BIT
3403
3404 014022 032765 075027 000014 1$: BIT #HRDERR,ER(R5) ;BRANCH IF ANY HARD ERROR BIT SETS
3405 014030 001024 BNE 99$
3406 014032 022710 001440 CMP #800.,(R0) ;WAIT FOR FIRST 800 FRAMES
3407 014036 101371 BHI 1$ ;TO BE READ
3408
3409 014040 004737 005122 JSR PC,TIMON ;TURN TIMER ON
3410 014044 023710 014014 2$: CMP #10$, (R0) ;WAIT FOR READING TO FINISH
3411 014050 103402 BLO 3$
3412 014052 000163 005212 JMP TIMER(R3) ;GO TO TIMER & RETURN VIA R2
3413
3414 014056 012700 000005 3$: MOV #5,R0 ;DIVIDE TIME BY 32.
3415 014062 006204 4$: ASR R4
3416 014064 005300 DEC R0
3417 014066 001375 BNE 4$
3418 014070 004737 005736 JSR PC,RHINIT ;INIT DRIVE
3419 014074 004737 005250 JSR PC,TIMOK ;CHECK TIME
3420 014100 000401 BR 100$
3421
3422 014102 104400 99$: HLT
3423 014104 104000 100$: SCOPE
3424
3425 ;TEST 027-SKEW TAPE SPEED TEST-REVERSE
3426 ;THIS TEST READS FORWARD 40" (32000. FRAMES) OF TAPE, THEN READS REVERSE
3427 ;32" (26400.-800. * 25600. FRAMES) OF TAPE. THE TIME IS THEN DIVIDED BY
3428 ;32. TO GET TIME TO READ 1" (800. FRAMES) OF TAPE.
3429 014106 112737 000027 001126 TST027: MOV# #27,@#TSTNUM
3430 014114 012702 014252 MOV #3$,R2 ;SET RETURN PC FROM TIMER
3431 014120 004737 006100 JSR PC,.REWIND ;REWIND SLAVE
(1) 014124 102471 BVS 99$ ;BRANCH IF ERROR ON REWIND
3432 014126 052765 001700 000032 BIS #NORM11,TC(R5)
3433 014134 052765 000010 000010 BIS #BAI,CS2(R5)
3434 014142 004337 006316 JSR R3,@#TMCMD ;READ FORWARD 32000. FRAMES
3435 014146 017572 .WORD RDBUF
3436 014150 177777 .WORD 1. ;WORD COUNT
3437 014152 076400 10$: .WORD 32000. ;FRAME COUNT
3438 014154 000070 .WORD RDFWD ;READ FORWARD

```

```

3439 014156 005215          INC      (R5)          ;SET 'GO' BIT
3440
3441 014160 032711 040000    1$:     BIT      @ERR,(R1)    ;BRANCH IF ERROR BITS SETS
3442 014164 001051          BNE     99$
3443 014166 023710 014152    CMP     @#10$,(R0)
3444 014172 101372          BHI     1$
3445
3446 014174 004737 005736    JSR     PC,RHINIT      ;INIT DRIVE
3447 014200 004737 075502    JSR     PC,DELA;      ;WAIT FOR TAPE MOTION TO STOP
3448 014204 052765 000010 000010  BIS     @BAI,CS2(R5)   ;INHIBIT BUS ADDRESS INCREMENT
3449 014212 004337 006316    JSR     R3,@#TMCMD     ;READ REVERSE 32" OF TAPE
3450 014216 017572          .WORD  RDBUF          ;READ BUFFER
3451 014220 177777          .WORD  -1.            ;WORD COUNT
3452 014222 063440          11$:   .WORD  26400.    ;FRAME COUNT
3453 014224 000076          .WORD  RDREV          ;READ REVERSE
3454 014226 005215          INC      (R5)          ;SET 'GO' BIT
3455
3456 014230 032765 075027 000014  2$:     BIT      @HRDERR,ER(R5) ;EXIT TEST IF ERROR BIT SETS
3457 014236 001024          BNE     99$
3458 014240 022710 001440    CMP     @#800.,(R0)   ;WAIT FOR FIRST 800 FRAMES
3459 014244 101371          BHI     2$            ;TO BE READ
3460
3461 014246 004737 005122          JSR     PC,TIMON      ;TURN TIMER ON
3462 014252 023710 014222    3$:     CMP     @#11$,(R0)  ;WAIT FOR ALL FRAMES TO BE READ
3463 014256 103402          BLO     4$
3464 014260 000163 005212    JMP     TIMER(R3)     ;GO TO TIMER & RETURN VIA R2
3465
3466 014264 012700 000005    4$:     MOV     @5,R0     ;DIVIDE TIME BY 32.
3467 014270 006204    5$:     ASR     R4
3468 014272 005300          DEC     R0
3469 014274 001375          BNE     5$
3470 014276 004737 005736    JSR     PC,RHINIT
3471 014302 004737 005250    JSR     PC,TIMOK
3472 014306 000401          BR      100$
3473
3474 014310 104400          99$:   HLT
3475 014312          100$:
(1) 014312 004737 006100    JSR     PC,.REWIND    ;REWIND SLAVE
(1) 014316 102774          BVS     99$          ;BRANCH IF ERROR ON REWIND
3476 014320 104000          SCOPE
3477
3478 014322 000137 013522    JMP     @#FINISH
3479
3480
3481

```

3483					.SBTTL PROGRAM MESSAGES
3484					OPERATOR INSTRUCTIONS
3485	014326	005015	046524	031460	M.NAM: .ASCII <CR><LF>'TM03-TE16/TU77 DRIVE FUNCTION TIMER (CZTEEE0)';..B
	014334	052055	030505	027466	
	014342	052524	033467	042040	
	014350	044522	042526	043040	
	014356	047125	052103	047511	
	014364	020116	044524	042515	
	014372	020122	041450	052132	
	014400	042505	030105	051	
3486	014405	015	052012	050131	.ASCIZ <CR><LF>'TYPE <CR> TO TERMINATE RESPONSE & ^C TO RESTART
	014412	020105	041474	037122	
	014420	052040	020117	042524	
	014426	046522	047111	052101	
	014434	020105	042522	050123	
	014442	047117	042523	023040	
	014450	057040	020103	047524	
	014456	051040	051505	040524	
	014464	052122	000		
3487	014467	015	042412	042116	M.EOP: .ASCIZ <CR><LF>'END OF PASS '
	014474	047440	020106	040520	
	014502	051523	000040		
3488	014506	005015	040510	042122	M.HSWR: .ASCIZ <CR><LF>'HARDWARE SWR IN USE'<CR><LF>
	014514	040527	042522	051440	
	014522	051127	044440	020116	
	014530	051525	006505	000012	
3489	014536	005015	042522	047515	I.REM: .ASCIZ <CR><LF>'REMOVE TM03 FROM SLAVE 0. CLEAR LOC. 40.
	014544	042526	052040	042115	
	014552	020120	051106	046517	
	014560	051440	040514	042526	
	014566	030040	020056	046103	
	014574	040505	020122	047514	
	014602	027103	032040	027060	
	014610	000			
3490	014611	015	052012	050131	I.REG: .ASCIZ <CR><LF>'TYPE FIRST ADDRESS OF CONTROLLER
	014616	020105	044506	051522	
	014624	020124	042101	051104	
	014632	051505	020123	043117	
	014640	041440	047117	051124	
	014646	046117	042514	020122	
	014654	000040			
3491	014656	054524	042520	052040	I.DRVS: .ASCIZ 'TYPE TM03 DRIVE #'S TO BE TESTED: ALL #
	014664	030115	020063	051104	
	014672	053111	020105	023443	
	014700	020123	047524	041040	
	014706	020105	042524	052123	
	014714	042105	020072	040440	
	014722	046114	000040		
3492	014726	047506	020122	046524	I.SLVS: .ASCII FOR TM03 DRIVE '
	014734	031460	042040	044522	
	014742	042526	040		
3493	014745	060	020055	054524	I.DRV: .ASCIZ '0- TYPE SLAVE #'S TO BE TESTED: ALL #
	014752	042520	051440	040514	
	014760	042526	021440	051447	
	014766	052040	020117	042522	
	014774	052040	051505	042524	

C/

CZTEEF0 TM03 TE16 TU77 DF1
CZTEEF.P11 06 APR 84 11:05

MACY11 30(1046) 06 APR 84 11:07 PAGE 25 35
PROGRAM MESSAGES

SEQ 0080

	015002	035104	040440	046114	
	015010	000040			
3494	015012	050123	042505	020104	I.SPD: .ASCIZ 'SPEED TESTS? (YES/NO): NO
	015020	042524	052123	037523	
	015026	024040	042531	027523	
	015034	047516	035051	047040	
	015042	020117	000		
3495	015045	015	042412	042116	M.EOT: .ASCIZ <CR><LF>'END OF TAPE'<CR><LF>
	015052	047440	020106	040524	
	015060	042520	005015	000	
3496					
3497					ERROR MESSAGES
3498	015065	015	052012	040522	E.TRP4: .ASCIZ <CR><LF>'TRAPPED TO 4'
	015072	050120	042105	052040	
	015100	020117	000064		
3499	015104	047516	041440	047117	E.NCON: .ASCIZ 'NO CONTROLLER AT ADDRESS SPECIFIED'<CR><LF>
	015112	051124	046117	042514	
	015120	020122	052101	040440	
	015126	042104	042522	051523	
	015134	051440	042520	044503	
	015142	044506	042105	005015	
	015150	000			
3500	015151	124	030115	020063	E.NDRV: .ASCIZ 'TM03 DRIVE '
	015156	051104	053111	020105	
	015164	000			
3501	015165	104	044522	042526	E.NSLV: .ASCII 'DRIVE '
	015172	040			
3502	015173	060	051440	040514	E.DRV: .ASCII 'O SLAVE '
	015200	042526	040		
3503	015203	060	047040	052117	E.NAVA: .ASCIZ 'O NOT AVAILABLE FOR TEST'<CR><LF>
	015210	040440	040526	046111	
	015216	041101	042514	043040	
	015224	051117	052040	051505	
	015232	006524	000012		
3504	015236	047516	052040	030115	E.UNIT: .ASCIZ 'NO TM03-TE16/TU77 UNIT FOUND TO TEST'<CR><LF>
	015244	026463	042524	033061	
	015252	052057	033525	020067	
	015260	047125	052111	043040	
	015266	052517	042116	052040	
	015274	020117	042524	052123	
	015302	005015	000		
3505	015305	123	043117	020124	E.SFT: .ASCIZ 'SOFT ERROR (DATA)'<CR><LF>
	015312	051105	047522	020122	
	015320	042050	052101	024501	
	015326	005015	000		
3506	015331	124	051505	020124	E.HDR: .ASCIZ 'TEST # '
	015336	020043	000		
3507	015341	040	042504	044526	E.HDR1: .ASCII 'DEVICE ERROR'<CR><LF>
	015346	042503	042440	051122	
	015354	051117	005015		
3508	015360	051503	004461	041527	.ASCIZ 'CS1 <HT>'WC'<HT>'BA <HT>'FC <HT>'CS2'<HT>'DS'<HT>'ER <HT>'TC <CR><LF>
	015366	041011	004501	041506	
	015374	041411	031123	042011	
	015402	004523	051105	052011	
	015410	006503	000012		
3509	015414	047440	052125	047440	E.HDR2: .ASCIZ 'OUT OF RANGE ERROR'<CR><LF>

D7

	015422	020106	040522	043516	
	015430	020105	051105	047522	
	015436	006522	000012		
3510	015442	044440	020106	044124	E.77T5: .ASCII ' IF THE M8940 IS REV J3 , L OR HIGHER THEN CHECK ACTUAL VALUE ' <CR><LF>
	015450	020105	034115	032071	
	015456	020060	051511	051040	
	015464	053105	045040	020063	
	015472	020054	020114	051117	
	015500	044040	043511	042510	
	015506	020122	044124	047105	
	015514	041440	042510	045503	
	015522	040440	052103	040525	
	015530	020114	040526	052514	
	015536	020105	005015		
3511	015542	040440	040507	047111	.ASCII ' AGAINST THIS RANGE : < 011250 009215 > ' <CR><LF>
	015550	052123	052040	044510	
	015556	020123	040522	043516	
	015564	020105	004472	036040	
	015572	030040	030461	032462	
	015600	020060	020055	030060	
	015606	031071	032461	037040	
	015614	006440	012		
3512	015617	040	043111	044440	.ASCIIZ ' IF IT FALLS INTO THIS RANGE THEN YOU HAVE NO PROBLEM.' <CR><LF><LF>
	015624	020124	040506	046114	
	015632	020123	047111	047524	
	015640	052040	044510	020123	
	015646	040522	043516	020105	
	015654	044124	047105	054440	
	015662	052517	044040	053101	
	015670	020105	047516	050040	
	015676	047522	046102	046505	
	015704	006456	005012	000	
3513	015711	040	043111	052040	E.77T16: .ASCII ' IF THE REV IS LOWER THAN M2 THEN CHECK ACTUAL VALUE ' <CR><LF>
	015716	042510	051040	053105	
	015724	044440	020123	047514	
	015732	042527	020122	044124	
	015740	047101	046440	020062	
	015746	044124	047105	041440	
	015754	042510	045503	040440	
	015762	052103	040525	020114	
	015770	040526	052514	020105	
	015776	005015			
3514	016000	040440	040507	047111	.ASCII ' AGAINST THIS RANGE : < 004500 003570 > ' <CR><LF>
	016006	052123	052040	044510	
	016014	020123	040522	043516	
	016022	020105	004472	036040	
	016030	030040	032060	030065	
	016036	020060	020055	030060	
	016044	032463	030067	037040	
	016052	006440	012		
3515	016055	040	043111	044440	.ASCIIZ ' IF IT FALLS INTO THIS RANGE THEN YOU HAVE NO PROBLEM. ' <CR><LF><LF>
	016062	020124	040506	046114	
	016070	020123	047111	047524	
	016076	052040	044510	020123	
	016104	040522	043516	020105	
	016112	044124	047105	054440	

F /

CZTEEE0 TM03-TE16/TU77 DFT
CZTEEE.P11 06-APR-84 11:05

MACY11 30(1046) 06 APR 84 11:07 PAGE 25 37
PROGRAM MESSAGES

SEQ 0082

	016120	052517	044040	053101		
	016126	020105	047516	050040		
	016134	047522	046102	0 0505		
	016142	006456	005012	000		
3516	016147	015	052012	046511	E.TIMOV:	.ASCIZ <CR><LF>'TIMER OVERFLOWED'<CR><LF>
	016154	051105	047440	042526		
	016162	043122	047514	042527		
	016170	006504	000012			
3517	016174	005015	044524	042515	E.TIMEX:	.ASCIZ <CR><LF>'TIME EXPIRED WAITING FOR RDY'<CR><LF>
	016202	042440	050130	051111		
	016210	042105	053440	044501		
	016216	044524	043516	043040		
	016224	051117	051040	054504		
	016232	005015	000			
3518	016235	040	040507	020120	E.GAP:	.ASCIZ 'GAP #'
	016242	020043	000			
3519						
3520						
3521	016245	015	025012	025052		:TIME DOCUMENT LINES
	016252	025052	025052	025052	L.HDR1:	.ASCIZ <CR><LF>'*****'
	016260	025052	025052	025052		
	016266	025052	025052	025052		
	016274	025052	025052	025052		
	016302	025052	025052	025052		
	016310	025052	025052	025052		
	016316	025052	025052	025052		
	016324	025052	025052	025052		
	016332	025052	025052	025052		
	016340	025052	025052	025052		
	016346	025052	025052	025052		
	016354	025052	006452	000012		
3522	016362	020052	046524	031460	L.HDR2:	.ASCII '* TM03 DRIVE FUNCTION TIMES DRIVE #
	016370	042040	044522	042526		
	016376	043040	047125	052103		
	016404	047511	020116	044524		
	016412	042515	026523	042040		
	016420	044522	042526	021440		
	016426	040				
3523	016427	060	051440	040514	L.DRV:	.ASCII '0 SLAVE #'
	016434	042526	021440	040		
3524	016441	060	020040	000	L.SLV:	.ASCIZ '0'
3525	016445	124	030505	020066	L.TE16:	.ASCIZ 'TE16'
	016452	000				
3526	016453	124	033525	020067	L.TU77:	.ASCIZ 'TU77'
	016460	000				
3527	016461	123	051105	040511	L.SER:	.ASCIZ 'SERIAL #'
	016466	020114	020043	000		
3528	016473	040	005015	006452	L.HDR3:	.ASCII ' '<CR><LF>' '<CR><LF>
	016500	012				
3529	016501	052	043040	047125		.ASCIZ '* FUNCTION'<HT><HT>' TIME(SPECIFICATION)'<HT>' TIME(ACTUAL)'<CR><LF>
	016506	052103	047511	004516		
	016514	052011	046511	024105		
	016522	050123	041505	043111		
	016530	041511	052101	047511		
	016536	024516	052011	046511		
	016544	024105	041501	052524		

3530	016552	046101	006451	000012	
3531	016560	040522	043516	036505	L.RNG: .ASCIZ 'RANGE*'
	016566	000074			
3532	016570	041501	052524	046101	L.ACT: .ASCIZ 'ACTUAL*'
	016576	000075			
3533					
3534					;TEST DESCRIPTOR HEADERS
3535	016600	025052	044440	044516	A.T000: .ASCIZ '** INITIALIZATION ERROR'
	016606	044524	046101	055111	
	016614	052101	047511	020116	
	016622	051105	047522	000122	
3536	016630	020052	051127	052111	A.T001: .ASCIZ '* WRITE FROM BOT' <HT>
	016636	020105	051106	046517	
	016644	041040	052117	000011	
3537	016652	020052	051127	052111	A.T002: .ASCIZ '* WRITE START' <HT><HT>
	016660	020105	052123	051101	
	016666	004524	000011		
3538	016672	020052	051127	052111	A.T003: .ASCIZ '* WRITE SHUTDOWN' <HT>
	016700	020105	044123	052125	
	016706	047504	047127	000011	
3539	016714	020052	051127	052111	A.T004: .ASCIZ '* WRITE SETTLEDOWN' <HT>
	016722	020105	042523	052124	
	016730	042514	047504	047127	
	016736	000011			
3540	016740	020052	042522	042101	A.T005: .ASCIZ '* READ FROM BOT' <HT><HT>
	016746	043040	047522	020115	
	016754	047502	004524	000011	
3541	016762	020052	042522	042101	A.T006: .ASCIZ '* READ START' <HT><HT>
	016770	051440	040524	052122	
	016776	004411	000		
3542	017001	052	051040	040505	A.T007: .ASCIZ '* READ SHUTDOWN' <HT><HT>
	017006	020104	044123	052125	
	017014	047504	047127	004411	
	017022	000			
3543	017023	052	051040	040505	A.T010: .ASCIZ '* READ SETTLEDOWN' <HT>
	017030	020104	042523	052124	
	017036	042514	047504	047127	
	017044	000011			
3544	017046	020052	042522	042101	A.T011: .ASCIZ '* READ REV START' <HT>
	017054	051040	053105	051440	
	017062	040524	052122	000011	
3545	017070	020052	042522	042101	A.T012: .ASCIZ '* READ REV SHUTDOWN' <HT>
	017076	051040	053105	051440	
	017104	052510	042124	053517	
	017112	004516	000		
3546	017115	052	051040	040505	A.T013: .ASCIZ '* READ REV SETTLEDOWN' <HT>
	017122	020104	042522	020126	
	017130	042523	052124	042514	
	017136	047504	047127	000011	
3547	017144	020052	052524	047122	A.T014: .ASCIZ '* TURN AROUND DELAY F-R' <HT>
	017152	040440	047522	047125	
	017160	020104	042504	040514	
	017166	020131	026506	004522	
	017174	000			
3548	017175	052	052040	051125	A.T015: .ASCIZ '* TURN AROUND DELAY R-F' <HT>

G/

CZTEEE0 IM03 TE16, IU77 DFT
CZTEEE.P11 06-APR 84 11:05

MACY11 30(1046) 06 APR-84 11:07 PAGE 25 39
PROGRAM MESSAGES

SEQ 0084

	017202	020116	051101	052517			
	017210	042116	042040	046105			
	017216	054501	051040	043055			
	017224	000011					
3549	017226	020052	040507	020120	A.T016:	.ASCIZ	' * GAP SIZE-STOP HALF' <HT>
	017234	044523	042532	051455			
	017242	047524	020120	040510			
	017250	043114	000011				
3550	017254	020052	040507	020120	A.T017:	.ASCIZ	' * GAP SIZE-START HALF' <HT>
	017262	044523	042532	051455			
	017270	040524	052122	044040			
	017276	046101	004506	000			
3551	017303	052	043440	050101	A.T020:	.ASCIZ	' * GAP SIZE-INTERRECORD' <HT>
	017310	051440	055111	026505			
	017316	047111	042524	051122			
	017324	041505	051117	004504			
	017332	000					
3552	017333	052	043440	050101	A.T021:	.ASCIZ	' * GAP CONSISTANCY' <HT>
	017340	041440	047117	044523			
	017346	052123	047101	054503			
	017354	000011					
3553	017356	020052	040504	040524	A.T022:	.ASCIZ	' * DATA TIME-800BPI' <HT>
	017364	052040	046511	026505			
	017372	030070	041060	044520			
	017400	000011					
3554	017402	020052	040504	040524	A.T023:	.ASCIZ	' * DATA TIME-1600BPI' <HT>
	017410	052040	046511	026505			
	017416	033061	030060	050102			
	017424	004511	000				
3555	017427	052	042440	040522	A.T024:	.ASCIZ	' * ERASE GAP TIME' <HT>
	017434	042523	043440	050101			
	017442	052040	046511	004505			
	017450	000					
3556	017451	052	053440	044522	A.T025:	.ASCIZ	' * WRITE FILE MARK' <HT>
	017456	042524	043040	046111			
	017464	020105	040515	045522			
	017472	000011					
3557	017474	020052	040524	042520	A.T026:	.ASCIZ	' * TAPE SPEED FWD' <HT>
	017502	051440	042520	042105			
	017510	043055	042127	000011			
3558	017516	020052	040524	042520	A.T027:	.ASCIZ	' * TAPE SPEED-REV' <HT>
	017524	051440	042520	042105			
	017532	051055	053105	000011			
3559							
3560	017540	005015	043536	000	L.CNTG:	.ASCIZ	<CR><LF>' ?G'
3561	017545	015	051412	051127	L.SWR:	.ASCIZ	<CR><LF>' SWR='
	017552	000075					
3562	017554	020040	042516	036527	L.NEW:	.ASCIZ	' NEW='
	017562	000040					
3563	017564	005015	006477	000012	L.QUEST:	.ASCIZ	<CR><LF>' ?' <CR><LF>
3564							
3565		017572				.EVEN	
3566		017572				RDBUF =	
3567	017572	000200				WTBUF =	
3568		000001				.BLKW	128.
						.END	

		2844	2878	2904	2934	2970	3002	3032	3056	3084	3116	3173	3235	3272
		3305	3333	3412	3464									
TIMERR	005222	2018	2020#											
TIMERO	005236	2026#												
TIMER1	005166	2011#	2015											
TIMOK	005250	2038#	2701	2720	2746	2777	2798	2820	2848	2882	2908	2938	2974	3006
		3036	3060	3088	3120	3205	3243	3278	3309	3337	3419	3471		
TIMON	005122	1988#	2107	2121	2692	2711	2738	2769	2789	2811	2840	2875	2899	2931
		2967	2997	3027	3051	3079	3111	3168	3232	3269	3300	3328	3409	3461
TKB	* 177562	1088#	1550	1786	1815									
TKISR	004216	1242	1815#											
TKS	* 177560	1088#	1550	1783	2632*									
TKVEC	* 000060	1089#	1241											
TMBASE	001010	1264#	1938	2371	2378*	2379	2598	2625						
TMCMD	006316	2230	2251	2261	2271	2303#	3219	3256	3295	3323	3397	3434	3449	
TMC51	* 172440	1092#	1264											
TMK	* 000004	1159#												
TPB	* 177566	1088#												
TPS	* 177564	1088#												
TPVEC	* 000064	1089#												
TRAPVE	* 000034	1089#												
TRE	* 040000	1132#												
TRIVEC	* 000014	1089#												
TSTNUM	001126	1275#	1766	1771	1864	1871	1876	1944	2054	2624*	2688*	2708*	2728*	2753*
		2785*	2806*	2828*	2856*	2891*	2916*	2946*	2989*	3014*	3043*	3067*	3096*	3138*
		3215*	3251*	3286*	3317*	3392*	3429*							
TSTTBL	002444	1504#	2655	2656										
TST000	010172	1504	1946	2624#	2677									
TST001	010466	1505	2678	2688#										
TST002	010552	1506	2708#											
TST003	010630	1507	2728#											
TST004	010720	1508	2753#											
TST005	011026	1509	2785#											
TST006	011112	1510	2806#											
TST007	011176	1511	2828#											
TST010	011276	1512	2856#											
TST011	011416	1513	2891#											
TST012	011514	1514	2916#											
TST013	011624	1515	2946#											
TST014	011764	1516	2989#											
TST015	012056	1517	3014#											
TST016	012164	1518	3043#											
TST017	012260	1519	3067#											
TST020	012370	1520	3096#											
TST021	012510	1521	3138#											
TST022	013014	1522	3215#											
TST023	013144	1523	3251#											
TST024	013274	1524	3286#											
TST025	013414	1525	3317#											
TST026	013750	1526	3387	3392#										
TST027	014106	1527	3429#											
TU77GT	002024	1403#												
TU77TT	001664	1370#	2613											
TYPDEC	003332	1623#	1669	1672	1676	1697	1700	1704						
TYPE	* 000004	1211#	1535	1538	1550	1607	1640	1667	1670	1673	1674	1677	1695	1698
		1701	1702	1705	1708	1769	1791	1796	1806	1808	1833	1863	1870	1875

CZTEEE0 TM03 TE16 TU77 DFT
CZTEEE.P11 06 APR 84 11:05

MACY11 30(1046) 06-APR-84 11:07 PAGE 27
CROSS REFERENCE TABLE MACRO NAMES

SEQ 0092

INPLT	10700	1539	2374	2399	2468	2585										
RESTOR	10670	1608	1641	1732	1773	1918	2062	2102								
REWIND	10730	2213	2638	2690	2786	2981	3140	3217	3253	3288	3341	3394	3431	3475		
SAVE	10640	1587	1624	1716	1741	2038	2074									
SETGO	10830	2215	2235	2286	2642	2812	2832	2861	2894	2900	2919	2923	2949	2953	2992	
	2998	3017	3021	3028	3046	3052	3070	3074	3080	3099	3103	3107	3112	3144	3154	
	3163	3169	3224	3261	3291	3301	3320	3329	3402	3439	3454					
TIMCHA	10800	2110	2124	2697	2716	2742	2773	2794	2816	2844	2878	2904	2934	2970	3002	
	3032	3056	3084	3116	3173	3235	3272	3305	3333	3412	3464					
TIMEON	10770	2121	2692	2738	2769	2789	2811	2840	2875	2899	2931	2967	2997	3027	3051	
	3079	3111	3168	3232	3269	3300	3328	3409	3461							
\$CATCH	10200	1226														
\$CHAIN	10200	2361														
\$CPREG	10200	1088														
\$CPVEC	10200	1089														
\$TYPE	10200	1550														
.\$ACT1	10200	1240														
.\$EOP	10200	3375														

. ABS. 020172 000

ERRORS DETECTED: 0

CZTEEE.CZTEEE/CRF=CZTEAE.SML/ML.CZTEEE.P11
RUN-TIME: 4 6 .7 SECONDS
RUN TIME RATIO: 17/11=1.4
CORE USED: 11K (22 PAGES)