B 1

.REM %

IDENTIFICATION

PRODUCT CODE: AC-E499B-MC

PRODUCT NAME: CZTUSB0 TM03/TU45 DR FCTN TMR

DATE CREATED: 25 MAY 1978

UPDATE INFORMATION:	DATE	AUTHOR
	29-FEB-80	VIJAY ANANDWALA

MAINTAINER: COMPUTER SPECIAL SYSTEMS

AUTHOR: CSS DIAGNOSTICS

TABLE OF CONTENTS

ABSTRACT

CHAPTER 1 REQUIREMENTS

1.1 EQUIPMENT

1.2 MEMORY STORAGE

1.3 PRELIMINARY PROGRAMS

CHAPTER 2 LOADING AND STARTING PROCEDURE

2.1 ACT11 OPERATION

CHAPTER 3 SWITCH SETTINGS

CHAPTER 4 ERRORS

4.1 ERROR TYPEOUT FORMAT (HARDWARE)

4.2 ERROR TYPEOUT FORMAT (FUNCTION OUT OF RANGE)

CHAPTER 5 SUBROUTINE ABSTRACTS

CHAPTER 6 MISCELLANEOUS

6.1 STACK POINTER

6.2 EXECUTION TIME

CHAPTER 7 PROGRAM DESCRIPTION

7.1 FUNCTION TIME DOCUMENT

7.2 TEST SEQUENCE / RELATED ADJUSTMENTS / ASSOCIATED HARDWARE

7.3 SUBTEST DESCRIPTIONS

ABSTRACT

PROGRAM CZTUSB0 MEASURES THE TIME REQUIRED AND GAP SIZES PRODUCED BY THE TM03/TU45 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.

CHAPTER 1

REQUIREMENTS

PDP-11 FAMILY CENTRAL PROCESSOR WITH 4K MEMORY WITH UP TO 64 TM03/TU45 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)

CZTUOA CONTROL LOGIC TEST(PART 1)
CZTUPA0 BASIC FUNCTION TEST

CHAPTER 2

LOADING AND STARTING PROCEDURE

2.0 LOAD & START PROCEDURE:

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

THE PROGRAM WILL THEN REQUEST THE FIRST BUS ADDRESS OF THE RHXX CONTROLLER, TM03 DRIVES TO BE TESTED, TU45 SLAVES TO BE TESTED, AND IF SPEED TESTS ARE TO BE RUN. IN ADDITON TO EACH REQUEST A DEFAULT ANSWER WILL BE TYPED. TO INVOKE THE DEFAULT TYPE A CARRIAGE RETURN.

THE REQUESTS & THEIR DEFAULTS ARE:

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

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

IF MORE THAN 1 DRIVE OR SLAVE IS TO BE TESTED, TYPE A COMMA 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/TU45 COMBINATIONS ARE TESTED. ADDITIONALLY THE SOFTWARE
SWR IS INVOKED WITH A SWITCH SETTING OF 100000 IF LOADED VIA ACT11.
NO OPERATOR INTERVENTION IS REQUIRED

** EXCEPTION: IF LOADED VIA TMDP TM03 DRIVE 0 TU45 SLAVE 0 IS
NOT TESTED.

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.
SW08 (000900)	PRINT TIME	THIS SWITCH WHEN SET WILL PRINT A TIME LINE AFTER EACH ITERATION.
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.

CHAPTER 4

ERRORS

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

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

```
TEST # XXXXXX DEVICE ERROR

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

WHERE:

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

4.2 ERROR TYPEOUT FORMAT (FUNCTION TIME OUT OF RANGE)

```
TEST # XXXXXX OUT OF RANGE ERROR

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

CHAPTER 5

SUBROUTINE ABSTRACTS

5.1 .SCOPE

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

1. LOADS R5 WITH BASE ADDRESS
2. TYPES TIME LINE <SW08>=1
3. PROVIDES CONTINUOUS LOOP <SW14>
4. MOVES FUNCTION TIME INTO TABLE
5. OUTPUTS LINE ITEM <SW10>=1
6. DELAYS 350MS BEFORE STARTING TEST
7. INIT'S DRIVE/SLAVE
8. CLEARS THE ERROR FLAG (ERFLG)
9. CHECK FOR CONTROL G (^G>

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

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.

CHAPTER 6

MISCELLANEOUS

6.1 STACK POINTER

THE STACK POINTER IS INITAILLY SET TO 500.

6.2 EXECUTION TIME

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

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

CHAPTER 7

PROGRAM DESCRIPTION

7.1 SAMPLE TIME DOCUMENT

```
TYPE FIRST ADDRESS OF CONTROLER: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 9 CHAN. SER. # 5009
*
* FUNCTION                TIME(SPECIFICATION)     TIME(ACTUAL)
* WRITE FROM BOT          RANGE=<063000-059000>       ACTUAL=061000
* WRITE START             RANGE=<004500-003700>       ACTUAL=004100
* WRITE SHUTDOWN          RANGE=<003600-002600>       ACTUAL=003100
* WRITE SETTLEDOWN        RANGE=<007600-001400>       ACTUAL=4500
* READ FROM BOT           RANGE=<009200-005200>       ACTUAL=007200
* READ START              RANGE=<001700-000900>       ACTUAL=001300
* READ SHUTDOWN           RANGE=<001450-000350>       ACTUAL=000900
* READ SETTLEDOWN         RANGE=<007600-001400>       ACTUAL=004500
* READ REV START          RANGE=<001700-000900>       ACTUAL=001300
* READ REV SHUTDOWN       RANGE=<001900-001500>       ACTUAL=001700
* READ REV SETTLEDOWN     RANGE=<007600-001400>       ACTUAL=004500
* TURN AROUND DELAY F-R   RANGE=<008800-002800>       ACTUAL=005800
* TURN AROUND DELAY R-F   RANGE=<008800-002800>       ACTUAL=005800
* GAP SIZE-STOP HALF      RANGE=<007900-004500>       ACTUAL=006200
* GAP SIZE-START HALF     RANGE=<008550-005250>       ACTUAL=006900
* GAP SIZE-INTERRECORD    RANGE=<006150-004450>       ACTUAL=005300
* GAP CONSISANCY          RANGE=<007050-005350>       ACTUAL=006200
* DATA TIME-800BPI        RANGE=<012000-011000>       ACTUAL=011500
* DATA TIME-1600BPI       RANGE=<012500-011500>       ACTUAL=012000
* ERASE GAP TIME          RANGE=<049900-046900>       ACTUAL=048400
* WRITE FILE MARK         RANGE=<051200-049200>       ACTUAL=050200
```

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 9 CHAN. SER. # 5009
*
*FUNCTION               TIME(SPECIFICATION)     TIME(ACTUAL)
*TAPE SPEED FWD         RANGE=<011300-010300>       ACTUAL=010800
*TAPE SPEED REV         RANGE=<011300-010300>       ACTUAL=010800
```

7.2 TEST SEQUENCE WITH RELATED ADJUSTMENTS AND ASSOCIATED HARDWARE

TEST NO./NAME	RELATED ADJUSTMENTS	ASSOCIATED HARDWARE
1. WRITE FROM BOT	*NONE	*M8928 ROM*M8903 ACCL CNTR
2. WRITE START	* ''	* '' * ''
3. WRITE SHUTDOWN	* ''	* '' * ''
4. WRITE SETTLEDOWN	* ''	*M8928 SETTLEDOWN ONE SHOT
5. READ FROM BOT	* ''	*M8928 ROM*M8903 ACCL CNTR
6. READ START	* ''	* '' * ''
7. READ SHUTDOWN	* ''	* '' * ''
10. READ SETTLEDOWN	* ''	*M8928 SETTLEDOWN ONE SHOT
11. READ REVERSE START	* ''	*M8928 ROM*M8903 ACCL CNTR
12. READ REVERSE SHUTDOWN	* ''	* '' * ''
13. READ REVERSE SETTLEDOWN	* ''	*M8928 SETTLEDOWN ONE SHOT
14. TURN AROUND F-R	* ''	*M8928 ROM*M8903 ACCL CNTR
15. TURN AROUND R-F	* ''	* '' * ''
16. GAP SIZE-STOP HALF	*FWRD/REV SPEED-START/STOP-RAMPS	*CAPSTAN SERVO LOOP
17. GAP SIZE-START HALF	*SAME AS IN TEST 16	* '' '' ''
20. GAP SIZE INTERRECORD	*FWD/REV SPEED	* '' '' ''

TM03 DRIVE FUNCTION TIMER PAGE 14

21. GAP CONSISTENCY	*SAME AS IN TEST 16	*WRITE CLOCK
22. DATA TIME 800 BPI	*NONE	* '' ''
23. DATA TIME 1600 BPI	* ''	* '' ''
24. ERASE GAP TIME	* ''	*M8928 ROM*M8903 ACCL CNTR
25. WRITE FILE MARK	* ''	* '' '' * '' '' ''
26. TAPE SPEED-FORWARD	*FWD SPEED	*CAPSTAN SERVO LOOP
27. TAPE SPEED-REVERSE	*REVERSE SPEED	*CAPSTAN SERVO LOOP

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

7.3 SUBTEST DESCRIPTIONS:

THE FIRST THIRTEEN (13) TESTS (T1 - T15) ARE CHECKS OF THE ROM CIRCUITS IN THE TU45 (M9811), THE ACCL COUNTER IN THE TM03 (M8903), AND THE SETTLEDOWN ONE SHOT (M8928).

T1. WRITE FROM BOT:

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

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

T2. WRITE START:

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

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

T3. WRITE SHUTDOWN:

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

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

T4. WRITE SETTLEDOWN:

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

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

T5. READ FROM BOT

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

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

T6. READ START

THIS TEST MEASURES THE SAME DELAY AS IN T2.

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

T7. READ SHUTDOWN:

THIS TEST MEASURES THE SAME DELAY AS IN T3.

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

T10. READ SETTLEDOWN:

THIS TEST MEASURES THE SAME DELAY AS IN T4.

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

TM03 DRIVE FUNCTION TIMER PAGE 17

T11. READ REVERSE START:

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

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

T12. READ REVERSE SHUTDOWN

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

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

T13. READ REVERSE SETTLEDOWN:

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

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

T14. TURN AROUND DELAY-FORWARD TO REVERSE

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

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

TM03 DRIVE FUNCTION TIMER PAGE 18

T15. TURN AROUND DELAY-REVERSE TO FORWARD

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

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

GAP MEASUREMENTS:

THE PREVIOUS THIRTEEN (13) TESTS WERE MEASUREMENTS OF LOGIC DELAYS PERFORMED BY THE TM03 OR TU45 IN ORDER TO ALLOW FOR PROPER ACCELERATION AND DECELLERATION OF TAPE ACCORDING TO THE DESIRED INTEROCORD GAP (.6 INCHES). THIS TEST, HOWEVER, WILL MEASURE THE PHYSICAL SIZE OF THE INTERCORD GAP THAT EXISTS ON TAPE AS A RESULT OF THE START/STOP TIMES OF THE CAPSTAN ITSELF. BECAUSE THE INTERRCORD GAP IS CREATED BY TWO ACTIONS, THE START OF MOTION AND THE STOP OF MOTION IT IS NECESSARY TO MAKE TWO SEPERATE MEASUREMENTS. A THIRD MEASUREMENT, MADE ON THE FLY, OF THE ENTIRE LENGTH OF THE GAP WILL ALSO BE MADE.

T16. GAP SIZE (STOP HALF)

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

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

T17. GAP SIZE (START HALF)

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

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

T20. GAP SIZE (INTERRECORD)

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

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

SEQ 0023

7. STOP

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

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

T24. ERASE:

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

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

T25. TAPE MARK:

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

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

TM03 DRIVE FUNCTION TIMER PAGE 22

T26. TAPE SPEED FORWARD:

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

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

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.

```
;
2297                                    .LIST   BIN,LOC,SEQ
2298
2299                                    .NLIST  MC
2300                                    .NLIST  TOC
2301                                    .LIST   ME
2302                                    .ENABLE ABS,AMA
2303                                    .MCALL  $CPVEC,$CPREG,$CATCH,$TYPE,..$ACT11,..$EOP,$CHAIN
2304                                    .TITLE  CZTUSB0 TM03/TU45 DRIVE FUNCTION TIMER
2305                                    .SBTTL  STARTING INSTRUCTIONS
2306                            ;LOADING AND STARTING PROCEDURE
2307                            ;       LOAD PROGRAM USING ABS LOADER
2308                            ;       LOAD ADDRESS 200
2309                            ;       SET SWITCH OPTIONS
2310                            ;       PRESS START
2311
2312                            ;RESTART PROCEDURE
2313                            ;       LOAD ADDRESS 210
2314                            ;       SET SWITCH OPTIONS
2315                            ;       PRESS START
2316
2317                            ·SWITCH REGISTER SWITCH ASSIGNMENTS
2318            100000                  SW15=   100000          ;HALT ON ERROR
2319            040000                  SW14=   040000          ;LOOP SUBTEST
2320            020000                  SW13=   020000          ;INHIBIT ERROR TYPE OUT
```

```
                        SW11=   004000          ;INHIBIT SUBTEST ITERATION
                        SW10=   002000          ;INHIBIT PUBLISHING TIME SPECIFICATION
                        SW09=   001000          ;RING BELL ON ERROR
                        SW08=   000400          ;TYPE LINE ITEM AFTER EACH ITERATION
                        SW07=   00200           ;NOT USED
                        SW06=   000100          ;CONTINUOUS CYCLE
                ;       SW05-SW00               ;RUN TEST SELECTED
                ;       **NOTE: IF <SW15-SW00> = 177777 AT STARTUP USE SOFTWARE
                ;               SWITCH REGISTER.

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

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

                        .SBTTL  MACRO DEFINITIONS
                        .MACRO  SAVE
                        JSR     PC,..SAVE               ;SAVE REGISTERS ON THE STACK
                        .ENDM   SAVE
                        .MACRO  RESTORE
                        JSR     PC,..RESTORE            ;RESTORE REGISTERS FROM THE STACK
                        .ENDM   RESTORE
                        .MACRO  INPUT
                        JSR     PC,..INPUT              ;GET USER INPUT
                        .ENDM   INPUT
                        .MACRO  REWIND
                        JSR     PC,..REWIND             ;REWIND SLAVE
                        BVS     99$                     ;BRANCH IF ERROR ON REWIND
                        .ENDM   REWIND
                        .MACRO  TIMEON
                        JSR     PC,TIMON                ;TURN TIMER ON
                        .ENDM
                        .MACRO  TIMCHK
                        JMP     TIMER(R3)               ;GO TO TIMER & RETURN VIA R2
                        .ENDM
                        .MACRO  SETGO
                        INC     (R5)                    ;SET 'GO' BIT
                        .ENDM

                        .SBTTL  REGISTER ASSIGNMENTS
                ;;DEFINITIONS AND REGISTER ASSIGNMENTS
                ;;GENERAL REGISTER ASSIGNMENTS
  (1)                   R0=%0
  (1)   000000          R1=%1
  (1)   000001          R2=%2
  (1)   000002          R3=%3
  (1)   000003          R4=%4
  (1)   000004
```

```
(1)          000005                  R5=%5
(1)          000006                  SP=%6
(1)          000007                  PC=%7
(1)          000000                  R10=%0
(1)          000001                  R11=%1
(1)          000002                  R12=%2
(1)          000003                  R13=%3
(1)          000004                  R14=%4
(1)          000005                  R15=%5
(1)
(1)                                  ;;REGISTER ADDRESSES
(1)          177776                          PSW=    177776          ;;PROCESSER STATUS WORD
(1)          177774                          SLR=    177774          ;;STACK LIMIT REGISTER (11/40,11/45)
(1)          177772                          PIRQ=   177772          ;;PROGRAM INTERRUPT REQ. (11/45)
(1)          177770                          UBREAK= 177770          ;;MICRO-BREAK REGISTER (11/45)
(1)          177560                          TKS=    177560          ;;KEYBOARD CSR
(1)          177562                          TKB=    177562          ;;KEYBOARD DATA BUFFER REGISTER
(1)          177564                          TPS=    177564          ;;TELEPRINTER CSR
(1)          177566                          TPB=    177566          ;;TELEPRINTER DATA BUFFER REGISTER
(1)
2371                                 ;;VECTOR ADDRESSES
(1)          000004                          ERRVEC=4                ;;ADDRESS OF ERROR VECTOR
(1)          000010                          RESVEC=10               ;;ADDRESS OF RESERVED INST. TRAP VECTOR
(1)          000014                          TBITVEC=14              ;;ADDRESS OF 'T' BIT TRAP VECTOR
(1)          000014                          TRTVEC=14               ;;ADDRESS OF 'TRACE' TRAP VECTOR
(1)          000014                          BPTVEC=14               ;;ADDRESS OF 'BREAKPOINT' TRAP VECTOR
(1)          000020                          IOTVEC=20               ;;ADDRESS OF IOT TRAP VECTOR
(1)          000024                          PFVEC=24                ;;ADDRESS OF POWER FAIL TRAP VECTOR
(1)          000030                          EMTVEC=30               ;;ADDRESS OF EMT VECTOR
(1)          000034                          TRAPVEC=34              ;;ADDRESS OF TRAP VECTOR
(1)          000060                          TKVEC=  60              ;;ADDRESS OF TTY KEYBOARD INT. VECTOR
(1)          000064                          TPVEC=64                ;;ADDRESS OF TTY PRINTER INTERRUPT VECTOR
(1)          000114                          PARVEC= 114             ;;ADDRESS OF MA/MF PARITY ERROR VECTOR
(1)          000240                          PIRVEC=240              ;;ADDRESS OF PIRQ VECTOR
(1)          000244                          FPEVEC=244              ;;ADDRESS OF FLOATING POINT INT. VECTOR
(1)          000250                          MMVEC=250               ;;ADDRESS OF MEM MGMT ERROR TRAP VECTOR
(1)
2372                                 ;CLOCK ADDRESS AND VECTORS
2373         172540                          PLKCSR= 172540          ;KW11-P
2374         000104                          PLKVEC= 104
2375         177546                          LKS=    177546          ;KW11-L
2376         000100                          LKVEC=  100
2377         177514                          LPS=    177514          ;LP11
2378         177516                          LPB=    177516
2379
2380                                 ;RH,TM03/TU45 REGISTERS
2381         172440                          TMCS1=  172440
2382
2383                                 ;TM03/TU45 INDEX VALUES
2384         000000                          CS1=    00                      ;CONTROL STATUS #1
2385         000002                          WC=     02
2386         000004                          BA=     04                      ;BUS ADDRESS REGISTER
2387         000006                          FC=     06                      ;FRAME COUNT
2388         000010                          CS2=    10                      ;CONTROL STATUS #2
2389         000012                          DS=     12                      ;DRIVE STATUS
2390         000014                          ER=     14                      ;ERROR REG #1
```

```
2391    000016                  AS=     16                      ;ATTENTION SUMMARY
2392    000022                  DB=     22                      ;DATA BUFFER REG
2393    000024                  MR=     24                      ;MAINTENANCE REG
2394    000026                  DT=     26                      ;DRIVE TYPE REG
2395    000030                  SN=     30                      ;SERIAL NUMBER REGISTER
2396    000032                  TC=     32                      ;TAPE CONTROL REG
2397
2398                            .SBTTL  TM03/TU45 REGISTER BITS
2399                    ;RHCS1-CS1(R5)
2400    000001                  GO=     1
2401    000000                  NOP=    0
2402    000002                  RWDOFF= 2
2403    000006                  RWD=    6
2404    000010                  DRYCLR= 10
2405    000026                  WFMK=   26
2406    000024                  ERASE=  24
2407    000030                  SPCFWD= 30
2408    000032                  SPCREV= 32
2409    000050                  WCHKF=  50
2410    000056                  WCHKR=  56
2411    000060                  WFWD=   60
2412    000070                  RDFWD=  70
2413    000076                  RDREV=  76
2414    000100                  IE=     100
2415    000200                  RDY=    200
2416    000400                  A16=    400
2417    001000                  A17=    1000
2418    002000                  PSEL=   2000
2419    004000                  DVA=    4000
2420    020000                  MCPE=   20000
2421    040000                  TRE=    40000
2422    100000                  SC=     100000
2423                    ;RHCS2-CS2(R5)
2424    000000                  DV0=    0
2425    000001                  DV1=    1
2426    000002                  DV2=    2
2427    000003                  DV3=    3
2428    000004                  DV4=    4
2429    000005                  DV5=    5
2430    000006                  DV6=    6
2431    000007                  DV7=    7
2432    000010                  BAI=    10
2433    000020                  PAT=    20
2434    000040                  CLR=    40
2435    000100                  IR=     100
2436    000200                  OR=     200
2437    000400                  MDPE=   400
2438    001000                  MXF=    1000
2439    002000                  PGE=    2000
2440    004000                  NEM=    4000
2441    010000                  NED=    10000
2442    020000                  UPE=    20000
2443    040000                  WCE=    40000
2444    100000                  DLT=    100000
2445                    ;RHDS-DS(R5)
2446    000001                  SLA=    1
```

```
                                        BOT=    2
                                        TMK=    4
                                        IDB=    10
                                        SDWN=   20
                                        PES=    40
                                        SSC=    100
                                        DRY=    200
                                        DPR=    400
                                        EOT=    2000
                                        WRL=    4000
                                        MOL=    10000
                                        PIP=    20000
                                        ERR=    40000
                                        ATA=    100000
                                ;RHER-ER(R5)
                                        ILF=    1
                                        ILR=    2
                                        RMR=    4

                                        FMT=    20
                                        INCVAE= 100
                                        PEFLRC= 200
                                        NSG=    400
                                        FCE=    1000
                                        CSITM=  2000
                                        NEF=    4000
                                        DTE=    10000
                                        OPI=    20000
                                        UNS=    40000

                                ;RHMR-MR(R5)
                                        OSC=    100

                                ;RHDT-DT(R5)
                                        SPR=    2000
                                        CH7=    10000
                                        TAP=    40000

                                ;RHTC-TC(R5)
                                        NORM11= 1700
                                        CDM11=  320
                                        BPI200= 0
                                        BPI556= 000400
                                        BPI800= 001000
                                        PE1600= 002300
                                        ACCL=   100000



                                ;INSTRUCTION EQUATES
                                        HLT=    TRAP
                                        SCOPE=  EMT
                                        TYPE=   IOT

                                ;MISCELLANEOUS EQUATES
                                        OUTBUF=INIT             ;OUTPUT BUFFER STARTS AT BEG OF PROGRAM
```

```
2503            177400                  FRMCNT= -256.                   ;FRAME COUNT
2504            177600                  WRDCNT- -128.                   ;WORD COUNT
2505                                    ;ASCII EQUATES
2506            000001                  CNTRLA= 1                       ;ASCII CODE FOR CONTROL A (^A)
2507            000003                  CNTRLC= 3                       ;ASCII CODE FOR CONTROL C (^C)
2508            000007                  CNTRLG= 7                       ;ASCII CODE FOR CONTROL G (^G)
2509            000011                  HT=     11                      ;ASCII CODE FOR HORIZONTAL TAB
2510            000012                  LF=     12                      ;ASCII CODE FOR LINE FEED
2511            000015                  CR=     15                      ;ASCII CODE FOR CARRIAGE RETURN
2512            000017                  CNTRLO= 17                      ;ASCII CODE FOR CONTROL O (^O)
2513            000025                  CNTRLU= 25                      ;ASCII CODE FOR CONTROL U (^U)
```

```
2516                                    ;SETUP TRAP VECTORS
2517            000014                          .=TBITVEC
2518  000014   000016                          .WORD   .+2                     ;SET 'T' TRAP TO TIMER ROUTINE
2519  000016   000000                          .WORD   HALT                    ;PRIORITY LEVEL 7
2520  000020   002130                          .WORD   .TYPE                   ;SET IOT TRAP TO .TYPE ROUTINE
2521  000022   000000                          .WORD   0                       ;PRIOTITY LEVEL 0
2522  000024   000026                          .WORD   PFVEC+2                 ;POWER FAIL TRAP TO HALT
2523  000026   000000                          .WORD   HALT                    ;AT PFVEC+2
2524  000030   004010                          .WORD   .SCOPE                  ;SET EMT TRAP TO .SCOPE ROUTINE
2525  000032   000340                          .WORD   340                     ;PRIORITY LEVEL 7
2526  000034   003544                          .WORD   .HLT                    ;SET TRAP TRAP TO .HLT ROUTINE
2527  000036   000340                          .WORD   340                     ;PRIORITY LEVEL 7
2528
 (1)                                    ;ACT11 HOOK **********
 (1)            000040                          $SVPC=.                 ;SAVE CURRENT LOCATION CTR
 (1)            000046                          .=46
 (1)  000046   012712                          .WORD   $ENDAD          ;SET LOCATION 46
 (1)            000052                          .=52
 (1)  000052   000000                          .WORD   0               ;SET LOCATION 52 = 0
 (1)            000040                          .=$SVPC                 ;RESTORE LOCATION CTR
 (1)
2529            000060                          .=TKVEC
2530  000060   003420                          .WORD   TKISR
2531  000062   000200                          .WORD   200
2532
2533                                            ;SOFTWARE SWITCH REGISTER LOC. 176
2534            000176                          .=176
2535  000176   000100                          SWREG:  .WORD   SW06            ;SOFTWARE SWITCH REGISTER
2536
2537            000200                          .=200
2538  000200   000137  005650                  JMP     @#INIT                  ;GO TO START OF PROGRAM
2539            000210                          .=210
2540  000210   000137  006756                  JMP     @#RSTRT         ;RESTART ADDRESS
2541
2542            000500                          .=500
2543            000600                  STKPTR= 600                     ;STACK
2544
2545            001000                          .=1000
2546                                    ;PROGRAM TAGS
2547  001000   177570                  SWR:    177570                  ;SWITCH REGISTER
2548  001002   000000                  SCPADR: .WORD   0
2549  001004      000                   DRVNUM: .BYTE   0               ;TM03 DRIVE UNDER TEST
2550  001005      000                   SLVNUM: .BYTE   0               ;TU45 SLAVE UNDER TEST
2551  001006   000000                  SLVPTR: .WORD   0               ;POINTER TO SLAVE TABLE (SLVTBL) BELOW
2552  001010   172440                  TMBASE: .WORD   TMCS1           ;BASE ADDRESS OF TM03/TU45 REGISTERS
2553  001012   000000                  ATIME:  .WORD   0                       ;CONTAINS 'TICK' COUNT
2554  001014   000020                  ATIMTBL:.BLKW   16.             ;EACH ENTRY CONTAINS TIME FOR FUNCTION
2555                                                                   ;ENTRIES ARE MADE BY 'SCOPE' ROUTINE
2556  001054   000020                  GAPTBL: .BLKW   16.             ;TIMES RECORDED BY 'GAP CONSISTANCY' TEST
2557  001114   000000                  DELTIM: .WORD   0               ;VARIBLE DELAY
2558  001116   000000                  OCTALO: .WORD   0
2559  001120      000                   GAP:    .BYTE   0               ;CONTAINS GAP # (USED FOR TST 021)
2560  001121      000                   ITCNT:  .BYTE   0               ;ITERATION COUNT
2561  001122      000                   TSTNUM: .BYTE   0               ;TEST #
2562  001123      000                   ERFLG:  .BYTE   0               ;ERROR FLAG
2563  001124      000                   PRGFLG: .BYTE   0               ;PROGRAM FLAG
```

```
2564  001125      000                 UNTFND: .BYTE   0               ;UNIT FOUND INDICATOR
2565  001126      000                 TYPFLG: .BYTE   0
2566  001127      000                 PSCNT:  .BYTE   0               ;CONTAINS PASS COUNT
2567  001130      000                 ASFLG:  .BYTE   0               ;1/0 = YES/NO.
2568              001132                      .EVEN
2569  001132  030460                 DIGTAB: ''01
2570  001134  031462                         ''23
2571  001136  032464                         ''45
2572  001140  033466                         ''67
2573  001142  034470                         ''89
2574  001144  000006                 ODIGITS:.BLKB   6               ;RESERVE SPACE FOR CONVERTED DIGITS
2575  001152      000                         .BYTE   0               ;TERMINATOR
2576              001154                      .EVEN
2577  001154  000010                 DRVTBL: .BLKB   8.              ;A 0/-1 = DRIVE NOT TO BE/TO BE TESTED
2578  001164  000100                 SLVTBL: .BLKB   64.             ;A 0/-1 = SLAVE NOT TO BE/TO BE TESTED
2579  001264  000110                 INBUF:  .BLKB   72.             ;TELETYPE INPUT BUFFER
2580  001374  005015      000        CRLF:   .ASCIZ  <CR><LF>        ;MISCELLANEOUS ASCII CHARACTERS
2581  001377      134      000       BKSLSH: .ASCIZ  '\'
2582  001401      060      000       ECHO:   .ASCIZ  '0'
2583  001403      007      000       BELL:   .ASCIZ  <7>
2584  001405      055      000       DASH:   .ASCIZ  '-'
2585  001407      040                SPACE2: .ASCII  ' '
2586  001410  000040                 SPACE:  .ASCIZ  ' '
2587  001412  004476      000        ANGTAB: .ASCIZ  '>'<HT>
2588              001416                      .EVEN
```

```
2590                                    .SBTTL  TIME SPECIFICATION TABLE
2591                                    ;THE BELOW TABLE CONTAINS THE SPECIFIED FUNCTION TIMES IN TENS OF
2592                                    ;MICROCESONDS. NOTE THAT WHEN TIMES ARE TYPED THAT THEY ARE TYPED IN
2593                                    ;MICROSECONDS (BY APPENDING A 0).
2594                                    ;FORMAT IS
2595                                    ;       .WORD   MAX,MIN         ;TIME IN MS     FUNCTION        TEST #
2596
2597  001416  000000  000000  STIMTBL:.WORD   0,0                     ;SPARE
2598  001422  014234  013414          .WORD   6300.,5900.             ;63.0-59.0      WRITE FROM BOT  TST001
2599  001426  000702  000562          .WORD   00450.,00370.           ;4.5-3.7        WRITE START     TST002
2600  001432  000550  000404          .WORD   00360.,00260.           ;3.6-2.6        WRITE SHUTDOWN  TST003
2601  001436  001370  000214          .WORD   00760.,00140.           ;7.6-1.4        WRITE STLDOWN   TST004
2602  001442  001630  001010          .WORD   00920.,00520.           ;9.2-5.2        READ FROM BOT   TST005
2603  001446  000252  000132          .WORD   00170.,00090.           ;1.7-0.9        READ START      TST006
2604  001452  000221  000043          .WORD   00145.,00035.           ;1.45-0.35      READ SHUTDOWN   TST007
2605  001456  001370  000214          .WORD   00760.,00140.           ;7.6-1.4        READ SETTLEDOWN TST010
2606  001462  000252  000132          .WORD   00170.,00090.           ;1.7-0.9        RD REV START    TST011
2607  001466  000276  000226          .WORD   00190.,00150.           ;1.9-1.5        RD REV SHTDWN   TST012
2608  001472  001370  000214          .WORD   00760.,00140.           ;7.6-1.4        RD REV STLDWN   TST013
2609  001476  001560  000430          .WORD   00880.,00280.           ;8.8-2.8        TRN RND DLY F-R TST014
2610  001502  001560  000430          .WORD   00880.,00280.           ;8.8-2.8        TRN RND DLY R-F TST015
2611  001506  001426  000702          .WORD   00790.,00450.           ;7.9-4.5        GAP SIZE STOP   TST016
2612  001512  001527  001015          .WORD   00855.,00525.           ;8.55-5.25      GAP SIZE STRT   TST017
2613  001516  001147  000675          .WORD   00615.,00445.           ;6.15-4.45      GAP SIZE INTER  TST020
2614  001522  001301  001027          .WORD   00705.,00535.           ;7.05-5.35      GAP CONSISANCY  TST021
2615  001526  002260  002114          .WORD   01200.,01100.           ;12.0-11.0      DAT TIME 800BPI TST022
2616  001532  002342  002176          .WORD   01250.,01150.           ;12.5-11.5      DAT TIME 1600PE TST023
2617  001536  011576  011122          .WORD   04990.,04690.           ;49.9-46.9      ERASE           TST024
2618  001542  012000  011470          .WORD   05120.,04920.           ;51.2-49.2      WRT FILE MARK   TST025
2619  001546  002152  002006          .WORD   01130.,01030.           ;11.3-10.3      TAPE SPEED FWD  TST026
2620  001552  002152  002006          .WORD   01130.,01030.           ;11.3-10.3      TAPE SPEED REV  TST027
2621
2622                                    ;NOTE: TEST 26 AND 27 REQUIRE PRERECORDED 800BPI SKEW TAPE.
```

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

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

001556  001274  001041       GTIMTBL:.WORD   00700.,00545.   ;7.00-5.45      GAP-0   0 MS
001562  001344  001200               .WORD   00740.,00640.   ;7.4-6.4       GAP-1   1.0 MS
001566  001370  001154               .WORD   00760.,00620.   ;7.6-6.2       GAP-2   2.0 MS
001572  001274  001060               .WORD   00700.,00560.   ;7.0-5.6       GAP-3   3.0 MS
001576  001306  000776               .WORD   00710.,00510.   ;7.1-5.1       GAP-4   4.0 MS
001602  001351  000707               .WORD   00745.,00455.   ;7.45-4.55      GAP-5   5.0 MS
001606  001351  000733               .WORD   00745.,00475.   ;7.45-4.75      GAP-6   6.0 MS
001612  001351  000733               .WORD   00745.,00475.   ;7.45-4.75      GAP-7   7.0 MS
001616  001274  000764               .WORD   00700.,00500.   ;7.0-5.0        GAP-10  8.0 MS
001622  001262  000764               .WORD   00690.,00500.   ;6.9-5.0        GAP-11  9.0 MS
001626  001262  000764               .WORD   00690.,00500.   ;6.9-5.0        GAP-12  10.0 MS
001632  001262  000764               .WORD   00690.,00500.   ;6.9-5.0        GAP-13  11.0 MS
001636  001262  000764               .WORD   00690.,00500.   ;6.9-5.0        GAP-14  12.0 MS
001642  001262  000764               .WORD   00690.,00500.   ;6.9-5.0        GAP-15  13.1 MS
001646  001262  000764               .WORD   00690.,00500.   ;6.9-5.0        GAP-16  14.1 MS
001652  001262  000764               .WORD   00690.,00500.   ;6.9-5.0        GAP-17  15.1 MS
```

```
2648                                            .SBTTL  TEST HEADER POINTERS
2649                                    ;THE BELOW TABLE CONTAINS POINTERS TO EACH TEST'S DESCRIPTOR
2650  001656  000000                    NAMPTR: .WORD   0
2651  001660  015103                            .WORD   A.T001
2652  001662  015125                            .WORD   A.T002
2653  001664  015145                            .WORD   A.T003
2654  001666  015167                            .WORD   A.T004
2655  001670  015213                            .WORD   A.T005
2656  001672  015235                            .WORD   A.T006
2657  001674  015254                            .WORD   A.T007
2658  001676  015276                            .WORD   A.T010
2659  001700  015321                            .WORD   A.T011
2660  001702  015343                            .WORD   A.T012
2661  001704  015370                            .WORD   A.T013
2662  001706  015417                            .WORD   A.T014
2663  001710  015450                            .WORD   A.T015
2664  001712  015501                            .WORD   A.T016
2665  001714  015527                            .WORD   A.T017
2666  001716  015556                            .WORD   A.T020
2667  001720  015606                            .WORD   A.T021
2668  001722  015631                            .WORD   A.T022
2669  001724  015655                            .WORD   A.T023
2670  001726  015702                            .WORD   A.T024
2671  001730  015724                            .WORD   A.T025
2672  001732  015747                            .WORD   A.T026
2673  001734  015771                            .WORD   A.T027
2674
2675                                    ;TABLE OF TEST STARTING ADDRESSES
2676  001736  007224                    TSTTBL: .WORD   TST000
2677  001740  007474                            .WORD   TST001
2678  001742  007560                            .WORD   TST002
2679  001744  007636                            .WORD   TST003
2680  001746  007726                            .WORD   TST004
2681  001750  010034                            .WORD   TST005
2682  001752  010120                            .WORD   TST006
2683  001754  010204                            .WORD   TST007
2684  001756  010304                            .WORD   TST010
2685  001760  010424                            .WORD   TST011
2686  001762  010522                            .WORD   TST012
2687  001764  010632                            .WORD   TST013
2688  001766  010772                            .WORD   TST014
2689  001770  011064                            .WORD   TST015
2690  001772  011172                            .WORD   TST016
2691  001774  011266                            .WORD   TST017
2692  001776  011376                            .WORD   TST020
2693  002000  011516                            .WORD   TST021
2694  002002  012022                            .WORD   TST022
2695  002004  012152                            .WORD   TST023
2696  002006  012302                            .WORD   TST024
2697  002010  012422                            .WORD   TST025
2698  002012  012764                            .WORD   TST026
2699  002014  013112                            .WORD   TST027
```

```
2701  002016  000000                  TIB:    .WORD   0
2702                                  ;ROUTINE TO LOAD SSOFTWARE SWR
2703
2704  002020  022737  000176  001000  GTSWR:  CMP     #SWREG,SWR      ;BRANCH IF SOFTWARE SWR
2705  002026  001027                  BNE     2$              ;NOT INVOKED
2706  002030  004737  002354          JSR     PC,..SAVE       ;SAVE REGISTERS ON THE STACK
2707  002034  000004  016020          TYPE,L.SWR
2708  002040  017702  176734          MOV     @SWR,R2
2709  002044  004737  002426          JSR     PC,TYPOCT
2710  002050  000004  016027          TYPE,L.NEW
2711  002054  004737  003272          JSR     PC,..INPUT              ;GET USER INPUT
2712  002060  122737  000015  001264  CMPB    #CR,@#INBUF             ;EXIT IF FIRST CHAR IS <CR>
2713  002066  001405                  BEQ     1$
2714  002070  004737  003056          JSR     PC,CNVTAO               ;CONERT ASCII TO OCTAL
2715  002074  013777  001116  176676  MOV     @#OCTALO,@SWR           ;SET NEW SWITCH REG CONTENTS
2716  002102  004737  002376  1$:     JSR     PC,..RESTORE
2717  002106  000207          2$:     RTS     PC
2718
2719
2720                                          .SBTTL  PROGRAM SUBROUTINES
2721                                          .SBTTL          TYPE SUBROUTINE
2722                                  ;;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)   002110     000                  $NULL:  .BYTE   0               ;;CONTAINS NULL CHARACTER
 (1)   002111     002                  $FILL:  .BYTE   2               ;;CONTAINS # OF FILLER CHARACTERS
 (1)   002112     000                  $TPFLG: .BYTE   0               ;;CONTAINS TELEPRINTER AVAILABLE FLAG
 (1)                                                                   ;;0/377 = AVAIL/NOT AVAIL
 (1)   002113     000                  $TKFLG: .BYTE   0               ;;CONTAINS KEYBOARD AVAILABLE FLAG
 (1)   002114  177564                  $TPS:   .WORD   177564          ;;ADDRESS OF TLEPRINTER STATUS REGISTER
 (1)   002116  177566                  $TPB:   .WORD   177566          ;;ADDRESS OF TELEPRINTER DATA BUFFER
 (1)   002120     000                  $CHARCNT:.BYTE  0               ;;CONTAINS # OF CHARS TYPED
 (1)   002121     000                  $CNTRLO:.BYTE   0               ;;CONTAINS CONTROL O CHAR (IF TYPED)
 (1)   002122  005015      000         $CRLF:  .ASCIZ  <15><12>
 (1)           002126                          .EVEN
 (1)   002126  000000                  RDSW:   .WORD   0
 (1)
 (1)   002130  010046                  .TYPE:  MOV     R0,-(SP)        ;;SAVE R0
 (1)   002132  017600  000002          MOV     @2(SP),R0       ;;GET MESSAGE ADDRESS
 (1)   002136  062766  000002  000002  ADD     #2,2(SP)        ;;ADJUST RETURN PC
 (1)   002144  105037  002121          CLRB    $CNTRLO
 (1)
 (1)   002150  105737  002121          TYPE1:  TSTB    $CNTRLO         ;;BRANCH IF CONTROL O(^O) WASN'T TYPED
 (1)   002154  001410                  BEQ     TYPE2
 (1)   002156  000004  002122          TCRLF:  TYPE,$CRLF              ;;TYPE <CR><LF>
 (1)   002162  105737  002126          TSTB    RDSW
 (1)   002166  100006                  BPL     TYPE3
 (1)   002170  005037  002126          CLR     RDSW
 (1)   002174  000207                  RTS     PC
 (1)   002176  112046                  TYPE2:  MOVB    (R0)+,-(SP)     ;;PUSH CHARACTER TO BE TYPED ONTO STACK
 (1)   002200  001003                  BNE     TYPE4           ;;BRANCH IF NOT THE TERMINATOR
 (1)   002202  005726                  TST     (SP)+           ;;POP TERMINATOR CHAR OFF THE STACK
```

```
(1)  002204  012600                TYPE3:  MOV     (SP)+,R0        ;;RESTORE R0
(1)  002206  000002                        RTI                     ;;RETURN TO CALLER
(1)
(1)  002210  122716  000011        TYPE4:  CMPB    #$HT,(SP)       ;;BRANCH IF HORIZONTAL TAB <HT>
(1)  002214  001445                        BEQ     9$
(1)  002216  004737  002250                JSR     PC,5$           ;;TYPE CHARACTER
(1)  002222  122726  000012        3$:     CMPB    #12,(SP)+       ;;CHECK IF CHARACTER WAS A LINE FEED
(1)  002226  001350                        BNE     TYPE1           ;;BRANCH IF NOT LINE FEED
(1)  002230  013746  002110                MOV     $NULL,-(SP)     ;;GET # OF FILLERS REQUIRED AND FILLER
(1)                                                                ;;CHARACTER.
(1)
(1)  002234  105366  000001        4$:     DECB    1(SP)           ;;DECREMENT FILLERS REQ. COUNT
(1)  002240  002770                        BLT     3$              ;;BRANCH IF NO MORE FILLERS ARE REQUIRED
(1)  002242  004737  002250                JSR     PC,5$           ;;TYPE FILLER CHARACTER
(1)  002246  000772                        BR      4$
(1)
(1)  002250  105777  177640        5$:     TSTB    @$TPS           ;;WAIT FOR OUTPUT DEVICE
(1)  002254  100375                        BPL     .-4
(1)  002256  122737  000017  002121        CMPB    #17,@#$CNTRLO   ;;CHECK IF CONTROL O WAS TYPED
(1)  002264  001403                        BEQ     6$              ;;STOP TYPING MESSAGE IF ^O WAS TYPED
(1)  002266  116677  000002  177622        MOVB    2(SP),@$TPB     ;;OUTPUT CHARACTER
(1)  002274  122766  000015  000002  6$:     CMPB    #15,2(SP)       ;;BRANCH IF NOT <CR>
(1)  002302  001003                        BNE     7$
(1)  002304  105037  002120                CLRB    $CHARCNT        ;;CLEAR CHARACTERS TYPED COUNT
(1)  002310  000406                        BR      8$
(1)  002312  122766  000012  000002  7$:     CMPB    #12,2(SP)       ;;BRANCH IF <LF> OR 'NULL'
(1)  002320  002002                        BGE     8$
(1)  002322  105237  002120                INCB    $CHARCNT        ;;INCREMENT CHARACTER TYPED COUNT
(1)  002326  000207                8$:     RTS     PC
(1)
(1)                                        ;;HORIZONTAL TAB <HT> PROCESSER
(1)  002330  112716  000040        9$:     MOVB    #40,(SP)        ;;LOAD 'SPACE'
(1)  002334  004737  002250        10$:    JSR     PC,5$           ;;TYPE 'SPACE'
(1)  002340  132737  000007  002120        BITB    #7,$CHARCNT     ;;TYPE SPACES UNTIL A MULTIPLE
(1)  002346  001372                        BNE     10$             ;;OF 8 CHARACTERS HAVE BEEN TYPED
(1)  002350  105726                        TSTB    (SP)+           ;;POP SPACE
(1)  002352  000676                        BR      TYPE1           ;;GET NEXT CHARACTER
2723
2724                                       ;SUBROUTINE TO SAVE GENERAL REGISTERS ON THE STACK
2725                                       ;CALL:  SAVE
2726  002354  010546              .SAVE:  MOV     R5,-(SP)                ;SAVE REGISTERS ON THE STACK
2727  002356  010446                      MOV     R4,-(SP)
2728  002360  010346                      MOV     R3,-(SP)
2729  002362  010246                      MOV     R2,-(SP)
2730  002364  010146                      MOV     R1,-(SP)
2731  002366  010046                      MOV     R0,-(SP)
2732  002370  016646  000014              MOV     14(SP),-(SP)            ;GET RETURN PC
2733  002374  000207                      RTS     PC                      ;RETURN
2734
2735                                       ;SUBROUTINE TO RESTORE GENERAL REGISTERS FROM THE STACK
2736                                       ;CALL:  RESTORE
2737  002376  012666  000014      .RESTORE:MOV    (SP)+,14(SP)            ;MOVE RETURN PC
2738  002402  012600                      MOV     (SP)+,R0                ;RESTORE REGISTERS
2739  002404  012601                      MOV     (SP)+,R1
2740  002406  012602                      MOV     (SP)+,R2
2741  002410  012603                      MOV     (SP)+,R3
```

```
2742  002412  012604                  MOV     (SP)+,R4
2743  002414  012605                  MOV     (SP)+,R5
2744  002416  000207                  RTS     PC                      ;RETURN
2745
2746                                ;SUBROUTINE TO CONVERT OCTAL DATA TO ASCII
2747                                ;CALL:  MOV     NUMBER,R2               ;MOVE NUMBER TO R2
2748                                ;       JSR     PC,CNVOCT
2749
2750  002420  110637  001126        CNVOCT: MOVB    SP,TYPFLG               ;SET DO NOT TYPE FLAG
2751  002424  000402                  BR      CNVTO
2752
2753                                        .SBTTL          OCTAL TO ASCII & TYPE ROUTINE
2754                                ;SUBROUTINE TO CONVERT OCTAL NUMBER TO ASCII AND TYPE IT OUT
2755                                ;CALL:  MOV     NUMBER,R2               ;PUT # IN R2
2756                                ;       JSR     PC,TYPOCT               ;CALL ROUTINE
2757
2758  002426  105037  001126        TYPOCT: CLRB    @#TYPFLG                ;SET TYPE FLAG
2759  002432                        CNVTO:
 (1)  002432  004737  002354                  JSR     PC,.SAVE                ;SAVE REGISTERS ON THE STACK
2760  002436  012704  001144                  MOV     #ODIGITS,R4             ;SET PTR TO OUTPUT
2761  002442  005003                  CLR     R3                      ;R3 WILL CONTAIN OCTAL DIGIT
2762  002444  010201                  MOV     R2,R1                   ;GET # TO BE TYPED
2763  002446  006302                  1$:     ASL     R2                      ;SHIFT #
2764  002450  006103                  ROL     R3
2765  002452  012700  000006                  MOV     #6,R0                   ;SET DIGIT COUNTER
2766  002456  000404                  BR      3$
2767
2768  002460  006302                  2$:     ASL     R2                      ;SHIFT # 3 PLACES LEFT
2769  002462  006103                  ROL     R3
2770  002464  005301                  DEC     R1
2771  002466  001374                  BNE     2$
2772  002470  012701  000003          3$:     MOV     #3,R1                   ;SET SHIFT COUNTER
2773  002474  116324  001132                  MOVB    DIGTAB(R3),(R4)+        ;MOVE ASCII EQUIV TO OUTPUT
2774  002500  005003                  CLR     R3
2775  002502  005300                  DEC     R0                      ;DECREMENT DIGIT COUNT
2776  002504  001365                  BNE     2$                      ;GET NEXT DIGIT
2777  002506  105737  001126                  TSTB    @#TYPFLG                ;BRANCH IF ASCII IS
2778  002512  001002                  BNE     4$                      ;NOT TO BE TYPED
2779  002514  000004  001144                  TYPE,ODIGITS
2780  002520                          4$:
 (1)  002520  004737  002376                  JSR     PC,.RESTORE             ;RESTORE REGISTERS FROM THE STACK
2781  002524  000207                  RTS     PC
2782
2783
2784                                ;SUBROUTINE TO CONVERT OCTAL DATA TO DECIMAL ASCII
2785                                ;CALL:  MOV     NUMBER,R2               ;MOVE NUMBER TO R2
2786                                ;       JSR     PC,CNVDEC
2787
2788  002526  110637  001126        CNVDEC: MOVB    SP,@#TYPFLG             ;SET DO NOT TYPE FLAG
2789  002532  000402                  BR      CNVTD
2790                                        .SBTTL          OCTAL TO DECIMAL & TYPE ROUTINE
2791                                ;THIS ROUTINE CONVERTS AN OCTAL # TO DECIMAL ASCII AND TYPES IT OUT
2792                                ;CALL:  MOV     NUMBER,R2               ;PUT # IN R2
2793                                ;       JSR     PC,TYPDEC               ;CALL ROUTINE
2794
2795  002534  105037  001126        TYPDEC: CLRB    @#TYPFLG                ;SET TYPE FLAG
```

```
2796  002540                        CNVTD:
 (1)  002540  004737  002354                JSR     PC,..SAVE               ;SAVE REGISTERS ON THE STACK
2797  002544  005000                        CLR     R0                      ;R0 IS INDEX TO DECIMAL CONSTANT
2798  002546  012704  001144                MOV     #ODIGITS,R4             ;SET OUTPUT PTR
2799  002552  005003                1$:     CLR     R3                      ;R3 CONTAINS DECIMAL DIGIT
2800  002554  166002  002634        2$:     SUB     DCONST(R0),R2           ;SUBTRACT DECIMAL CONSTANT UNTIL
2801  002560  103402                        BLO     3$                      ;INPUT # GOES NEGATIVE
2802  002562  005203                        INC     R3                      ;KEEPING TRACK OF SUBTRACTIONS
2803  002564  000773                        BR      2$
2804  002566  066002  002634        3$:     ADD     DCONST(R0),R2           ;ADD BACK CONSTANT WHEN NEGATIVE
2805  002572  116324  001132                MOVB    DIGTAB(R3),(R4)+        ;MOVE ASCII EQUIVILENT
2806  002576  062700  000002                ADD     #2,R0                   ;NEXT CONSTANT
2807  002602  005760  002634                TST     DCONST(R0)              ;UNTIL ALL CONSTANTS DONE
2808  002606  001361                        BNE     1$
2809  002610  112724  000060                MOVB    #'0,(R4)+               ;LAST DIGIT IS 0
2810  002614  105737  001126                TSTB    @#TYPFLG                ;BRANCH IF ASCII IS
2811  002620  001002                        BNE     4$                      ;NOT TO BE TYPED
2812  002622  000004  001144                TYPE,ODIGITS
2813  002626                        4$:
 (1)  002626  004737  002376                JSR     PC,..RESTORE            ;RESTORE REGISTERS FROM THE STACK
2814  002632  000207                        RTS     PC
2815
2816  002634  023420                DCONST: .WORD   10000.
2817  002636  001750                        .WORD   1000.
2818  002640  000144                        .WORD   100.
2819  002642  000012                        .WORD   10.
2820  002644  000001                        .WORD   1.
2821  002646  000000                        .WORD   0                       ;TERMINATOR
2822
2823                                        .SBTTL          TYPE SPECIFIED TIMES ROUTINE
2824                                ;THIS SUBROUTINE OUTPUTS THE TIME SPECIFICATIONS FOR THE TEST
2825                                ;AND ALSO THE ACTUAL TIME RECORDED (ATIME)
2826                                ;FORMAT OF LINE TYPED
2827                                ;RANGE=<AAAAAA-BBBBBB>          ACTUAL=CCCCCC
2828                                ;WHERE:         AAAAAA IS MAXIMUM TIME FOR TEST (STIMTBL(TSTNUMX4)).
2829                                ;               BBBBBB IS MINIMUM TIME FOR TEST (STIMTBL(TSTNUMX4+2)).
2830                                ;               CCCCCC IS ACTUAL TIME RECORDED BY TEST (ATIME).
2831                                ;CALL:  MOVB    TEST NUMBER,R2  ;LOAD TEST NUMBER
2832                                ;       MOV     #TIME,@#ATIME           ;MOVE TIME TO ATIME
2833                                ;       JSR     PC,OUTSPC
2834  002650  010246                OUTSPC: MOV     R2,-(SP)                ;SAVE R2 & R3 ON THE STACK
2835  002652  010346                        MOV     R3,-(SP)
2836  002654  006302                        ASL     R2                      ;MULTIPLY TEST # TIMES 4
2837  002656  006302                        ASL     R2                      ;TO FORM INDEX INTO STIMTBL
2838  002660  010203                        MOV     R2,R3                   ;R3 CONTAINS INDEX INTO TABLE
2839  002662  000004  015063                TYPE,L.RNG
2840  002666  016302  001416                MOV     STIMTBL(R3),R2          ;GET MAXIMUM SPEC TIME
2841  002672  004737  002534                JSR     PC,TYPDEC               ;CONVERT TO DECIMAL & TYPE
2842  002676  000004  001405                TYPE,DASH
2843  002702  016302  001420                MOV     STIMTBL+2(R3),R2        ;GET MINIMUM TIME
2844  002706  004737  002534                JSR     PC,TYPDEC               ;CONVERT TO DECIMAL & TYPE
2845  002712  000004  001412                TYPE,ANGTAB
2846  002716  000004  015073                TYPE,L.ACT
2847  002722  013702  001012                MOV     @#ATIME,R2              ;GET ACTUAL TIME
2848  002726  004737  002534                JSR     PC,TYPDEC               ;CONVERT TO DECIMAL & TYPE
2849  002732  000004  001374                TYPE,CRLF
```

```
2850  002736  012603                        MOV     (SP)+,R3
2851  002740  012602                        MOV     (SP)+,R2
2852  002742  000207                        RTS     PC                      ;RETURN
2853
2854                                          .SBTTL          TYPE GAP TIMES SUBROUTINE
2855                                  ;THIS SUBROUTINE IS USED TO TYPE THE SPECIFIED GAP SIZES (RECORDED IN
2856                                  ;TST021). IT IS CALLED BY THE GAPOK ROUTINE IF THE GAP SIZE IS OUT OF
2857                                  ;RANGE VIA THE HLT ROUTINE (HLT+2).
2858                                  ;CALL:  MOVB    #GAP,GAP                ;LOAD GAP # INTO GAP
2859                                  ;       MOV     #TIME,ATIME             ;LOAD ACTUAL TIME INTO ATIME
2860                                  ;       JSR     PC,OUTGAP
2861
2862  002744  010246                  OUTGAP: MOV     R2,-(SP)                ;SAVE R2 AND R3
2863  002746  010346                          MOV     R3,-(SP)
2864  002750  113703  001120                 MOVB    GAP,R3                  ;GET GAP #
2865  002754  006303                          ASL     R3
2866  002756  006303                          ASL     R3
2867  002760  000004  015063                 TYPE,L.RNG
2868  002764  016302  001556                 MOV     GTIMTBL(R3),R2          ;GET MAX TIME
2869  002770  004737  002534                 JSR     PC,TYPDEC               ;CONVERT TO DECIMAL & TYPE
2870  002774  000004  001405                 TYPE,DASH
2871  003000  016302  001560                 MOV     GTIMTBL+2(R3),R2        ;GET MIN TIME
2872  003004  004737  002534                 JSR     PC,TYPDEC               ;CONVERT TO DECIMAL & TYPE
2873  003010  000004  001412                 TYPE,ANGTAB                     ;TYPE <
2874  003014  000004  015073                 TYPE,L.ACT
2875  003020  013702  001012                 MOV     @#ATIME,R2              ;GET ACTUAL TIME
2876  003024  004737  002534                 JSR     PC,TYPDEC               ;CONVERT TO DECIMAL & TYPE
2877  003030  000004  014552                 TYPE,E.GAP
2878  003034  113702  001120                 MOVB    @#GAP,R2                ;GET GAP #
2879  003040  004737  002426                 JSR     PC,TYPOCT               ;TYPE GAP #
2880  003044  000004  001374                 TYPE,CRLF
2881  003050  012603                          MOV     (SP)+,R3                ;RESTORE R3 AND R2
2882  003052  012602                          MOV     (SP)+,R2
2883  003054  000207                          RTS     PC
2884
2885                                          .SBTTL          ASCII TO OCTAL CONVERT SUBROUTINE
2886                                  ;SUBROUTINE TO CONVERT ASCII DATA TO OCTAL. CONVERTED OCTAL DATA
2887                                  ;IS LEFT IN OCTALO <15-00>.
2888  003056                          CNVTAO:
 (1)  003056  004737  002354                 JSR     PC,.SAVE                ;SAVE REGISTERS ON THE STACK
2889  003062  012700  001264                 MOV     #INBUF,R0               ;SET PTR TO ASCII DATA
2890  003066  012701  001116                 MOV     #OCTALO,R1              ;GET ADDRESS OF OCTAL DATA
2891  003072  005011                          CLR     (R1)                    ;CLEAR OUT OLD OCTAL DATA
2892  003074  005061  000002                 CLR     2(R1)
2893  003100  122710  000015         1$:     CMPB    #CR,(R0)                ;<CR> TERMINATES INPUT
2894  003104  001414                          BEQ     3$
2895  003106  112002                          MOVB    (R0)+,R2                ;GET 'OCTAL' DATA
2896  003110  042702  177770                 BIC     #177770,R2              ;STRIP UNUSED BITS
2897  003114  012703  000003                 MOV     #3,R3                   ;SET SHIFT COUNT
2898  003120  006311                  2$:     ASL     (R1)                    ;SHIFT LAST
2899  003122  006161  000002                 ROL     2(R1)                   ;OCTAL DIGIT
2900  003126  005303                          DEC     R3
2901  003130  001373                          BNE     2$
2902  003132  050211                          BIS     R2,(R1)                 ;AND INSERT THIS DIGIT
2903  003134  000761                          BR      1$                      ;GO GET NEXT DIGIT
2904  003136                          3$:
```

```
  (1)  003136  004737  002376                JSR     PC,.RESTORE             ;RESTORE REGISTERS FROM THE STACK
2905   003142  000207                        RTS     PC                      ;RETURN
2906
2907                                        .SBTTL          PUBLISH SUBROUTINE
2908                                ;THE PUBLISH SUBROUTINE AVERAGES THE RECORDED TIMES FOR EACH TEST IT-
2909                                ;ERATION (IF 16. ITERATIONS) AND PLACES THE AVERAGE RESULT IN 'ATIME'.
2910                                ;IT TYPES THE NAME OF THE FUNCTION THAT WAS TIMED,THE TIME SPEC-
2911                                ;IFICATION AND THE ACTUAL TIME .
2912
2913   003144                       PUBLISH:
  (1)  003144  004737  002354                JSR     PC,.SAVE                ;SAVE REGISTERS ON THE STACK
2914   003150  012700  001014                MOV     #ATIMTBL,R0     ;GET TABLE ADDRESS CONTAINING TIMES
2915   003154  113701  001121                MOVB    @#ITCNT,R1      ;GET # OF ENTRIES (GIVEN BY ITERATION COUNT)
2916   003160  122701  000001                CMPB    #1,R1                   ;BRANCH IF SINGLE ITERATION
2917   003164  001423                        BEQ     4$
2918   003166  005002                        CLR     R2                      ;CLEAR 'SUM' REGISTERS
2919   003170  005003                        CLR     R3
2920   003172  122701  000004                CMPB    #4.,R1                  ;BRANCH IF 4. ITERATIONS
2921   003176  001402                        BEQ     1$
2922   003200  000000                        HALT                            ;ITERATION COUNT MUST BE 1 OR 4.
2923   003202  000777                        BR      .                       ;DO NOT CHANGE POSIT OF SW11
2924                                                                         ;WHEN TEST IS RUNNING.
2925
2926   003204  062002                1$:     ADD     (R0)+,R2                ;SUM INDIVIDUAL TIMES
2927   003206  005503                        ADC     R3
2928   003210  005301                        DEC     R1
2929   003212  001374                        BNE     1$
2930
2931   003214  012700  000002        2$:     MOV     #2,R0
2932   003220  006203                3$:     ASR     R3                      ;SHIFT TIME IN R3 & R2 2 PLACES
2933   003222  006002                        ROR     R2                      ;RIGHT = DIVIDE BY 4.
2934   003224  005300                        DEC     R0
2935   003226  001374                        BNE     3$
2936   003230  010237  001012                MOV     R2,@#ATIME              ;MOVE AVERAGED TIMES
2937
2938   003234  113700  001122        4$:     MOVB    @#TSTNUM,R0             ;GET TEST #
2939   003240  006300                        ASL     R0
2940   003242  016037  001656  003252        MOV     NAMPTR(R0),5$           ;GET TEST NAME STRING ADDRESS
2941   003250  000004                        TYPE
2942   003252  000000                5$:     .WORD   0
2943   003254  113702  001122                MOVB    @#TSTNUM,R2             ;GET TEST #
2944   003260  004737  002650                JSR     PC,OUTSPC               ;OUTPUT TIMES
2945   003264  004737  002376                JSR     PC,.RESTORE             ;RESTORE REGISTERS FROM THE STACK
2946   003270  000207                        RTS     PC
2947
2948                                        .SBTTL          INPUT SUBROUTINE
2949                                ;SUBROUTINE TO GET TTY INPUT
2950                                ;CALL:  JSR     PC,.INPUT
2951                                ;INPUT DATA IS RETUNED IN BUFFER BEGINNING AT INBUF.
2952
2953   003272  010046                .INPUT: MOV     R0,-(SP)        ;SAVE R0 ON THE STACK
2954   003274  012700  001264        1$:     MOV     #INBUF,R0
2955   003300  105737  177560        2$:     TSTB    @#TKS
2956   003304  100375                        BPL     2$
2957
2958   003306  113746  177562                MOVB    @#TKB,-(SP)     ;GET CHARACTER
```

```
2959  003312  042716  000200                BIC     #200,(SP)
2960  003316  122716  000177                CMPB    #177,(SP)       ;CHECK RUBOUT
2961  003322  001004                        BNE     3$
2962  003324  124026                        CMPB    -(R0),(SP)+     ;REMOVE CHARACTER FROM INPUT
2963  003326  000004  001377                TYPE,BKSLSH
2964  003332  000762                        BR      2$              ;WAIT FOR NEXT CHARACTER
2965  003334  122716  000025        3$:     CMPB    #CNTRLU,(SP)    ;CHECK CONTROL U (^U)
2966  003340  001004                        BNE     4$
2967  003342  005726                        TST     (SP)+
2968  003344  000004  001374                TYPE,CRLF
2969  003350  000751                        BR      1$
2970  003352  122716  000003        4$:     CMPB    #CNTRLC,(SP)    ;BRANCH IF NOT CONTROL C
2971  003356  001003                        BNE     40$
2972  003360  000005                        RESET                   ;RESET I/O
2973  003362  000137  005650                JMP     @#INIT          ;RESTART PROGRAM
2974  003366  111637  001401        40$:    MOVB    (SP),@#ECHO
2975  003372  111620                        MOVB    (SP),(R0)+
2976  003374  122726  000015                CMPB    #CR,(SP)+
2977  003400  001403                        BEQ     5$
2978  003402  000004  001401                TYPE,ECHO
2979  003406  000734                        BR      2$
2980  003410  000004  001374        5$:     TYPE,CRLF
2981  003414  012600                        MOV     (SP)+,R0
2982  003416  000207                        RTS     PC
2983
2984                                ;KEYBOARD INTERRUPT SERVICE ROUTINE
2985  003420  113746  177562        TKISR:  MOVB    @#TKB,-(SP)     ;GET TYPED CHARACTER
2986  003424  042716  000200                BIC     #200,(SP)       ;STRIP PARITY BIT
2987  003430  122716  000017                CMPB    #CNTRLO,(SP)    ;BRANCH IF NOT CONTROL O (^O)
2988  003434  001002                        BNE     1$
2989  003436  111637  002121                MOVB    (SP),$CNTRLO    ;SET CONTROL O INDICATOR IN TYPE ROUTINE
2990
2991  003442  122716  000003        1$:     CMPB    #3,(SP)         ;BRANCH IF NOT CONTROL C (^C)
2992  003446  001007                        BNE     2$
2993  003450  023727  000042  012712        CMP     @#42,#$ENDAD    ;INHIBIT ^C IF ACT11 QV OR AA
2994  003456  001403                        BEQ     2$
2995  003460  000005                        RESET
2996  003462  000137  005650                JMP     @#INIT          ;RESTART PROGRAM
2997
2998  003466  122716  000001        2$:     CMPB    #CNTRLA,(SP)    ;BRANCH IF NOT ^A
2999  003472  001011                        BNE     3$
3000  003474  022737  000176  001000        CMP     #SWREG,SWR      ;BRANCH IF HARDWARE SWR IS INVOKED
3001  003502  001010                        BNE     4$
3002  003504  012737  177570  001000        MOV     #177570,SWR     ;INVOKE HARDWARE SWR
3003  003512  000004  013475                TYPE,M.HSWR
3004  003516  122716  000007        3$:     CMPB    #CNTRLG,(SP)    ;BRANCH IF NOT ^G
3005  003522  001005                        BNE     5$
3006  003524  012737  000176  001000  4$:   MOV     #SWREG,SWR      ;INVOKE SOFTWARE SWR
3007  003532  004737  002020                JSR     PC,GTSWR        ;GET NEW SWITCH REGISTER
3008  003536  005726                5$:     TST     (SP)+           ;POP CHARACTER OFF THE STACK
3009  003540  000002                        RTI                     ;RETURN TO CALLER
```

```
                                        .SBTTL          ERROR SERVICE ROUTINES
                                ;ROUTINE TO PROCESS ERROR TRAPS (TRAPS TO 4)
003542  000000                ERRTRP: HALT

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

003544  004737  002354          .HLT:   JSR     PC,..SAVE               ;SAVE REGISTERS ON THE STACK
003550  110637  001123          1$:     MOVB    SP,@#ERFLG              ;SET ERROR FLAG
003554  032777  020000  175216          BIT     #SW13,@SWR              ;BRANCH IF NO TYPOUT
003562  001075                        BNE     4$
003564  000004  014353                TYPE,E.HDR
003570  113702  001122                MOVB    @#TSTNUM,R2             ;GET TEST #
003574  004737  002426                JSR     PC,TYPOCT               ;AND TYPE IT
003600  016600  000016                MOV     16(SP),R0               ;GET RETURN PC
003604  162700  000002                SUB     #2,R0                   ;NOW PC OF HLT CALL
003610  111000                        MOVB    (R0),R0                 ;NOW HLT CALL ITSELF
003612  001417                        BEQ     2$                      ;BRANCH IF HLT
003614  000004  014436                TYPE,E.HDR2
003620  122700  000002                CMPB    #2,R0                   ;BRANCH IF NOT HLT+2
003624  001005                        BNE     10$
003626  004737  002744                JSR     PC,OUTGAP               ;TYPE GAP SPECIFIED TIMES
003632  000004  001374                TYPE,CRLF
003636  000447                        BR      4$
003640  004737  002650          10$:    JSR     PC,OUTSPC               ;TYPE SPECIFIED TIMES
003644  000004  001374                TYPE,CRLF
003650  000442                        BR      4$
003652  016500  000014          2$:     MOV     ER(R5),R0
003656  032765  002300  000032        BIT     #PE1600,TC(R5)
003664  001403                        BEQ     20$
003666  042700  102100                BIC     #102100,R0
003672  000402                        BR      21$
003674  042700  102300          20$:    BIC     #102300,R0
003700  005700                21$:    TST     R0
003702  001003                        BNE     22$
003704  000004  014327                TYPE,E.SFT              ;TYPE SOFT ERROR MESSAGE
003710  000434                        BR      6$

003712  000004  014363          22$:    TYPE,E.HDR1
003716  010500                        MOV     R5,R0                   ;GET FIRST ADDRESS OF REGS.
003720  012701  000007                MOV     #7.,R1                  ;TYPE FIRST 7 REGS.
003724  012002                3$:     MOV     (R0)+,R2                ;GET REG CONTENTS
003726  004737  002426                JSR     PC,TYPOCT               ;AND TYPE IT
003732  000004  001407                TYPE,SPACE2
003736  005301                        DEC     R1
003740  001371                        BNE     3$
003742  016502  000032                MOV     TC(R5),R2               ;GET CONTENTS OF TC REGISTER
003746  004737  002426                JSR     PC,TYPOCT
003752  000004  001374                TYPE,CRLF

003756  032777  001000  175014  4$:     BIT     #SW09,@SWR              ;BRANCH IF NO RING THE BELL
003764  001402                        BEQ     5$
003766  000004  001403                TYPE,BELL
003772  005777  175002          5$:     TST     @SWR                    ;HALT ON ERROR?
```

```
 3067  003776  100001                        BPL     6$
 3068  004000  000000                        HALT
 3069  004002                          6$:
  (1)  004002  004737  002376                JSR     PC,.RESTORE             ;RESTORE REGISTERS FROM THE STACK
 3070  004006  000002                        RTI                             ;RETURN
 3071
 3072
```

```
                                        .SBTTL          SCOPE SUBROUTINE
                                ;SCOPE ROUTINE
                                ;THIS ROUTINE IS ENTERED UPON COMPLETION OF EACH SUBTEST
                                ;THE SCOPE ROUTINE:
                                ;       OUTPUTS TIME SPEC ON EACH ITERATION IF SW08 IS SET
                                ;       REPEATS TEST IF SW14 IS SET
                                ;       STORES ACTUAL TIME FOR FUNCTION IN TIME TABLE (ATIMTBL)
                                ;       PUBLISHES TIME IF SW10=0
                                ;       UPDATES ITERATION COUNT AND IF ITERATIONS COMPLETE CONTINUES
                                ;       TO NEXT TEST, OTHERWISE REPEATS TEST.
                                ;       DELAYS BEFORE CONTINUING OR REPEATING TEST.
                                ;       INITIALIZES DRIVE
                                ;RETURNS:       R5=BASE ADDRESS OF TM03 REGISTERS (ADDRESS OF CS1)
                                ;               R1='DS' REG ADDRESS
                                ;               R0='FC' REG ADDRESS

004010  013705  001010          .SCOPE: MOV     @#TMBASE,R5             ;SET R5 TO FIRST TM REG
004014  032777  000400  174756          BIT     #SW08,@SWR              ;BRANCH IF SPECIFICATION LINE
004022  001404                          BEQ     10$                     ;NOT DESIRED ON EACH ITERATION
004024  113702  001122                  MOVB    @#TSTNUM,R2             ;GET TEST NUMBER
004030  004737  002650                  JSR     PC,OUTSPC               ;OUTPUT TIME  RECORDED
004034  032777  040000  174736  10$:    BIT     #SW14,@SWR              ;BRANCH IF CONTINUOS LOOP
004042  001432                          BEQ     2$                      ;NOT DESIRED
004044  017701  174730          1$:     MOV     @SWR,R1                 ;GET SWITCHES
004050  042701  177740                  BIC     #177740,R1              ;CLEAR ALL BUT TEST #
004054  001406                          BEQ     11$                     ;BRANCH IF ALL SELECTED
004056  020137  001122                  CMP     R1,@#TSTNUM             ;BRANCH IF RUNNING SELECTED TEST
004062  001403                          BEQ     11$
004064  012737  007224  001002          MOV     #TST000,SCPADR          ;RESTART AT TST000
004072  004737  004634          11$:    JSR     PC,DELAY                ;DELAY 350 MS
004076  004737  005120                  JSR     PC,RHINIT               ;INIT
004102  105037  001123                  CLRB    @#ERFLG                 ;CLEAR ERROR FLAG
004106  013716  001002                  MOV     SCPADR,(SP)
004112  010501                          MOV     R5,R1
004114  062701  000012                  ADD     #DS,R1                  ;ADDRESS OF 'DS' REG IS IN R1
004120  010500                          MOV     R5,R0
004122  062700  000006                  ADD     #FC,R0                  ;ADDRESS OF 'FC' REG IS IN R0
004126  000002                          RTI

004130  105737  001123          2$:     TSTB    @#ERFLG                 ;BRANCH IF ERROR FLAG IS SET
004134  001006                          BNE     3$
004136  113700  001121                  MOVB    @#ITCNT,R0              ;GET ITERATION COUNT
004142  006300                          ASL     R0                      ;STORE TIME IN TABLE
004144  013760  001012  001014          MOV     @#ATIME,ATIMTBL(R0)
004152  105237  001121          3$:     INCB    @#ITCNT                 ;INCREMENT ITERATION COUNT
004156  105737  001127                  TSTB    @#PSCNT                 ;INHIBIT ITERATIONS ON
004162  001410                          BEQ     4$                      ;ON FIRST PASS
004164  032777  004000  174606          BIT     #SW11,@SWR              ;BRANCH IF SINGLE ITERATION DESIRED
004172  001004                          BNE     4$
004174  122737  000004  001121          CMPB    #4.,@#ITCNT             ;BRANCH IF ITERATIONS INCOMPLETE
004202  001320                          BNE     1$
004204  032777  000037  174566  4$:     BIT     #37,@SWR        ;IF TEST SELECTED IS TEST 0
004212  001002                          BNE     42$             ;TREAT AS ALL TESTS
004214  011637  001002          40$:    MOV     (SP),@#SCPADR   ;SET SCOPE ADDRESS TO NEXT TEST
004220  032777  002000  174552  42$:    BIT     #SW10,@SWR      ;BRANCH IF NO PUBLICATION DESIRED
004226  001005                          BNE     5$
```

```
3130  004230  005737  005746                TST     CHNFLG                  ;BRANCH IF IN CHAIN MODE
3131  004234  001002                        BNE     5$
3132  004236  004737  0031?4                JSR     PC,PUBLISH              ;GO PUBLISH TEST DATA
3133  004242  105037  001121        5$:     CLRB    @#ITCNT                 ;RESET ITERATION COUNT
3134  004246  000676                        BR      1$
3135
3136                                        .SBTTL  TIMER SUBROUTINES
3137
3138                                ;SUBROUTINE TO SYNCHRONIZE THE TIMER AND TURN IT ON.
3139                                ;REGISTER 4 IS CLEARED, AND THE OSCILLATOR POLARITY IS MONITORED
3140                                ;THE ROUTINE IS EXITED WHEN THE OSCILLATOR POLARITY CHANGES WITH R3
3141                                ;SET TO INDICATE THE POLARITY OF THE OSCILLATOR.
3142                                ;CALL:  JSR     PC,TIMON
3143                                ;RETURNS:       R3 SET TO INDICATE LAST POLARITY (+24/-24=0/1)
3144                                ;               R4 = 0
3145
3146  004250  005004                TIMON:  CLR     R4                      ;CLEAR TIME COUNT
3147  004252  012703  000024                MOV     #24,R3                  ;SET POLARITY TO '0' STATE
3148  004256  032765  000100  000024        BIT     #OSC,MR(R5)             ;BRANCH IF POLARITY IS '0'
3149  004264  001405                        BEQ     2$
3150  004266  032765  000100  000024  1$:   BIT     #OSC,MR(R5)             ;WAIT FOR OSCILLATOR TO RETURN
3151  004274  001374                        BNE     1$
3152  004276  000405                        BR      4$
3153
3154  004300  005403                2$:     NEG     R3                      ;NEGATE PREV POLARITY INDICATOR
3155  004302  032765  000100  000024  3$:   BIT     #OSC,MR(R5)             ;WAIT FOR OSCILLATOR TO RETURN
3156  004310  001774                        BEQ     3$                      ;TO '1' STATE
3157  004312  000207                4$:     RTS     PC
3158
3159                                ;SUBROUTINE TO COUNT TIME
3160                                ;EACH TIME THE OSCILLATOR TOGGLES (BIT <06> IN MR REG) REGISTER
3161                                ;R4 IS INCREMENTED, AND THE REGISTER R3 IS NEGATED TO INDICATE
3162                                ;THE LAST STATE OF THE OSCILLATOR.
3163                                ;CALL   JMP     TIMER(R3)               ;R3 IS SET BY TIMON ROUTINE
3164                                ;       R2=RETURN ADDRESS TO CALLER
3165                                ;NOTE: THE TIME TO EXECUTE THIS ROUTINE IS VERY CRITICAL. IT MUST BE
3166                                ;LESS THAN 40 US.
3167
3168                                ;ENTER HERE VIA JMP  TIMER(R3) WHEN R3=-24 (PREV STATE=1)
3169  004314  032765  000100  000024  TIMER1: BIT     #OSC,MR(R5)             ;BRANCH IF CURRENT STATE IS '0'
3170  004322  001406                        BEQ     TIMER                   ;GO INCREMENT TIME
3171  004324  000112                        JMP     (R2)                    ;RETURN TO TEST
3172
3173          004340                        .=TIMER1+24
3174  004340  005403                TIMER:  NEG     R3                      ;NEGATE PREV STATE INDICATOR
3175  004342  005204                        INC     R4                      ;INCREMENT 'TICK' COUNT
3176  004344  100401                        BMI     TIMERR                  ;BRANCH ON OVERFLOW
3177  004346  000112                        JMP     (R2)                    ;RETURN TO TEST
3178  004350  000004  014464        TIMERR: TYPE,E.TIMOV                    ;TYPE 'TIMER OVERFLOWED'
3179  004354  104400                        HLT                             ;REPORT HARDWARE ERROR
3180  004356  000177  174420                JMP     @SCPADR                 ;RETURN TO BEGINNING OF TEST
3181
3182          004364                        .=TIMER+24
3183                                ;ENTER HERE VIA JMP  TIMER(R3) WHEN R3=+24 (PREV STATE=0)
3184  004364  032765  000100  000024  TIMER0: BIT     #OSC,MR(R5)             ;BRANCH IF CURRENT STATE - '1'
3185  004372  001362                        BNE     TIMER
```

```
3186  004374  000112                          JMP     (R2)
3187
3188                                  ;SUBROUTINE TO CHECK TIME RECORDED BY SUBTEST.
3189                                  ;THIS SUBROUTINE COMPUTES THE ACTUAL TIME (IN MICROSECONDS) AND CHECKS
3190                                  ;THAT THE TIME RECORDED BY THE SUBTEST IS CORRECT BY COMPARING THE TIME
3191                                  ;WITH THE HIGH LIMIT (STIMTBL(R0)) AND THE LOW LIMIT (STIMTBL+2(R0)).
3192                                  ;IF THE TIME IS OUT OF RANGE AN OUT OF RANGE ERROR TYPEOUT RESULTS.
3193                                  ;THE SUBROUTINE IS ENTERED WITH:
3194                                  ;       R4=TICK COUNT
3195
3196  004376                          TIMOK:
 (1)  004376  004737  002354                  JSR     PC,.SAVE                ;SAVE REGISTERS ON THE STACK
3197  004402  012700  000070                  MOV     #56.,R0                 ;GET TIME PER TICK
3198  004406  010401                          MOV     R4,R1                   ;GET TICKS COUNT
3199  004410  005002                          CLR     R2                      ;CLEAR SUMMING REGISTERS
3200  004412  005003                          CLR     R3
3201  004414  060002                  1$:     ADD     R0,R2                   ;MULTIPLY TIME PER TICK
3202  004416  005503                          ADC     R3                      ;BY TICK COUNT
3203  004420  005301                          DEC     R1
3204  004422  001374                          BNE     1$
3205  004424  010246                          MOV     R2,-(SP)                ;DIVIDE COUNT BY 10.
3206
3207  004426  010346                          MOV     R3,-(SP)
3208  004430  012746  000012                  MOV     #10.,-(SP)
3209  004434  004737  004722                  JSR     PC,DIVIDE
3210  004440  005726                          TST     (SP)+                   ;DISCARD REMAINDER
3211  004442  012637  001012                  MOV     (SP)+,@#ATIME           ;STORE QUOTIENT
3212  004446  113700  001122                  MOVB    @#TSTNUM,R0             ;GET TEST #
3213  004452  006300                          ASL     R0
3214  004454  006300                          ASL     R0
3215  004456  023760  001012  001416          CMP     @#ATIME,STIMTBL(R0)     ;CHECK THAT TIME IS WITHIN
3216  004464  101004                          BHI     2$                      ;LIMITS SPECIFIED
3217  004466  023760  001012  001420          CMP     @#ATIME,STIMTBL+2(R0)
3218  004474  101001                          BHI     3$
3219  004476  104401                  2$:     HLT+1                           ;CALL ERROR ROUTINE
3220  004500                          3$:
 (1)  004500  004737  002376                  JSR     PC,.RESTORE             ;RESTORE REGISTERS FROM THE STACK
3221  004504  000207                          RTS     PC                      ;RETURN
3222
3223                                  ;SUBROUTINE TO CHECK INDIVIDUAL GAP TIMES (PRODUCED BY TST021)
3224                                  ;SUBROUTINE COMPUTES THE ACTUAL TIME (IN MICROSECONDS) AND CHECKS
3225                                  ;THAT THE GAP TIME RECORDED BY THE SUBTEST (TST021) BY COMPARING THE
3226                                  ;TIME WITH THE MAX LIMIT (GTIMTBL-GAPTBL(R1)) AND THE MIN LIMIT
3227                                  ;(GTIMTBL+2-GAPTBL(R1)).
3228                                  ;CALL:  MOV     #TICK COUNT,R4          ;R4 CONTAINS TICK COUNT
3229                                  ;       MOVB    #GAP,@#GAP              ;LOCATION GAP CONTAINS GAP #
3230                                  ;       JSR     PC,GAPOK
3231
3232  004506                          GAPOK:
 (1)  004506  004737  002354                  JSR     PC,.SAVE                ;SAVE REGISTERS ON THE STACK
3233  004512  012700  000070                  MOV     #56.,R0                 ;GET TIME PER TICK
3234  004516  010401                          MOV     R4,R1                   ;GET TICK COUNT
3235  004520  005002                          CLR     R2                      ;CLEAR SUMMING REGISTERS
3236  004522  005003                          CLR     R3
3237  004524  060002                  1$:     ADD     R0,R2                   ;MULTIPLY TICK COUNT
3238  004526  005503                          ADC     R3                      ;BY TIME PER TICK
```

```
3239  004530  005301                          DEC     R1
3240  004532  001374                          BNE     1$
3241
3242  004534  010246                          MOV     R2,-(SP)                ;DIVIDE TIME BY 10.
3243  004536  010346                          MOV     R3,-(SP)
3244  004540  012746  000012                  MOV     #10.,-(SP)
3245  004544  004737  004722                  JSR     PC,DIVIDE
3246  004550  005726                          TST     (SP)+                   ;DISCARD REMAINDER
3247  004552  012637  001012                  MOV     (SP)+,@#ATIME           ;STORE QUOTIENT
3248  004556  113703  001120                  MOVB    @#GAP,R3                ;GET GAP #
3249  004562  006303                          ASL     R3                      ;MULTPLY BY 4
3250  004564  006303                          ASL     R3                      ;TO GET AT TABLE ENTRY
3251  004566  023763  001012  001556          CMP     @#ATIME,GTIMTBL(R3)     ;CHECK TIME (MAX)
3252  004574  101004                          BHI     2$
3253  004576  023763  001012  001560          CMP     @#ATIME,GTIMTBL+2(R3)   ;CHECK TIME (MIN)
3254  004604  101002                          BHI     3$
3255  004606  104402                  2$:     HLT+2                           ;REPORT OUT OF RANGE ERROR
3256  004610  000406                          BR      100$
3257  004612  032777  000400  174160  3$:     BIT     #SW08,@SWR              ;BRANCH IF TIMES NOT WANTED
3258  004620  001402                          BEQ     100$
3259  004622  004737  002744                  JSR     PC,OUTGAP               ;TYPE GAP TIMES
3260
3261  004626                          100$:
 (1)  004626  004737  002376                  JSR     PC,.RESTORE             ;RESTORE REGISTERS FROM THE STACK
3262  004632  000207                          RTS     PC                              ;RETURN TO TEST
3263
3264                                          .SBTTL          DELAY SUBROUTINES
3265                                  ;THIS SUBROUTINE CAUSES A DELAY OF 350 MS.
3266  004634  004737  004250          DELAY:  JSR     PC,TIMON
3267  004640  010246                          MOV     R2,-(SP)                ;SAVE R2 ON THE STACK
3268  004642  012702  004652                  MOV     #2$,R2                  ;SET RETURN ADDRESS FOR TIMER
3269  004646                          1$:
 (1)  004646  000163  004340                  JMP     TIMER(R3)               ;GO TO TIMER & RETURN VIA R2
3270  004652  032704  004000          2$:     BIT     #4000,R4
3271  004656  001773                          BEQ     1$
3272  004660  012602                          MOV     (SP)+,R2                ;RESTORE R2
3273  004662  000207                          RTS     PC
3274
3275                                  ;THIS SUBROUTINE ALLOWS A CALLER SPECIFIED DELAY (UP TO 65MS.)
3276                                  ;CALL:  MOV     DELAY TIME,DELTIM       ;LOAD DELAY TIME (IN US)
3277                                  ;       JSR     PC,DELAYV
3278  004664  005737  001114          DELAYV: TST     DELTIM                  ;BRANCH IF 0 DELAY
3279  004670  001413                          BEQ     3$
3280  004672  004737  004250                  JSR     PC,TIMON                ;TURN TIMER ON
3281  004676  010246                          MOV     R2,-(SP)                ;SAVE R2 ON THE STACK
3282  004700  012702  004710                  MOV     #2$,R2                  ;SET RETURN ADDRESS FROM TIMER
3283  004704                          1$:
 (1)  004704  000163  004340                  JMP     TIMER(R3)               ;GO TO TIMER & RETURN VIA R2
3284  004710  023704  001114          2$:     CMP     @#DELTIM,R4
3285  004714  101373                          BHI     1$
3286  004716  012602                          MOV     (SP)+,R2                ;RESTORE R2
3287  004720  000207                  3$:     RTS     PC
3288
3289                                          .SBTTL  DIVIDE SUBROUTINE
3290                                  ;THIS SUBROUTINE DIVIDES A DOUBLE PRECISION # AND RETURNS THE RESULT
3291                                  ;TO THE CALLER ON THE STACK. BOTH DIVIDEND & DIVISOR MUST BE POSITIVE.
```

```
3292                                    ;CALL:  MOV     LEAST SIGNIFICANT HALF DIVIDEND,-(SP)
3293                                    ;       MOV     #MOST SIGNIFICANT HALF DIVIDEND,-(SP)
3294                                    ;       MOV     #DIVISOR,-(SP)
3295                                    ;       JSR     PC,DIVIDE
3296                                    ;RETURN
3297                                    ;       (SP)=REMAINDER ON STACK
3298                                    ;       2(SP)=QUOTIENT
3299
3300                                    ;NOTE:  THIS SUBROUTINE DESTROYS PREVIOUS CONTENTS OF R0,R1,R2 & R3.
3301
3302  004722  005046                    DIVIDE: CLR     -(SP)                   ;SAVE LOC FOR SIGNS
3303  004724  012746  000021                    MOV     #17.,-(SP)              ;SET ITERATION COUNT
3304  004730  016601  000012                    MOV     12(SP),R1               ;GET LSH DIVIDEND
3305  004734  016600  000010                    MOV     10(SP),R0               ;GET MSH DIVIDEND
3306  004740  016602  000006                    MOV     6(SP),R2                ;GET DIVISOR
3307  004744  005402                            NEG     R2                      ;NEGATE DIVISOR
3308  004746  000241                            CLC                             ;CLEAR 'C' BIT IN PSW
3309  004750  000405                            BR      2$
3310  004752  006100                    1$:     ROL     R0                      ;ROTATE MSH DIVIDEND
3311  004754  010003                            MOV     R0,R3                   ;SAVE IN R3
3312  004756  060203                            ADD     R2,R3                   ;SUBTRACT DIVISOR FROM MSH DIVIDEND
3313  004760  103001                            BCC     2$                      ;BRANCH IF DIVIDEND > DIVISOR
3314  004762  010300                            MOV     R3,R0                   ;SAVE REMAINDER IN R0
3315  004764  006101                    2$:     ROL     R1                      ;ROTATE LSH DIVIDEND
3316  004766  005316                            DEC     (SP)                    ;DECREMENT ITERATION COUNT
3317  004770  001370                            BNE     1$
3318  004772  005726                            TST     (SP)+                   ;POP ITERATION COUNTER
3319  004774  005726                            TST     (SP)+                   ;POP SIGN CORRECTION
3320  004776  010166  000006                    MOV     R1,6(SP)                ;PUSH REMAINDER ON STACK
3321  005002  010066  000004                    MOV     R0,4(SP)                ;PUSH QUOTIENT ONTO STACK
3322  005006  012616                            MOV     (SP)+,(SP)
3323  005010  000207                            RTS     PC
3324
3325                                            .SBTTL  DRIVE SUBROUTINES
3326                                    ;SUBROUTINE TO CHECK IF DRIVE IS AVAILABLE
3327                                    ;CALL:  MOVB    DRIVE#,DRVNUM
3328                                    ;       JSR     PC,DRVAVA
3329                                    ;RETURN:        'C' BIT SET IF NOT AVAILABLE
3330  005012  113765  001004  000010    DRVAVA: MOVB    @#DRVNUM,CS2(R5)        ;LOAD DRIVE #
3331  005020  032765  040000  000026            BIT     #TAP,DT(R5)             ;CHECK IF TAPE UNIT
3332  005026  001003                            BNE     1$
3333  005030  004737  005120                    JSR     PC,RHINIT
3334  005034  000262                            SEV                             ;SET 'V' TO IND NOT AVAIL
3335  005036  000207                    1$:     RTS     PC                      ;RETURN
3336
3337                                    ;SUBROUTINE TO CHECK IF TU45 SLAVE IS AVAILABLE FOR TEST
3338                                    ;CALL:  MOVB    DRIVE #,@#DRVNUM        ;PASS DRIVE # VIA DRVNUM
3339                                    ;       MOVB    SLAVE #,@#SLVNUM        ;PASS SLAVE # VIA SLVNUM
3340                                    ;       JSR     PC,SLVAVA               ;CALL SUBROUTINE
3341  005040  113765  001004  000010    SLVAVA: MOVB    @#DRVNUM,CS2(R5)        ;LOAD DRIVE #
3342  005046  113765  001005  000032            MOVB    @#SLVNUM,TC(R5)         ;AND SLAVE #
3343  005054  032765  002000  000026            BIT     #SPR,DT(R5)             ;BRANCH IF SLAVE NOT PRESENT
3344  005062  001414                            BEQ     1$
3345  005064  032765  010000  000012            BIT     #MOL,DS(R5)             ;BRANCH IF ON LINE
3346  005072  001011                            BNE     2$                      ;
3347  005074  116037  001132  014234            MOVB    DIGTAB(R0),@#E.NSL1     ;SET DRIVE # IN MESSAGE
```

```
3348  005102  116137  001132  014245          MOVB    DIGTAB(R1),@#E.MSG      ;SET SLAVE # IN MESSAGE
3349  005110  000004  014226                  TYPE,E.NDR1                     ;PRINT DRIVE #,SLAVE # NOT
3350                                                                          ;ON LINE
3351  005114  000262                  1$:     SEV                             ;SET 'V' TO INDICATE NO SLAVE
3352  005116  000207                  2$:     RTS     PC
3353
3354                                  ;SUBROUTINE TO  INITIALIZE RH CONTROLLER
3355                                  ;CALL:  JSR PC,RHINIT
3356
3357  005120  012765  000040  000010  RHINIT: MOV     #40,CS2(R5)
3358  005126  113765  001004  000010          MOVB    @#DRVNUM,CS2(R5)
3359  005134  005046                          CLR     -(SP)
3360  005136  113716  001005                  MOVB    @#SLVNUM,(SP)
3361  005142  012665  000032                  MOV     (SP)+,TC(R5)            ;LOAD SLAVE # INTO TC REG
3362  005146  052765  001700  000032          BIS     #NORM11,TC(R5)
3363  005154  000207                          RTS     PC
3364
3365                                  ;SUBROUTINE TO WAIT FOR DRIVE READY (DRY)
3366  005156  005027                  WAITRDY:CLR     (PC)+                   ;CLEAR WAIT TIMER
3367  005160  000000                  WAITTIM:.WORD   0
3368  005162  105765  000012          1$:     TSTB    DS(R5)                  ;WAIT FOR READY TO SET
3369  005166  100406                          BMI     2$
3370  005170  005237  005160                  INC     WAITTIM                 ;INCREMENT WAIT TIMER
3371  005174  001372                          BNE     1$                      ;BRANCH IF TIME HAS NOT EXPIRED
3372  005176  000004  014511                  TYPE,E.TIMEXP                   ;TYPE 'TIME EXPIRED WAITING FOR RDY'
3373  005202  000425                          BR      99$                     ;TAKE ERROR EXIT
3374  005204  032765  002000  000012  2$:     BIT     #EOT,DS(R5)             ;CHECK FOR END OF TAPE
3375  005212  001415                          BEQ     3$                      ;BRANCH IF NO EOT
3376  005214  000004  013322                  TYPE,M.NAM
3377  005220  000004  014035                  TYPE,M.EOT                      ;TYPE 'END OF TAPE'
3378  005224  004737  005262                  JSR     PC,.REWIND              ;REWIND SLAVE
 (1)  005230  102412                          BVS     99$                     ;BRANCH IF ERROR ON REWIND
3379  005232  004737  005344                  JSR     PC,WRITE                ;WRITE A RECORD
3380  005236  005215                          INC     (R5)                    ;SET 'GO' BIT
3381  005240  004737  005156                  JSR     PC,WAITRDY              ;WAIT FOR READY
3382  005244  000404                          BR      99$                     ;TAKE ERROR EXIT
3383  005246  032765  040000  000012  3$:     BIT     #ERR,DS(R5)             ;CHECK ERROR EXIT
3384  005254  001401                          BEQ     100$
3385  005256  000262                  99$:    SEV
3386  005260  000207                  100$:   RTS     PC
3387                                  ;SUBROUTINE TO REWIND A UNIT (DRIVE/SLAVE COMBINATION)
3388                                  ;CALL   MOVB    DRIVE #,@#DRVNUM
3389                                  ;       MOVB    SLAVE #,@#SLVNUM
3390                                  ;       JSR     PC,.REWIND
3391                                  ;SUBROUTINE RETURNS TO CALLER WITH SELECTED SLAVE AT 'BOT', & 'V' SET IF
3392                                  ;AN ERROR OCCURS.
3393
3394  005262  004737  005120  .REWIND:JSR     PC,RHINIT               ;INITIALIZE CONTROLLER
3395  005266  004337  005500          JSR     R3,TMCMD                ;GO TO TM COMMAND SUBROUTINE
3396  005272  000000                  .WORD   0                       ;BUS ADDRESS (NOT USED)
3397  005274  000000                  .WORD   0                       ;WORD COUNT (NOT USED)
3398  005276  000000                  .WORD   0                       ;FRAME COUNT (NOT USED)
3399  005300  000006                  .WORD   RWD                     ;REWIND COMMAND
3400  005302  005215                  INC     (R5)                    ;SET 'GO' BIT
3401  005304  032765  000002  000012  1$:     BIT     #BOT,DS(R5)             ;BRANCH IF 'BOT' SET
3402  005312  001005                          BNE     2$
```

```
3403  005314  032765  040000  000012          BIT     #ERR,DS(R5)             ;CHECK ERROR BIT
3404  005322  001006                          BNE     99$                     ;BRANCH IF ERROR BIT SET
3405  005324  000767                          BR      1$
3406
3407  005326  032765  020000  000012  2$:     BIT     #PIP,DS(R5)             ;WAIT FOR TAPE MOTION TO STOP
3408  005334  001374                          BNE     2$
3409  005336  000401                          BR      100$
3410  005340  000262                  99$:    SEV
3411  005342  000207                  100$:   RTS     PC
3412
3413                                  ;SUBROUTINE TO WRITE 256. WORD RECORD
3414                                  ;CALL:  JSR     PC,WRITE
3415
3416  005344  004337  005500          WRITE:  JSR     R3,TMCMD                ;GO TO TM COMMAND SUBROUTINE
3417  005350  016046                          .WORD   WTBUF                   ;BUS ADDRESS
3418  005352  177600                          .WORD   WRDCNT                  ;WORD COUNT
3419  005354  177400                          .WORD   FRMCNT                  ;FRAME COUNT
3420  005356  000060                          .WORD   WFWD                    ;WRITE FORWARD COMMAND
3421  005360  000207                          RTS     PC
3422
3423                                  ;SUBROUTINE TO READ A 256. WORD RECORD.
3424                                  ;CALL:  JSR     PC,READ
3425
3426  005362  004337  005500          READ:   JSR     R3,@#TMCMD
3427  005366  016046                          .WORD   RDBUF                   ;ADDRESS OF READ BUFFER
3428  005370  177600                          .WORD   WRDCNT                  ;2'S COMPLEMENT OF WORD COUNT
3429  005372  177400                          .WORD   FRMCNT                  ;2'S COMPLEMENT OF FRAME COUNT
3430  005374  000070                          .WORD   RDFWD                   ;READ FORWARD COMMAND
3431  005376  000207                          RTS     PC
3432
3433                                  ;SUBROUTINE TO INITIATE READ REVERSE COMMAND
3434                                  ;CALL:  JSR     PC,REVRD
3435                                  ;
3436  005400  004337  005500          REVRD:  JSR     R3,TMCMD
3437  005404  016446                          .WORD   RDBUF+256.              ;ADDRESS OF READ REVERSE BUFFER
3438  005406  177600                          .WORD   WRDCNT                  ;2'S COMPLEMENT OF WORD COUNT
3439  005410  177400                          .WORD   FRMCNT                  ;2'S COMPLEMENT OF FRAME COUNT
3440  005412  000076                          .WORD   RDREV                   ;READ REVERSE COMMAND
3441  005414  000207                          RTS     PC
3442
3443                                  ;SUBROUTINE TO SPACE FORWARD 1 RECORD
3444  005416  012765  177777  000006  FWDSPC: MOV     #-1,FC(R5)              ;LOAD RECORD COUNT
3445  005424  012715  000031                  MOV     #SPCFWD+1,(R5)          ;LOAD COMMAND
3446  005430  004737  005156                  JSR     PC,WAITRDY              ;WAIT FOR READY
3447  005434  000207                          RTS     PC                      ;RETURN
3448
3449                                  ;SUBROUTINE TO WRITE A RECORD AND BACK SPACE OVER  THE RECORD.
3450  005436  004737  005344          WRT.BK: JSR     PC,WRITE                ;WRITE THE RECORD
3451  005442  005215                          INC     (R5)                    ;SET 'GO' BIT
3452  005444  004737  005156                  JSR     PC,WAITRDY
3453  005450  102412                          BVS     2$
3454  005452  012765  177777  000006          MOV     #-1,FC(R5)              ;LOAD RECORD COUNT
3455  005460  012715  000033                  MOV     #SPCREV+1,(R5)          ;LOAD COMMAND
3456  005464  004737  005156                  JSR     PC,WAITRDY
3457  005470  102402                          BVS     2$
3458  005472  004737  004634          1$:     JSR     PC,DELAY                ;WAIT FOR TAPE MOTION TO STOP
```

```
3459  005476  000207                  2$:     RTS     PC
3460
3461                                  ;SUBROUTINE TO LOAD A COMMAND
3462                                  ;CALL:  JSR     R3,TMCMD
3463                                  ;       .WORD   BUS ADDRESS
3464                                  ;       .WORD   WORD COUNT (2'S COMPLEMENT)
3465                                  ;       .WORD   FRAME COUNT (2'S COMPLEMENT)
3466                                  ;       .WORD   COMMAND
3467
3468  005500  012365  000004          TMCMD:  MOV     (R3)+,BA(R5)            ;LOAD BUS ADDRESS
3469  005504  012365  000002                  MOV     (R3)+,WC(R5)            ;LOAD WORD COUNT
3470  005510  012365  000006                  MOV     (R3)+,FC(R5)            ;LOAD FRAME COUNT
3471  005514  012315                          MOV     (R3)+,(R5)              ;LOAD COMMAND
3472  005516  000203                          RTS     R3                      ;RETURN
3473
3474                                  ;SUBROUTINE TO PRINT TU45 SERIAL NUMBER
3475                                  ;JSR     PC,SNPT
3476
3477  005520  016503  000030          SNPT:   MOV     SN(R5),R3
3478  005524  012701  001144                  MOV     #ODIGITS,R1
3479  005530  000303                          SWAB    R3
3480  005532  006003                          ROR     R3
3481  005534  006003                          ROR     R3
3482  005536  006003                          ROR     R3
3483  005540  006003                          ROR     R3                      ;GET FIRST DIGIT
3484  005542  042703  177760                  BIC     #177760,R3
3485  005546  052703  000260                  BIS     #260,R3
3486  005552  110321                          MOVB    R3,(R1)+                ;FILL FIRST DIGIT
3487  005554  016503  000030                  MOV     SN(R5),R3
3488  005560  000303                          SWAB    R3
3489  005562  042703  177760                  BIC     #177760,R3
3490  005566  052703  000260                  BIS     #260,R3
3491  005572  110321                          MOVB    R3,(R1)+                ;GET SECOND DIGIT
3492  005574  016503  000030                  MOV     SN(R5),R3
3493  005600  006003                          ROR     R3
3494  005602  006003                          ROR     R3
3495  005604  006003                          ROR     R3
3496  005606  006003                          ROR     R3
3497  005610  042703  177760                  BIC     #177760,R3
3498  005614  052703  000260                  BIS     #260,R3
3499  005620  110321                          MOVB    R3,(R1)+                ;GET THIRD DIGIT
3500  005622  016503  000030                  MOV     SN(R5),R3
3501  005626  042703  177760                  BIC     #177760,R3
3502  005632  052703  000260                  BIS     #260,R3
3503  005636  110321                          MOVB    R3,(R1)+                ;GET FOURTH DIGIT
3504  005640  105011                          CLRB    (R1)
3505  005642  000004  001144                  TYPE,ODIGITS                    ;TYPE SERIAL NUMBER
3506  005646  000207                          RTS     PC                      ;RETURN
3507
```

```
 3509                                           .SBTTL  PROGRAM INITIALIZATION
 3510  005650  012706  000600          INIT:    MOV     #STKPTR,SP              ;SET STACK PTR
 3511  005654  005037  001264                   CLR     @#INBUF
 3512
 3513  005660  013746  000006                   MOV     @#6,-(SP)               ;SAVE VECTORS
 3514  005664  013746  000004                   MOV     @#4,-(SP)
 3515  005670  012737  005710  000004           MOV     #61$,@#4                        ;SET UP FOR TIMEOUT
 3516  005676  022777  177777  173074           CMP     #-1,@SWR                ;REFERENCE HARDWARE SWITCH REGISTER
 3517  005704  001402                           BEQ     60$
 3518  005706  000404                           BR      62$
 3519  005710  022626                   61$:    CMP     (SP)+,(SP)+             ;ADJUST STACK
 3520  005712  012737  000176  001000   60$:    MOV     #SWREG,SWR              ;POINT TO SOFTWARE SWITCH REG
 3521  005720  012637  000004           62$:    MOV     (SP)+,@#4               ;RESTORE VECTORS
 3522  005724  012637  000006                   MOV     (SP)+,@#6
 3523  005730  105037  001124                   CLRB    @#PRGFLG                ;CLEAR PROGRAM FLAG
 3524  005734  105037  001130                   CLRB    @#ASFLG                 ;CLEAR ASK FLAG
 3525  005740  105037  001127                   CLRB    @#PSCNT         ;SET PASS COUNT = 0
 3526  005744  005027                           CLR     (PC)+           ;;CLEAR CHAIN INDICATOR
  (1)  005746  000000                   CHNFLG: .WORD   0               ;;CHAIN MODE INDICATOR
  (1)                                           ;;1/0 = CHAIN/NOT CHAIN MODE
  (1)  005750  022737  012712  000042           CMP     #$ENDAD,@#42    ;;BRANCH IF LOADED VIA ACT11 CHAIN MODE
  (1)  005756  001404                           BEQ     50$
  (1)  005760  005737  000042                   TST     @#42            ;;BRANCH IF IN DUMP MODE
  (1)  005764  001413                           BEQ     52$
  (1)  005766  000406                           BR      51$
  (1)  005770  012737  000176  001000   50$:    MOV     #SWREG,SWR      ;;INVOKE SOFTWARE SWR
  (1)  005776  012777  100000  172774           MOV     #100000,@SWR    ;;WITH HALT ON ERROR SET
  (1)  006004  005237  005746           51$:    INC     CHNFLG          ;;SET CHNFLG = CHAIN MODE
  (1)  006010  000137  006106                   JMP     5$              ;;GO TO CHAIN ADDRESS
  (1)  006014                           52$:
 3527  006014  122737  000006  000041           CMPB    #6,@#41         ;BRANCH IF NOT LOADED VIA TMDP
 3528  006022  001002                           BNE     1$
 3529  006024  000004  013525                   TYPE,I.REM              ;ADVISE USER TO REMOVE TMDP
 3530  006030  000004  013322           1$:     TYPE,M.NAM                      ;TYPE TITLE
 3531  006034  105037  013322                   CLRB    M.NAM                   ;DO NOT TYPE TITLE ON RESTART
 3532  006040  000004  013572                   TYPE,I.REG                      ;ASK USER TO TYPE CONT BASE ADRS
 3533  006044  013702  001010                   MOV     @#TMBASE,R2             ;GET CURRENT CONT BASE ADDRESS
 3534  006050  004737  002426                   JSR     PC,TYPOCT               ;AND TYPE IT
 3535  006054  000004  001410                   TYPE,SPACE
 3536  006060  004737  003272                   JSR     PC,.INPUT               ;GET USER INPUT
 3537  006064  122737  000015  001264           CMPB    #CR,@#INBUF             ;DO NOT CHANGE CURRENT VALUE
 3538  006072  001405                           BEQ     5$                      ;IF USER TYPES <CR>
 3539  006074  004737  003056           4$:     JSR     PC,CNVTAO               ;CONVERT ASCII TO OCTAL
 3540  006100  013737  001116  001010           MOV     @#OCTALO,@#TMBASE       ;SET NEW ADDRESS
 3541  006106  013705  001010           5$:     MOV     @#TMBASE,R5
 3542
 3543                                   ;ROUTINE TO CHECK IF CONTROLLER (RH11) IS AVAILAABLE
 3544  006112  000261                           SEC                             ;SET 'C' IN PSW
 3545  006114  005715                           TST     (R5)                    ;BRANCH IF CONTROLLER AVAIL
 3546  006116  103003                           BCC     6$
 3547  006120  000004  014074                   TYPE,E.NCON
 3548  006124  000651                           BR      INIT
 3549  006126  012737  003542  000004   6$:     MOV     #ERRTRP,@#ERRVEC        ;SET ERROR TRAP VECTOR
```

```
3551                                    ;ROUTINE TO GET TM03 DRIVES USER DESIRES TO TEST
3552  006134  105037  001123          DRIVES: CLRB    @#ERFLG                 ;CLEAR ERROR FLAG
3553  006140  012701  001154                  MOV     #DRVTBL,R1              ;MARK ALL DRIVES AS NOT TO
3554  006144  012700  000004                  MOV     #4,R0                   ;BE TESTED. A '0' INDICATES
3555  006150  005021                  1$:     CLR     (R1)+                   ;THAT A DRIVE IS NOT TO BE
3556  006152  005300                          DEC     R0                      ;TESTED
3557  006154  001375                          BNE     1$
3558  006156  005737  005746                  TST     CHNFLG                  ;BRANCH IF IN CHAIN MODE
3559  006162  001014                          BNE     2$
3560  006164  000004  013637                  TYPE,I.DRVS
3561  006170  004737  003272                  JSR     PC,.INPUT               ;GET USER INPUT
3562  006174  012700  001264                  MOV     #INBUF,R0
3563  006200  122710  000101                  CMPB    #'A,(R0)                ;IF USER RESPONDS WITH 'A' OR
3564  006204  001403                          BEQ     2$                      ;<CR> THEN ALL AVAILABLE DRIVES
3565  006206  122710  000015                  CMPB    #CR,(R0)                ;ARE TO BE TESTED
3566  006212  001013                          BNE     4$
3567  006214  110637  001124          2$:     MOVB    SP,PRGFLG               ;SET FLAG TO IND ALL DRIVES
3568  006220  012701  001154                  MOV     #DRVTBL,R1              ;MARK ALL DRIVES TO BE TESTED
3569  006224  012700  000004                  MOV     #4,R0                   ;A '-1' INDICATES THAT A DRIVE
3570  006230  012721  177777          3$:     MOV     #-1,(R1)+               ;IS TO BE TESTED
3571  006234  005300                          DEC     R0
3572  006236  001374                          BNE     3$
3573  006240  000417                          BR      CHKDRV                  ;GO CHECK DRIVE AVAILABILITY
3574
3575                                    ;GET USER SELECTED DRIVES AND MARK EACH DRIVE SELECTED TO BE TESTED
3576  006242  122710  000015          4$:     CMPB    #CR,(R0)
3577  006246  001414                          BEQ     CHKDRV
3578  006250  121027  000054                  CMPB    (R0),#',                ;CHECK IF 'COMMA'
3579  006254  001001                          BNE     5$
3580  006256  105720                          TSTB    (R0)+                   ;STEP PTR PAST 'COMMA'
3581  006260  112001                  5$:     MOVB    (R0)+,R1
3582  006262  042701  177770                  BIC     #177770,R1
3583  006266  112761  177777  001154          MOVB    #-1,DRVTBL(R1)
3584  006274  000240                          NOP
3585  006276  000761                          BR      4$
3586
3587                                    ;ASCERTAIN THAT DRIVES (TM03'S) SPECIFIED ARE AVAILABLE
3588  006300  005000                  CHKDRV: CLR     R0                      ;A (0) IN DRVTBL(R0) INDICATES
3589  006302  105760  001154          1$:     TSTB    DRVTBL(R0)              ;THE DRIVE IS NOT TO BE TESTED
3590  006306  001005                          BNE     3$                      ;A '1' INDICATES TO BE TESTED
3591  006310  005200                  2$:     INC     R0
3592  006312  122700  000010                  CMPB    #8.,R0
3593  006316  001371                          BNE     1$
3594  006320  000424                          BR      5$
3595  006322  110037  001004          3$:     MOVB    R0,@#DRVNUM             ;GET DRIVE #
3596  006326  004737  005012                  JSR     PC,@#DRVAVA             ;AND CHECK IF AVAILABLE
3597  006332  102366                          BVC     2$                      ;'V' BIT SET INDICATES NOT AVAIL
3598  006334  105737  001124                  TSTB    @#PRGFLG                ;DO NOT TYPE NOT AVAILABLE
3599  006340  001011                          BNE     4$                      ;MESSAGE IF ALL SELECTED
3600  006342  000004  014141                  TYPE,E.NDRV
3601  006346  116037  001132  014173          MOVB    DIGTAB(R0),@#E.NAVA     ;SET DRIVE # IN MESSAGE
3602  006354  000004  014173                  TYPE,E.NAVA
3603  006360  110637  001123                  MOVB    SP,@#ERFLG              ;SET 'ERROR' FLAG
3604  006364  105060  001154          4$:     CLRB    DRVTBL(R0)              ;MARK DRIVE UNAVAILABLE
3605  006370  000747                          BR      2$                      ;CHECK NEXT DRIVE
3606  006372  105737  001123          5$:     TSTB    @#ERFLG                 ;GO GET SLAVES IF NO ERROR
```

```
3607  006376  001256                          BNE     DRIVES                  ;ELSE ASK USER TO RETYPE DRIVES
3608
3609                                  ;ROUTINE TO GET SLAVES (TU45'S) USER DESIRES TO TEST
3610  006400  105037  001123          SLAVES: CLRB    @#ERFLG         ;CLEAR ERROR INDICATOR
3611  006404  012701  001164                  MOV     #SLVTBL,R1              ;MARK ALL SLAVES (64.) AS NOT
3612  006410  012700  000040                  MOV     #32.,R0                 ;TO BE TESTED.A 0 INDICATES THAT
3613  006414  005021                  1$:     CLR     (R1)+                   ;A DRIVE'S SLAVE IS NOT TO BE
3614  006416  005300                          DEC     R0                      ;TESTED
3615  006420  001375                          BNE     1$
3616  006422  012701  001164                  MOV     #SLVTBL,R1              ;R1 POINTS TO DRIVE'S SLAVE
3617  006426  105760  001154          2$:     TSTB    DRVTBL(R0)              ;BRANCH IF DRIVE IS TO BE TESTED
3618  006432  001007                          BNE     4$                      ;& IS AVAILABLE
3619  006434  062701  000010          3$:     ADD     #8.,R1                  ;STEP SLAVE PTR TO NEXT DRIVE'S
3620  006440  005200                          INC     R0                      ;SLAVES AND INCREMENT DRIVE #
3621  006442  122700  000010                  CMPB    #8.,R0                  ;CHECK ALL DRIVES
3622  006446  001367                          BNE     2$                      ;AND WHEN ALL DRIVES CHECKED
3623  006450  000457                          BR      CHKSLV                  ;GO CHECK SLAVE AVAILABILITY
3624
3625  006452  105737  001124          4$:     TSTB    @#PRGFLG                ;BRANCH IF USER SELECTED ALL
3626  006456  001021                          BNE     5$                      ;DRIVES
3627  006460  110037  001004                  MOVB    R0,DRVNUM               ;GET DRIVE #
3628  006464  116037  001132  013726          MOVB    DIGTAB(R0),@#I.DRV      ;PREPARE USER ACTION MESSAGE
3629  006472  000004  013707                  TYPE,I.SLVS
3630  006476  004737  003272                  JSR     PC,.INPUT               ;GET USER INPUT
3631  006502  012703  001264                  MOV     #INBUF,R3               ;SET PTR TO USER INPUT
3632  006506  122713  000101                  CMPB    #'A,(R3)                ;AN 'A' OR <CR> AS FIRST CHAR
3633  006512  001403                          BEQ     5$                      ;INDICATES TEST ALL SLAVES
3634  006514  122713  000015                  CMPB    #CR,(R3)
3635  006520  001015                          BNE     7$
3636  006522  110637  001124          5$:     MOVB    SP,@#PRGFLG             ;SET 'ALL' INDICATOR
3637  006526  012701  001164                  MOV     #SLVTBL,R1              ;MARK ALL SLAVES FOR ALL
3638  006532  012700  000040                  MOV     #32.,R0                 ;DRIVES AS TO BE TESTED
3639  006536  012721  177777          6$:     MOV     #-1,(R1)+
3640  006542  005300                          DEC     R0
3641  006544  001374                          BNE     6$
3642  006546  105737  001124                  TSTB    @#PRGFLG                ;BRANCH IF ALL WAS SELECTED
3643  006552  001016                          BNE     CHKSLV
3644
3645  006554  122713  000015          7$:     CMPB    #CR,(R3)                ;GET USER SELECTED SLAVES FOR
3646  006560  001725                          BEQ     3$                      ;DRIVE
3647  006562  121327  000054                  CMPB    (R3),#',                ;STEP PTR PAST 'COMMA'
3648  006566  001001                          BNE     8$
3649  006570  105723                          TSTB    (R3)+
3650  006572  112304                  8$:     MOVB    (R3)+,R4                ;AND MARK SELECED SLAVE
3651  006574  042704  177770                  BIC     #177770,R4              ;AS TO BE TESTED
3652  006600  060104                          ADD     R1,R4
3653  006602  112714  177777                  MOVB    #-1,(R4)
3654  006606  000762                          BR      7$
3655
3656                                  ;ASCERTAIN THAT SLAVES (TU45'S) SELECTED ARE AVAILABLE
3657  006610  005000          CHKSLV: CLR     R0                      ;R0 WILL CONTAIN THE DRIVE #
3658  006612  005100                  COM     R0                      ;COMPLEMENT DRIVE #
3659  006614  012702  001164          MOV     #SLVTBL,R2              ;SET PTR TO SLAVE TABLE
3660  006620  005200                  1$:     INC     R0              ;INCREMENT DRIVE NUMBER
3661  006622  022700  000010                  CMP     #8.,R0          ;CHECK FOR END
3662  006626  001445                          BEQ     8$              ;IF SO:BR
```

```
3663  006630  105760  001154                  TSTB    DRVTBL(R0)              ;BRANCH IF DRIVE SELECTED
3664  006634  001003                          BNE     3$                      ;AND AVAILABLE FOR TEST
3665  006636  062702  000010                  ADD     #8.,R2                  ;ELSE STEP SLAVE PTR TO
3666  006642  000766                          BR      1$                      ;NEXT DRIVES SLAVES AND CHECK
3667                                                                           ;FOR DRIVE
3668
3669  006644  005001                  3$:     CLR     R1                      ;SET SLAVE # 0
3670  006646  105712                  4$:     TSTB    (R2)                    ;BRANCH IF DRIVE'S SLAVE IS SEL-
3671  006650  001006                          BNE     6$                      ;ECTED FOR TEST
3672  006652  005201                  5$:     INC     R1                      ;INCREMENT SLAVE #
3673  006654  005202                          INC     R2                      ;STEP PTR TO NEXT SLAVE
3674  006656  022701  000010                  CMP     #8.,R1                  ;GO TO 4$ IF ALL SLAVES NOT
3675  006662  001371                          BNE     4$                      ;CHECKED
3676  006664  000755                          BR      1$                      ;OTHERWISE GO TO 1$ ABOVE
3677
3678  006666  110037  001004          6$:     MOVB    R0,@#DRVNUM             ;PASS DRIVE & SLAVE #
3679  006672  110137  001005                  MOVB    R1,@#SLVNUM
3680  006676  004737  005040                  JSR     PC,@#SLVAVA             ;AND CHECK IF AVAILABLE
3681  006702  102363                          BVC     5$                      ;'V' SET INDCATES ERROR
3682  006704  105737  001124                  TSTB    @#PRGFLG                ;DO NOT TYPE ERROR MSG IF ALL
3683  006710  001012                          BNE     7$                      ;SLAVES SELECTED
3684  006712  116037  001132  014163          MOVB    DIGTAB(R0),@#E.DRV      ;ICATES ERROR. PREPARE ERROR
3685  006720  116137  001132  014173          MOVB    DIGTAB(R1),@#E.NAVA     ;MESSAGE
3686  006726  000004  014155                  TYPE,E.NSLV
3687  006732  110637  001123                  MOVB    SP,@#ERFLG              ;SET ERROR INDICATOR
3688  006736  105012                  7$:     CLRB    (R2)                    ;CLEAR SLAVE TABLE ENTRY
3689  006740  000744                          BR      5$                      ;GET NEXT SLAVE
3690
3691  006742  105737  001123          8$:     TSTB    @#ERFLG                 ;BRANCH IF ERROR
3692  006746  001214                          BNE     SLAVES                  ;ASK USER TO RETYPE SLAVES
3693  006750  012737  003542  000004  100$:   MOV     #ERRTRP,@#ERRVEC
3694
3695                                          ;SCAN DRIVE AND SLAVE TABLE FOR DRIVE/SLAVE COMBINATION TO TEST.
3696                                          ;RESTART ADDRESS--PROGRAM STARTS HERE WHEN START ADDRESS = 210 AND
3697                                          ;AFTER EACH PASS
3698  006756  105037  001004          RSTRT:  CLRB    @#DRVNUM                ;SET DRIVE AND SLAVE # 0
3699  006762  105037  001005                  CLRB    @#SLVNUM
3700  006766  012737  001164  001006          MOV     #SLVTBL,@#SLVPTR        ;SET PTR TO SLAVE TABLE
3701  006774  105037  001125                  CLRB    @#UNTFND                ;CLEAR 'UNIT FOUND' IND.
3702
3703                                          ;PROGRAM RESTARTS HERE AFTER EACH DRIVE/SLAVE HAS BEEN TESTED.
3704  007000  113700  001004          BEGIN:  MOVB    @#DRVNUM,R0             ;GET DRIVE #
3705  007004  113701  001005                  MOVB    @#SLVNUM,R1             ;AND SLAVE #
3706  007010  013702  001006                  MOV     @#SLVPTR,R2             ;GET SLAVE PTR
3707  007014  122737  000006  000041          CMPB    #6,@#41                 ;BRANCH IF LOADED VIA TMDP
3708  007022  001004                          BNE     1$
3709  007024  005737  001004                  TST     @#DRVNUM                ;TEST FOR DRIVE#0,SLAVE#0
3710  007030  001001                          BNE     1$                      ;IF NOT : BR
3711  007032  105012                          CLRB    (R2)                    ;SET DRIVE #0,SLAVE #0 NOT TO
3712                                                                           ;BE TESTED.
3713  007034  105760  001154          1$:     TSTB    DRVTBL(R0)              ;BRANCH IF DRIVE AVAIL TO TEST
3714  007040  001011                          BNE     3$
3715  007042  005001                          CLR     R1                      ;CLEAR SLAVE #
3716  007044  062702  000010                  ADD     #8.,R2                  ;AND STEP PTR TO NEXT DRIVE'S
3717  007050  005200                  2$:     INC     R0                      ;SLAVES AND INCREMENT DRIVE #
3718  007052  022700  000010                  CMP     #8.,R0                  ;EXIT TEST IF ALL DRIVES
```

```
3719  007056  001366                  BNE     1$                      ;CHECKED OTHERWISE CONTINUE
3720  007060  000137  012632          JMP     @#END                   ;SCAN FOR NEXT 'UNIT'
3721
3722  007064  105712          3$:     TSTB    (R2)                    ;BRANCH IF SLAVE ON DRIVE IS
3723  007066  001007                  BNE     4$                      ;AVAILABLE THERWISE STEP
3724  007070  005202                  INC     R2                      ;PTR TO NEXT SLAVE
3725  007072  005201                  INC     R1                      ;INCREMENT SLAVE #
3726  007074  122701  000010          CMPB    #8.,R1                  ;UNTIL ALL SLAVES CHECKED
3727  007100  001371                  BNE     3$                      ;WHEN ALL SLAVES CHECKED
3728  007102  005001                  CLR     R1                      ;SET SLAVE # 0
3729  007104  000761                  BR      2$                      ;AND CONTINUE SCAN
3730
3731  007106  110637  001125  4$:     MOVB    SP,@#UNTFND             ;INDICATE THAT A 'UNIT' IS FOUND
3732  007112  110037  001004          MOVB    R0,@#DRVNUM             ;SET DRIVE 3
```

```
3737  007116  110137  001005                  MOVB    R1,@#SLVNUM             ;SET SLAVE #
3738  007122  010237  001006                  MOV     R2,@#SLVPTR             ;SAVE SLAVE PTR
3739
3740  007126  105737  001130          5$:     TSTB    @#ASFLG
3741  007132  001034                          BNE     7$
3742  007134  112737  000001  001130          MOVB    #1,ASFLG
3743  007142  105037  001124                  CLRB    @#PRGFLG                ;CLEAR PROGRAM INDICATOR
3744  007146  005737  005746                  TST     CHNFLG                  ;BRANCH IF IN CHAIN MODE
3745  007152  001024                          BNE     7$
3746  007154  000004  013773                  TYPE,I.SPD              ;ASK USER IF HE WANTS TO RUN SPEED TESTS
3747  007160  004737  003272                  JSR     PC,.INPUT               ;GET USER INPUT
3748  007164  012703  001264                  MOV     #INBUF,R3       ;GET REPLY
3749  007170  122713  000015                  CMPB    #CR,(R3)        ;DO NOT DO SKEW TESTS IF <CR> IS FIRST
3750  007174  001405                          BEQ     6$
3751  007176  132713  000001                  BITB    #1,(R3)         ;BRANCH IF 'N'
3752  007202  001402                          BEQ     6$
3753  007204  111337  001124                  MOVB    (R3),@#PRGFLG   ;SET INDICATOR
3754  007210  022737  000176  001000  6$:     CMP     #SWREG,SWR              ;BRANCH IF SOFTWARE SWR
3755  007216  001002                          BNE     7$                      ;NOT INVOKED
3756  007220  004737  002020                  JSR     PC,GTSWR                ;GET SWITCH REGISTER
3757  007224                          7$:
3758
3759                                  ;NOTE THIS IS NOT A TEST
3760                                  ;INITIALIZE PROGRAM FLAGS
3761  007224  105037  001122          TST000: CLRB    @#TSTNUM
3762  007230  013705  001010                  MOV     @#TMBASE,R5             ;SET ADDRESS OF FIRST TM03 REG
3763  007234  010500                          MOV     R5,R0
3764  007236  062700  000006                  ADD     #FC,R0                  ;R0 CONTAINS ADDRESS OF FC REG
3765  007242  010501                          MOV     R5,R1
3766  007244  062701  000012                  ADD     #DS,R1                  ;R1 CONTAINS ADDRESS OF DS REG
3767  007250  012703  004340                  MOV     #TIMER,R3               ;SET JUMP ADDRESS TO TIMER
3768  007254  105037  001121                  CLRB    @#ITCNT                 ;CLEAR SUBTEST ITERATION COUNT
3769  007260  052737  000100  177560          BIS     #100,@#TKS              ;SET KEYBOARD IE BIT
3770
3771                                  ;GET USER RUN PROCEDURE
3772                                  ;IF SWR <05::00> IS NOT 0 THEN RUN TEST IN SWR<05::00>
3773                                  ;OTHERWISE RUN ALL TESTS
3774  007266  105737  001124                  TSTB    @#PRGFLG                ;SPEED TESTS?
3775  007272  001042                          BNE     2$                      ;IF SO: BR
3776
3777  007274  004737  005262                  JSR     PC,.REWIND              ;REWIND SLAVE
 (1)  007300  102471                          BVS     99$                     ;BRANCH IF ERROR ON REWIND
3778  007302  004737  005344                  JSR     PC,WRITE                ;WRITE A RECORD
3779  007306  005215                          INC     (R5)                    ;SET 'GO' BIT
3780  007310  004737  005156                  JSR     PC,WAITRDY              ;WAIT FOR READY
3781  007314  102463                          BVS     99$
3782  007316  005737  005746                  TST     CHNFLG                  ;BRANCH IF IN CHAIN MODE
3783  007322  001061                          BNE     100$
3784  007324  117702  171450                  MOVB    @SWR,R2                 ;GET SWITCHES
3785  007330  042702  177740                  BIC     #177740,R2              ;CLEAR ALL BUT TEST #
3786  007334  001421                          BEQ     2$                      ;& BRANCH IF TEST 0 WAS SELECTED
3787  007336  000004  014353                  TYPE,E.HDR                      ;TYPE TEST #
3788  007342  004737  002426                  JSR     PC,TYPOCT
3789  007346  006302                          ASL     R2                      ;FORM INDEX VALUE
3790  007350  016237  001656  007360          MOV     NAMPTR(R2),1$           ;GET ADDRESS OF TEST'S NAME
3791  007356  000004                          TYPE                            ;AND TYPE IT
```

```
3792  007360  000000                  1$:     .WORD   0
3793  007362  000004  001374                  TYPE,CRLF
3794  007366  016237  001736  001002          MOV     TSTTBL(R2),@#SCPADR     ;SET SCOPE ADDRESS FOR TEST
3795  007374  000172  001736                  JMP     @TSTTBL(R2)             ;GO TO TEST
3796  007400  000004  014562          2$:     TYPE,L.HDR1
3797  007404  113702  001004                  MOVB    DRVNUM,R2               ;GET DRIVE #
3798  007410  113704  001005                  MOVB    SLVNUM,R4               ;AND SLAVE #
3799  007414  116237  001132  014742          MOVB    DIGTAB(R2),@#L.DRV      ;SET DRIVE AND SLAVE #'S
3800  007422  116437  001132  014754          MOVB    DIGTAB(R4),@#L.SLV      ;INTO L.HDR2 MESSAGE
3801  007430  112737  000071  014757          MOVB    #'9,@#L.CHAN            ;GET SLAVES CHANNEL TYPE
3802  007436  000004  014675                  TYPE,L.HDR2
3803  007442  004737  005520                  JSR     PC,SNPT                 ;PRINT SLAVE SERIAL #
3804  007446  000004  014776                  TYPE,L.HDR3
3805  007452  105737  001124                  TSTB    @#PRGFLG                ;BRANCH IF SPEED TESTS NOT
3806  007456  001403                          BEQ     100$                    ;SELECTED
3807  007460  000137  012756                  JMP     @#SKEWTST               ,GO DO SPEED TESTS
3808  007464  104400                  99$:    HLT
3809  007466  012737  007474  001002  100$:   MOV     #TST001,@#SCPADR        ;SET SCOPE LOOP ADDRESS
3810
```

```
3812                                            .SBTTL  START OF TESTS
3813                                    ;TEST 001 - WRITE FROM BOT
3814                                    ;THIS TEST WILL MEASURE ACCELERATION DELAY REQUIRED TO
3815                                    ;MOVE THE TAPE APPROXIMATELY SEVEN (7) INCHES FORWARD
3816                                    ;FROM DEAD STOP BEFORE STARTING TO TRANSFER DATA.
3817
3818                                    ;THIS TEST MEASURES TIME FROM 'GO'=1 TO 'ACCL'=0.
3819  007474  112737  000001  001122  TST001: MOVB    #1,@#TSTNUM             ;SET TEST #
3820  007502  012702  007526                 MOV     #1$,R2                  ;SET RETURN PC FROM TIMER
3821  007506  004737  005262                 JSR     PC,.REWIND              ;REWIND SLAVE
 (1)  007512  102420                         BVS     99$                     ;BRANCH IF ERROR ON REWIND
3822  007514  004737  005344                 JSR     PC,WRITE                ;GO SETUP WRITE COMMAND
3823  007520  004737  004250                 JSR     PC,TIMON                ;TURN TIMER ON
3824  007524  005215                         INC     (R5)                    ;SET 'GO' BIT
3825
3826  007526  005765  000032          1$:    TST     TC(R5)                  ;BRANCH WHEN 'ACCL'=0
3827  007532  100002                         BPL     2$
3828  007534  000163  004340                 JMP     TIMER(R3)               ;GO TO TIMER & RETURN VIA R2
3829
3830  007540  004737  005156          2$:    JSR     PC,WAITRDY              ;WAIT FOR COMMAND TO FINISH
3831  007544  102403                         BVS     99$                     ;BRANCH IF ERROR
3832  007546  004737  004376                 JSR     PC,TIMOK                ;GO CHECK TIME
3833  007552  000401                         BR      100$
3834  007554  104400                  99$:   HLT
3835  007556  104000                  100$:  SCOPE
3836
3837                                    ;TEST 002 - WRITE START
3838                                    ;THIS TST MEASURES TIME FROM 'GO'=1 TO 'ACCL'=0.
3839  007560  112737  000002  001122  TST002: MOVB    #2,@#TSTNUM             ;SET TEST # 2
3840  007566  004737  005344                 JSR     PC,WRITE                ;INITIATE WRITE COMMAND
3841  007572  012702  007604                 MOV     #1$,R2                  ;SET RETURN PC FROM TIMER
3842  007576  004737  004250                 JSR     PC,TIMON
3843  007602  005215                         INC     (R5)                    ;SET 'GO' BIT
3844
3845  007604  005765  000032          1$:    TST     TC(R5)                  ;BRANCH WHEN 'ACCL'=0
3846  007610  100002                         BPL     2$
3847  007612  000163  004340                 JMP     TIMER(R3)               ;GO TO TIMER & RETURN VIA R2
3848
3849  007616  004737  005156          2$:    JSR     PC,WAITRDY              ;WAIT FOR READY
3850  007622  102403                         BVS     99$                     ;BRANCH IF ERROR
3851  007624  004737  004376                 JSR     PC,TIMOK                ;GO CHECK TIME RECORDED
3852  007630  000401                         BR      100$                    ;EXIT VIA SCOPE
3853
3854  007632  104400                  99$:   HLT                             ;REPORT ERROR
3855  007634  104000                  100$:  SCOPE
3856
3857                                    ;TEST 003- WRITE SHUTDOWN
3858                                    ;THIS TEST MEASURES TIME FROM 'FC REG'=0 TO 'SWDN'=1.
3859  007636  112737  000003  001122  TST003: MOVB    #3,@#TSTNUM             ;SET TEST#3
3860  007644  004737  005344                 JSR     PC,WRITE                ;INITIATE WRITE COMMAND
3861  007650  005215                         INC     (R5)                    ;SET 'GO' BIT
3862
3863  007652  005710                  1$:    TST     (R0)                    ;BRANCH WHEN WRITING FINISHED
3864  007654  001404                         BEQ     2$
3865  007656  032711  040000                 BIT     #ERR,(R1)               ;MONITOR ERROR BIT
3866  007662  001017                         BNE     99$
```

```
3867  007664  000772                        BR      1$
3868
3869  007666                          2$:
 (1)  007666  004737  004250                JSR     PC,TIMON                ;TURN TIMER ON
3870  007672  010702                        MOV     PC,R2                   ;LOAD RETURN PC FROM TIMER
3871  007674  032711  000020          3$:   BIT     #SDWN,(R1)              ;BRANCH WHEN DS <SDWN> SETS
3872  007700  001002                        BNE     4$
3873  007702  000163  004340                JMP     TIMER(R3)               ;GO TO TIMER & RETURN VIA R2
3874
3875  007706  004737  005156          4$:   JSR     PC,WAITRDY              ;WAIT FOR READY
3876  007712  102403                        BVS     99$
3877  007714  004737  004376                JSR     PC,TIMOK                ;GO CHECK TIME RECORDED
3878  007720  000401                        BR      100$
3879  007722  104400                99$:    HLT                             ;REPORT ERROR
3880  007724  104000                100$:   SCOPE
3881
3882                                  ;TEST 004 - WRITE SETTLEDOWN
3883                                  ;THIS TEST MEASURES TIME FROM 'SWDN'=1 TO 'SWDN'=0.
3884  007726  112737  000004  001122  TST004: MOVB   #4,@#TSTNUM
3885  007734  004737  005344                JSR     PC,WRITE
3886  007740  005215                        INC     (R5)                    ;SET 'GO' BIT
3887
3888  007742  005710                1$:     TST     (R0)                    ;BRANCH WHEN WRITING FINISHED
3889  007744  001404                        BEQ     2$
3890  007746  032711  040000                BIT     #ERR,(R1)               ;CHECK ERROR BIT
3891  007752  001026                        BNE     99$
3892  007754  000772                        BR      1$
3893
3894  007756  032711  000020          2$:   BIT     #SDWN,(R1)              ;WAIT FOR ASSERTION OF 'SDWN'
3895  007762  001004                        BNE     3$
3896  007764  032711  040000                BIT     #ERR,(R1)               ;MONITOR ERROR BIT
3897  007770  001017                        BNE     99$
3898  007772  000771                        BR      2$
3899
3900  007774                          3$:
 (1)  007774  004737  004250                JSR     PC,TIMON                ;TURN TIMER ON
3901  010000  010702                        MOV     PC,R2                   ;SET RETURN PC FROM TIMER
3902  010002  032711  000020                BIT     #SDWN,(R1)              ;BRANCH WHEN SWDN CLEARS
3903  010006  001402                        BEQ     5$
3904  010010  000163  004340                JMP     TIMER(R3)               ;GO TO TIMER & RETURN VIA R2
3905
3906  010014  004737  005156          5$:   JSR     PC,WAITRDY              ;WAIT FOR READY
3907  010020  102403                        BVS     99$
3908  010022  004737  004376                JSR     PC,TIMOK
3909  010026  000401                        BR      100$
3910
3911  010030  104400                99$:    HLT
3912  010032  104000                100$:   SCOPE
3913
3914                                  ;TEST 005 - READ FROM BOT
3915                                  ;THIS TEST MEASURES TIME FROM 'GO'=1 TO 'ACCL'=0.
3916  010034  112737  000005  001122  TST005: MOVB   #5,@#TSTNUM             ;SET TEST #5
3917  010042  004737  005262                JSR     PC,.REWIND              ;REWIND SLAVE
 (1)  010046  102422                        BVS     99$                     ;BRANCH IF ERROR ON REWIND
3918  010050  004737  005362                JSR     PC,READ
3919  010054  012702  010066                MOV     #1$,R2                  ;SET RETURN PC FROM TIMER
```

```
3920  010060  004737  004250            JSR     PC,TIMON                ;TURN TIMER ON
3921  010064  005215                    INC     (R5)                    ;SET 'GO' BIT
3922
3923  010066  005765  000032    1$:     TST     TC(R5)                  ;BRANCH WHEN 'ACCL' RESETS
3924  010072  100002                    BPL     2$
3925  010074  000163  004340            JMP     TIMER(R3)               ;GO TO TIMER & RETURN VIA R2
3926
3927  010100  004737  005156    2$:     JSR     PC,WAITRDY              ;WAIT FOR READY
3928  010104  102403                    BVS     99$                     ;BRANCH IF ERROR
3929  010106  004737  004376            JSR     PC,TIMOK                ;CHECK RECORDED TIME
3930  010112  000401                    BR      100$
3931
3932  010114  104400            99$:    HLT
3933  010116  104000            100$:   SCOPE
3934
3935                            ;TEST 006 - READ START
3936                            ;THIS TEST MEASURES TIME FROM 'GO'=1 TO 'ACCL'=0.
3937  010120  112737  000006  001122  TST006: MOVB    #6,@#TSTNUM             ;SET TEST #6
3938  010126  004737  005436            JSR     PC,WRT.BK               ;WRITE A RECORD & BACK SPACE
3939  010132  102422                    BVS     99$
3940  010134  004737  005362            JSR     PC,READ
3941  010140  012702  010152            MOV     #1$,R2                  ;SET RETURN PC FROM TIMER
3942  010144  004737  004250            JSR     PC,TIMON                ;TURN TIMER ON
3943  010150  005215                    INC     (R5)                    ;SET 'GO' BIT
3944
3945  010152  005765  000032    1$:     TST     TC(R5)                  ;BRANCH WHEN 'ACCL' RESETS
3946  010156  100002                    BPL     2$
3947  010160  000163  004340            JMP     TIMER(R3)               ;GO TO TIMER & RETURN VIA R2
3948
3949  010164  004737  005156    2$:     JSR     PC,WAITRDY
3950  010170  102403                    BVS     99$
3951  010172  004737  004376            JSR     PC,TIMOK
3952  010176  000401                    BR      100$
3953
3954  010200  104400            99$:    HLT
3955  010202  104000            100$:   SCOPE
3956
3957                            ;TEST 007 - READ SHUTDOWN
3958                            ;THIS TEST MEASURES TIME FROM 'FC REG'=FRAME COUNT TO 'SWDN'=1.
3959  010204  112737  000007  001122  TST007: MOVB    #7,@#TSTNUM             ;SET TEST #7
3960  010212  004737  005436            JSR     PC,WRT.BK               ;WRITE A RECORD & BACK SPACE
3961  010216  102430                    BVS     99$                     ;BRANCH IF ERROR
3962  010220  004737  005362            JSR     PC,READ
3963  010224  005215                    INC     (R5)                    ;SET 'GO' BIT
3964
3965  010226  022710  000400    1$:     CMP     #-FRMCNT,(R0)           ;WAIT FOR FRAME COUNT TO
3966  010232  001404                    BEQ     2$                      ;= # OF FRAMES WRITTEN
3967  010234  032711  040000            BIT     #ERR,(R1)               ;MONITOR ERROR BIT
3968  010240  001017                    BNE     99$
3969  010242  000771                    BR      1$
3970
3971  010244                    2$:
 (1)  010244  004737  004250            JSR     PC,TIMON                ;TURN TIMER ON
3972  010250  010702                    MOV     PC,R2                   ;SET RETURN PC FROM TIMER
3973  010252  032711  000020            BIT     #SDWN,(R1)              ;BRANCH WHEN SDWN SETS
3974  010256  001002                    BNE     3$
```

```
 3975  010260  000163  004340                  JMP     TIMER(R3)               ;GO TO TIMER & RETURN VIA R2
 3976
 3977  010264  004737  005156          3$:     JSR     PC,WAITRDY
 3978  010270  102403                          BVS     99$
 3979  010272  004737  004376                  JSR     PC,TIMOK
 3980  010276  000401                          BR      100$
 3981
 3982  010300  104400                  99$:    HLT                             ;REPORT ERROR
 3983  010302  104000                  100$:   SCOPE
 3984
 3985                                  ;TEST 010 - READ SETTLEDOWN
 3986                                  ;THIS TEST MEASURES TIME FROM 'SWDN'=1 TO 'SWDN'=0.
 3987  010304  112737  000010  001122  TST010: MOVB    #10,@#TSTNUM            ;SET TEST #10
 3988  010312  012702  010370                  MOV     #4$,R2                  ;SET RETURN PC FROM TIMER
 3989  010316  004737  005436                  JSR     PC,WRT.BK               ;WRITE A RECORD & BACK SPACE
 3990  010322  102436                          BVS     99$
 3991  010324  004737  005362                  JSR     PC,READ
 3992  010330  005215                          INC     (R5)                    ;SET 'GO' BIT
 3993
 3994  010332  105711                  1$:     TSTB    (R1)                    ;WAIT FOR READY
 3995  010334  100404                          BMI     2$                      ;BRANCH WHEN SET
 3996  010336  032711  040000                  BIT     #ERR,(R1)               ;CHECK ERROR BIT
 3997  010342  001026                          BNE     99$
 3998  010344  000772                          BR      1$
 3999
 4000  010346  032711  000020          2$:     BIT     #SDWN,(R1)              ;WAIT FOR ASSERTION OF 'SDWN'
 4001  010352  001004                          BNE     3$
 4002  010354  032711  040000                  BIT     #ERR,(R1)               ;MONITOR ERROR BIT
 4003  010360  001017                          BNE     99$
 4004  010362  000771                          BR      2$
 4005
 4006  010364                          3$:
  (1)  010364  004737  004250                  JSR     PC,TIMON                ;TURN TIMER ON
 4007  010370  032765  000020  000012  4$:     BIT     #SDWN,DS(R5)            ;WAIT FOR NEGATION OF SDWN
 4008  010376  001402                          BEQ     5$
 4009  010400  000163  004340                  JMP     TIMER(R3)               ;GO TO TIMER & RETURN VIA R2
 4010
 4011  010404  004737  005156          5$:     JSR     PC,WAITRDY
 4012  010410  102403                          BVS     99$
 4013  010412  004737  004376                  JSR     PC,TIMOK
 4014  010416  000401                          BR      100$
 4015
 4016  010420  104400                  99$:    HLT
 4017  010422  104000                  100$:   SCOPE
 4018
 4019
 4020                                  ;TEST 011-READ REVERSE START
 4021                                  ;THIS TEST MEASURES TIME FROM 'GO'=1 TO 'ACCL'-0.
 4022  010424  112737  000011  001122  TST011: MOVB    #11,@#TSTNUM
 4023  010432  012702  010470                  MOV     #1$,R2                  ;SET RETURN PC FROM TIMER
 4024  010436  004737  005344                  JSR     PC,WRITE                ;WRITE A RECORD
 4025  010442  005215                          INC     (R5)                    ;SET 'GO' BIT
 4026  010444  004737  005156                  JSR     PC,WAITRDY
 4027  010450  102422                          BVS     99$
 4028  010452  004737  004634                  JSR     PC,DELAY                ;WAIT FOR TAPE MOTION TO STOP
 4029  010456  004737  005400                  JSR     PC,REVRD
```

```
 4030  010462  004737  004250              JSR     PC,TIMON                ;TURN TIMER ON
 4031  010466  005215                      INC     (R5)                    ;SET 'GO' BIT
 4032
 4033  010470  005765  000032      1$:     TST     TC(R5)                  ;BRANCH WHEN 'ACCL' = 0
 4034  010474  100002                      BPL     2$
 4035  010476  000163  004340              JMP     TIMER(R3)               ;GO TO TIMER & RETURN VIA R2
 4036
 4037  010502  004737  005156      2$:     JSR     PC,WAITRDY
 4038  010506  102403                      BVS     99$                     ;BRANCH IF ERROR
 4039  010510  004737  004376              JSR     PC,TIMOK
 4040  010514  000401                      BR      100$
 4041
 4042  010516  104400              99$:    HLT
 4043  010520  104000              100$:   SCOPE
 4044
 4045                                      ;TEST 012-READ REVERSE SHUTDOWN
 4046                                      ;THIS TEST MEASURES TIME FROM 'FC REG' = FRAME COUNT TO 'SDWN'=1.
 4047  010522  112737  000012  001122  TST012: MOVB    #12,@#TSTNUM
 4048  010530  012702  010600              MOV     #3$,R2                  ;SET RETURN PC FROM TIMER
 4049  010534  004737  005344              JSR     PC,WRITE                ;WRITE A RECORD
 4050  010540  005215                      INC     (R5)                    ;SET 'GO' BIT
 4051  010542  004737  005156              JSR     PC,WAITRDY
 4052  010546  102427                      BVS     99$
 4053  010550  004737  005400              JSR     PC,REVRD
 4054  010554  005215                      INC     (R5)                    ;SET 'GO' BIT
 4055
 4056  010556  022710  000400      1$:     CMP     #-FRMCNT,(R0)           ;BRANCH WHEN FRAME COUNT
 4057  010562  001404                      BEQ     2$                      ;= # OF RECORD WRITTEN
 4058  010564  032711  040000              BIT     #ERR,(R1)               ;MONITOR ERROR BIT IN 'DS' REG
 4059  010570  001016                      BNE     99$
 4060  010572  000771                      BR      1$
 4061
 4062  010574                      2$:
  (1)  010574  004737  004250              JSR     PC,TIMON                ;TURN TIMER ON
 4063  010600  032711  000020      3$:     BIT     #SDWN,(R1)              ;BRANCH WHEN SDWN SETS
 4064  010604  001002                      BNE     4$
 4065  010606  000163  004340              JMP     TIMER(R3)               ;GO TO TIMER & RETURN VIA R2
 4066
 4067  010612  004737  005156      4$:     JSR     PC,WAITRDY              ;WAIT FOR READY
 4068  010616  102403                      BVS     99$
 4069  010620  004737  004376              JSR     PC,TIMOK
 4070  010624  000401                      BR      100$
 4071
 4072  010626  104400              99$:    HLT
 4073  010630  104000              100$:   SCOPE
 4074
 4075                                      ;TEST 013-READ REVERSE SETTLEDOWN
 4076                                      ;THIS TEST MEASURES TIME FROM 'SDWN'=1 TO 'SDWN'=0.
 4077  010632  112737  000013  001122  TST013: MOVB    #13,@#TSTNUM
 4078  010640  012702  010724              MOV     #4$,R2                  ;SET RETURN PC FROM TIMER
 4079  010644  004737  005344              JSR     PC,WRITE                ;WRITE A RECORD
 4080  010650  005215                      INC     (R5)                    ;SET 'GO' BIT
 4081  010652  004737  005156              JSR     PC,WAITRDY
 4082  010656  102435                      BVS     99$
 4083  010660  004737  005400              JSR     PC,REVRD
 4084  010664  005215                      INC     (R5)                    ;SET 'GO' BIT
```

```
 4085
 4086  010666  105711                   1$:     TSTB    (R1)                    ;BRANCH WHEN
 4087  010670  100404                           BMI     2$                      ;READY SETS
 4088  010672  032711  040000                   BIT     #ERR,(R1)
 4089  010676  001025                           BNE     99$
 4090  010700  000772                           BR      1$
 4091
 4092  010702  032711  000020           2$:     BIT     #SDWN,(R1)
 4093  010706  001004                           BNE     3$
 4094  010710  032711  040000                   BIT     #ERR,(R1)
 4095  010714  001016                           BNE     99$
 4096  010716  000771                           BR      2$
 4097
 4098  010720                           3$:
  (1)  010720  004737  004250                   JSR     PC,TIMON                ;TURN TIMER ON
 4099  010724  032711  000020           4$:     BIT     #SDWN,(R1)              ;BRANCH WHEN SWDN = 0
 4100  010730  001402                           BEQ     5$
 4101  010732  000163  004340                   JMP     TIMER(R3)               ;GO TO TIMER & RETURN VIA R2
 4102
 4103  010736  004737  005156           5$:     JSR     PC,WAITRDY              ;WAIT FOR READY
 4104  010742  102403                           BVS     99$
 4105  010744  004737  004376                   JSR     PC,TIMOK
 4106  010750  000401                           BR      100$
 4107
 4108  010752  104400                   99$:    HLT
 4109  010754  104000                   100$:   SCOPE
 4110
 4111                                   ;REWIND DRIVE
 4112  010756                           A:
  (1)  010756  004737  005262                   JSR     PC,.REWIND              ;REWIND SLAVE
  (1)  010762  102401                           BVS     99$                     ;BRANCH IF ERROR ON REWIND
 4113  010764  102002                           BVC     100$
 4114  010766  104400                   99$:    HLT
 4115  010770  000772                           BR      A
 4116  010772                           100$:
 4117
 4118                                   ;TEST 014-TURN AROUND DELAY (FORWARD-REVERSE)
 4119                                   ;THIS TEST MEASURES TIME FROM 'GO'=1 (READ REVERSE) TO 'ACCL'=0
 4120  010772  112737  000014  001122   TST014: MOVB    #14,@#TSTNUM
 4121  011000  012702  011032                   MOV     #2$,R2                  ;SET RETURN PC FROM TIMER
 4122  011004  004737  005344                   JSR     PC,WRITE                ;WRITE A RECORD
 4123  011010  005215                           INC     (R5)                    ;SET 'GO' BIT
 4124  011012  004737  005156                   JSR     PC,WAITRDY
 4125  011016  102420                           BVS     99$
 4126
 4127  011020  004737  005400           1$:     JSR     PC,REVRD                ;READ THE RECORD (REVERSE)
 4128  011024  004737  004250                   JSR     PC,TIMON                ;TURN TIMER ON
 4129  011030  005215                           INC     (R5)                    ;SET 'GO' BIT
 4130
 4131  011032  005765  000032           2$:     TST     TC(R5)                  ;WAIT FOR 'ACCL' - 0
 4132  011036  100002                           BPL     3$
 4133  011040  000163  004340                   JMP     TIMER(R3)               ;GO TO TIMER & RETURN VIA R2
 4134
 4135  011044  004737  005156           3$:     JSR     PC,WAITRDY
 4136  011050  102403                           BVS     99$
 4137  011052  004737  004376                   JSR     PC,TIMOK
```

```
4138  0'1056  000401                          BR      100$
4139
4140  011060  104400                  99$:    HLT
4141  011062  104000                  100$:   SCOPE
4142
4143                                  ;TEST 015- TURN AROUND DELAY (REVERSE-FORWARD)
4144                                  ;THIS TEST MEASURES TIME FROM 'GO'=1 (READ) TO 'ACCL'=0.
4145  011064  112737  000015  001122  TST015: MOVB    #15,@#TSTNUM
4146  011072  012702  011140                  MOV     #2$,R2                  ;SET RETURN PC FROM TIMER
4147  011076  004737  005344                  JSR     PC,WRITE                ;WRITE A RECORD
4148  011102  005215                          INC     (R5)                    ;SET 'GO' BIT
4149  011104  004737  005156                  JSR     PC,WAITPDY              ;WAIT FOR READY
4150  011110  102426                          BVS     99$
4151  011112  004737  005400                  JSR     PC,REVRD                ;READ A RECORD IN THE
4152  011116  005215                          INC     (R5)                    ;SET 'GO' BIT
4153
4154  011120  004737  005156                  JSR     PC,WAITRDY
4155  011124  102420                          BVS     99$
4156
4157  011126  004737  005362          1$:     JSR     PC,READ                 ;READ RECORD FORWARD
4158  011132  004737  004250                  JSR     PC,TIMON                ;TURN TIMER ON
4159  011136  005215                          INC     (R5)                    ;SET 'GO' BIT
4160
4161  011140  005765  000032          2$:     TST     TC(R5)                  ;WAIT FOR 'ACCL' = 0
4162  011144  100002                          BPL     3$
4163  011146  000163  004340                  JMP     TIMER(R3)               ;GO TO TIMER & RETURN VIA R2
4164
4165  011152  004737  005156          3$:     JSR     PC,WAITRDY
4166  011156  102403                          BVS     99$
4167  011160  004737  004376                  JSR     PC,TIMOK
4168  011164  000401                          BR      100$
4169
4170  011166  104400                  99$:    HLT
4171  011170  104000                  100$:   SCOPE
4172
4173                                  ;TEST 016-GAP SIZE (STOP HALF)
4174  011172  112737  000016  001122  TST016: MOVB    #16,@#TSTNUM
4175  011200  012702  011236                  MOV     #1$,R2                  ;SET RETURN PC FROM TIMER
4176  011204  004737  005344                  JSR     PC,WRITE                ;WRITE A RECORD
4177  011210  005215                          INC     (R5)                    ;SET 'GO' BIT
4178  011212  004737  005156                  JSR     PC,WAITRDY
4179  011216  102421                          BVS     99$
4180  011220  004737  004634                  JSR     PC,DELAY                ;DELAY 350 MS
4181  011224  004737  005400                  JSR     PC,REVRD                ;READ REVERSE RECORD
4182  011230  004737  004250                  JSR     PC,TIMON                ;TURN TIMER ON
4183  011234  005215                          INC     (R5)                    ;SET 'GO' BIT
4184
4185  011236  005710          1$:     TST     (R0)                    ;WAIT FOR FRAME COUNT > 0
4186  011240  001002                          BNE     2$
4187  011242  000163  004340                  JMP     TIMER(R3)               ;GO TO TIMER & RETURN VIA R2
4188
4189  011246  004737  005156          2$:     JSR     PC,WAITRDY              ;WAIT FOR READY BIT TO SET
4190  011252  102403                          BVS     99$
4191  011254  004737  004376                  JSR     PC,TIMOK                ;CHECK TIME
4192  011260  000401                          BR      100$
4193
```

```
 4194  011262  104400                  99$:    HLT
 4195  011264  104000                  100$:   SCOPE
 4196
 4197                                  ;TEST 017-GAP SIZE (START HALF)
 4198  011266  112737  000017  001122  TST017: MOVB    #17,@#TSTNUM
 4199  011274  012702  011346                  MOV     #1$,R2                  ;SET RETURN PC FROM TIMER
 4200  011300  004737  005344                  JSR     PC,WRITE                ;WRITE A RECORD
 4201  011304  005215                          INC     (R5)                    ;SET 'GO' BIT
 4202  011306  004737  005156                  JSR     PC,WAITRDY              ;WAIT FOR READY
 4203  011312  102427                          BVS     99$
 4204  011314  004737  005400                  JSR     PC,REVRD                ;READ REVERSE THE RECORD
 4205  011320  005215                          INC     (R5)                    ;SET 'GO' BIT
 4206  011322  004737  005156                  JSR     PC,WAITRDY              ;WAIT FOR READY
 4207  011326  102421                          BVS     99$                     ;BRANCH ON ERROR
 4208  011330  004737  004634                  JSR     PC,DELAY                ;WAIT FOR TAPE MOTION TO STOP
 4209  011334  004737  005362                  JSR     PC,READ                 ;READ RECORD
 4210  011340  004737  004250                  JSR     PC,TIMON                ;TURN TIMER ON
 4211  011344  005215                          INC     (R5)                    ;SET 'GO' BIT
 4212
 4213  011346  005710                  1$:     TST     (R0)                    ;WAIT FOR FRAME COUNT > 0
 4214  011350  001002                          BNE     2$
 4215  011352  000163  004340                  JMP     TIMER(R3)               ;GO TO TIMER & RETURN VIA R2
 4216
 4217  011356  004737  005156          2$:     JSR     PC,WAITRDY              ;WAIT FOR READY
 4218  011362  102403                          BVS     99$
 4219  011364  004737  004376                  JSR     PC,TIMOK                ;CHECK TIME
 4220  011370  000401                          BR      100$
 4221
 4222  011372  104400                  99$:    HLT
 4223  011374  104000                  100$:   SCOPE
 4224
 4225                                  ;TEST 020- GAP SIZE (INTERRECORD)
 4226                                  ;THIS TEST MEASURES TIME FROM 'GO'=1 TO 'FC REG' >0.
 4227  011376  112737  000020  001122  TST020: MOVB    #20,@#TSTNUM
 4228  011404  012702  011466                  MOV     #1$,R2                  ;SET RETURN PC FROM TIMER
 4229  011410  004737  005344                  JSR     PC,WRITE                ;WRITE A RECORD
 4230  011414  005215                          INC     (R5)                    ;SET 'GO' BIT
 4231  011416  004737  005156                  JSR     PC,WAITRDY              ;WAIT FOR READY
 4232  011422  102433                          BVS     99$
 4233  011424  004737  005344                  JSR     PC,WRITE                ;WRITE SECOND RECORD
 4234  011430  005215                          INC     (R5)                    ;SET 'GO' BIT
 4235  011432  004737  005156                  JSR     PC,WAITRDY              ;WAIT FOR READY
 4236  011436  102425                          BVS     99$
 4237  011440  004737  005400                  JSR     PC,REVRD                ;READ REVERSE SECOND RECORD
 4238  011444  005215                          INC     (R5)                    ;SET 'GO' BIT
 4239  011446  004737  005156                  JSR     PC,WAITRDY              ;WAIT FOR READY
 4240  011452  102417                          BVS     99$
 4241  011454  004737  005400                  JSR     PC,REVRD                ;READ REVERSE FIRST RECORD
 4242  011460  004737  004250                  JSR     PC,TIMON                ;TURN TIMER ON
 4243  011464  005215                          INC     (R5)                    ;SET 'GO' BIT
 4244
 4245  011466  005710                  1$:     TST     (R0)                    ;WAIT FOR FRAME COUNT > 0
 4246  011470  001002                          BNE     2$
 4247  011472  000163  004340                  JMP     TIMER(R3)               ;GO TO TIMER & RETURN VIA R2
 4248
 4249  011476  004737  005156          2$:     JSR     PC,WAITRDY              ;WAIT FOR READY
```

```
 4250  011502  102403                          BVS     99$
 4251  011504  004737  004376                  JSR     PC,TIMOK
 4252  011510  000401                          BR      100$
 4253
 4254  011512  104400                  99$:    HLT
 4255  011514  104000                  100$:   SCOPE
 4256
 4257                                  ;TEST 021- GAP CONSISTANCY
 4258                                  ;THIS TEST MEASURES TIME FROM 'GO'=1 TO 'FC REG' > 0.
 4259                                  ;THE TEST REWINDS THE TAPE,WRITES 17 RECORDS WITH A DELAY FROM 1-16 MS
 4260                                  ;BETWEEN EACH WRITE COMMAND. AFTER THE 17. RECORDS ARE WRITTTEN THE
 4261                                  ;PROGRAM READ REVERSES 16 RECORDS. AT THIS POINT THE TAPE IS STOPPED BE-
 4262                                  ;TWEEN THE FIRST AND SECOND RECORD. A READ COMMAND IS EXECUTED TO READ
 4263                                  ;THE 16 RECORDS WITH THE TIME BETEWEN GO=1 TO FC > 0 STORED IN 'GAPTBL'
 4264                                  ;FOR EACH RECORD READ. AFTER 16 RECORDS HAVE BEEN READ THE TIME IS VER-
 4265                                  ;IFIED FOR EACH READ. AFTER ALL RECORD TIMES ARE VERIFIED THEY ARE AVER-
 4266                                  ;AGED AND PLACED IN THE 'ATIMTBL' (BY SCOPE). THE ABOVE PROCESS IS RE-
 4267                                  ;PEATED FOR EACH ITERATION.
 4268
 4269  011516  112737  000021  001122  TST021: MOVB    #21,@#TSTNUM
 4270  011524  012702  011662                  MOV     #4$,R2          ;SET RETURN PC FROM TIMER
 4271  011530  004737  005262                  JSR     PC,.REWIND      ;REWIND SLAVE
  (1)  011534  102530                          BVS     99$             ;BRANCH IF ERROR ON REWIND
 4272  011536  005037  001114                  CLR     DELTIM          ;CLEAR VARIABLE DELAY TIME
 4273  011542  012700  000021                  MOV     #17.,R0         ;SET # OF RECORDS TO WRITE
 4274  011546  004737  005344          1$:     JSR     PC,WRITE        ;WRITE 17. RECORDS
 4275  011552  005215                          INC     (R5)            ;SET 'GO' BIT
 4276  011554  004737  005156                  JSR     PC,WAITRDY      ;WAIT FOR READY
 4277  011560  102516                          BVS     99$
 4278  011562  004737  004664                  JSR     PC,DELAYV       ;DELAY BEFORE WRITING NEXT REC.
 4279  011566  062737  000022  001114          ADD     #18.,DELTIM     ;SET NEXT DELAY TIME
 4280  011574  005300                          DEC     R0              ;DECREMENT RECORDS WRITTEN COUNT
 4281  011576  001363                          BNE     1$
 4282
 4283  011600  012700  000021                  MOV     #17.,R0         ;SET # OF RECS. TO REVERSE READ
 4284  011604  004737  005400          2$:     JSR     PC,REVRD        ;REVERSE READ 17. RECORDS
 4285  011610  005215                          INC     (R5)            ;SET 'GO' BIT
 4286  011612  004737  005156                  JSR     PC,WAITRDY      ;WAIT FOR READY
 4287  011616  102477                          BVS     99$
 4288  011620  005300                          DEC     R0              ;DECREMENT RECORD COUNT
 4289  011622  001370                          BNE     2$
 4290
 4291  011624  012700  000020                  MOV     #16.,R0         ;SET # OF RECORDS TO READ
 4292  011630  012701  001054                  MOV     #GAPTBL,R1      ;SET PTR TO GAP TABLE FOR TEST
 4293  011634  004737  005362                  JSR     PC,READ         ;READ A RECORD
 4294  011640  005215                          INC     (R5)            ;SET 'GO' BIT
 4295
 4296  011642  004737  005156          3$:     JSR     PC,WAITRDY      ;WAIT FOR READY
 4297  011646  102463                          BVS     99$
 4298  011650  004737  005362                  JSR     PC,READ         ;READ NEXT RECORD
 4299  011654  004737  004250                  JSR     PC,TIMON        ;TURN TIMER ON
 4300  011660  005215                          INC     (R5)            ;SET 'GO' BIT
 4301
 4302  011662  005765  000006          4$:     TST     FC(R5)          ;WAIT FOR FRAME COUNT > 0
 4303  011666  001002                          BNE     5$
 4304  011670  000163  004340                  JMP     TIMER(R3)       ;GO TO TIMER & RETURN VIA R2
```

```
4305
4306  011674  004737  005156          5$:     JSR     PC,WAITRDY              ;WAIT FOR READY
4307  011700  102446                          BVS     99$
4308  011702  010421                          MOV     R4,(R1)+                ;STORE TIME IN GAPTBL
4309  011704  005300                          DEC     R0                      ;DECREMENT # OF RECORDS READ
4310  011706  001355                          BNE     3$
4311
4312  011710  105037  001120                CLRB    @#GAP                   ;SET GAP # 0
4313  011714  012700  000020                MOV     #16.,R0
4314  011720  012701  001054                MOV     #GAPTBL,R1
4315
4316  011724  012104                  6$:     MOV     (R1)+,R4                ;GET GAP TICK COUNT
4317  011726  004737  004506                  JSR     PC,GAPOK                ;CHECK TIME
4318  011732  105237  001120                INCB    @#GAP                   ;INCREMENT GAP #
4319  011736  122737  000020  001120        CMPB    #16.,@#GAP              ;BRANCH IF ALL GAPS NOT CHECKED
4320  011744  001367                          BNE     6$
4321
4322  011746  012700  000020                MOV     #16.,R0                 ;SETUP TO AVERAGE GAP SIZES
4323  011752  012701  001054                MOV     #GAPTBL,R1              ;SET PTR TO TABLE
4324  011756  005002                          CLR     R2                      ;CLEAR 'SUM' REGISTERS
4325  011760  005003                          CLR     R3
4326  011762  062102                  7$:     ADD     (R1)+,R2                ;ADD ALL GAP SIZES TOGETHER
4327  011764  005503                          ADC     R3
4328  011766  005300                          DEC     R0
4329  011770  001374                          BNE     7$
4330  011772  012700  000004                MOV     #4,R0                   ;NOW DIVIDE BY 16.
4331  011776  006203                  8$:     ASR     R3                      ;BY SHIFTING 4 PLACES RIGHT
4332  012000  006002                          ROR     R2
4333  012002  005300                          DEC     R0
4334  012004  001374                          BNE     8$
4335  012006  010204                          MOV     R2,R4                   ;MOVE AVERAGED TIMES TO R4
4336  012010  004737  004376                JSR     PC,TIMOK                ;CHECK AVERAGED TIMES
4337  012014  000401                          BR      100$
4338
4339  012016  104400                  99$:    HLT
4340  012020  104000                  100$:   SCOPE
4341
4342
4343
4344                                  ;TEST 022-DATA TIME (800BPI)
4345                                  ;THIS TEST MEASURES THE TIME FROM FC REG >-6400 TO 'RDY' = 1.
4346  012022  112737  000022  001122  TST022: MOVB   #022,@#TSTNUM
4347  012030  012702  012110                  MOV     #3$,R2                  ;SET RETURN PC FROM TIMER
4348  012034  004737  005262                  JSR     PC,.REWIND              ;REWIND SLAVE
 (1)  012040  102442                          BVS     99$                     ;BRANCH IF ERROR ON REWIND
4349  012042  052765  001700  000032          BIS     #NORM11,TC(R5)          ;SET 800 BPI
4350  012050  004337  005500                  JSR     R3,TMCMD                ;WRITE 3200. WORD RECORD
4351  012054  016046                          .WORD   WTBUF
4352  012056  171600                          .WORD   -3200.
4353  012060  163400                          .WORD   -6400.
4354  012062  000060                          .WORD   WFWD
4355  012064  005215                          INC     (R5)                    ;SET 'GO' BIT
4356
4357  012066  022710  163400          1$:     CMP     #-6400.,(R0)            ;WAIT FOR WRITING TO START
4358  012072  001004                          BNE     2$
4359  012074  032711  040000                  BIT     #ERR,(R1)               ;MONITOR ERROR BIT
```

```
 4360  012100  001022                    BNE     99$
 4361  012102  000771                    BR      1$
 4362
 4363  012104                          2$:
  (1)  012104  004737  004250                JSR     PC,TIMON                ;TURN TIMER ON
 4364  012110  105711                  3$:   TSTB    (R1)                    ;BRANCH WHEN READY SETS
 4365  012112  100402                        BMI     4$
 4366  012114  000163  004340                JMP     TIMER(R3)               ;GO TO TIMER & RETURN VIA R2
 4367
 4368  012120  012700  000003          4$:   MOV     #3,R0                   ;SET SHIFT COUNT
 4369  012124  006204                  5$:   ASR     R4
 4370  012126  005300                        DEC     R0
 4371  012130  001375                        BNE     5$
 4372  012132  004737  005156                JSR     PC,WAITRDY
 4373  012136  102403                        BVS     99$
 4374  012140  004737  004376                JSR     PC,TIMOK                ;CHECK TIME
 4375  012144  000401                        BR      100$
 4376
 4377  012146  104400                  99$:  HLT
 4378  012150  104000                  100$: SCOPE
 4379
 4380                                  ;TEST 023-DATA TIME (1600BPI)
 4381                                  ;THIS TEST MEASURES THE TIME FROM FC REG >-6400 TO 'RDY' = 1.
 4382  012152  112737  000023  001122  TST023: MOVB    #023,@#TSTNUM
 4383  012160  012702  012246                MOV     #3$,R2                  ;SET RETURN PC FROM TIMER
 4384  012164  004737  005262                JSR     PC,.REWIND              ;REWIND SLAVE
  (1)  012170  102442                        BVS     99$                     ;BRANCH IF ERROR ON REWIND
 4385  012172  042765  003700  000032        BIC     #3700,TC(R5)            ;CLEAR CURRENT DENSITY
 4386  012200  052765  002300  000032        BIS     #PE1600,TC(R5)          ;SET 1600 BPI
 4387  012206  004337  005500                JSR     R3,TMCMD                ;WRITE 3200. WORD RECORD
 4388  012212  016046                        .WORD   WTBUF
 4389  012214  171600                        .WORD   -3200.
 4390  012216  163400                        .WORD   -6400.
 4391  012220  000060                        .WORD   WFWD
 4392  012222  005215                        INC     (R5)                    ;SET 'GO' BIT
 4393
 4394  012224  022710  163400          1$:   CMP     #-6400.,(R0)            ;BRANCH WHEN WRITING STARTS
 4395  012230  001004                        BNE     2$
 4396  012232  032711  040000                BIT     #ERR,(R1)               ;MONITOR ERROR BIT
 4397  012236  001017                        BNE     99$
 4398  012240  000771                        BR      1$
 4399
 4400  012242                          2$:
  (1)  012242  004737  004250                JSR     PC,TIMON                ;TURN TIMER ON
 4401  012246  105711                  3$:   TSTB    (R1)                    ;BRANCH WHEN READY SETS
 4402  012250  100402                        BMI     4$
 4403  012252  000163  004340                JMP     TIMER(R3)               ;GO TO TIMER & RETURN VIA R2
 4404
 4405  012256  006204                  4$:   ASR     R4                      ;DIVIDE TIME BY 4
 4406  012260  006204                        ASR     R4
 4407  012262  004737  005156                JSR     PC,WAITRDY
 4408  012266  102403                        BVS     99$
 4409  012270  004737  004376                JSR     PC,TIMOK                ;CHECK TIME
 4410  012274  000401                        BR      100$
 4411
 4412  012276  104400                  99$:  HLT
```

```
 4413  012300  104000                  100$:   SCOPE
 4414
 4415                                  ;TEST 024-ERASE
 4416                                  ;THIS TEST MEASURES TIME FROM 'GO'=1 TO 'RDY'=1.
 4417  012302  112737  000024  001122  TST024: MOVB    #24,@#TSTNUM
 4418  012310  012702  012372                  MOV     #2$,R2                  ;SET RETURN PC FROM TIMER
 4419  012314  004737  005262                  JSR     PC,.REWIND              ;REWIND SLAVE
  (1)  012320  102436                          BVS     99$                     ;BRANCH IF ERROR ON REWIND
 4420  012322  004737  005120                  JSR     PC,RHINIT               ;SET NRZ
 4421  012326  004737  005344                  JSR     PC,WRITE                ;WRITE A RECORD
 4422  012332  005215                          INC     (R5)                    ;SET 'GO' BIT
 4423  012334  004737  005156                  JSR     PC,WAITRDY
  .24  012340  102426                          BVS     99$
 .425  012342  012737  012350  001002          MOV     #1$,@#SCPADR
 4426  012350  004337  005500          1$:     JSR     R3,@#TMCMD
 4427  012354  000000                          .WORD   0
 4428  012356  000000                          .WORD   0
 4429  012360  000000                          .WORD   0
 4430  012362  000024                          .WORD   ERASE
 4431  012364  004737  004250                  JSR     PC,TIMON                ;TURN TIMER ON
 4432  012370  005215                          INC     (R5)                    ;SET 'GO' BIT
 4433
 4434  012372  105711                  2$:     TSTB    (R1)                    ;BRANCH WHEN READY SETS
 4435  012374  100402                          BMI     3$
 4436  012376  000163  004340                  JMP     TIMER(R3)               ;GO TO TIMER & RETURN VIA R2
 4437
 4438  012402  004737  005156          3$:     JSR     PC,WAITRDY
 4439  012406  102403                          BVS     99$
 4440  012410  004737  004376                  JSR     PC,TIMOK
 4441  012414  000401                          BR      100$
 4442
 4443  012416  104400                  99$:    HLT
 4444  012420  104000                  100$:   SCOPE
 4445
 4446                                  ;TEST 025 TAPE MARK
 4447                                  ;THIS TEST MEASURES TIME FROM 'GO'=1 TO 'RDY'=1.
 4448  012422  112737  000025  001122  TST025: MOVB    #25,@#TSTNUM
 4449  012430  012702  012472                  MOV     #1$,R2                  ;SET RETURN PC FROM TIMER
 4450  012434  004737  005344                  JSR     PC,WRITE                ;WRITE A RECORD
 4451  012440  005215                          INC     (R5)                    ;SET 'GO' BIT
 4452  012442  004737  005156                  JSR     PC,WAITRDY
 4453  012446  102423                          BVS     99$
 4454  012450  004337  005500                  JSR     R3,@#TMCMD
 4455  012454  000000                          .WORD   0
 4456  012456  000000                          .WORD   0
 4457  012460  000000                          .WORD   0
 4458  012462  000026                          .WORD   WFMK
 4459  012464  004737  004250                  JSR     PC,TIMON                ;TURN TIMER ON
 4460  012470  005215                          INC     (R5)                    ;SET 'GO' BIT
 4461
 4462  012472  105711                  1$:     TSTB    (R1)                    ;BRANCH WHEN READY SETS
 4463  012474  100402                          BMI     2$
 4464  012476  000163  004340                  JMP     TIMER(R3)               ;GO TO TIMER & RETURN VIA R2
 4465
 4466  012502  004737  005156          2$:     JSR     PC,WAITRDY
 4467  012506  102403                          BVS     99$
```

```
 4468  012510  004737  004376                JSR     PC,TIMOK
 4469  012514  000401                        BR      100$
 4470
 4471  012516  104400                99$:    HLT
 4472  012520                      100$:
  (1)  012520  004737  005262                JSR     PC,.REWIND              ;REWIND SLAVE
  (1)  012524  102774                        BVS     99$                     ;BRANCH IF ERROR ON REWIND
 4473  012526  104000                        SCOPE
 4474
```

```
 4476  012530  032777  002000  166242  FINISH: BIT     #SW10,@SWR              ;DO NOT SPACE PAPER
 4477  012536  001011                          BNE     2$                      ;IF USER SELECTED NO OUTPUT
 4478  012540  005737  005746                  TST     CHNFLG                  ;OR IF IN CHAIN MODE
 4479  012544  001006                          BNE     2$
 4480  012546  012700  000012                  MOV     #10.,R0                 ;SET LINE FEED COUNT
 4481  012552  000004  001374          1$:     TYPE,CRLF
 4482  012556  005300                          DEC     R0
 4483  012560  001374                          BNE     1$
 4484
 4485
 4486  012562  105237  001005          2$:     INCB    @#SLVNUM                ;SET NEXT SLAVE #
 4487  012566  005237  001006                  INC     @#SLVPTR                ;AND ITS POINTER
 4488  012572  122737  000010  001005          CMPB    #8.,@#SLVNUM            ;BRANCH IF LAST SLAVE (7)
 4489  012600  001402                          BEQ     3$
 4490  012602  000137  007000                  JMP     @#BEGIN                 ;BEGIN TEST ON NEXT SLAVE
 4491  012606  105037  001005          3$:     CLRB    @#SLVNUM                ;SET SLAVE #0
 4492  012612  105237  001004                  INCB    @#DRVNUM                ;AND INCREMENT DRIVE #
 4493  012616  122737  000010  001004          CMPB    #8.,@#DRVNUM            ;AND CHECK IF LAST DRIVE
 4494  012624  001402                          BEQ     END
 4495  012626  000137  007000                  JMP     @#BEGIN
 4496
 4497  012632  105737  001125          END:    TSTB    @#UNTFND                ;BRANCH IF A UNIT WAS FOUND
 4498  012636  001007                          BNE     1$
 4499  012640  000004  014265                  TYPE,E.UNIT
 4500  012644  005737  005746                  TST     CHNFLG                  ;CHAIN MODE ?
 4501  012650  001002                          BNE     1$                      ;YES BRANCH
 4502  012652  000137  005650                  JMP     @#INIT
 4503  012656  105237  001127          1$:     INCB    @#PSCNT         ;INCREMENT PASS COUNT
 4504  012662  000004  013456                  TYPE,M.EOP
 4505  012666  113702  001127                  MOVB    @#PSCNT,R2      ;GET PASSCOUNT
 4506  012672  004737  002426                  JSR     PC,TYPOCT       ;AND TYPE IT
 4507  012676  000004  001374                  TYPE,CRLF
 4508  012702  013700  000042                  MOV     @#42,R0         ;GET ACT11 RETURN ADDRESS
  (1)  012706  001405                          BEQ     HERE            ;BRANCH IF NOT ACT11
  (1)  012710  000005                          RESET
  (1)  012712  004710                  $ENDAD: JSR     PC,(R0)
  (1)  012714  000240                          NOP
  (1)  012716  000240                          NOP
  (1)  012720  000240                          NOP
  (1)  012722  000240                  HERE:   NOP
 4509  012724  005737  005746                  TST     CHNFLG          ;BRANCH IF CHAIN MODE
 4510  012730  001004                          BNE     1$
 4511  012732  032777  000100  166040          BIT     #SW06,@SWR              ;BRANCH IF NOT CONTINOUS LOOP
 4512  012740  001402                          BEQ     2$
 4513  012742  000137  006756          1$:     JMP     @#RSTRT                 ;RESTART
 4514  012746  000000                  2$:     HALT
 4515  012750  000005                          RESET
 4516  012752  000137  005650                  JMP     @#INIT                  ;RESTART
```

```
 4518                                   ;SKEW TAPE TIMING TESTS
 4519                                   ;THE FOLLOWING TESTS REQUIRE A SPECIALLY WRITTEN 800 BPI SKEW TAPE
 4520  012756  012737  012764  001002  SKEWTST:MOV     #TST026,@#SCPADR        ;SET SCOPE POINTER
 4521
 4522                                   ;TEST 026- SKEW TAPE SPEED TEST-FORWARD
 4523                                   ;THIS TEST READS 32" OF TAPE (26400.-800. = 25600. FRAMES),THEN
 4524                                   ;DIVIDES TIME BY 32. TO GET TIME TO READ 1" (800. FRAMES) OF TAPE.
 4525  012764  112737  000026  001122  TST026: MOVB    #26,@#TSTNUM
 4526  012772  012702  013050                 MOV     #2$,R2                  ;SET RETURN PC FROM TIMER
 4527  012776  004737  005262                 JSR     PC,.REWIND              ;REWIND SLAVE
  (1)  013002  102441                         BVS     99$                     ;BRANCH IF ERROR ON REWIND
 4528  013004  052765  001700  000032         BIS     #NORM11,TC(R5)  ;SET 800 BPI
 4529  013012  052765  000010  000010         BIS     #BAI,CS2(R5)            ;INHIBIT BUS ADDRESS INCREMENT
 4530  013020  004337  005500                 JSR     R3,@#TMCMD              ;READ 32" OF TAPE-FORWARD
 4531  013024  016046                         .WORD   RDBUF
 4532  013026  177777                         .WORD   -1.
 4533  013030  063440                 10$:    .WORD   26400.                  ;FRAME COUNT
 4534  013032  000070                         .WORD   RDFWD
 4535  013034  005215                         INC     (R5)                    ;SET 'GO' BIT
 4536
 4537  013036  022710  001440          1$:    CMP     #800.,(R0)              ;WAIT FOR FIRST 800 FRAMES
 4538  013042  101375                         BHI     1$                      ;TO BE READ
 4539
 4540  013044  004737  004250                 JSR     PC,TIMON                ;TURN TIMER ON
 4541  013050  023710  013030          2$:    CMP     @#10$,(R0)              ;WAIT FOR READING TO FINISH
 4542  013054  10340?                         BLO     3$
 4543  013056  00016_  004340                 JMP     TIMER(R3)               ;GO TO TIMER & RETURN VIA R2
 4544
 4545  013062  012700  000005          3$:    MOV     #5,R0                   ;DIVIDE TIME BY 32.
 4546  013066  006204                  4$:    ASR     R4
 4547  013070  005300                         DEC     R0
 4548  013072  001375                         BNE     4$
 4549  013074  004737  005120                 JSR     PC,RHINIT               ;INIT DRIVE
 4550  013100  004737  004376                 JSR     PC,TIMCK                ;CHECK TIME
 4551  013104  000401                         BR      100$
 4552
 4553  013106  104400                  99$:   HLT
 4554  013110  104000                  100$:  SCOPE
 4555
 4556                                   ;TEST 027-SKEW TAPE SPEED TEST-REVERSE
 4557                                   ;THIS TEST READS FORWARD 40" (32000. FRAMES) OF TAPE, THEN READS REVERSE
 4558                                   ;32" (26400.-800. = 25600. FRAMES) OF TAPE. THE TIME IS THEN DIVIDED BY
 4559                                   ;32. TO GET TIME TO READ 1" (800. FRAMES) OF TAPE.
 4560  013112  112737  000027  001122  TST027: MOVB    #27,@#TSTNUM
 4561  013120  012702  013246                 MOV     #3$,R2                  ;SET RETURN PC FROM TIMER
 4562  013124  004737  005262                 JSR     PC,.REWIND              ;REWIND SLAVE
  (1)  013130  102465                         BVS     99$                     ;BRANCH IF ERROR ON REWIND
 4563  013132  052765  001700  000032         BIS     #NORM11,TC(R5)
 4564  013140  052765  000010  000010         BIS     #BAI,CS2(R5)
 4565  013146  004337  005500                 JSR     R3,@#TMCMD              ;READ FORWARD 32000. FRAMES
 4566  013152  016046                         .WORD   RDBUF
 4567  013154  177777                         .WORD   -1.                     ;WORD COUNT
 4568  013156  076400                 10$.    .WORD   32000.                  ;FRAME COUNT
 4569  013160  000070                         .WORD   RDFWD                   ;READ FORWARD
 4570  013162  005215                         INC     (R5)                    ;SET 'GO' BIT
 4571
```

```
4572  013164  023710  013156          1$:     CMP     @#10$,(R0)
4573  013170  101375                          BHI     1$
4574
4575  013172  004737  005120                  JSR     PC,RHINIT               ;INIT DRIVE
4576  013176  004737  004634                  JSR     PC,DELAY                ;WAIT FOR TAPE MOTION TO STOP
4577  013202  052765  001700  000032          BIS     #NORM11,TC(R5)          ;SET 800 BPI
4578  013210  052765  000010  000010          BIS     #BAI,CS2(R5)            ;INHIBIT BUS ADDRESS INCRENENT
4579  013216  004337  005500                  JSR     R3,@#TMCMD              ;READ REVERSE 32'' OF TAPE
4580  013222  016046                          .WORD   RDBUF                   ;READ BUFFER
4581  013224  177777                          .WORD   -1.                     ;WORD COUNT
4582  013226  063440                  11$:    .WORD   26400.                  ;FRAME COUNT
4583  013230  000076                          .WORD   RDREV                   ;READ REVERSE
4584  013232  005215                          INC     (R5)                    ;SET 'GO' BIT
4585
4586  013234  022710  001440          2$:     CMP     #800.,(R0)              ;WAIT FOR FIRST 800 FRAMES
4587  013240  101375                          BHI     2$                      ;TO BE READ
4588
4589  013242  004737  004250                  JSR     PC,TIMON                ;TURN TIMER ON
4590  013246  023710  013226          3$:     CMP     @#11$,(R0)              ;WAIT FOR ALL FRAMES TO BE READ
4591  013252  103402                          BLO     4$
4592  013254  000163  004340                  JMP     TIMER(R3)               ;GO TO TIMER & RETURN VIA R2
4593
4594  013260  012700  000005          4$:     MOV     #5,R0                   ;DIVIDE TIME BY 32.
4595  013264  006204                  5$:     ASR     R4
4596  013266  005300                          DEC     R0
4597  013270  001375                          BNE     5$
4598  013272  004737  005120                  JSR     PC,RHINIT
4599  013276  004737  004376                  JSR     PC,TIMOK
4600  013302  000401                          BR      100$
4601
4602  013304  104400                  99$:    HLT
4603  013306                          100$:
 (1)  013306  004737  005262                  JSR     PC,.REWIND              ;REWIND SLAVE
 (1)  013312  102774                          BVS     99$                     ;BRANCH IF ERROR ON REWIND
4604  013314  104000                          SCOPE
4605
4606  013316  000137  012530                  JMP     @#FINISH
4607
4608
4609
```

```
 4611                                                   .SBTTL    PROGRAM MESSAGES
 4612                                                  ;OPERATOR INSTRUCTIONS
 4613  013322  005015  046524  031460  M.NAM:   .ASCII  <CR><LF>'TM03/TU45 DRIVE FUNCTION TIMER (CZTUSB0)'
       013330  052057  032125  020065
       013336  051104  053111  020105
       013344  052506  041516  044524
       013352  047117  052040  046511
       013360  051105  024040  055103
       013366  052524  041123  024460
 4614  013374  005015  054524  042520           .ASCIZ  <CR><LF>'TYPE <CR> TO TERMINATE RESPONSE & ^C TO RESTART'
       013402  036040  051103  020076
       013410  047524  052040  051105
       013416  044515  040516  042524
       013424  051040  051505  047520
       013432  051516  020105  020046
       013440  041536  052040  020117
       013446  042522  052123  051101
       013454  000124
 4615  013456  005015  047105  020104  M.EOP:   .ASCIZ  <CR><LF>'END OF PASS '
       013464  043117  050040  051501
       013472  020123     000
 4616  013475     015  044012  051101  M.HSWR: .ASCIZ  <CR><LF>'HARDWARE SWR IN USE'<CR><LF>
       013502  053504  051101  020105
       013510  053523  020122  047111
       013516  052440  042523  005015
       013524     000
 4617  013525     015  051012  046505  I.REM:   .ASCIZ  <CR><LF>'REMOVE TMDP FROM TU45 TO BE TESTED'
       013532  053117  020105  046524
       013540  050104  043040  047522
       013546  020115  052524  032464
       013554  052040  020117  042502
       013562  052040  051505  042524
       013570  000104
 4618  013572  005015  054524  042520  I.REG:   .ASCIZ  <CR><LF>'TYPE FIRST ADDRESS OF CONTROLLER  '
       013600  043040  051111  052123
       013606  040440  042104  042522
       013614  051523  047440  020106
       013622  047503  052116  047522
       013630  046114  051105  020040
       013636     000
 4619  013637     124  050131  020105  I.DRVS: .ASCIZ  %TYPE TM03 DRIVE #'S TO BE TESTED:  ALL %
       013644  046524  031460  042040
       013652  044522  042526  021440
       013660  051447  052040  020117
       013666  042502  052040  051505
       013674  042524  035104  020040
       013702  046101  020114     000
 4620  013707     106  051117  052040  I.SLVS: .ASCII  'FOR TM03 DRIVE '
       013714  030115  020063  051104
       013722  053111  020105
 4621  013726  026460  052040  050131  I.DRV:   .ASCIZ  %0- TYPE SLAVE #'S TO BE TESTED: ALL %
       013734  020105  046123  053101
       013742  020105  023443  020123
       013750  047524  041040  020105
       013756  042524  052123  042105
       013764  020072  046101  020114
```

```
                013772     000
 4622           013773     123  042520  042105  I.SPD:  .ASCIZ  'SPEED TESTS? (YES=1/NO=<CR>): NO '
                014000  052040  051505  051524
                014006  020077  054450  051505
                014014  030475  047057  036517
                014022  041474  037122  035051
                014030  047040  020117     000
 4623           014035     015  042412  042116  M.EOT:  .ASCIZ  <CR><LF>'END OF TAPE'<CR><LF>
                014042  047440  020106  040524
                014050  042520  005015     000
 4624
 4625                                            ;ERROR MESSAGES
 4626           014055     015  052012  040522  E.TRP4: .ASCIZ  <CR><LF>'TRAPPED TO 4'
                014062  050120  042105  052040
                014070  020117  000064
 4627           014074  047516  041440  047117  E.NCON: .ASCIZ  'NO CONTROLLER AT ADDRESS SPECIFIED'<CR><LF>
                014102  051124  046117  042514
                014110  020122  052101  040440
                014116  042104  042522  051523
                014124  051440  042520  044503
                014132  044506  042105  005015
                014140     000
 4628           014141     124  030115  020063  E.NDRV: .ASCIZ  'TM03 DRIVE '
                014146  051104  053111  020105
                014154     000
 4629           014155     104  044522  042526  E.NSLV: .ASCII  'DRIVE '
                014162     040
 4630           014163     060  051440  040514  E.DRV:  .ASCII  '0 SLAVE '
                014170  042526     040
 4631           014173     060  047040  052117  E.NAVA: .ASCIZ  '0 NOT AVAILABLE FOR TEST'<CR><LF>
                014200  040440  040526  046111
                014206  041101  042514  043040
                014214  051117  052040  051505
                014222  006524  000012
 4632           014226  051104  053111  020105  E.NDR1:.ASCII   'DRIVE '
 4633           014234  026060  051440  040514  E.NSL1:.ASCII   '0, SLAVE '
                014242  042526     040
 4634           014245     060  047040  052117  E.MSG:  .ASCIZ  '0 NOT ON LINE'<CR><LF>
                014252  047440  020116  044514
                014260  042516  005015     000
 4635           014265     116  020117  046524  E.UNIT: .ASCIZ  'NO TM03/TU45 UNIT FOUND TO TEST'<CR><LF>
                014272  031460  052057  032125
                014300  020065  047125  052111
                014306  043040  052517  042116
                014314  052040  020117  042524
                014322  052123  005015     000
 4636           014327     123  043117  020124  E.SFT:  .ASCIZ  'SOFT ERROR (DATA)'<CR><LF>
                014334  051105  047522  020122
                014342  042050  052101  024501
                014350  005015     000
 4637           014353     124  051505  020124  E.HDR:  .ASCIZ  'TEST # '
                014360  020043     000
 4638           014363     040  042504  044526  E.HDR1: .ASCII  ' DEVICE ERROR'<CR><LF>
                014370  042503  042440  051122
                014376  051117  005015
 4639           014402  051503  004461  041527          .ASCIZ  'CS1'<HT>'WC'<HT>'BA'<HT>'FC'<HT>'CS2'<HT>'DS'<HT>'ER'<HT>'TC'<CR><LF>
```

```
                014410  041011  004501  041506
                014416  041411  031123  042011
                014424  004523  051105  052011
                014432  006503  000012
 4640           014436  047440  052125  047440  E.HDR2: .ASCIZ  ' OUT OF RANGE ERROR'<CR><LF>
                014444  020106  040522  043516
                014452  020105  051105  047522
                014460  006522  000012
 4641           014464  005015  044524  042515  E.TIMOV:        .ASCIZ  <CR><LF>'TIMER OVERFLOWED'<CR><LF>
                014472  020122  053117  051105
                014500  046106  053517  042105
                014506  005015     000
 4642           014511     015  052012  046511  E.TIMEX:        .ASCIZ  <CR><LF>'TIME EXPIRED WAITING FOR RDY'<CR><LF>
                014516  020105  054105  044520
                014524  042522  020104  040527
                014532  052111  047111  020107
                014540  047506  020122  042122
                014546  006531  000012
 4643           014552  043440  050101  021440  E.GAP:  .ASCIZ  ' GAP # '
                014560  000040
 4644
 4645                                            ;TIME DOCUMENT LINES
 4646           014562  025052  025052  025052  L.HDR1: .ASCIZ  '*******************************************************************************
                014570  025052  025052  025052
                014576  025052  025052  025052
                014604  025052  025052  025052
                014612  025052  025052  025052
                014620  025052  025052  025052
                014626  025052  025052  025052
                014634  025052  025052  025052
                014642  025052  025052  025052
                014650  025052  025052  025052
                014656  025052  025052  025052
                014664  025052  025052  025052
                014672  005015     000
 4647           014675     052  052040  030115  L.HDR2: .ASCII  '* TM03 DRIVE FUNCTION TIMES- DRIVE # '
                014702  020063  051104  053111
                014710  020105  052506  041516
                014716  044524  047117  052040
                014724  046511  051505  020055
                014732  051104  053111  020105
                014740  020043
 4648           014742  020060  046123  053101  L.DRV:  .ASCII  '0 SLAVE # '
                014750  020105  020043
 4649           014754  020060     040          L.SLV:  .ASCII  '0  '
 4650           014757     071  041440  040510  L.CHAN: .ASCIZ  '9 CHAN. SER # '
                014764  027116  051440  051105
                014772  021440  000040
 4651           014776  006440  025012  005015  L.HDR3: .ASCII  ' '<CR><LF>'*'<CR><LF>
 4652           015004  020052  052506  041516          .ASCIZ  '* FUNCTION'<HT><HT>'TIME(SPECIFICATION)'<HT>'TIME(ACTUAL)'<CR><LF>
                015012  044524  047117  004411
                015020  044524  042515  051450
                015026  042520  044503  044506
                015034  040503  044524  047117
                015042  004451  044524  042515
                015050  040450  052103  040525
```

```
        015056  024514  005015     000
4653
4654    015063     122  047101  042507  L.RNG:   .ASCIZ  'RANGE=<'
        015070  036075     000
4655    015073     101  052103  040525  L.ACT:   .ASCIZ  'ACTUAL='
        015100  036514     000
4656
4657                                    ;TEST DESCRIPTOR HEADERS
4658    015103     052  053440  044522  A.T001:  .ASCIZ  '* WRITE FROM BOT'<HT>
        015110  042524  043040  047522
        015116  020115  047502  004524
        015124     000
4659    015125     052  053440  044522  A.T002:  .ASCIZ  '* WRITE START'<HT><HT>
        015132  042524  051440  040524
        015140  052122  004411     000
4660    015145     052  053440  044522  A.T003:  .ASCIZ  '* WRITE SHUTDOWN'<HT>
        015152  042524  051440  052510
        015160  042124  053517  004516
        015166     000
4661    015167     052  053440  044522  A.T004:  .ASCIZ  '* WRITE SETTLEDOWN'<HT>
        015174  042524  051440  052105
        015202  046124  042105  053517
        015210  004516     000
4662    015213     052  051040  040505  A.T005:  .ASCIZ  '* READ FROM BOT'<HT><HT>
        015220  020104  051106  046517
        015226  041040  052117  004411
        015234     000
4663    015235     052  051040  040505  A.T006:  .ASCIZ  '* READ START'<HT><HT>
        015242  020104  052123  051101
        015250  004524  000011
4664    015254  020052  042522  042101  A.T007:  .ASCIZ  '* READ SHUTDOWN'<HT><HT>
        015262  051440  052510  042124
        015270  053517  004516  000011
4665    015276  020052  042522  042101  A.T010:  .ASCIZ  '* READ SETTLEDOWN'<HT>
        015304  051440  052105  046124
        015312  042105  053517  004516
        015320     000
4666    015321     052  051040  040505  A.T011:  .ASCIZ  '* READ REV START'<HT>
        015326  020104  042522  020126
        015334  052123  051101  004524
        015342     000
4667    015343     052  051040  040505  A.T012:  .ASCIZ  '* READ REV SHUTDOWN'<HT>
        015350  020104  042522  020126
        015356  044123  052125  047504
        015364  047127  000011
4668    015370  020052  042522  042101  A.T013:  .ASCIZ  '* READ REV SETTLEDOWN'<HT>
        015376  051040  053105  051440
        015404  052105  046124  042105
        015412  053517  004516     000
4669    015417     052  052040  051125  A.T014:  .ASCIZ  '* TURN AROUND DELAY F-R'<HT>
        015424  020116  051101  052517
        015432  042116  042040  046105
        015440  054501  043040  051055
        015446  000011
4670    015450  020052  052524  047122  A.T015:  .ASCIZ  '* TURN AROUND DELAY R-F'<HT>
        015456  040440  047522  047125
```

```
        015464  020104  042504  040514
        015472  020131  026522  004506
        015500     000
 4671   015501     052  043440  050101  A.T016: .ASCIZ  '* GAP SIZE-STOP HALF'<HT>
        015506  051440  055111  026505
        015514  052123  050117  044040
        015522  046101  004506     000
 4672   015527     052  043440  050101  A.T017: .ASCIZ  '* GAP SIZE-START HALF'<HT>
        015534  051440  055111  026505
        015542  052123  051101  020124
        015550  040510  043114  000011
 4673   015556  020052  040507  020120  A.T020: .ASCIZ  '* GAP SIZE-INTERRECORD'<HT>
        015564  044523  042532  044455
        015572  052116  051105  042522
        015600  047503  042122  000011
 4674   015606  020052  040507  020120  A.T021: .ASCIZ  '* GAP CONSISTANCY'<HT>
        015614  047503  051516  051511
        015622  040524  041516  004531
        015630     000
 4675   015631     052  042040  052101  A.T022: .ASCIZ  '* DATA TIME-800BPI'<HT>
        015636  020101  044524  042515
        015644  034055  030060  050102
        015652  004511     000
 4676   015655     052  042040  052101  A.T023: .ASCIZ  '* DATA TIME-1600BPI'<HT>
        015662  020101  044524  042515
        015670  030455  030066  041060
        015676  044520  000011
 4677   015702  020052  051105  051501  A.T024: .ASCIZ  '* ERASE GAP TIME'<HT>
        015710  020105  040507  020120
        015716  044524  042515  000011
 4678   015724  020052  051127  052111  A.T025: .ASCIZ  '* WRITE FILE MARK'<HT>
        015732  020105  044506  042514
        015740  046440  051101  004513
        015746     000
 4679   015747     052  052040  050101  A.T026: .ASCIZ  '* TAPE SPEED-FWD'<HT>
        015754  020105  050123  042505
        015762  026504  053506  004504
        015770     000
 4680   015771     052  052040  050101  A.T027: .ASCIZ  '* TAPE SPEED-REV'<HT>
        015776  020105  050123  042505
        016004  026504  042522  004526
        016012     000
 4681
 4682   016013     015  057012  000107  L.CNTG: .ASCIZ  <CR><LF>'^G'
 4683   016020  005015  053523  036522  L.SWR:  .ASCIZ  <CR><LF>'SWR='
        016026     000
 4684   016027     040  047040  053505  L.NEW:  .ASCIZ  '  NEW= '
        016034  020075     000
 4685   016037     015  037412  005015  L.QUEST:        .ASCIZ  <CR><LF>'?'<CR><LF>
        016044     000
 4686           016046                          .EVEN
 4687           016046                  RDBUF=.
 4688           016046                  WTBUF=.
 4689   016046  000200                  .BLKW   128.
 4690           000001                  .END
```

```
A       010756          4112#   4115
ACCL    = 100000        2492#
ANGTAB  001412          2587#   2845    2873
AS      = 000016        2391#
ASFLG   001130          2567#   3524*   3740    3742*
ATA     = 100000        2460#
ATIME   001012          2553#   2847    2875    2936*   3117    3211*   3215    3217    3247*   3251    3253
ATIMTB  001014          2554#   2914    3117*
A.T001  015103          2651    4658#
A.T002  015125          2652    4659#
A.T003  015145          2653    4660#
A.T004  015167          2654    4661#
A.T005  015213          2655    4662#
A.T006  015235          2656    4663#
A.T007  015254          2657    4664#
A.T010  015276          2658    4665#
A.T011  015321          2659    4666#
A.T012  015343          2660    4667#
A.T013  015370          2661    4668#
A.T014  015417          2662    4669#
A.T015  015450          2663    4670#
A.T016  015501          2664    4671#
A.T017  015527          2665    4672#
A.T020  015556          2666    4673#
A.T021  015606          2667    4674#
A.T022  015631          2668    4675#
A.T023  015655          2669    4676#
A.T024  015702          2670    4677#
A.T025  015724          2671    4678#
A.T026  015747          2672    4679#
A.T027  015771          2673    4680#
A16     = 000400        2416#
A17     = 001000        2417#
BA      = 000004        2386#   3468*
BAI     = 000010        2432#   4529    4564    4578
BEGIN   007000          3704#   4490    4495
BELL    001403          2583#   3065
BKSLSH  001377          2581#   2963
BOT     = 000002        2447#   3401
BPI200= 000000          2488#
BPI556= 000400          2489#
BPI800= 001000          2490#
BPTVEC= 000014          2371#
CDM11 - 000320          2487#
CHKDRV  006300          3573    3577    3588#
CHKSLV  006610          3623    3643    3657#
CHNFLG  005746          3130    3526#*  3558    3744    3782    4478    4500    4509
CH7     = 010000        2482#
CLR     = 000040        2434#
CNTRLA= 000001          2506#   2998
CNTRLC= 000003          2507#   2970
CNTRLG= 000007          2508#   3004
CNTRLO= 000017          2512#   2987
CNTRLU= 000025          2513#   2965
CNVDEC  002526          2788#
CNVOCT  002420          2750#
```

```
CNVTAO  003056          2714    2888#   3539
CNVTD   002540          2789    2796#
CNVTO   002432          2751    2759#
CR    = 000015          2511#   2580    2712    2893    2976    3537    3565    3576    3634    3645    3749    4613    4614
                        4615    4616    4617    4618    4623    4626    4627    4631    4634    4635    4636    4638    4639
                        4640    4641    4642    4646    4651    4652    4682    4683    4685
CRLF    001374          2580#   2849    2880    2968    2980    3035    3038    3061    3793    4481    4507
CSITM = 002000          2471#
CS1   = 000000          2384#
CS2   = 000010          2388#   3330*   3341*   3357*   3358*   4529*   4564*   4578*
DASH    001405          2584#   2842    2870
DB    = 000022          2392#
DCONST  002634          2800    2804    2807    2816#
DELAY   004634          3103    3266#   3458    4028    4180    4208    4576
DELAYV  004664          3278#   4278
DELTIM  001114          2557#   3278    3284    4272*   4279*
DIGTAB  001132          2569#   2773    2805    3347    3348    3601    3628    3684    3685    3799    3800
DIVIDE  004722          3209    3245    3302#
DLT   = 100000          2444#
DPR   = 000400          2454#
DRIVES  006134          3552#   3607
DRVAVA  005012          3330#   3596
DRVNUM  001004          2549#   3330    3341    3358    3595*   3627*   3678*   3698*   3704    3709    3732*   3797    4492*
                        4493
DRVTBL  001154          2577#   3553    3568    3583*   3589    3604*   3617    3663    3713
DRY   = 000200          2453#
DRYCLR= 000010          2404#
DS    = 000012          2389#   3108    3345    3368    3374    3383    3401    3403    3407    3766    4007
DT    = 000026          2394#   3331    3343
DTE   = 010000          2473#
DVA   = 004000          2419#
DV0   = 000000          2424#
DV1   = 000001          2425#
DV2   = 000002          2426#
DV3   = 000003          2427#
DV4   = 000004          2428#
DV5   = 000005          2429#
DV6   = 000006          2430#
DV7   = 000007          2431#
ECHO    001401          2582#   2974*   2978
EMTVEC= 000030          2371#
END     012632          3720    4494    4497#
EOT   = 002000          2455#   3374
ER    = 000014          2390#   3040
ERASE = 000024          2406#   4430
ERFLG   001123          2562#   3021*   3105*   3113    3552*   3603*   3606    3610*   3687*   3691
ERR   = 040000          2459#   3383    3403    3865    3890    3896    3967    3996    4002    4058    4088    4094    4359
                        4396
ERRTRP  003542          3013#   3549    3693
ERRVEC= 000004          2371#   3549*   3693*
E.DRV   014163          3684*   4630#
E.GAP   014552          2877    4643#
E.HDR   014353          3024    3787    4637#
E.HDR1  014363          3051    4638#
E.HDR2  014436          3031    4640#
E.MSG   014245          3348*   4634#
```

```
E.NAVA  014173          3601*  3602   3685*  4631#
E.NCON  014074          3547   4627#
E.NDRV  014141          3600   4628#
E.NDR1  014226          3349   4632#
E.NSLV  014155          3686   4629#
E.NSL1  014234          3347*  4633#
E.SFT   014327          3048   4636#
E.TIME  014511          3372   4642#
E.TIMO  014464          3178   4641#
E.TRP4  014055          4626#
E.UNIT  014265          4499   4635#
FC    = 000006          2387#  3110   3444*  3454*  3470*  3764   4302
FCE   = 001000          2470#
FINISH  012530          4476#  4606
FMT   = 000020          2466#
FPEVEC= 000244          2371#
FRMCNT= 177400          2503#  3419   3429   3439   3965   4056
FWDSPC  005416          3444#
GAP     001120          2559#  2864   2878   3248   4312*  4318*  4319
GAPOK   004506          3232#  4317
GAPTBL  001054          2556#  4292   4314   4323
GO    = 000001          2400#
GTIMTB  001556          2631#  2868   2871   3251   3253
GTSWR   002020          2704#  3007   3756
HERE    012722          4508#
HLT   = 104400          2497#  3179   3219   3255   3808   3834   3854   3879   3911   3932   3954   3982   4016
                        4042   4072   4108   4114   4140   4170   4194   4222   4254   4339   4377   4412   4443
                        4471   4553   4602
HT    = 000011          2509#  2587   4639   4652   4658   4659   4660   4661   4662   4663   4664   4665   4666
                        4667   4668   4669   4670   4671   4672   4673   4674   4675   4676   4677   4678   4679
                        4680
IDB   = 000010          2449#
IE    = 000100          2414#
ILF   = 000001          2462#
ILR   = 000002          2463#
INBUF   001264          2579#  2712   2889   2954   3511*  3537   3562   3631   3748
INCVAE= 000100          2467#
INIT    005650          2502   2538   2973   2996   3510#  3548   4502   4516
IOTVEC= 000020          2371#
IR    = 000100          2435#
ITCNT   001121          2560#  2915   3115   3118*  3123   3133*  3768*
I.DRV   013726          3628*  4621#
I.DRVS  013637          3560   4619#
I.REG   013572          3532   4618#
I.REM   013525          3529   4617#
I.SLVS  013707          3629   4620#
I.SPD   013773          3746   4622#
LF    = 000012          2510#  2580   4613   4614   4615   4616   4617   4618   4623   4626   4627   4631   4634
                        4635   4636   4638   4639   4640   4641   4642   4646   4651   4652   4682   4683   4685
LKS   = 177546          2375#
LKVEC = 000100          2376#
LPB   = 177516          2378#
LPS   = 177514          2377#
L.ACT   015073          2846   2874   4655#
L.CHAN  014757          3801*  4650#
L.CNTG  016013          4682#
```

```
L.DRV   014742           3799*   4648#
L.HDR1  014562           3796    4646#
L.HDR2  014675           3802    4647#
L.HDR3  014776           3804    4651#
L.NEW   016027           2710    4684#
L.QUES  016037           4685#
L.RNG   015063           2839    2867    4654#
L.SLV   014754           3800*   4649#
L.SWR   016020           2707    4683#
MCPE  = 020000           2420#
MDPE  = 000400           2437#
MMVEC = 000250           2371#
MOL   = 010000           2457#   3345
MR    = 000024           2393#   3148    3150    3155    3169    3184
MXF   = 001000           2438#
M.EOP   013456           4504    4615#
M.EOT   014035           3377    4623#
M.HSWR  013475           3003    4616#
M.NAM   013322           3376    3530    3531*   4613#
NAMPTR  001656           2650#   2940    3790
NED   = 010000           2441#
NEF   = 004000           2472#
NEM   = 004000           2440#
NOP   = 000000           2401#
NORM11= 001700           2486#   3362    4349    4528    4563    4577
NSG   = 000400           2469#
OCTAL0  001116           2558#   2715    2890    3540
ODIGIT  001144           2574#   2760    2779    2798    2812    3478    3505
OPI   = 020000           2474#
OR    = 000200           2436#
OSC   = 000100           2478#   3148    3150    3155    3169    3184
OUTBUF= 005650           2502#
OUTGAP  002744           2862#   3034    3259
OUTSPC  002650           2834#   2944    3037    3094
PARVEC= 000114           2371#
PAT   = 000020           2433#
PEFLRC= 000200           2468#
PES   = 000040           2451#
PE1600= 002300           2491#   3041    4386
PFVEC = 000024           2371#   2522
PGE   = 002000           2439#
PIP   = 020000           2458#   3407
PIRQ  = 177772           2370#
PIRVEC= 000240           2371#
PLKCSR= 172540           2373#
PLKVEC= 000104           2374#
PRGFLG  001124           2563#   3523*   3567*   3598    3625    3636*   3642    3682    3743*   3753*   3774    3805
PSCNT   001127           2566#   3119    3525*   4503*   4505
PSEL  = 002000           2418#
PSW   = 177776           2370#
PUBLIS  003144           2913#   3132
RDBUF = 016046           3427    3437    4531    4566    4580    4687#
RDFWD = 000070           2412#   3430    4534    4569
RDREV = 000076           2413#   3440    4583
RDSW    002126           2722#*
RDY   = 000200           2415#
```

```
READ    005362          3426#  3918   3940   3962   3991   4157   4209   4293   4298
RESVEC= 000010          2371#
REVRD   005400          3436#  4029   4053   4083   4127   4151   4181   4204   4237   4241   4284
RHINIT  005120          3104   3333   3357#  3394   4420   4549   4575   4598
RMR   = 000004          2464#
RSTRT   006756          2540   3698#  4513
RWD   = 000006          2403#  3399
RWDOFF= 000002          2402#
SC    = 100000          2422#
SCOPE = 104000          2498#  3835   3855   3880   3912   3933   3955   3983   4017   4043   4073   4109   4141
                        4171   4195   4223   4255   4340   4378   4413   4444   4473   4554   4604
SCPADR  001002          2548#  3102*  3106   3127*  3180   3794*  3809*  4425*  4520*
SDWN  = 000020          2450#  3871   3894   3902   3973   4000   4007   4063   4092   4099
SKEWTS  012756          3807   4520#
SLA   = 000001          2446#
SLAVES  006400          3610#  3692
SLR   = 177774          2370#
SLVAVA  005040          3341#  3680
SLVNUM  001005          2550#  3342   3360   3679*  3699*  3705   3737*  3798   4486*  4488   4491*
SLVPTR  001006          2551#  3700*  3706   3738*  4487*
SLVTBL  001164          2578#  3611   3616   3637   3659   3700
SN    = 000030          2395#  3477   3487   3492   3500
SNPT    005520          3477#  3803
SPACE   001410          2586#  3535
SPACE2  001407          2585#  3056
SPCFWD= 000030          2407#  3445
SPCREV= 000032          2408#  3455
SPR   = 002000          2481#  3343
SSC   = 000100          2452#
STIMTB  001416          2597#  2840   2843   3215   3217
STKPTR= 000600          2543#  3510
SWR     001000          2547#  2704   2708   2715*  3000   3002*  3006*  3022   3063   3066   3091   3095   3097
                        3121   3125   3128   3257   3516   3520*  3526*  3754   3784   4476   4511
SWREG   000176          2535#  2704   3000   3006   3520   3526   3754
SW06  = 000100          2326#  2535   4511
SW07  = 000200          2325#
SW08  = 000400          2324#  3091   3257
SW09  = 001000          2323#  3063
SW10  = 002000          2322#  3128   4476
SW11  = 004000          2321#  3121
SW13  = 020000          2320#  3022
SW14  = 040000          2319#  3095
SW15  = 100000          2318#
TAP   = 040000          2483#  3331
TBITVE= 000014          2371#  2517
TC    = 000032          2396#  3041   3059   3342*  3361*  3362*  3826   3845   3923   3945   4033   4131   4161
                        4349*  4385*  4386*  4528*  4563*  4577*
TCRLF   002156          2722#
TIB     002016          2701#
TIMER   004340          3170   3174#  3182   3185   3269   3283   3767   3828   3847   3873   3904   3925   3947
                        3975   4009   4035   4065   4101   4133   4163   4187   4215   4247   4304   4366   4403
                        4436   4464   4543   4592
TIMERR  004350          3176   3178#
TIMER0  004364          3184#
TIMER1  004314          3169#  3173
TIMOK   004376          3196#  3832   3851   3877   3908   3929   3951   3979   4013   4039   4069   4105   4137
```

```
TIMON   004250  4167  4191  4219  4251  4336  4374  4409  4440  4468  4550  4599
                3146# 3266  3280  3823  3842  3869  3900  3920  3942  3971  4006  4030  4062
                4098  4128  4158  4182  4210  4242  4299  4363  4400  4431  4459  4540  4589
TKB    = 177562 2370# 2958  2985
TKISR   003420  2530  2985#
TKS    = 177560 2370# 2955  3769*
TKVEC  = 000060 2371# 2529
TMBASE  001010  2552# 3090  3533  3540* 3541  3762
TMCMD   005500  3395  3416  3426  3436  3468# 4350  4387  4426  4454  4530  4565  4579
TMCS1  = 172440 2381# 2552
TMK    = 000004 2448#
TPB    = 177566 2370#
TPS    = 177564 2370#
TPVEC  = 000064 2371#
TRAPVE= 000034  2371#
TRE    = 040000 2421#
TRTVEC= 000014  2371#
TSTNUM  001122  2561# 2938  2943  3025  3093  3100  3212  3761* 3819* 3839* 3859* 3884* 3916*
                3937* 3959* 3987* 4022* 4047* 4077* 4120* 4145* 4174* 4198* 4227* 4269* 4346*
                4382* 4417* 4448* 4525* 4560*
TSTTBL  001736  2676# 3794  3795
TST000  007224  2676  3102  3761#
TST001  007474  2677  3809  3819#
TST002  007560  2678  3839#
TST003  007636  2679  3859#
TST004  007726  2680  3884#
TST005  010034  2681  3916#
TST006  010120  2682  3937#
TST007  010204  2683  3959#
TST010  010304  2684  3987#
TST011  010424  2685  4022#
TST012  010522  2686  4047#
TST013  010632  2687  4077#
TST014  010772  2688  4120#
TST015  011064  2689  4145#
TST016  011172  2690  4174#
TST017  011266  2691  4198#
TST020  011376  2692  4227#
TST021  011516  2693  4269#
TST022  012022  2694  4346#
TST023  012152  2695  4382#
TST024  012302  2696  4417#
TST025  012422  2697  4448#
TST026  012764  2698  4520  4525#
TST027  013112  2699  4560#
TYPDEC  002534  2795# 2841  2844  2848  2869  2872  2876
TYPE   = 000004 2499# 2707  2710  2722  2779  2812  2839  2842  2845  2846  2849  2867  2870
                2873  2874  2877  2880  2941  2963  2968  2978  2980  3003  3024  3031  3035
                3038  3048  3051  3056  3061  3065  3178  3349  3372  3376  3377  3505  3529
                3530  3532  3535  3547  3560  3600  3602  3629  3686  3746  3787  3791  3793
                3796  3802  3804  4481  4499  4504  4507
TYPE1   002150  2722#
TYPE2   002176  2722#
TYPE3   002204  2722#
TYPE4   002210  2722#
TYPFLG  001126  2565# 2750* 2758* 2777  2788* 2795* 2810
```

```
TYPOCT  002426  2709  2758#  2879  3026  3055  3060  3534  3788  4506
UBREAK= 177770  2370#
UNS   = 040000  2475#
UNTFND  001125  2564#  3701*  3731*  4497
UPE   = 020000  2442#
WAITRD  005156  3366#  3381  3446  3452  3456  3780  3830  3849  3875  3906  3927  3949  3977
                4011  4026  4037  4051  4067  4081  4103  4124  4135  4149  4154  4165  4178
                4189  4202  4206  4217  4231  4235  4239  4249  4276  4286  4296  4306  4372
                4407  4423  4438  4452  4466
WAITTI  005160  3367#  3370*
WC    = 000002  2385#  3469*
WCE   = 040000  2443#
WCHKF = 000050  2409#
WCHKR = 000056  2410#
WFMK  = 000026  2405#  4458
WFWD  = 000060  2411#  3420  4354  4391
WRDCNT= 177600  2504#  3418  3428  3438
WRITE   005344  3379  3416#  3450  3778  3822  3840  3860  3885  4024  4049  4079  4122  4147
                4176  4200  4229  4233  4274  4421  4450
WRL   = 004000  2456#
WRT.BK  005436  3450#  3938  3960  3989
WTBUF = 016046  3417  4351  4388  4688#
$CHARC  002120  2722#*
$CNTRL  002121  2722#*  2989*
$CRLF   002122  2722#
$ENDAD  012712  2528  2993  3526  4508#
$FILL   002111  2722#
$HT   = 000011  2722#
$NULL   002110  2722#
$SVPC - 000040  2528#
$TKFLG  002113  2722#
$TPB    002116  2722#*
$TPFLG  002112  2722#
$TPS    002114  2722#
.     = 016446  2514#  2517#  2518  2528#  2529#  2534#  2557#  2539#  2542#  2545#  2554#  2556#  2568#
                2574#  2576#  2577#  2578#  2579#  2588#  2722#  2923  3173#  3182#  4686#  4687  4688#
                4689#
.HLT    003544  2526  3020#
.INPUT  003272  2711  2953#  3536  3561  3630  3747
.RESTO  002376  2716  2737#  2780  2813  2904  2945  3069  3220  3261
.REWIN  005262  3378  3394#  3777  3821  3917  4112  4271  4348  4384  4419  4472  4527  4562
                4603
.SAVE   002354  2706  2726#  2759  2796  2888  2913  3020  3196  3232
.SCOPE  004010  2524  3090#
.TYPE   002130  2520  2722#
```

```
DTBOOT  1243#
GETANS   707#
INPUT   2352#  2711  3536  3561  3630  3747
LDPDR    515#
LPDP11  1268#
PSPTAG   740#
REGBOX   132#
RESLDR   873#
RESTOR  2349#  2780  2813  2904  2945  3069  3220  3261
REWIND  2355#  3378  3777  3821  3917  4112  4271  4348  4384  4419  4472  4527  4562  4603
SAVE    2346#  2759  2796  2888  2913  3196  3232
SAVLDR   855#
SETGO   2365#  3380  3400  3451  3779  3943  3963  3992  4025  4031  4050  4054  4080  4084  4123
        4129   4148  4152  4159  4177  4183  4201  4205  4211  4230  4234  4238  4243  4275  4285
        4294   4300  4355  4392  4422  4432  4451  4460  4535  4570  4584
SVTK$   1141#
TIMCHK  2362#  3269  3283  3828  3847  3873  3904  3925  3947  3975  4009  4035  4065  4101  4133
        4163   4187  4215  4247  4304  4366  4403  4436  4464  4543  4592
TIMEON  2359#  3280  3823  3869  3900  3920  3942  3971  4006  4030  4062  4098  4128  4158  4182
        4210   4242  4299  4363  4400  4431  4459  4540  4589
$CATCH  1124#  2303#  2514
$CHAIN    89#  2303#  3526
$CHNMO   105#
$CNV16   606#
$CNV18   635#
$CNV48   704#
$CPCHK   897#
$CPREG    17#  2303#  2370
$CPVEC   167#  2303#  2371
$FPREG    46#
$GETAN   771#
$KMMRE   347#
$KWPDR   998#
$KW11    929#
$LCTRL     2#
$LPREG   186#
$MAMFO  1176#
$MMBIT   207#
$MMREG   264#
$PDRBI   385#
$POWER   439#
$PSWBI   147#
$RECO    795#
$RESLD   876#
$RESTO   476#
$SAVE    464#
$SAVLD   858#
$SETTB   508#
$SHIFT   489#
$SMMRE   309#
$STINS     8#
$STKPT   202#
$ST200  1136#
$SVTK   1146#
$SWOPT    56#
$TCDRV  1029#
```

```
$TCREG    192#
$TRAPS    402#
$TYPE     518#    2303#    2722
$TYPEF    592#
$UMMRE    272#
$VECTA   1163#
.$ACT1     67#    2303#    2528
.$EOP      78#    2303#    4508


. ABS.  016446     000


 ERRORS DETECTED:  0

 CZTUSB,CZTUSB/CRF=CZTUSB.P11
 RUN-TIME: 16 26 2 SECONDS
 RUN-TIME RATIO: 100/45=2.2
 CORE USED:  15K  (29 PAGES)
```