

TA11

MOTION TEST
MD-11-DZTAD-C

EP-DZTAD-C-DL-A
COPYRIGHT © 1976
FICHE 1 OF 1

NOV 1976
digital
MADE IN U.S.A.

IDENTIFICATION

PRODUCT CODE: MAINDEC-11-DZTAD-C-D
PRODUCT NAME: TALL MOTION TEST.
DATE REVISED: 21 JUNE 76
MAINTAINER: DIAGNOSTIC ENGINEERING
AUTHOR: JIM LACEY

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

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

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

COPYRIGHT (C) 1973, 1976 BY DIGITAL EQUIPMENT CORPORATION

CONTENTS

1. ABSTRACT
2. REQUIREMENTS
 - 2.1 EQUIPMENT
 - 2.2 STORAGE
 - 2.3 PRELIMINARY PROGRAMS
3. LOADING PROCEDURE
4. STARTING PROCEDURE
 - 4.1 CONTROL SWITCH SETTINGS
 - 4.2 STARTING ADDRESS
 - 4.3 PROGRAM & OPERATOR ACTION
5. OPERATING PROCEDURE
 - 5.1 OPERATIONAL SWITCH SETTINGS
 - 5.2 SUBROUTINE ABSTRACTS
6. ERRORS
7. RESTRICTIONS
8. MISCELLANEOUS
 - 8.1 EXECUTION TIME
 - 8.2 STACK POINTER
 - 8.3 PASS COUNTER
 - 8.4 ITERATIONS
 - 8.5 SPECIAL REGISTERS
9. PROGRAM DESCRIPTION

1. ABSTRACT

THIS PROGRAM CONTAINS A SERIES OF TESTS THAT CHECK THE TUBO DRIVE FOR PROPER OPERATION.

2. REQUIREMENTS

2.1 EQUIPMENT

PDP-11 COMPUTER WITH OR WITHOUT HARDWARE SWITCH REGISTER WITH CONSOLE TELETYPE, AND A TA11 CASSETTE

2.2 STORAGE

THIS PROGRAM REQUIRES APPROX. 4K STORAGE.

2.3 PRELIMINARY PROGRAMS

MAINDEC-11-DZTAA
MAINDEC-11-DZTAB
MAINDEC-11-DZTAC

3. LOADING PROCEDURE

USE STANDARD PROCEDURE FOR LOADING .ABS TAPES OR A CASSETTE TAPE.

4. STARTING PROCEDURE

4.1 CONTROL SWITCH SETTINGS

SEE 5.1.

4.2 STARTING ADDRESSES

200 NORMAL STARTING ADDRESS
204 SELECT DRIVE(S) BEFORE STARTING TEST
210 SELECT DRIVE(S) AND ADDRESSES BEFORE STARTING TEST
214 SETUP FOR MANUAL LOOPING
220 WRITE FILE GAP FROM BOT TO EOT
224 WRITE CONTINUOUS BLOCKS OF DATA
230 READ CONTINUOUS BLOCKS OF DATA
234 WRITE FILE GAP AND A BLOCK OF DATA
240 READ BLOCK OF DATA AND INTO A FILE GAP
244 SPACE FWD FILE GAP FROM BOT TO EOT
250 BACK SPACE FILE GAPS

4.3 PROGRAM & OPERATOR ACTION

1. LOAD PROGRAM INTO MEMORY (SEE SECTION 3.)
2. LOAD A WRITE ENABLED CASSETTE IN BOTH DRIVES
3. REWIND BOTH DRIVES
4. LOAD ADDRESS 200.
5. SET SWITCHES (SEE SECTION 5.1)
6. PRESS START.
7. THE PROGRAM WILL LOOP & TTY BELL WILL RING ONCE EVERY PASS, IF SW<10>=0.

*** NOTE: IF USING THE SOFTWARE SWITCH REGISTER THE PROGRAM WILL TYPE "SWR=XXXXXX NEW=" AFTER TYPING THE NAME OF THE PROGRAM.

4.3.1 DRIVE SELECTION

STARTING THE PROGRAM AT 200 WILL RESULT IN AUTOMATIC SELECTION OF DRIVES "A" AND "B" TO BE TESTED.

NOTE: IF LOAD MEDIUM IS CASSETTE WITH STANDARD VECTOR PROGRAM WILL RESPOND AS IF STARTED AT 210.

STARTING THE PROGRAM AT 204, 210, OR 214 ALLOWS THE OPERATOR TO SELECT THE DRIVE(S) TO BE TESTED.

THE PROGRAM WILL TYPE "DRIVE(S)?".

EITHER OR BOTH DRIVES CAN BE SELECTED BY TYPING "A" AND/OR "B" FOLLOWED BY A CARRIAGE RETURN.

4.3.1.1 DRIVE SELECTION EXAMPLES

DRIVE(S)? A,B
DRIVE(S)? AB
DRIVE(S)? B,A
DRIVE(S)? B

4.3.2 ADDRESS SELECTION

STARTING THE PROGRAM AT 210 OR 214 ALLOWS THE OPERATOR TO CHANGE THE "CONTROL AND STATUS" AND "DATA BUFFER" REGISTER ADDRESSES, THE VECTOR ADDRESS AND THE PRIORITY LEVEL.

THE PROGRAM WILL ASK FOR THE DRIVES TO BE TESTED AS PER 4.3.1. AFTER THE DRIVES HAVE BEEN SELECTED IT WILL ASK FOR:

1. BUS ADDRESS OF THE CONTROL AND STATUS REGISTER (TACS)
2. VECTOR ADDRESS
3. PRIORITY LEVEL

AND THE OPERATOR MUST RESPOND WITH THE DESIRED PARAMETER OR A CARRIAGE RETURN (WHICH IMPLIES LEAVE AS IS). WHEN ALL PARAMETERS HAVE BEEN DEFINED THE PROGRAM WILL TYPE THEM BACK OUT AND ASK IF THEY ARE OK AT WHICH TIME THE OPERATOR RESPONDES WITH A "Y" OR A "CARRIAGE RETURN" FOR "YES" ANYTHING ELSE IS A "NO".

4.3.2.1 ADDRESS SELECTION EXAMPLES

DRIVES(S) A
TACS? 177500
VECTOR? 260
PRIORITY? 6
TACS=177500 TADB=177502 VECTOR=000260 PRIORITY=000300
OK?

DRIVES(S) A,B
TACS? 470
VECTOR?
PRIORITY?
TACS=177470 TADB=177472 VECTOR=000260 PRIORITY=000300
OK?

4.3.3 SUBTEST LOOPING

THE SCOPE ROUTINE (REFER 5.2.1) PROVIDES A MEANS BY WHICH THE OPERATOR CAN SPECIFY THE FIRST ADDRESS OF A SCOPE LOOP.

THE OPERATOR TYPES A "CONTROL C" (↑C) AND WHEN THE NEXT SCOPE STATEMENT IS EXECUTED THE PROGRAM WILL ASK FOR:

1. TEST PC--- THE FIRST ADDRESS OF THE TEST (MUST BE A SCOPE)
2. LOOP PC--- THE ADDRESS TO LOOP BACK TO

4.3.3.1 SUBTEST LOOPING EXAMPLES

:OPERATOR TYPES "↑C"
 TEST PC? 2242
 LOOP PC?

:OPERATOR TYPES "2242""CARRIAGE RETURN "(CR)"
 :OPERATOR TYPES "CR" WHICH
 :IMPLIES "LOOP PC"="TEST PC"

:OPERATOR TYPES "↑C"
 TEST PC? 3000
 LOOP PC? 3020

:OPERATOR TYPES "3000""CR"
 :OPERATOR TYPES "3020""CR"
 :PROGRAM USES THIS AS THE
 :FIRST ADDRESS OF THE SCOPE
 :LOOP

:OPERATOR TYPES "↑C"
 TEST PC?

:OPERATOR TYPES "CR"
 :PROGRAM CONTINUES
 :FROM THIS POINT

5. OPERATING PROCEDURE

SEQ 0007

5.1 OPERATIONAL SWITCH SETTINGS

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

CONTROL:

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

- 1) TYPE CONTROL G <↑G>: THIS WILL ALLOW THE TTY TO ENTER DATA INTO LOC. 176 AT SELECTED POINTS WITHIN THE PROGRAM.
- 2) THE MACHINE WILL THEN TYPE: SWR=XXXXXXNEW= (XXXXXX IS THE OCTAL CONTENTS OF THE SOFTWARE SWITCH REGISTER.)
- 3) AFTER THE ''NEW='' HAS BEEN TYPED THEN THE OPERATOR CAN DO ONE OF THE FOLLOWING AT THE TTY:
 - A) TYPE A NUMBER TO BE LOADED INTO LOC. 176 FOLLOWED BY A <CR>. (ONLY NUMBERS BETWEEN 0-7 WILL BE ACCEPTED AND ONLY 6 NUMBERS WILL BE ALLOWED)
IF A <CR> IS THE FIRST KEY DEPRESSED THE SOFTWARE SWITCH REGISTER CONTENTS WILL NOT BE CHANGED.
 - B) IF A CONTROL U <↑U> IS DEPRESSED THEN THE PROGRAM WILL SEND YOU BACK TO STEP 2.

WITH SW<15:08>=0 THE PROGRAM WILL PRINT OUT ON ERRORS AND CONTINUE IN TEST. BELL WILL RING AT COMPLETION OF A PASS.
THE SWITCH SETTINGS ARE:

SW<15>=1...HALT ON ERROR
 SW<14>=1...LOOP ON TEST
 SW<13>=1...INHIBIT ERROR TYPEOUTS
 SW<11>=1...INHIBIT ITERATIONS
 SW<10>=1...RING BELL ON ERROR
 SW<10>=0...RING BELL ON PASS COMPLETE
 SW<09>=1...LOOP ON ERROR
 SW<07>=1...LOCK ON CURRENT DRIVE
 SW<06>=1...DELAY AT END OF EACH FUNCTION
 SW<05>=1...RUN WITHOUT INTERRUPTS
 SW<04>=1...IGNORE BLOCK CHECK ERRORS
 SW<03>=1...INHIBIT DATA COMPARE

5.2 SUBROUTINE ABSTRACTS

5.2.1 SCOPE

THIS SUBROUTINE CALL (VIA AN IOT INSTRUCTION) IS PLACED BETWEEN EACH TEST IN THE INSTRUCTION SECTION. IT RECORDS THE STARTING ADDRESS OF EACH TEST IN LOCATION "\$LPADR" AND "\$LPERR" AS IT IS BEING ENTERED.

THIS ROUTINE SUPPORTS THE S/W SWITCH REG FUNCTIONS
NOTE: THIS ROUTINE CHECKS THE TTY INPUT BUFFER FOR A "CONTROL C" (REFER TO 4.3.3)

5.2.2 TRAPCATCHER

A ".+2" - "HALT" SEQUENCE IS REPEATED FROM LOC. 0 TO LOC. 776 TO CATCH ANY UNEXPECTED TRAPS. THUS, ANY UNEXPECTED TRAPS WILL HALT AT THE DEVICE TRAP VECTOR +2.

5.2.3 ERROR

THIS SUBROUTINE CALL (VIA A EMT INSTRUCTION) IS USED TO REPORT ALL ERRORS. (REFER TO 6.)

*** THIS ROUTINE SUPPORTS THE S/W SWITCH REG FUNCTIONS
*** IF THE PROCESSOR HALTS (BIT 15=1), OPERATOR CAN RESET S/W SWITCH REGISTER BY HITTING A "CONTROL G" (↑G) BEFORE HITTING CONTINUE.

5.2.4 TRAP

A NUMBER OF SUBROUTINES ARE CALLED BY THE TRAP INSTRUCTION. FOLLOWING IS THE CALLS USED AND THE STARTING ADDRESS OF THE ROUTINE.

5.2.4.1 TYPE (\$TYPE)

TYPE AN ASCIZ STRING ON THE TTY

5.2.4.2 RDCHR (\$RDCHR)

READ A SINGLE ASCII CHARACTER FROM THE TTY

5.2.4.3 RDLIN (\$RDLIN)

READ AN ASCII STRING FROM THE TTY

5.2.4.4 SETLOOP (T.SETLOOP)

SETUP TO LOOP AS PER THE TTY

5.2.5 THE FOLLOWING "TRAP" CALLS ARE WHAT ARE USED TO PERFORM THE TESTS.

THE ROUTINES THAT ARE CALLED IS WHAT MAKES UP THE HEART OF THIS PROGRAM.

5.2.5.1 WFG (T.WFG) WRITE A FILE GAP

5.2.5.2 WRITE (T.WRIT) WRITE A BLOCK OF DATA

5.2.5.3 READ (T.READ) READ A BLOCK OF DATA

5.2.5.4 BSFG (T.BSFG) BACK SPACE A FILE GAP

5.2.5.5 BSBG (T.BSBG) BACK SPACE A BLOCK GAP

5.2.5.6 SFFG (T.SFFG) SPACE FORWARD A FILE GAP

5.2.5.7 SFBG (T.SFBG) SPACE FORWARD A BLOCK GAP

5.2.5.8 REWIND (T.RWIND) REWIND THE TAPE TO BOT

5.2.5.9 SELDRV (T.SELDRV) SELECT A DRIVE

5.2.5.10 BLKCMP (T.BLKCMP) COMPARE READ AND WRITE BUFFERS

123

5.2.6 THE FOLLOWING SUBROUTINES ARE CALLED BY A JSR

5.2.6.1 DO.CMD

THIS ROUTINE WILL LOAD THE LOW BYTE OF THE "TACS" WITH THE FIRST BYTE FOLLOWING THE CALL.

WHEN THE FUNCTION IS TO BE PERFORMED WITH "INTERRUPT ENABLE"=1 THE TAIL VECTOR IS SET TO "SERV". WHEN THE FUNCTION IS TO BE PERFORMED WITH "INTERRUPT ENABLE"=0 THE VECTOR IS SET TO "BADINT"

NOTE SWR<5> PROVIDES OVERRIDE CAPABILITIES OF "INTERRUPT ENABLE"=1.

5.2.6.2 WAITFLAG

THIS ROUTINE IS CALLED AFTER A COMMAND HAS BEEN SENT TO THE TAIL. IT WAITS A PREDETERMINED AMOUNT OF TIME A TAKES ONE OF THREE EXITS.

THE EXITS FOLLOW THE CALL AND ARE:

1. ERROR NO FLAGS OCCURRED
2. ERROR NO INTERRUPT OCCURRED
3. NORMAL RETURN

5.2.6.3 FLAGS

THIS ROUTINE IS CALLED TO DETERMINE WHAT FLAGS ARE UP. THIS ROUTINE WILL TAKE ONE OF FOUR RETURNS DEPENDING ON THE FLAGS.

THE RETURNS FOLLOW THE CALL AND ARE:

1. "TRANSFER REQUEST"=0 AND "READY"=0
2. "TRANSFER REQUEST"=1
3. "READY"=1 AND "ERROR"=0
4. "READY"=1 AND "ERROR"=1

5.2.6.3 DO.CRC

THIS ROUTINE IS USED TO CALCULATE "CRC". IT WORKS ON ONE BYTE AT A TIME WHICH MUST BE IN R0 WHEN CALLED.

5.2.6.4 A2OCT

THIS ROUTINE CHANGES AN ASCII STRING TO AN OCTAL NUMBER.

5.2.6.5 GETDRV

THIS ROUTINE IS USED TO ASK THE OPERATOR WHICH DRIVE(S)
ARE TO BE TESTED

5.2.6.6 ASKADR

THIS ROUTINE IS USED TO INPUT THE ADDRESSES FOR THE "TACS",
"TADB" AND THE VECTOR AND THE PRIORITY LEVEL TO USE.

5.2.6.7 TYPERR

THIS ROUTINE IS USED TO TYPE OUT THE "ERROR" DATA

5.2.6.8 EXAM

THIS ROUTINE IS USED TO DETERMINE WHICH DRIVE(S) ARE AVAILABLE
FOR TESTING.

5.2.7 THE FOLLOW ROUTINES ARE USED TO MAKE ADJUSTMENTS TO
THE TUBO. BEFORE USING ANY OF THEM LOAD AND START 214.

5.2.7.1 WFGSUB

WRITE FILE GAPS FROM "BOT" TO "EOT"
START AT 220
THIS ROUTINE CAN BE USED TO ADJUST THE "WRITE GAP MONO" AND
THE "WRITE DELAY MONO".

5.2.7.2 WRTSUB

WRITE CONTINUOUS BLOCKS OF DATA
START AT 224
THE PROGRAM WILL HALT THREE(3) TIMES
AFTER EACH HALT SET THE SWR AND PRESS CONTINUE
HALT 1 --- SWR<7:0> = NUMBER OF BYTES PER BLOCK
HALT 2 ---SWR<7:0> = PATTERN DESIRED
HALT 3 --- SWR<15:0> = OPERATIONAL SWITCH SETTINGS
THIS ROUTINE CAN BE USED TO ADJUST THE "GAP TIME MONO"
** IF USING SOFTWARE SWITCH REGISTER, AFTER
EACH HALT OPERATOR WILL BE PROMPTED
FOR THE VALUE WITH "SWR=XXXXXX NEW="

5.2.7.3 RDSUB

READ CONTINUOUS BLOCKS OF DATA
START AT 230
THIS ROUTINE CAN BE USED TO ADJUST THE "SIGNAL MONO"
AND THE "THRESHOLD POT"

5.2.7.4 WGPBLK

WRITE A FILE GAP AND A BLOCK OF DATA FROM BOT TO ECT
START AT 234
THE PROGRAM WILL HALT THREE (3) TIMES
AFTER EACH HALT SET THE SWR AND PRESS CONTINUE
HALT 1 --- SWR<7:0> = NUMBER OF BYTES PER BLOCK
HALT 2 --- SWR<7:0> = PATTERN DESIRED
HALT 3 --- SWR<15:0> = OPERATIONAL SWITCH SETTINGS
THIS ROUTINE CAN BE USED TO ADJUST THE "WRITE GAP MONO"
AND THE "GAP TIME MONO".

** IF USING SOFTWARE SWITCH REGISTER, AFTER
EACH HALT OPERATOR WILL BE PROMPTED
FOR THE VALUE WITH "SWR=XXXXXX NEW="

5.2.7.5 RGBLK

READ A BLOCK OF DATA AND A FILE GAP
START AT 240
THIS ROUTINE IS USED AFTER "WRITE A BLOCK AND A FILE GAP" ROUTINE
IT CAN BE USED TO ADJUST THE "SIGNAL MON". THE THRESHOLD POT"
AND THE "TAPE BLANK MONO".

5.2.7.6 SFFGSB

SPACE FORWARD FILE GAP FROM "BOT" TO "EOT"
START AT 244
THIS ROUTINE CAN BE USED AFTER "WRITE FILE GAP" FOR LOW SPEED
SPACE FOWARD (TAPE BLANK MONO CAN BE ADJUSTED). OR AFTER READ OR
WRITE A FILE GAP AND A BLOCK OF DATA FOR HIGH SPEED SPACE FORWARD
(SIGNAL MONO CAN BE CHECKED).

5.2.7.7 BSFGSB

BACK SPACE FILE GAP
START AT 250
THIS ROUTINE CAN BE USED TO ADJUST OR CHECK THE "SIGNAL MONO".

6. ERRORS

THERE ARE A NUMBER OF ERRORS THAT CAN OCCUR IN THIS PROGRAM. WHEN AN ERROR IS ENCOUNTERED THE CALL TO THE ERROR (ERROR) ROUTINE IS MADE AND IF SW<13> IS NOT SET AN ERROR MESSAGE PERTAINING TO THE ERROR WILL BE TYPED. EACH ERROR TYPE OUT WILL CONTAIN THE FOLLOWING:

1. THE FUNCTION BEING PERFORM
2. AN ERROR MESSAGE
3. A DATA HEADER
4. A DATA STRING

REFER TO THE LISTING UNDER \$ERRTB FOR THE DIFFERENT ERRORS THAT CAN OCCUR.

7. RESTRICTIONS

BEFORE STARTING THE PROGRAM THE OPERATOR MUST INSURE THAT A CASSETTE IS LOADED IN THE DRIVE(S) TO BE TESTED AND IS WRITE ENABLED.

8. MISCELLANEOUS

8.1 EXECUTION TIME

THE FIRST PASS TAKES APPROXIMATELY 4 MINUTES ALL SUBSEQUENT PASSES TAKE APPROXIMATELY 8 MINUTES

8.2 STACK POINTER

STACK IS INITIALLY SET TO 1100.

8.3 PASS COUNT

A PROGRAM PASS THRU COUNT IS KEPT IN "\$PASS".

8.4 ITERATIONS

THE FIRST PASS OF THE PROGRAM WILL AUTOMATICALLY INHIBIT ITERATIONS. ALL SUBSEQUENT PASSES WILL PERFORM FULL, (ONE PER DRIVE), ITERATIONS.

8.5 SPECIAL REGISTERS

R4 AND R5 ARE RESERVED FOR "TACS" AND "TADB" THROUGH OUT THE PROGRAM.

9. PROGRAM DESCRIPTION

THIS PROGRAM IS A SEQUENCE OF INDEPENDENT TESTS THAT CHECK THE TAIL FOR PROPER OPERATION.

EACH TESTS IS A SERIES OF "TRAP" CALLS TO ROUTINES THAT PERFORM THE DESIRED FUNCTION.

THE PROGRAM STARTS WITH A SIMPLE SEQUENCE OF FUNCTIONS AND BUILTS IN COMPLEXITY INTRODUCING ONE NEW FUNCTION AT A TIME, UNTIL ALL LEGAL COMBINATIONS HAVE BEEN PERFORMED. THEN, MULTI SPACING IS PERFORMED TO TRY AND GENERATE NOISE THAT MIGHT CAUSE PROBLEMS DUE TO SPEED CHANGES AND FREQUENT START STOPPING.

C02

TALL MOTION TEST
DZTADC.NEWMAINDEC-11-DZTAD-C
24-MAY-76 00:00MACY11 27(1006)
TABLE OF CONTENTS

23-SEP-76 10:33

SEQ 0015

```

GENERAL INFORMATION
OPERATIONAL SWITCH SETTINGS
BASIC DEFINITIONS
STARTING ADDRESSES
TRAP CATCHER
STARTING ADDRESS(ES)
COMMON TAGS
ERROR POINTER TABLE
START OF TEST
INITIALIZE THE COMMON TAGS
TYPE PROGRAM NAME
GET VALUE FOR SOFTWARE SWITCH REGISTER
T1 REWIND WRITE REWIND READ
T2 TEST BSBG AFTER WRITE AND READ AFTER BSBG
T3 TEST BSFG AFTER WRITE AND READ AFTER BSFG
T4 TEST BSBG AFTER READ
T5 TEST BSFG AFTER READ
T6 TEST SFBG AFTER REWIND AND BSFG AFTER SFBG
T7 TEST WRITE AFTER READ
T10 TEST WRITE AFTER WRITE AND READ AFTER READ
T11 TEST BSBG AFTER BSBG
T12 TEST READ AFTER SFBG
T13 TEST SFBG AFTER BSBG
T14 TEST SFBG AFTER BSFG
T15 REWRITE THE LAST BLOCK 3 TIMES AFTER DOING A WRITE AFTER WRITE
T16 REWRITE THE LAST BLOCK 3 TIMES AFTER DOING A WRITE AFTER READ
T17 TEST WFG AFTER WRITE AND BSFG AFTER WFG
T20 TEST BSBG AFTER WFG
T21 TEST WRITE AFTER WFG
T22 TEST WFG AFTER READ AND BSFG AFTER WRITE
T23 TEST SFFG AFTER REWIND AND READ AFTER SFFG
T24 TEST BSBG AFTER SFFG
T25 TEST SFBG AFTER SFFG
T26 TEST SFBG AFTER SFBG
T27 TEST BSFG AFTER SFBG AND BSFG AFTER BSFG
T30 TEST BSBG THRU A FILE GAP
T31 TEST BSFG AFTER BSBG
T32 TEST BSBG AFTER BSFG
T33 TEST BSFG AFTER SFFG AND SFFG AFTER READ
T34 TEST SFFG AFTER SFFG
T35 TEST SFFG AFTER SFBG
T36 TEST WFG AFTER SFBG
T37 TEST WRITE AFTER SFBG
*****THE FOLLOWING TESTS TRY TO GENERATE NOISE***
T40 3 ONE BLOCK FILES---MULTI SPACING & TEST SFFG AFTER BSFG
T41 TWO BLOCK FILES---MULTI SPACING FUNCTIONS
T42 MULTI SPACING & TEST SFBG AFTER READ
T43 TWO BLOCK FILES--MULTI SPACING & TEST SFFG AFTER BSBG
T44 6 ONE BLOCK FILES---MULTI SPACING
END OF PASS ROUTINE
ROUTINE TO EXAMINE DRIVE(S) FOR AVAILABILITY
SCOPE HANDLER ROUTINE
ERROR HANDLER ROUTINE
ERROR TIMEOUT ROUTINE
ROUTINE TO ASK THE OPERATOR WHAT DRIVE(S) TO TEST
ROUTINE TO INPUT CSR,DBR, AND VECTOR ADDRESS AND PRIORITY

```


002

TA11 MOTION TEST MAINDEC-11-DZTAD-C MACY11 27(1006) 23-SEP-76 10:33
DZTADC.NEW 24-MAY-76 00:00 TABLE OF CONTENTS

SEQ 0016

1851	ROUTINE TO CHANGE AN ASCII STRING TO OCTAL
1888	ROUTINE TO GET "TEST" AND "SCOPE LOOP" ADDRESSES FROM THE TTY
1925	ROUTINE TO SELECT THE DRIVE TO BE TESTED
1962	ROUTINE TO COMPARE THE READ AND WRITE BUFFERS
1999	ROUTINE TO "WRITE-FILE-GAP"
2036	ROUTINE TO "WRITE" A RECORD ON THE TAPE
2073	ROUTINE TO "READ" A RECORD
2110	ROUTINE TO "BACK-SPACE-FILE-GAP"
2147	ROUTINE TO "BACK-SPACE-BLOCK-GAP"
2184	ROUTINE TO "SPACE-FORWARD-FILE-GAP"
2221	ROUTINE TO "SPACE-FORWARD-BLOCK-GAP"
2258	ROUTINE TO REWIND THE TAPE
2295	ROUTINE TO WAIT FOR A FLAG
2332	ROUTINE TO EXAMINE THE FLAGS
2369	ROUTINE TO CALCULATE THE CRC
2406	ROUTINE TO PERFORM THE REQUESTED FUNCTION
2443	ROUTINE TO SERVICE INTERRUPTS
2480	ROUTINE TO SERVICE ILLGAL INTERRUPTS FROM THE TA11
2517	***** MANUAL ADJUSTMENT ROUTINES *****
2554	WRITE FILE GAPS FROM "BOT" TO "EOT"
2591	WRITE CONTINUOUS BLOCKS OF DATA
2628	READ AND COMPARE CONTINUOUS BLOCKS OF DATA
2665	WRITE A FILE GAP AND A BLOCK OF DATA FROM BOT TO EOT
2702	READ AND COMPARE A BLOCK OF DATA AND A FILE GAP
2739	SPACE FORWARD FILE GAP FROM "BOT" TO "EOT"
2776	BACK SPACE FILE GAP
2813	SETUP READ AND WRITE LINKS FOR SUBROUTINES
2850	TYPE ROUTINE
2887	TTY INPUT ROUTINE
2924	READ AN OCTAL NUMBER FROM THE TTY
2961	BINARY TO OCTAL (ASCII) AND TYPE
2998	TRAP DECODER
3035	TRAP TABLE
3072	POWER DOWN AND UP ROUTINES
3109	DATA TABLE POINTERS FOR ERRORS
3146	ASCII MESSAGES
3183	READ AND WRITE BUFFERS AND LINKS


```

.TITLE TA11 MOTION TEST
*COPYRIGHT (C) 1973, 1976
*DIGITAL EQUIPMENT CORP.
*MAYNARD, MASS. 01754
*
*PROGRAM BY JIM LACEY
*
*THIS PROGRAM WAS ASSEMBLED USING THE PDP-11 MAINDEC SYSMAC
*PACKAGE (MAINDEC-11-DZQAC-C2), SEPT 14, 1976.
*
*****
*****
*****
.REM!

```

MAINDEC-11-DZTAD-C

GENERAL INFORMATION ABOUT THE TA11/TU60 CASSETTE

ADDRESS MNEMONIC DESCRIPTION

```

777500 TACS CONTROL AND STATUS REGISTER
777502 TADB DATA BUFFER REGISTER
260 TAVEC INTERRUPT VECTOR

```

TACS REGISTER DESCRIPTION

BIT	NAME	INIT STATE	READ AND/OR WRITE?
15	ERROR	?	READ ONLY
14	BLOCK CHECK ERROR	0	READ ONLY
13	CLEAR LEADER	?	READ ONLY
12	WRITE LOCK	?	READ ONLY
11	FILE GAP	0	READ ONLY
10	TIMING ERROR	0	READ ONLY
09	OFF LINE	?	READ ONLY
08	UNIT SELECT	0	READ/WRITE
07	TRANSFER REQUEST	0	READ ONLY
06	INTERRUPT ENABLE	0	READ/WRITE
05	READY	1	READ ONLY
04	ILBS	0	READ/WRITE
03	FUNCTION BIT 02	0	READ/WRITE
02	FUNCTION BIT 01	0	READ/WRITE
01	FUNCTION BIT 00	0	READ/WRITE

0=WRITE-FILE-GAP
1=WRITE
2=READ
3=BACK SPACE FILE GAP
4=BACK SPACE BLOCK GAP
5=SPACE FORWARD FILE GAP
6=SPACE FORWARD BLOCK GAP
7=REWIND

F02

TALL MOTION TEST
DZTADC.NEW 24-MAY-76 00:00

MACY11 27(1005) 23-SEP-76 10:33 PAGE 2
GENERAL INFORMATION

SEQ 0018

57

00

GO BIT

0

WRITE ONLY!

.SBTTL OPERATIONAL SWITCH SETTINGS

SWITCH	USE
15	HALT ON ERROR
14	LOOP ON TEST
13	INHIBIT ERROR TYPEOUTS
11	INHIBIT ITERATIONS
10	BELL ON ERROR
9	LOOP ON ERROR
7	LOCK ON CURRENT DRIVE
6	DELAY AT END OF EACH FUNCTION
5	RUN WITHOUT INTERRUPTS
4	IGNORE BLOCK CHECK ERRORS
3	INHIBIT DATA COMPARE

.SBTTL BASIC DEFINITIONS

;*INITIAL ADDRESS OF THE STACK POINTER *** 1100 ***

001100

STACK= 1100

.EQUIV EMT,ERROR

;;BASIC DEFINITION OF ERROR CALL

.EQUIV IOT,SCOPE

;;BASIC DEFINITION OF SCOPE CALL

;*MISCELLANEOUS DEFINITIONS

000011

HT= 11

;;CODE FOR HORIZONTAL TAB

000012

LF= 12

;;CODE FOR LINE FEED

000015

CR= 15

;;CODE FOR CARRIAGE RETURN

000200

CRLF= 200

;;CODE FOR CARRIAGE RETURN-LINE FEED

177776

PS= 177776

;;PROCESSOR STATUS WORD

177774

.EQUIV PS,PSW

177772

STKLMT= 177774

;;STACK LIMIT REGISTER

177570

PIRQ= 177772

;;PROGRAM INTERRUPT REQUEST REGISTER

177570

DSWR= 177570

;;HARDWARE SWITCH REGISTER

177570

DDISP= 177570

;;HARDWARE DISPLAY REGISTER

;*GENERAL PURPOSE REGISTER DEFINITIONS

000000

R0= %0

;;GENERAL REGISTER

000001

R1= %1

;;GENERAL REGISTER

000002

R2= %2

;;GENERAL REGISTER

000003

R3= %3

;;GENERAL REGISTER

000004

R4= %4

;;GENERAL REGISTER

000005

R5= %5

;;GENERAL REGISTER

000006

R6= %6

;;GENERAL REGISTER

000007

R7= %7

;;GENERAL REGISTER

000006

SP= %6

;;STACK POINTER

000007

PC= %7

;;PROGRAM COUNTER

;*PRIORITY LEVEL DEFINITIONS

000000

PR0= 0

;;PRIORITY LEVEL 0

000040

PR1= 40

;;PRIORITY LEVEL 1

000100

PR2= 100

;;PRIORITY LEVEL 2

000140

PR3= 140

;;PRIORITY LEVEL 3

000200

PR4= 200

;;PRIORITY LEVEL 4

000240

PR5= 240

;;PRIORITY LEVEL 5

H02

TA11 MOTION TEST MAINDEC-11-DZTAD-C MACY11 27(1006) 23-SEP-76 10:33 PAGE 4
 DZTADC.NEW 24-MAY-76 00:00 BASIC DEFINITIONS

SEQ 0020

114 000300 PR6= 300 ::PRIORITY LEVEL 6
 115 000340 PR7= 340 ::PRIORITY LEVEL 7

:*"SWITCH REGISTER" SWITCH DEFINITIONS

116 100000 SW15= 100000
 117 040000 SW14= 40000
 118 020000 SW13= 20000
 119 010000 SW12= 10000
 120 004000 SW11= 4000
 121 002000 SW10= 2000
 122 001000 SW09= 1000
 123 000400 SW08= 400
 124 000200 SW07= 200
 125 000100 SW06= 100
 126 000040 SW05= 40
 127 000020 SW04= 20
 128 000010 SW03= 10
 129 000004 SW02= 4
 130 000002 SW01= 2
 131 000001 SW00= 1
 132 .EQUIV SW09,SW9
 133 .EQUIV SW08,SW8
 134 .EQUIV SW07,SW7
 135 .EQUIV SW06,SW6
 136 .EQUIV SW05,SW5
 137 .EQUIV SW04,SW4
 138 .EQUIV SW03,SW3
 139 .EQUIV SW02,SW2
 140 .EQUIV SW01,SW1
 141 .EQUIV SW00,SW0

:*DATA BIT DEFINITIONS (BIT00 TO BIT15)

142 100000 BIT15= 100000
 143 040000 BIT14= 40000
 144 020000 BIT13= 20000
 145 010000 BIT12= 10000
 146 004000 BIT11= 4000
 147 002000 BIT10= 2000
 148 001000 BIT09= 1000
 149 000400 BIT08= 400
 150 000200 BIT07= 200
 151 000100 BIT06= 100
 152 000040 BIT05= 40
 153 000020 BIT04= 20
 154 000010 BIT03= 10
 155 000004 BIT02= 4
 156 000002 BIT01= 2
 157 000001 BIT00= 1
 158 .EQUIV BIT09,BIT9
 159 .EQUIV BIT08,BIT8
 160 .EQUIV BIT07,BIT7
 161 .EQUIV BIT06,BIT6
 162 .EQUIV BIT05,BIT5
 163 .EQUIV BIT04,BIT4
 164 .EQUIV BIT03,BIT3
 165 .EQUIV BIT02,BIT2


```

170      .EQUIV BIT01,BIT1
171      .EQUIV BIT00,BIT0
172
173      ;*BASIC "CPU" TRAP VECTOR ADDRESSES
174      ERRVEC= 4      ;; TIME OUT AND OTHER ERRORS
175      RESVEC= 10     ;; RESERVED AND ILLEGAL INSTRUCTIONS
176      TBITVEC=14     ;; "T" BIT
177      TRTVEC= 14     ;; TRACE TRAP
178      BPTVEC= 14     ;; BREAKPOINT TRAP (BPT)
179      IOTVEC= 20     ;; INPUT/OUTPUT TRAP (IOT) **SCOPE**
180      PWRVEC= 24     ;; POWER FAIL
181      EMTVEC= 30     ;; EMULATOR TRAP (EMT) **ERROR**
182      TRAPVEC=34     ;; "TRAP" TRAP
183      TKVEC= 60      ;; TTY KEYBOARD VECTOR
184      TPVEC= 64      ;; TTY PRINTER VECTOR
185      PIRQVEC=240    ;; PROGRAM INTERRUPT REQUEST VECTOR
186      ;;*****
187
188      ;*****TAIL FUNCTIONS*****
189      XWFG= 0        ;; WRITE FILE GAP FUNCTION
190      XWRITE= 2      ;; WRITE FUNCTION
191      XREAD= 4       ;; READ FUNCTION
192      XBSFG= 6       ;; BACK SPACE FILE GAP FUNCTION
193      XBSBG= 10      ;; BACK SPACE BLOCK GAP FUNCTION
194      XSFFG= 12      ;; SPACE FWD FILE GAP FUNCTION
195      XSFBG= 14      ;; SPACE FWD BLOCK GAP FUNCTION
196      XRWND= 16      ;; REWIND FUNCTION
197
198      ;*****TAIL BIT ASSIGNMENT*****
199      ERROR= BIT15
200      CRCERR= BIT14
201      LEADER= BIT13
202      WRTLOCK=BIT12
203      FGAP= BIT11
204      TIMERR= BIT10
205      OFFLINE=BIT09
206      UNIT= BIT08
207      TR.REQ= BIT07
208      INT.EN= BIT06
209      READY= BIT05
210      ILBS= BIT04
211      FUNC2= BIT03
212      FUNC1= BIT02
213      FUNC0= BIT01
214      GO= BIT00
215      FUNCTION=      FUNC2+FUNC1+FUNC0
216
217      ;////////////////////
218      ;////////////////////
219      ;SPECIAL REGISTERS
220
221      TACS= %4      ;R4 IS USED AS A POINTER TO THE TACS REGISTER
222      TADB= %5      ;R5 IS USED AS A POINTER TO THE TADB REGISTER.
223      ;////////////////////

```



```

224      .SBTTL TRAP CATCHER
225
226      000000
227      .=0
228      ;*ALL UNUSED LOCATIONS FROM 4 - 776 CONTAIN A ".+2,HALT"
229      ;*SEQUENCE TO CATCH ILLEGAL TRAPS AND INTERRUPTS
230      ;*LOCATION 0 CONTAINS 0 TO CATCH IMPROPERLY LOADED VECTORS
231      000174 000174
232      000176 000000
233      .=174
234      DISPREG: .WORD 0      ;; SOFTWARE DISPLAY REGISTER
235      SWREG:   .WORD 0      ;; SOFTWARE SWITCH REGISTER
236      .SBTTL STARTING ADDRESS(ES)
237      JMP      @#BEGIN1      ;; JUMP TO STARTING ADDRESS OF PROGRAM
238      JMP      @#BEGIN2      ;SELECT DRIVE(S) BEFORE STARTING TEST
239      JMP      @#BEGIN3      ;SELECT DRIVE(S) AND ADDRESSES BEFORE TESTING
240      JMP      @#BEGINX      ;SETUP FOR MANUAL LOOPING
241      JMP      @#WFGSUB      ;WRITE FILE GAP FROM BOT TO EOT
242      JMP      @#WRTSUB      ;WRITE CONTINUOUS BLOCKS OF DATA
243      JMP      @#RDSUB       ;READ CONTINUOUS BLOCKS OF DATA
244      JMP      @#WGPBLK      ;WRITE FILE GAP AND A BLOCK OF DATA
245      JMP      @#RGPBLK      ;READ BLOCK OF DATA AND INTO A FILE GAP
246      JMP      @#SFFGSB      ;SPACE FWD FILE GAP FROM BOT TO EOT
247      JMP      @#BSFGSB      ;BACK SPACE FILE GAPS
248      ;*****

```

.SBTTL COMMON TAGS

*THIS TABLE CONTAINS VARIOUS COMMON STORAGE LOCATIONS
*USED IN THE PROGRAM.

247									
248									
249									
250									
251									
252									
253		001100							
254	001100								
255	001100	000000							
256	001102	000							
257	001103	000							
258	001104	000000							
259	001106	000000							
260	001110	000000							
261	001112	000000							
262	001114	000							
263	001115	001							
264	001116	000000							
265	001120	000000							
266	001122	000000							
267	001124	000000							
268	001126	000000							
269	001130	000000							
270	001132	000000							
271	001134	000							
272	001135	000							
273	001136	000000							
274	001140	177570							
275	001142	177570							
276	001144	177560							
277	001146	177562							
278	001150	177564							
279	001152	177566							
280	001154	000							
281	001155	002							
282	001156	012							
283	001157	000							
284	001160	000000							
285									
286	001162	000000							
287	001164	000000							
288	001166	000000							
289	001170	000000							
290	001172	000000							
291	001174	177607	000377						
292	001200	077							
293	001201	015							
294	001202	000012							
295									
296	001204	000000							
297	001206	000000							
298	001210	000000							
299	001212	000000							
300	001214	000000							
301									
302	001216	000001							

.=1100

\$CMTAG: .WORD 0
\$PASS: .WORD 0
\$TSTNM: .BYTE 0
\$ERFLG: .BYTE 0
\$ICNT: .WORD 0
\$LPADR: .WORD 0
\$LPERR: .WORD 0
\$ERTTL: .WORD 0
\$ITEMB: .BYTE 0
\$ERMAX: .BYTE 1
\$ERRPC: .WORD 0
\$GDADR: .WORD 0
\$BDADR: .WORD 0
\$GDDAT: .WORD 0
\$BDDAT: .WORD 0
\$SAUTOB: .BYTE 0
\$INTAG: .BYTE 0
\$SWR: .WORD DSWR
\$DISPLAY: .WORD DDISP
\$TKS: 177560
\$TKB: 177562
\$TPS: 177564
\$TPB: 177566
\$NULL: .BYTE 0
\$FILLS: .BYTE 2
\$FILLC: .BYTE 12
\$TPFLG: .BYTE 0
\$REGAD: .WORD 0

\$REGO: .WORD 0
\$REG1: .WORD 0
\$TMP0: .WORD 0
\$TIMES: 0
\$ESCAPE: 0
\$BELL: .ASCIZ <207><377><377>
\$QUES: .ASCII /?/
\$CRLF: .ASCII <15>
\$LF: .ASCIZ <12>

\$AVPC: .WORD 0
\$BYTNUM: .WORD 0
\$RCRC0: .WORD 0
\$RCRC1: .WORD 0
\$RCRC2: .WORD 0

\$MXCNT: 1

;; START OF COMMON TAGS
;; CONTAINS PASS COUNT
;; CONTAINS THE TEST NUMBER
;; CONTAINS ERROR FLAG
;; CONTAINS SUBTEST ITERATION COUNT
;; CONTAINS SCOPE LOOP ADDRESS
;; CONTAINS SCOPE RETURN FOR ERRORS
;; CONTAINS TOTAL ERRORS DETECTED
;; CONTAINS ITEM CONTROL BYTE
;; CONTAINS MAX. ERRORS PER TEST
;; CONTAINS PC OF LAST ERROR INSTRUCTION
;; CONTAINS ADDRESS OF 'GOOD' DATA
;; CONTAINS ADDRESS OF 'BAD' DATA
;; CONTAINS 'GOOD' DATA
;; CONTAINS 'BAD' DATA
;; RESERVED--NOT TO BE USED
;; AUTOMATIC MODE INDICATOR
;; INTERRUPT MODE INDICATOR
;; ADDRESS OF SWITCH REGISTER
;; ADDRESS OF DISPLAY REGISTER
;; TTY KBD STATUS
;; TTY KBD BUFFER
;; TTY PRINTER STATUS REG. ADDRESS
;; TTY PRINTER BUFFER REG. ADDRESS
;; CONTAINS NULL CHARACTER FOR FILLS
;; CONTAINS # OF FILLER CHARACTERS REQUIRED
;; INSERT FILL CHARS. AFTER A "LINE FEED"
;; "TERMINAL AVAILABLE" FLAG (BIT<07>=0=YES)
;; CONTAINS THE ADDRESS FROM
;; WHICH (\$REGO) WAS OBTAINED
;; CONTAINS ((\$REGAD)+0)
;; CONTAINS ((\$REGAD)+2)
;; USER DEFINED
;; MAX. NUMBER OF ITERATIONS
;; ESCAPE ON ERROR ADDRESS
;; CODE FOR BELL
;; QUESTION MARK
;; CARRIAGE RETURN
;; LINE FEED

;; STORAGE FOR THE PC
;; THE NUMBER OF BYTES "READ" OR "WRITTEN"
;; CRC CALCULATED FOR "WRITE"
;; CRC CALCULATED FOR "READ"
;; CRC FROM THE TA11 FOR "READ"
;; MAX. NUMBER OF ITERATIONS

L02

TA11 MOTION TEST MAINDEC-11-DZTAD-C MACY11 27(1006) 23-SEP-76 10:33 PAGE 8
DZTADC.NEW 24-MAY-76 00:00 COMMON TAGS

SEQ 0024

303	001220	177500		TACSL:	177500		;LOW BYTE ADDRESS OF TACS
304	001222	177501		TACSH:	177501		;HIGH BYTE ADDRESS OF TACS
305	001224	177502		TADBL:	177502		;LOW BYTE ADDRESS OF TADB
306	001226	177503		TADBH:	177503		;HIGH BYTE ADDRESS OF TADB
307	001230	000260	000262	TAVEC:	260,262		;TA11 VECTOR ADDRESS
308	001234	000300		TAPRIO:	300		;TA11 BR LEVEL 6
309	001236	000000		DRIVE:	0		;NEXT DRIVE TO TEST
310	001240	000000	000000	DRVKEY:	0,0		;DRIVE SELECT KEY:
311	001244	001240		DRVPT:	DRVKEY		
312	001246	000000		ASKKEY:	0		
313	001250	177777	000000	CURDRV:	-1,0		;CURRENT DRIVE BEING TESTED
314	001254	000000		WAITKEY:	0		;WAIT ON INTERRUPT KEY
315	001256	001260		RWDFLG:	RWDA		;REWIND FLAG POINTER
316	001260	000000		RWDA:	0		;DRIVE "A" REWIND FLAG
317	001262	000000		RWDB:	0		;DRIVE "B" REWIND FLAG
318	001264	000000	000000	STALL:	0,0		;STALL TIME

.SBTTL ERROR POINTER TABLE

;*THIS TABLE CONTAINS THE INFORMATION FOR EACH ERROR THAT CAN OCCUR.
;*THE INFORMATION IS OBTAINED BY USING THE INDEX NUMBER FOUND IN
;*LOCATION \$ITEMB. THIS NUMBER INDICATES WHICH ITEM IN THE TABLE IS PERTINENT.
;*NOTE1: IF \$ITEMB IS 0 THE ONLY PERTINENT DATA IS (\$ERRPC).
;*NOTE2: EACH ITEM IN THE TABLE CONTAINS 4 POINTERS EXPLAINED AS FOLLOWS:

;* EM ::POINTS TO THE ERROR MESSAGE
;* DH ::POINTS TO THE DATA HEADER
;* DT ::POINTS TO THE DATA
;* DF ::POINTS TO THE DATA FORMAT

\$ERRTB:

;NOTE: ALL NUMBERS WILL BE TYPED AS 6 DIGIT OCTAL NUMBERS

;ITEM

1

EM1

;IMPROPER FLAG SETTING

DH1

;TEST ERROR

DT1

;PC

PC

TACS

TADB

0

;SAVPC

\$ERRPC

\$REGO

\$REG1

;ITEM

2

EM2

;IMPROPER FLAG OCCURRED

DH1

;TEST ERROR

DT1

;PC

PC

TACS

TADB

0

;SAVPC

\$ERRPC

\$REGO

\$REG1

;ITEM

3

EM3

;MISSED A FLAG

DH1

;TEST ERROR

DT1

;PC

PC

TACS

TADB

0

;SAVPC

\$ERRPC

\$REGO

\$REG1

;ITEM

4

EM4

;INTERRUPT FAILED

DH1

;TEST ERROR

DT1

;PC

PC

TACS

TADB

0

;SAVPC

\$ERRPC

\$REGO

\$REG1

;ITEM

5

EM5

;PREMATURE READY OCCURRED

DH1

;TEST ERROR

DT1

;PC

PC

TACS

TADB

0

;SAVPC

\$ERRPC

\$REGO

\$REG1

;ITEM

6

EM6

;DIDN'T STOP IN A FILE GAP

DH1

;TEST ERROR

DT1

;PC

PC

TACS

TADB

319
320
321
322
323
324
325
326
327
328
329
330
331
332
333
334
335
336
337
338
339
340
341
342
343
344
345
346
347
348
349
350
351
352
353
354
355
356
357
358
359
360
361
362
363
364
365
366
367
368
369
370
371
372
373
374

001270

001270 015550

001272 016112

001274 015054

001276 000000

001300 015576

001302 016112

001304 015054

001306 000000

001310 015625

001312 016112

001314 015054

001316 000000

001320 015643

001322 016112

001324 015054

001326 000000

001330 015664

001332 016112

001334 015054

001336 000000

001340 015715

001342 016112

375	001344	015054	DT1	;	SAVPC	\$ERRPC	\$REGO	\$REG1			
376	001346	000000	0								
377											
378			;	ITEM	7						
379	001350	015747	EM7	;	DIDN'T STOP ON CLEAR LEADER						
380	001352	016112	DH1	;	TEST ERROR						
381				;	PC PC	TACS	TADB				
382	001354	015054	DT1	;	SAVPC	\$ERRPC	\$REGO	\$REG1			
383	001356	000000	0								
384											
385			;	ITEM	10						
386	001360	016003	EM10	;	BAD DATA READ						
387	001362	016172	DH2	;	TEST ERROR				EXPT'D	RCV'D	BYTE
388				;	PC PC	TACS			DATA	DATA	NUMBER
389	001364	015066	DT2	;	SAVPC	\$ERRPC	\$REGO		\$GDDAT	\$BDDAT	BYTNUM
390	001366	000000	0								
391											
392			;	ITEM	11						
393	001370	016021	EM11	;	ILLEGAL BUFFER						
394	001372	016611	DH5	;	TEST ERROR						
395				;	PC PC	TACS					
396	001374	015136	DT5	;	SAVPC	\$ERRPC	\$REGO				
397	001376	000000	0								
398											
399			;	ITEM	12						
400	001400	016045	EM12	;	CRC ERROR						
401	001402	016333	DH3	;	TEST ERROR				WRITE	READ	TU60
402				;	PC PC	TACS			CRC	CRC	CRC
403	001404	015104	DT3	;	SAVPC	\$ERRPC	\$REGO		RCRC0	RCRC1	RCRC2
404	001406	000000	0								
405											
406			;	ITEM	13						
407	001410	016057	EM13	;	SHORT RECORD						
408	001412	016471	DH4	;	TEST ERROR					BYTES	
409				;	PC PC	TACS			TADB	LEFT	
410	001414	015122	DT4	;	SAVPC	\$ERRPC	\$REGO		\$REG1	\$TMP0	
411	001416	000000	0								
412											
413			;	ITEM	14						
414	001420	016074	EM14	;	BAD INTERRUPT						
415	001422	016611	DH5	;	TEST ERROR						
416				;	PC PC	TACS					
417	001424	015136	DT5	;	SAVPC	\$ERRPC	\$REGO				
418	001426	000000	0								
419											
420	001430		ITEMS2:	;	ITEMS 201-202						
421											
422	001430	016674	EM201	;	TA11 FAILED TO RESPOND						
423	001432	016746	DH201	;	PC TACS						
424	001434	016662	DT201	;	\$ERRPC TACS						
425	001436	000000	0	;	BOTH NUMBERS ARE TYPED AS OCTAL NUMBERS						
426											
427	001440	016723	EM202	;	NO DRIVES AVAILABLE						
428	001442	016763	DH202	;	PC						
429	001444	016670	DT202	;	\$ERRPC						
430	001446	000000	0	;							

486
485
484
483
482
481
480
479
478
477
476
475
474
473
472
471
470
469
468
467
466
465
464
463
462
461
460
459
458
457
456
455
454
453
452
451
450
449
448
447
446
445
444
443
442
441
440
439
438
437
436
435
434
433
432
431
430
429
428
427
426
425
424
423
422
421
420
419
418
417
416
415
414
413
412
411
410
409
408
407
406
405
404
403
402
401
400
399
398
397
396
395
394
393
392
391
390
389
388
387
386
385
384
383
382
381
380
379
378
377
376
375
374
373
372
371
370
369
368
367
366
365
364
363
362
361
360
359
358
357
356
355
354
353
352
351
350
349
348
347
346
345
344
343
342
341
340
339
338
337
336
335
334
333
332
331
330
329
328
327
326
325
324
323
322
321
320
319
318
317
316
315
314
313
312
311
310
309
308
307
306
305
304
303
302
301
300
299
298
297
296
295
294
293
292
291
290
289
288
287
286
285
284
283
282
281
280
279
278
277
276
275
274
273
272
271
270
269
268
267
266
265
264
263
262
261
260
259
258
257
256
255
254
253
252
251
250
249
248
247
246
245
244
243
242
241
240
239
238
237
236
235
234
233
232
231
230
229
228
227
226
225
224
223
222
221
220
219
218
217
216
215
214
213
212
211
210
209
208
207
206
205
204
203
202
201
200
199
198
197
196
195
194
193
192
191
190
189
188
187
186
185
184
183
182
181
180
179
178
177
176
175
174
173
172
171
170
169
168
167
166
165
164
163
162
161
160
159
158
157
156
155
154
153
152
151
150
149
148
147
146
145
144
143
142
141
140
139
138
137
136
135
134
133
132
131
130
129
128
127
126
125
124
123
122
121
120
119
118
117
116
115
114
113
112
111
110
109
108
107
106
105
104
103
102
101
100
99
98
97
96
95
94
93
92
91
90
89
88
87
86
85
84
83
82
81
80
79
78
77
76
75
74
73
72
71
70
69
68
67
66
65
64
63
62
61
60
59
58
57
56
55
54
53
52
51
50
49
48
47
46
45
44
43
42
41
40
39
38
37
36
35
34
33
32
31
30
29
28
27
26
25
24
23
22
21
20
19
18
17
16
15
14
13
12
11
10
9
8
7
6
5
4
3
2
1
0

```

: ////////////////////////////////////////////////////////////////////
: ////////////////////////////////////////////////////////////////////
: *****
: BEGIN1 IS FOR NORMAL START
: BEGIN2 IS FOR DRIVE SELECTION
: BEGIN3 IS FOR DRIVE & ADDRESS SELECTION
: BEGIN4 IS FOR MANUAL OPERATION
: *****
: BEGIN1: CLR      R5                ;NORMAL START
:           MOV     #AB,3#DRVKEY
:           CMPB    #5,3#41          ;CASSETTE DDP?
:           BNE     BGNCMN           ;GO BEGIN COMMON CODE IF NO
:           CMP     #260,3#TAVEC     ;STANDARD VECTOR?
:           BNE     BGNCMN           ;GO BEGIN COMMON CODE IF NO
:           BR      BEGIN3           ;GET DRIVES AND ADDRESSES
: BEGIN2: MOV     #1,R5              ;ASK FOR DRIVES FLAG
:           BR      BGNCMN           ;BEGIN COMMON CODE
: BEGIN3: MOV     #2,R5              ;ASK FOR DRIVES AND ADDRESSES
:           BR      BGNCMN
: BEGIN4: MOV     #3,R5
: BGNCMN:
: .SBTTL  INITIALIZE THE COMMON TAGS
: ;;CLEAR THE COMMON TAGS (SCMTAG) AREA
:           MOV     #SCMTAG,R6       ;;FIRST LOCATION TO BE CLEARED
:           CLR     (R6)+             ;;CLEAR MEMORY LOCATION
:           CMP     #SWR,R6          ;;DONE?
:           BNE     -6               ;;LOOP BACK IF NO
:           MOV     #STACK,SP        ;;SETUP THE STACK POINTER
: ;;INITIALIZE A FEW VECTORS
:           MOV     #SSCOPE,3#IOTVEC ;;IOT VECTOR FOR SCOPE ROUTINE
:           MOV     #340,3#IOTVEC+2 ;;LEVEL 7
:           MOV     #SEERR,3#EMTVEC  ;;EMT VECTOR FOR ERROR ROUTINE
:           MOV     #340,3#EMTVEC+2 ;;LEVEL 7
:           MOV     #STRAP,3#TRAPVEC ;;TRAP VECTOR FOR TRAP CALLS
:           MOV     #340,3#TRAPVEC+2;LEVEL 7
:           MOV     #SPWRDN,3#PWRVEC ;;POWER FAILURE VECTOR
:           MOV     #340,3#PWRVEC+2 ;;LEVEL 7
:           MOV     SENDCT,SEOPCT    ;;SETUP END-OF-PROGRAM COUNTER
:           CLR     STIMES           ;;INITIALIZE NUMBER OF ITERATIONS
:           CLR     $ESCAPE          ;;CLEAR THE ESCAPE ON ERROR ADDRESS
:           MOVB    #1,$ERMAX        ;;ALLOW ONE ERROR PER TEST
:           MOV     #,$SLPADR        ;;INITIALIZE THE LOOP ADDRESS FOR SCOPE
:           MOV     #,$SLPERR        ;;SETUP THE ERROR LOOP ADDRESS
: ;;SIZE FOR A HARDWARE SWITCH REGISTER. IF NOT FOUND OR IT IS
: ;;EQUAL TO A "-1", SETUP FOR A SOFTWARE SWITCH REGISTER.
:           MOV     3#ERRVEC, -(SP)  ;;SAVE ERROR VECTOR
:           MOV     #64$,3#ERRVEC    ;;SET UP ERROR VECTOR
:           MOV     #DSWR,SWR        ;;SETUP FOR A HARDWARE SWICH REGISTER
:           MOV     #DDISP,DISPLAY   ;;AND A HARDWARE DISPLAY REGISTER
:           CMP     #-1,3#SWR        ;;TRY TO REFERENCE HARDWARE SWR
:           BNE     66$              ;;BRANCH IF NO TIMEOUT TRAP OCCURRED
:                                     ;;AND THE HARDWARE SWR IS NOT = -1
:           BR      65$              ;;BRANCH IF NO TIMEOUT

```


C03

TA11 MOTION TEST MAINDEC-11-DZTAD-C MACY11 27(1006) 23-SEP-76 10:33 PAGE 12
 DZTADC.NEW 24-MAY-76 00:00 INITIALIZE THE COMMON TAGS

SEQ 0028

```

487 001722 012716 001730 64$: MOV #65$, (SP) ;;SET UP FOR TRAP RETURN
488 001726 000002 RTI
489 001730 012767 000176 177202 65$: MOV #SWREG, SWR ;;POINT TO SOFTWARE SWR
490 001736 012767 000174 177176 MOV #DISPREG, DISPLAY
491 001744 012637 000004 66$: MOV (SP)+, 2*ERRVEC ;;RESTORE ERROR VECTOR
492
493
494
495
496
497 001750 005227 177777
498 001754 001036
499 001756 022737 006442 000042
500 001764 001432
501 001766 104401 002024
502
503
504
505
506
507 001772 005737 000042
508 001776 001006
509 002000 026727 177134 000176
510 002006 001005
511 002010 104405
512 002012 000403
513 002014 112767 000001 177112 67$: MOVB #1, SAUTOB ;;SET AUTO-MODE INDICATOR
514 002022 000413 68$:
515 002022 BR HERE
516 002052

```

.SBTTL TYPE PROGRAM NAME
 ;;TYPE THE NAME OF THE PROGRAM IF FIRST PASS
 INC #-1 ;;FIRST TIME?
 BNE HERE ;;BRANCH IF NO
 CMP #SENDAD, 2*42 ;;ACT-11?
 BEQ HERE ;;BRANCH IF YES
 TYPE MSGID ;;TYPE ASCIZ STRING
 .SBTTL GET VALUE FOR SOFTWARE SWITCH REGISTER
 TST 2*42 ;;ARE WE RUNNING UNDER XXDP/ACT?
 BNE 67\$;;BRANCH IF YES
 CMP SWR, #SWREG ;;SOFTWARE SWITCH REG SELECTED?
 BNE 68\$;;BRANCH IF NO
 GTSWR ;;GET SOFT-SWR SETTINGS
 BR 68\$
 BR 68\$
 BR HERE ;;GET OVER THE ASCIZ
 .MSGID: .ASCIZ <CRLF>/MAINDEC-11-DZTAD-C/<CRLF>
 HERE:

D03

TALL MOTION TEST
DZTADC.NEWMAINDEC-11-DZTAD-C
24-MAY-76 00:00MACY11 27(1006) 23-SEP-76 10:33 PAGE 13
GET VALUE FOR SOFTWARE SWITCH REGISTER

SEQ 0029

000000
 000001
 000002
 000003
 000004
 000005
 000006
 000007
 000008
 000009
 000010
 000011
 000012
 000013
 000014
 000015
 000016
 000017
 000018
 000019
 000020
 000021
 000022
 000023
 000024
 000025
 000026
 000027
 000028
 000029
 000030
 000031
 000032
 000033
 000034
 000035
 000036
 000037
 000038
 000039
 000040
 000041
 000042
 000043
 000044
 000045
 000046
 000047
 000048
 000049
 000050
 000051
 000052
 000053
 000054
 000055
 000056
 000057
 000058
 000059
 000060
 000061
 000062
 000063

THE CONTENTS OF R5 DETERMINES WHAT WILL BE DONE

R5=3 MANUAL OPERATIONS
 R5=2 ASK FOR DRIVE(S) AND ADDRESSES (TACS AND VECTOR)
 R5=1 ASK FOR DRIVE(S)
 R5=0 DON'T ASK FOR ANYTHING

002052 010504
 002054 005305
 002056 002406
 002060 004737 007432
 002064 005305
 002066 002402
 002070 004737 007542

BEGINX: MOV R5,R4 ;COPY R5
 DEC R5 ;ASK FOR DRIVES?
 BLT CHKADR ;BR IF NO
 JSR PC,3#ASKDRV ;GO GET DRIVES TO BE TESTED
 DEC R5 ;ASK FOR ADDRESSES?
 BLT CHKADR ;BR IF NO
 JSR PC,3#ASKADR ;GO GET TALL ADDRESSES

CHECK THAT "TACS" WILL RESPOND TO ADDRESSING

I. TIMEOUT OCCURRED
 A. TYPE ERROR MESSAGE
 B. EXAMINE R4
 1. R4>0 GOTO BEGINX
 2. R4=0 EXAMINE (42)
 A. (42)=0 GOTO BEGINX
 B. (42)>0 GOTO SENDAD

II. TIMEOUT DIDN'T OCCUR
 A. CONTINUE

002074 012737 002112 000004
 002102 005000
 002104 005777 177110
 002110 000402
 002112 005200
 002114 022626
 002116 012737 000006 000004
 002124 005700
 002126 001412
 002130 104201
 002132 012705 000002
 002136 005704
 002140 001344
 002142 013700 000042
 002146 001741
 002150 000137 006442
 002154

CHKADR: MOV #1\$,3#ERRVEC ;IN CASE OF TIMEOUTS
 CLR R0 ;USE AS A SWITCH
 TST 3TACSL ;SEE IF TALL RESPONDS
 BR 2\$;BR IF NO TIMEOUT
 1\$: INC R0 ;COME HERE ON TIMEOUT
 CMP (SP)+,(SP)+ ;CLEANUP THE STACK
 2\$: MOV #ERRVEC+2,3#ERRVEC ;RESTORE TIMEOUT VECTOR
 TST R0 ;DID A TIMEOUT OCCUR?
 BEQ 3\$;BR IF NO
 ERROR 201 ;TALL FAILED TO RESPOND
 MOV #2,R5 ;DRIVES & ADDRESSES
 TST R4 ;OPERATOR INPUTS?
 BNE BEGINX ;BR IF YES
 MOV 3#42,R0 ;GET MONITOR RETURN ADDRESS
 BEQ BEGINX ;BR IF NO MONITOR
 JMP 3#SENDAD ;GO TO END
 3\$:

E03

TA11 MOTION TEST
DZTADC.NEWMAINDEC-11-DZTAD-C
24-MAY-76 00:00MACY11 27(1006) 23-SEP-76 10:33 PAGE 14
GET VALUE FOR SOFTWARE SWITCH REGISTER

SEQ 0030

564
565
566
567
568
569
570
571
572
573
574
575
576
577
578
579
580
581
582
583
584
585
586
587
588
589
590
591
592
593
594
595
596
597
598
599
600
601
602
603
604
605
606
607
608
609
610
611
612
613
614
615
616
617
618
619:*****
:*****

:MAKE SURE THE DRIVES IN THE DRIVE TABLE CAN BE TESTED

:I. DESIRED DRIVES CAN NOT BE TESTED
:A. TYPE ERROR MESSAGE
:B. EXAMINE R4
:1. R4>0 GOTO BEGINX
:2. R4=0 EXAMINE (42)
:A. (42)=0 GOTO BEGINX
:B. (42)>0 GOTO \$ENDAD
:II. BOTH DRIVES IN THE TABLE BUT ONLY ONE OF THEM CAN BE TESTED
:A. CLEAR BAD DRIVE FROM THE DRIVE TABLE
:B. CONTINUE IN PROGRAM
:III. DESIRED DRIVE(S) CAN BE TESTED
:A. CONTINUE IN PROGRAM

:*****

CHKDRV: MOV #DRVKEY, R0 ; PICKUP ADDRESS OF ASCII DRIVE KEY
JSR PC, @#EXAM ; GO EXAMINE FIRST DRIVE
BR 1\$; OK TO TEST---GO CHECK NEXT
MOVB 1(R0), (R0) ; REPLACE 1ST WITH 2ND
BEQ 2\$; BR IF NO 2ND DRIVE SELECTED
JSR PC, @#EXAM ; GO EXAMINE DRIVE
BR 2\$; OK TO TEST
CLR (R0) ; CLEAR DRIVE CODES
BR 2\$
1\$: INC R0 ; POINT TO 2ND
JSR PC, @#EXAM ; GO EXAMINE DRIVE
BR 2\$; OK TO TEST
CLRB (R0) ; CLEAR 2ND
2\$: MOV #DRVKEY, R0 ; RESET ADDRESS POINTERS
MOV R0, @#DRVPNT
CMPB (R0), 1(R0) ; 1ST = 2ND?
BNE 3\$; BR IF NO
CLRB 1(R0) ; YES---CLEAR 2ND
3\$: TST (R0) ; ANY DRIVES?
BEQ 5\$; BR IF NO
BR MANUAL
5\$: ERROR 202 ; NO DRIVES AVAILABLE
MOV #2, R5 ; DRIVES & ADDRESS
TST R4 ; OPERATOR INPUTS?
BNE BEGINX ; BR IF YES
MOV @#42, R0 ; GET MONITOR RETURN ADDRESS
BEQ BEGINX ; NO MONITOR
JMP @#SENDAD ; GO TO END
MANUAL: CMP R4, #3
BNE OK
MOV -1, R4
OK: MOV R4, @#ASKKEY
BR START
PWRST: TYPE MSGID ; POWER FAIL RESTART
MOV #DRVKEY, @#DRVPNT

F03

TAIL MOTION TEST
DZTADC.NEWMAINDEC-11-DZTAD-C
24-MAY-76 00:00MACY11 27(1006) 23-SEP-76 10:33 PAGE 15
GET VALUE FOR SOFTWARE SWITCH REGISTER

SEQ 0031

```

620 002326 012777 012366 176674 START: MOV #BADINT, @TAVEC ;SETUP TAIL TRAP VECTOR
621 002334 013777 001234 176670 MOV @TAPRIO, @TAVEC+2
622 002342 012737 000340 177776 MOV #340, @PS ;LOCKOUT ALL I/O INT
623 002350 013704 001220 MOV @TACSL, TACS ;SETUP TACS
624 002354 013705 001224 MOV @TADBL, TADB ;SETUP TADB
625 002360 005037 001102 CLR @STSTNM ;ZERO THE TEST NUMBER
626 002364 005037 006624 CLR @LOOPKEY ;CLEAR THE LOOP UNDER TTY CONTROL KEY
627 002370 012737 000001 001216 MOV #1, @SMXCNT ;SET FOR ONE ITERATION PER TEST
628 002376 013701 001244 MOV @DRVPT, R1 ;GET DRIVE POINTER
629 002402 010167 176630 MOV R1, DRIVE ;SET DRIVE TO POINTER
630
631 ;////////////////////////////////////
632 ;TYPE THE DRIVE(S) THAT WILL BE TESTED
633 MOV (R1), @CURDRV ;GET CURRENT DRIVE FOR TYPE-OUT
634 TYPE ,MTSDRV ;TYPE "TESTING DRIVE"
635 TYPE ,CURDRV ;TYPE THE CURRENT DRIVE
636 BIT #SW07, @SWR ;LOCK ON ONE DRIVE?
637 BNE 2$ ;BR IF YES
638 TSTB @DRVKEY+1 ;TWO DRIVES AVAILABLE?
639 BEQ 2$ ;BR IF NO
640 INC @SMXCNT ;SET FOR TWO ITERATIONS PER TEST (ONE PER DRIVE)
641 INC R1 ;INCREMENT THE POINTER
642 MOV (R1), @CURDRV ;SET SECOND DRIVE FOR TYPE-OUT
643 BNE 1$ ;GO TYPE IT
644 MOV @DRVKEY, R1 ;RESET POINTER
645 MOV (R1), @CURDRV ;NOW GET THE SECOND DRIVE
646 TYPE ,MANDRV ;TYPE "AND DRIVE"
647 TYPE ,CURDRV ;TYPE DRIVE
648 MOV R1, @DRVPT ;SAVE POINTER FOR NEXT TRIP THRU
649 MOV #5$, @SLPADR ;SETUP THE SCOPE LOOP ADDRESS
650 TST @SPASS ;ONCE ONLY
651 BNE TST1
652 ;////////////////////////////////////
653 ;ON THE FIRST PASS OF THE PROGRAM DETERMINE THE AMOUNT OF TIME TO
654 ;WAIT ON A FLAG ("READY"/"TRANSFER REQUEST") AND THE STALL TIME TO USE
655 ;WHEN STALLING AFTER A FUNCTION IS COMPLETED
656 SELDRV ;SELECT DRIVE
657 MOV #FATAL, @SESCAPE ;SETUP ESCAPE ON ERROR
658 CLR @RWDFLG ;SET FOR NO REWIND ERRORS
659 MOV #1CBIT15, @MAXCNT ;SETUP IMPOSSIBLE COUNT
660 MOV @XRWD, @TACS ;INSURE NO ERROR DUE TO "CLEAR LEADER"
661 REWIND ;GO TO BOT
662 WFG ;NOW TIME A WFG
663 SUB @HICNT, @MAXCNT ;GET THE TIME IT TOOK TO WFG
664 INC @MAXCNT ;MAKE IT BIGGER
665 ROL @MAXCNT
666 ROL @MAXCNT
667 CLR R0
668 CLR R1
669 MOV #4$, @TAVEC ;SETUP TAIL VECTOR
670 MOV @XWFG!INT.EN!GO, @TACS ;START A "WFG"
671 CLR @PS ;ALLOW INTERRUPTS
672 SUB #1, R0 ;START TIMING HOW LONG
673 SBC R1 ;IT TAKES TO WRITE A FILE GAP
674 BNE 3$
675 MOV #340, @PS ;LOCK OUT INTERRUPTS
676 BR FATAL ;IT TOOK TO LONG

```


G03

TA11 MOTION TEST
DZTADC.NEW24-MAY-76 00:00
MAINDEC-11-DZTAD-CMACY11 27(1006) 23-SEP-76 10:33 PAGE 16
GET VALUE FOR SOFTWARE SWITCH REGISTER

SEQ 0032

676	002632	006201		4\$:	ASR	R1		;DIVIDE THE TIME BY 2
677	002634	006000			ROR	RO		
678	002636	010037	001264		MOV	RO, @#STALL		;AND SAVE IT AS THE STALL TIME
679	002642	010137	001266		MOV	R1, @#STALL+2		
680	002646	012777	012366	176354	MOV	#BADINT, @TAVEC		;SET TRAP CATCHER
681	002654	105737	001241		TSTB	@#DRVKEY+1		;TWO DRIVES?
682	002660	001407			BEQ	5\$		;BR IF NO
683	002662	104422			SELDRV			;SELECT THE OTHER DRIVE
684	002664	005077	176366		CLR	@RWDFLG		;SET REWIND FLAG
685	002670	112714	000016		MOVB	#XRWND, @TACS		;AVOID "CLEAR LEADER" ERROR
686	002674	104421			REWIND			;GO TO BOT
687	002676	104412			WFG			;GO TO OXIDE
688	002700	005737	001246	5\$:	TST	@#ASKKEY		;GO START TESTING IF NO MANUAL
689	002704	002034			BGE	TST1		; OPERATIONS REQUESTED
690	002706	005000			CLR	RO		
691	002710	006137	012022		ROL	@#MAXCNT		;INCREASE THE WAIT TIME
692	002714	000000			HALT			;GIVE CONTROL TO THE OPERATOR
693								
694	002716			FATAL:	TYPE	,65\$		::TYPE ASCIZ STRING
695	002716	104401	002724		BR	,64\$		::GET OVER THE ASCIZ
696	002722	000414		::65\$:	.ASCIZ	<15><12>*FATAL ERROR ON DRIVE *		
697				64\$:				
698	002754				TYPE	,CURDRV		;CURRENT DRIVE
699	002754	104401	001250		MOV	@#42, RO		;CHECK FOR A MONITOR
700	002760	013700	000042		BEQ	1\$		;BR IF NONE
701	002764	001402			JMP	@#SENDAD		;LEAVE
702	002766	000137	006442		HALT			;ERROR OCCURRED BEFORE TESTING STARTED
703	002772	000000		1\$:	BR	1\$		;HANGUP
704	002774	000776						

H03

TA11 MOTION TEST
DZTADC.NEW 24-MAY-76 00:00MAINDEC-11-DZTAD-C
T1MACY11 27(1006) 23-SEP-76 10:33 PAGE 17
REWIND,WRITE,REWIND,READ

SEQ 0033

```

705
706
707
708 002776 000004
709 003000 012767 003032 176164
710 003006 104422
711 003010 104421
712 003012 104413 017022
713 003016 104421
714 003020 104414 016766
715 003024 104423
716 003026 016766
717 003030 017022
718
719
720
721 003032 000004
722 003034 012767 003066 176130
723 003042 104422
724 003044 104421
725 003046 104413 017022
726 003052 104416
727 003054 104414 016766
728 003060 104423
729 003062 016766
730 003064 017022
731
732
733
734 003066 000004
735 003070 012767 003122 176074
736 003076 104422
737 003100 104421
738 003102 104413 017022
739 003106 104415
740 003110 104414 016766
741 003114 104423
742 003116 016766
743 003120 017022
744
745
746
747 003122 000004
748 003124 012767 003172 176040
749 003132 104422
750 003134 104421
751 003136 104413 017022
752 003142 104421
753 003144 104414 016766
754 003150 104423
755 003152 016766
756 003154 017022
757 003156 104416
758 003160 104414 016766
759 003164 104423
760 003166 016766

*****
*TEST 1 REWIND,WRITE,REWIND,READ
*****
TST1: SCOPE
MOV #TST2,$ESCAPE ;;ESCAPE TO TEST 2 ON ERROR
SELDV ;SELECT DRIVE FOR TESTING
REWIND ;GO TO "CLEAR LEADER"
WRITE ,WLNK1 ;WRITE ON TAPE
REWIND ;GO TO BEGINNING OF TAPE
READ ,RLNK1 ;READ
BLKCMP ;COMPARE THE READ AND WRITE BUFFERS
RLNK1 ;LINK TO THE READ BUFFER
WLNK1 ;LINK TO THE WRITE BUFFER
*****
*TEST 2 TEST BSBG AFTER WRITE AND READ AFTER BSBG
*****
TST2: SCOPE
MOV #TST3,$ESCAPE ;;ESCAPE TO TEST 3 ON ERROR
SELDV ;SELECT DRIVE FOR TESTING
REWIND ;GO TO "BOT"
WRITE ,WLNK1 ;WRITE ONE BLOCK
BSBG ;BACK OVER THE BLOCK
READ ,RLNK1 ;NOW READ
BLKCMP ;COMPARE THE READ AND WRITE BUFFERS
RLNK1 ;LINK TO THE READ BUFFER
WLNK1 ;LINK TO THE WRITE BUFFER
*****
*TEST 3 TEST BSFG AFTER WRITE AND READ AFTER BSFG
*****
TST3: SCOPE
MOV #TST4,$ESCAPE ;;ESCAPE TO TEST 4 ON ERROR
SELDV ;SELECT DRIVE FOR TESTING
REWIND ;GO TO "BOT"
WRITE ,WLNK1 ;WRITE ONE BLOCK
BSFG ;BACK OVER THE BLOCK
READ ,RLNK1 ;NOW READ
BLKCMP ;COMPARE THE READ AND WRITE BUFFERS
RLNK1 ;LINK TO THE READ BUFFER
WLNK1 ;LINK TO THE WRITE BUFFER
*****
*TEST 4 TEST BSBG AFTER READ
*****
TST4: SCOPE
MOV #TST5,$ESCAPE ;;ESCAPE TO TEST 5 ON ERROR
SELDV ;SELECT DRIVE FOR TESTING
REWIND ;GO TO "BOT"
WRITE ,WLNK1 ;WRITE ONE BLOCK
REWIND ;GO TO "BOT"
READ ,RLNK1 ;READ
BLKCMP ;COMPARE THE READ AND WRITE BUFFERS
RLNK1 ;LINK TO THE READ BUFFER
WLNK1 ;LINK TO THE WRITE BUFFER
BSBG ;BACK UP
READ ;READ
BLKCMP ;COMPARE THE READ AND WRITE BUFFERS
RLNK1 ;LINK TO THE READ BUFFER

```


TA11 MOTION TEST
DZTADC.NEW 24-MAY-76 00:00

MAINDEC-11-DZTAD-C
T4

MACY11 27(1006) 23-SEP-76 10:33 PAGE 18
TEST BSBG AFTER READ

SEQ 0034

```

761 003170 017022          WLNK1          ;LINK TO THE WRITE BUFFER
762          ;*****
763          ;*TEST 5      TEST BSFG AFTER READ
764          ;*****
765 003172 000004          TST5: SCOPE
766 003174 012767 003242 175770      MOV      #TST6,$ESCAPE    ;;ESCAPE TO TEST 6 ON ERROR
767 003202 104422          SELDRV          ;SELECT DRIVE FOR TESTING
768 003204 104421          REWIND          ;GO TO "BOT"
769 003206 104413 017022      WRITE      ,WLNK1          ;WRITE A BLOCK
770 003212 104421          REWIND          ;GO TO "BOT"
771 003214 104414 016766      READ       ,RLNK1          ;READ A BLOCK
772 003220 104423          BLKCMP          ;COMPARE THE READ AND WRITE BUFFERS
773 003222 016766          RLNK1          ;LINK TO THE READ BUFFER
774 003224 017022          WLNK1          ;LINK TO THE WRITE BUFFER
775 003226 104415          BSFG           ;BACK OVER THE DATA
776 003230 104414 016766      READ       ,RLNK1          ;NOW READ THE DATA
777 003234 104423          BLKCMP          ;COMPARE THE READ AND WRITE BUFFERS
778 003236 016766          RLNK1          ;LINK TO THE READ BUFFER
779 003240 017022          WLNK1          ;LINK TO THE WRITE BUFFER
780          ;*****
781          ;*TEST 6      TEST SFBG AFTER REWIND AND BSFG AFTER SFBG
782          ;*****
783 003242 000004          TST6: SCOPE
784 003244 012767 003302 175720      MOV      #TST7,$ESCAPE    ;;ESCAPE TO TEST 7 ON ERROR
785 003252 104422          SELDRV          ;SELECT DRIVE FOR TESTING
786 003254 104421          REWIND          ;GO TO "BOT"
787 003256 104413 017022      WRITE      ,WLNK1          ;WRITE ONE BLOCK OF DATA
788 003262 104421          REWIND          ;GO TO "BOT"
789 003264 104420          SFBG           ;SPACE OVER THE DATA
790 003266 104415          BSFG           ;BACK OVER THE DATA
791 003270 104414 016766      READ       ,RLNK1          ;READ THE BLOCK
792 003274 104423          BLKCMP          ;COMPARE THE READ AND WRITE BUFFERS
793 003276 016766          RLNK1          ;LINK TO THE READ BUFFER
794 003300 017022          WLNK1          ;LINK TO THE WRITE BUFFER
795          ;*****
796          ;*TEST 7      TEST WRITE AFTER READ
797          ;*****
798 003302 000004          TST7: SCOPE
799 003304 012767 003356 175660      MOV      #TST10,$ESCAPE  ;;ESCAPE TO TEST 10 ON ERROR
800 003312 104422          SELDRV          ;SELECT DRIVE FOR TESTING
801 003314 104421          REWIND          ;GO TO "CLEAR LEADER"
802 003316 104413 017022      WRITE      ,WLNK1          ;WRITE ONE BLOCK
803 003322 104416          BSBG           ;BACK OVER THE DATA BLOCK
804 003324 104414 016766      READ       ,RLNK1          ;AND READ IT
805 003330 104423          BLKCMP          ;COMPARE THE READ AND WRITE BUFFERS
806 003332 016766          RLNK1          ;LINK TO THE READ BUFFER
807 003334 017022          WLNK1          ;LINK TO THE WRITE BUFFER
808 003336 104413 017026      WRITE      ,WLNK2          ;WRITE A SECOND BLOCK
809 003342 104416          BSBG           ;BACK OVER IT
810 003344 104414 016772      READ       ,RLNK2          ;AND READ IT
811 003350 104423          BLKCMP          ;COMPARE THE READ AND WRITE BUFFERS
812 003352 016772          RLNK2          ;LINK TO THE READ BUFFER
813 003354 017026          WLNK2          ;LINK TO THE WRITE BUFFER
814          ;*****
815          ;*TEST 10      TEST WRITE AFTER WRITE AND READ AFTER READ
816          ;*****

```


TAII MOTION TEST
DZTADC.NEW

24-MAY-76 00:00
MAINDEC-11-DZTAD-C
T10

MACY11 27(1006) 23-SEP-76 10:33 PAGE 19
TEST WRITE AFTER WRITE AND READ AFTER READ

SEQ 0035

```

817 003356 000004 TST10: SCOPE
818 003360 012767 003422 175604 MOV #TST11,$ESCAPE ;;ESCAPE TO TEST 11 ON ERROR
819 003366 104422 SELDRV ;SELECT DRIVE FOR TESTING
820 003370 104421 REWIND ;GO TO "CLEAR LEADER"
821 003372 104413 017022 WRITE ,WLNK1 ;WRITE TWO(2) BLOCKS
822 003376 104413 017026 WRITE ,WLNK2
823 003402 104421 REWIND ;GO TO BOT
824 003404 104414 016766 READ ,RLNK1 ;READ BOTH BLOCKS
825 003410 104414 016772 READ ,RLNK2
826 003414 104423 BLKCMP ;COMPARE THE READ AND WRITE BUFFERS
827 003416 016772 RLNK2 ;LINK TO THE READ BUFFER
828 003420 017026 WLNK2 ;LINK TO THE WRITE BUFFER
829 *****
830 ;*TEST 11 TEST BSBG AFTER BSBG
831 *****
832 003422 000004 TST11: SCOPE
833 003424 012767 003464 175540 MOV #TST12,$ESCAPE ;;ESCAPE TO TEST 12 ON ERROR
834 003432 104422 SELDRV ;SELECT DRIVE FOR TESTING
835 003434 104421 REWIND ;GO TO "CLEAR LEADER"
836 003436 104413 017022 WRITE ,WLNK1 ;WRITE TWO(2) BLOCKS
837 003442 104413 017026 WRITE ,WLNK2
838 003446 104416 BSBG ;BACK OVER THE 2ND BLOCK
839 003450 104416 BSBG ;BACK OVER THE 1ST BLOCK
840 003452 104414 016766 READ ,RLNK1 ;READ THE 1ST BLOCK
841 003456 104423 BLKCMP ;COMPARE THE READ AND WRITE BUFFERS
842 003460 016766 RLNK1 ;LINK TO THE READ BUFFER
843 003462 017022 WLNK1 ;LINK TO THE WRITE BUFFER
844 *****
845 ;*TEST 12 TEST READ AFTER SFBG
846 *****
847 003464 000004 TST12: SCOPE
848 003466 012767 003526 175476 MOV #TST13,$ESCAPE ;;ESCAPE TO TEST 13 ON ERROR
849 003474 104422 SELDRV ;SELECT DRIVE FOR TESTING
850 003476 104421 REWIND ;GO TO "CLEAR LEADER"
851 003500 104413 017022 WRITE ,WLNK1 ;WRITE TWO(2) BLOCKS
852 003504 104413 017026 WRITE ,WLNK2
853 003510 104421 REWIND ;BOT
854 003512 104420 SFBG ;SPACE OVER THE 1ST BLOCK
855 003514 104414 016772 READ ,RLNK2 ;READ THE 2ND BLOCK
856 003520 104423 BLKCMP ;COMPARE THE READ AND WRITE BUFFERS
857 003522 016772 RLNK2 ;LINK TO THE READ BUFFER
858 003524 017026 WLNK2 ;LINK TO THE WRITE BUFFER
859 *****
860 ;*TEST 13 TEST SFBG AFTER BSBG
861 *****
862 003526 000004 TST13: SCOPE
863 003530 012767 003572 175434 MOV #TST14,$ESCAPE ;;ESCAPE TO TEST 14 ON ERROR
864 003536 104422 SELDRV ;SELECT DRIVE FOR TESTING
865 003540 104421 REWIND ;GO TO "CLEAR LEADER"
866 003542 104413 017022 WRITE ,WLNK1 ;WRITE TWO(2) BLOCKS
867 003546 104413 017026 WRITE ,WLNK2
868 003552 104416 BSBG ;BACK SPACE OVER BOTH BLOCKS
869 003554 104416 BSBG
870 003556 104420 SFBG ;SPACE FWD OVER 1ST BLOCK
871 003560 104414 016772 READ ,RLNK2 ;READ 2ND BLOCK
872 003564 104423 BLKCMP ;COMPARE THE READ AND WRITE BUFFERS

```


K03

TA11 MOTION TEST
DZTADC.NEW 24-MAY-76 00:00MAINDEC-11-DZTAD-C
T13MACY11 27(1006) 23-SEP-76 10:33 PAGE 20
TEST SFBG AFTER BSBG

SEQ 0036

```

873 003566 016772          RLNK2          ;LINK TO THE READ BUFFER
874 003570 017026          WLNK2          ;LINK TO THE WRITE BUFFER
875                                     ;*****
876                                     ;*TEST 14      TEST SFBG AFTER BSFG
877                                     ;*****
878 003572 000004          TST14: SCOPE
879 003574 012767 003634 175370      MOV      #TST15,$ESCAPE ;;ESCAPE TO TEST 15 ON ERROR
880 003602 104422          SELDRV          ;SELECT DRIVE FOR TESTING
881 003604 104421          REWIND          ;GO TO "CLEAR LEADER"
882 003606 104413 017022          WRITE      ,WLNK1          ;WRITE TWO BLOCKS
883 003612 104413 017026          WRITE      ,WLNK2
884 003616 104415          BSBG            ;SPACE REV FILE
885 003620 104420          SFBG            ;SPACE FWD BLOCK
886 003622 104414 016772          READ      ,RLNK2          ;READ THE 2ND BLOCK
887 003626 104423          BLKCMP          ;COMPARE THE READ AND WRITE BUFFERS
888 003630 016772          RLNK2          ;LINK TO THE READ BUFFER
889 003632 017026          WLNK2          ;LINK TO THE WRITE BUFFER
890                                     ;*****
891                                     ;*TEST 15-      REWRITE THE LAST BLOCK 3 TIMES AFTER DOING A WRITE AFTER WRITE
892                                     ;*****
893 003634 000004          TST15: SCOPE
894 003636 012767 004026 175326      MOV      #TST16,$ESCAPE ;;ESCAPE TO TEST 16 ON ERROR
895 003644 104422          SELDRV          ;SELECT DRIVE FOR TESTING
896 003646 104421          REWIND          ;GO TO "CLEAR LEADER"
897 003650 104413 017022          WRITE      ,WLNK1          ;WRITE 2 BLOCKS
898 003654 104413 017026          WRITE      ,WLNK2
899 003660 104416          BSBG            ;BACK OVER BOTH OF THEM
900 003662 104416          BSBG
901 003664 104414 016766          READ      ,RLNK1          ;READ THE FIRST ONE
902 003670 104423          BLKCMP          ;COMPARE THE READ AND WRITE BUFFERS
903 003672 016766          RLNK1          ;LINK TO THE READ BUFFER
904 003674 017022          WLNK1          ;LINK TO THE WRITE BUFFER
905 003676 104414 016772          READ      ,RLNK2          ;NOW READ THE 2ND ONE
906 003702 104423          BLKCMP          ;COMPARE THE READ AND WRITE BUFFERS
907 003704 016772          RLNK2          ;LINK TO THE READ BUFFER
908 003706 017026          WLNK2          ;LINK TO THE WRITE BUFFER
909 003710 104416          BSBG            ;BACK OVER THE 2ND ONE
910 003712 104413 017032          WRITE      ,WLNK3          ;WRITE OVER THE 2ND BLOCK
911 003716 104416          BSBG            ;NOW BACK UP AND
912 003720 104414 016776          READ      ,RLNK3          ;READ IT
913 003724 104423          BLKCMP          ;COMPARE THE READ AND WRITE BUFFERS
914 003726 016776          RLNK3          ;LINK TO THE READ BUFFER
915 003730 017032          WLNK3          ;LINK TO THE WRITE BUFFER
916 003732 104416          BSBG            ;BACK OVER IT AGAIN
917 003734 104413 017036          WRITE      ,WLNK4          ;AND WRITE OVER IT AGAIN
918 003740 104416          BSBG            ;BACK UP AND
919 003742 104414 017002          READ      ,RLNK4          ;READ THE BLOCK
920 003746 104423          BLKCMP          ;COMPARE THE READ AND WRITE BUFFERS
921 003750 017002          RLNK4          ;LINK TO THE READ BUFFER
922 003752 017036          WLNK4          ;LINK TO THE WRITE BUFFER
923 003754 104416          BSBG            ;BACK OVER THE BLOCK AGAIN
924 003756 104413 017026          WRITE      ,WLNK2          ;AND WRITE OVER IT AGAIN
925 003762 104416          BSBG            ;NOW BACK UP
926 003764 104414 016772          READ      ,RLNK2          ;AND READ IT
927 003770 104423          BLKCMP          ;COMPARE THE READ AND WRITE BUFFERS
928 003772 016772          RLNK2          ;LINK TO THE READ BUFFER

```


TA11 MOTION TEST MAINDEC-11-DZTAD-C
DZTADC.NEW 24-MAY-76 00:00 T15

MACY11 27(1006) 23-SEP-76 10:33 PAGE 21
REWRITE THE LAST BLOCK 3 TIMES AFTER DOING A WRITE AFTER WRITE

SEQ 0037

929	003774	017026		WLNK2		;LINK TO THE WRITE BUFFER
930	003776	104416		BSBG		;BACK OVER BOTH BLOCKS
931	004000	104416		BSBG		
932	004002	104414	016766	READ	,RLNK1	;AND READ THE 1ST ONE
933	004006	104423		BLKCMP		;COMPARE THE READ AND WRITE BUFFERS
934	004010	016766		RLNK1		;LINK TO THE READ BUFFER
935	004012	017022		WLNK1		;LINK TO THE WRITE BUFFER
936	004014	104414	016772	READ	,RLNK2	;AND THE 2ND ONE
937	004020	104423		BLKCMP		;COMPARE THE READ AND WRITE BUFFERS
938	004022	016772		RLNK2		;LINK TO THE READ BUFFER
939	004024	017026		WLNK2		;LINK TO THE WRITE BUFFER
940				*****		
941				;TEST 16 REWRITE THE LAST BLOCK 3 TIMES AFTER DOING A WRITE AFTER READ		
942				*****		
943	004026	000004		TST16: SCOPE		
944	004030	012767	004204 175134	MOV	#TST17,\$ESCAPE	;ESCAPE TO TEST 17 ON ERROR
945	004036	104422		SELDIV		;SELECT DRIVE FOR TESTING
946	004040	104421		REWIND		;GO TO "CLEAR LEADER"
947	004042	104413	017022	WRITE	,WLNK1	;WRITE A BLOCK OF DATA
948	004046	104416		BSBG		;BACK OVER IT
949	004050	104414	016766	READ	,RLNK1	;AND READ IT
950	004054	104423		BLKCMP		;COMPARE THE READ AND WRITE BUFFERS
951	004056	016766		RLNK1		;LINK TO THE READ BUFFER
952	004060	017022		WLNK1		;LINK TO THE WRITE BUFFER
953	004062	104413	017026	WRITE	,WLNK2	;WRITE 2ND BLOCK
954	004066	104416		BSBG		;BACK OVER IT
955	004070	104414	016772	READ	,RLNK2	;AND READ IT
956	004074	104423		BLKCMP		;COMPARE THE READ AND WRITE BUFFERS
957	004076	016772		RLNK2		;LINK TO THE READ BUFFER
958	004100	017026		WLNK2		;LINK TO THE WRITE BUFFER
959	004102	104416		BSBG		;BACK OVER IT AND
960	004104	104413	017036	WRITE	,WLNK4	;WRITE OVER 2ND BLOCK
961	004110	104416		BSBG		;BACK UP AND
962	004112	104414	017002	READ	,RLNK4	;READ IT
963	004116	104423		BLKCMP		;COMPARE THE READ AND WRITE BUFFERS
964	004120	017002		RLNK4		;LINK TO THE READ BUFFER
965	004122	017036		WLNK4		;LINK TO THE WRITE BUFFER
966	004124	104416		BSBG		;BACK OVER IT AGAIN
967	004126	104413	017042	WRITE	,WLNK5	;REWRITE 2ND BLOCK
968	004132	104416		BSBG		
969	004134	104414	017006	READ	,RLNK5	;READ 2ND BLOCK
970	004140	104423		BLKCMP		;COMPARE THE READ AND WRITE BUFFERS
971	004142	017006		RLNK5		;LINK TO THE READ BUFFER
972	004144	017042		WLNK5		;LINK TO THE WRITE BUFFER
973	004146	104416		BSBG		;BACK OVER IT AGAIN
974	004150	104413	017032	WRITE	,WLNK3	;WRITE ANOTHER BLOCK OVER IT
975	004154	104416		BSBG		;BACK OVER 2 BLOCKS
976	004156	104416		BSBG		
977	004160	104414	016766	READ	,RLNK1	;READ THE 1ST BLOCK
978	004164	104423		BLKCMP		;COMPARE THE READ AND WRITE BUFFERS
979	004166	016766		RLNK1		;LINK TO THE READ BUFFER
980	004170	017022		WLNK1		;LINK TO THE WRITE BUFFER
981	004172	104414	016776	READ	,RLNK3	;READ 2ND BLOCK
982	004176	104423		BLKCMP		;COMPARE THE READ AND WRITE BUFFERS
983	004200	016776		RLNK3		;LINK TO THE READ BUFFER
984	004202	017032		WLNK3		;LINK TO THE WRITE BUFFER

MO3

TA11 MOTION TEST
DZTADC.NEW 24-MAY-76 00:00

MAINDEC-11-DZTAD-C
T17

MACY11 27(1006) 23-SEP-76 10:33 PAGE 22
TEST WFG AFTER WRITE AND BSFG AFTER WFG

SEQ 0038

```

985
986
987
988 004204 000004
989 004206 012767 004242 174756
990 004214 104422
991 004216 104421
992 004220 104413 017022
993 004224 104412
994 004226 104415
995 004230 104414 016766
996 004234 104423
997 004236 016766
998 004240 017022
999
1000
1001
1002 004242 000004
1003 004244 012767 004300 174720
1004 004252 104422
1005 004254 104421
1006 004256 104413 017022
1007 004262 104412
1008 004264 104416
1009 004266 104414 016766
1010 004272 104423
1011 004274 016766
1012 004276 017022
1013
1014
1015
1016 004300 000004
1017 004302 012767 004356 174662
1018 004310 104422
1019 004312 104421
1020 004314 104413 017022
1021 004320 104421
1022 004322 104414 016766
1023 004326 104423
1024 004330 016766
1025 004332 017022
1026 004334 104412
1027 004336 104413 017026
1028 004342 104416
1029 004344 104414 016772
1030 004350 104423
1031 004352 016772
1032 004354 017026
1033
1034
1035
1036 004356 000004
1037 004360 012767 004434 174604
1038 004366 104422
1039 004370 104421
1040 004372 104413 017022

```

```

*****
*TEST 17      TEST WFG AFTER WRITE AND BSFG AFTER WFG
*****
TST17:  SCOPE
        MOV      #TST20,$ESCAPE ;;ESCAPE TO TEST 20 ON ERROR
        SELDRV   ;SELECT DRIVE FOR TESTING
        REWIND   ;GO TO "BOT"
        WRITE    ,WLNK1 ;WRITE ONE BLOCK
        WFG      ;FOLLOWED BY A FILE GAP
        BSFG     ;BACK OVER THE DATA BLOCK
        READ     ,RLNK1 ;READ THE DATA
        BLKCOMP  ;COMPARE THE READ AND WRITE BUFFERS
        RLNK1    ;LINK TO THE READ BUFFER
        WLNK1    ;LINK TO THE WRITE BUFFER
*****
*TEST 20      TEST BSFG AFTER WFG
*****
TST20:  SCOPE
        MOV      #TST21,$ESCAPE ;;ESCAPE TO TEST 21 ON ERROR
        SELDRV   ;SELECT DRIVE FOR TESTING
        REWIND   ;GO TO BOT
        WRITE    ,WLNK1 ;WRITE ONE BLOCK OF DATA
        WFG      ;WRITE A FILE GAP
        BSBG     ;BACK OVER THE DATA
        READ     ,RLNK1 ;READ ONE BLOCK
        BLKCOMP  ;COMPARE THE READ AND WRITE BUFFERS
        RLNK1    ;LINK TO THE READ BUFFER
        WLNK1    ;LINK TO THE WRITE BUFFER
*****
*TEST 21      TEST WRITE AFTER WFG
*****
TST21:  SCOPE
        MOV      #TST22,$ESCAPE ;;ESCAPE TO TEST 22 ON ERROR
        SELDRV   ;SELECT DRIVE FOR TESTING
        REWIND   ;GO TO BOT
        WRITE    ,WLNK1 ;WRITE ONE BOCK
        REWIND   ;GO TO BOT
        READ     ,RLNK1 ;READ ONE BLOCK
        BLKCOMP  ;COMPARE THE READ AND WRITE BUFFERS
        RLNK1    ;LINK TO THE READ BUFFER
        WLNK1    ;LINK TO THE WRITE BUFFER
        WFG      ;WRITE A FILE GAP
        WRITE    ,WLNK2 ;WRITE ANOTHER BLOCK
        BSBG     ;BACK OVER THE LAST BLOCK
        READ     ,RLNK2 ;READ THE SECOND BLOCK
        BLKCOMP  ;COMPARE THE READ AND WRITE BUFFERS
        RLNK2    ;LINK TO THE READ BUFFER
        WLNK2    ;LINK TO THE WRITE BUFFER
*****
*TEST 22      TEST WFG AFTER READ AND BSFG AFTER WRITE
*****
TST22:  SCOPE
        MOV      #TST23,$ESCAPE ;;ESCAPE TO TEST 23 ON ERROR
        SELDRV   ;SELECT DRIVE FOR TESTING
        REWIND   ;GO TO "BOT"
        WRITE    ,WLNK1 ;WRITE ONE BLOCK

```


TA11 MOTION TEST
DZTADC.NEW 24-MAY-76 00:00

MAINDEC-11-DZTAD-C
T22

MACY11 27(1006) 23-SEP-76 10:33 PAGE 23
TEST WFG AFTER READ AND BSFG AFTER WRITE

SEQ 0039

1041	004376	104421		REWIND		:GO TO "BOT"
1042	004400	104414	016766	READ ,RLNK1		:READ ONE BLOCK
1043	004404	104423		BLKCMP		:COMPARE THE READ AND WRITE BUFFERS
1044	004406	016766		RLNK1		:LINK TO THE READ BUFFER
1045	004410	017022		WLNK1		:LINK TO THE WRITE BUFFER
1046	004412	104412		WFG		:WRITE A FILE GAP
1047	004414	104413	017026	WRITE ,WLNK2		:WRITE A BLOCK
1048	004420	104415		BSFG		:BACK OVER THE DATA
1049	004422	104414	016772	READ ,RLNK2		:READ SECOND BLOCK
1050	004426	104423		BLKCMP		:COMPARE THE READ AND WRITE BUFFERS
1051	004430	016772		RLNK2		:LINK TO THE READ BUFFER
1052	004432	017026		WLNK2		:LINK TO THE WRITE BUFFER
1053				:*****		
1054				:TEST 23 TEST SFFG AFTER REWIND AND READ AFTER SFFG		
1055				:*****		
1056	004434	000004		TST23: SCOPE		
1057	004436	012767	004500 174526	MOV #TST24,\$ESCAPE ;	;;ESCAPE TO TEST 24 ON ERROR	
1058	004444	104422		SELDRV		:SELECT DRIVE FOR TESTING
1059	004446	104421		REWIND		:GO TO "BOT"
1060	004450	104413	017022	WRITE ,WLNK1		:WRITE A BLOCK OF DATA
1061	004454	104412		WFG		:WRITE A FILE GAP
1062	004456	104413	017026	WRITE ,WLNK2		:WRITE A BLOCK OF DATA
1063	004462	104421		REWIND		:GO TO "BOT"
1064	004464	104417		SFFG		:SPACE OVER FIRST BLOCK
1065	004466	104414	016772	READ ,RLNK2		:READ THE 2ND BLOCK
1066	004472	104423		BLKCMP		:COMPARE THE READ AND WRITE BUFFERS
1067	004474	016772		RLNK2		:LINK TO THE READ BUFFER
1068	004476	017026		WLNK2		:LINK TO THE WRITE BUFFER
1069				:*****		
1070				:TEST 24 TEST BSBG AFTER SFFG		
1071				:*****		
1072	004500	000004		TST24: SCOPE		
1073	004502	012767	004546 174462	MOV #TST25,\$ESCAPE ;	;;ESCAPE TO TEST 25 ON ERROR	
1074	004510	104422		SELDRV		:SELECT DRIVE FOR TESTING
1075	004512	104421		REWIND		:GO TO "BOT"
1076	004514	104413	017022	WRITE ,WLNK1		:WRITE DATA
1077	004520	104412		WFG		:WRITE FILE GAP
1078	004522	104413	017026	WRITE ,WLNK2		:WRITE DATA
1079	004526	104421		REWIND		:GO TO "BOT"
1080	004530	104417		SFFG		:SPACE OVER FIRST FILE
1081	004532	104416		BSBG		:BACKUP TO FIRST BLOCK
1082	004534	104414	016766	READ ,RLNK1		:READ THE BLOCK
1083	004540	104423		BLKCMP		:COMPARE THE READ AND WRITE BUFFERS
1084	004542	016766		RLNK1		:LINK TO THE READ BUFFER
1085	004544	017022		WLNK1		:LINK TO THE WRITE BUFFER
1086				:*****		
1087				:TEST 25 TEST SFBG AFTER SFFG		
1088				:*****		
1089	004546	000004		TST25: SCOPE		
1090	004550	012767	004622 174414	MOV #TST26,\$ESCAPE ;	;;ESCAPE TO TEST 26 ON ERROR	
1091	004556	104422		SELDRV		:SELECT DRIVE FOR TESTING
1092	004560	104421		REWIND		:GO TO "BOT"
1093	004562	104413	017022	WRITE ,WLNK1		:WRITE A BLOCK OF DATA
1094	004566	104412		WFG		:FOLLOWED BY A FILE GAP
1095	004570	104413	017026	WRITE ,WLNK2		:FOLLOWED BY 2 BLOCKS OF DATA
1096	004574	104413	017032	WRITE ,WLNK3		

B04

 TAIL MOTION TEST
 DZTADC.NEW 24-MAY-76 00:00

 MAINDEC-11-DZTAD-C
 T25

 MACY11 27(1006) 23-SEP-76 10:33 PAGE 24
 TEST SFBG AFTER SFFG

SEQ 0040

1097	004600	104421		REWIND		:GO TO "BOT"
1098	004602	104417		SFFG		:SPACE OVER 1ST FILE
1099	004604	104420		SFBG		:SPACE OVER BLOCK
1100	004606	104416		BSBG		:BACK OVER THE BLOCK
1101	004610	104414	016772	READ	,RLNK2	:READ THE LAST BLOCK
1102	004614	104423		BLKCMP		:COMPARE THE READ AND WRITE BUFFERS
1103	004616	016772		RLNK2		:LINK TO THE READ BUFFER
1104	004620	017026		WLNK2		:LINK TO THE WRITE BUFFER
:*****						
1105				*TEST 26	TEST SFBG AFTER SFBG	
:*****						
1106				†ST26:	SCOPE	
1107				MOV	#TST27,\$ESCAPE	::ESCAPE TO TEST 27 ON ERROR
1108	004622	000004		SELDRV		:SELECT DRIVE FOR TESTING
1109	004624	012767	004702 174340	REWIND		:GO TO "BOT"
1110	004632	104422		WRITE	,WLNK1	:WRITE A BLOCK OF DATA
1111	004634	104421		WFG		:WRITE A FILE GAP
1112	004636	104413	017022	WRITE	,WLNK2	:WRITE 3 BLOCKS OF DATA
1113	004642	104412		WRITE	,WLNK3	
1114	004644	104413	017026	WRITE	,WLNK4	
1115	004650	104413	017032	REWIND		:GO TO "BOT"
1116	004654	104413	017036	SFFG		:SPACE OVER THE 1ST FILE
1117	004660	104421		SFBG		:SPACE OVER TWO BLOCK
1118	004662	104417		SFBG		
1119	004664	104420		READ	,RLNK4	:READ LAST BLOCK OF DATA
1120	004666	104420		BLKCMP		:COMPARE THE READ AND WRITE BUFFERS
1121	004670	104414	017002	RLNK4		:LINK TO THE READ BUFFER
1122	004674	104423		WLNK4		:LINK TO THE WRITE BUFFER
1123	004676	017002		:*****		
1124	004700	017036		*TEST 27	TEST BSFG AFTER SFBG AND BSFG AFTER BSFG	
:*****						
1125				†ST27:	SCOPE	
1126				MOV	#TST30,\$ESCAPE	::ESCAPE TO TEST 30 ON ERROR
1127				SELDRV		:SELECT DRIVE FOR TESTING
1128	004702	000004		REWIND		:GO TO "BOT"
1129	004704	012767	004754 174260	WRITE	,WLNK1	:WRITE ONE BLOCK OF DATA
1130	004712	104422		WFG		:WRITE A FILE GAP
1131	004714	104421		WRITE	,WLNK2	:WRITE A BLOCK OF DATA
1132	004716	104413	017022	REWIND		:GO TO "BOT"
1133	004722	104412		SFFG		:SPACE OVER 1ST FILE
1134	004724	104413	017026	SFBG		:SPACE OVER 1ST BLOCK OF 2ND FILE
1135	004730	104421		BSFG		:BACK TO BEGINNING OF 2ND FILE
1136	004732	104417		BSFG		:BACK TO BEGINNING OF 1ST FILE
1137	004734	104420		READ	,RLNK1	:READ 1ST BLOCK OF 1ST FILE
1138	004736	104415		BLKCMP		:COMPARE THE READ AND WRITE BUFFERS
1139	004740	104415		RLNK1		:LINK TO THE READ BUFFER
1140	004742	104414	016766	WLNK1		:LINK TO THE WRITE BUFFER
1141	004746	104423		:*****		
1142	004750	016766		*TEST 30	TEST BSFG THRU A FILE GAP	
1143	004752	017022		:*****		
1144				†ST30:	SCOPE	
1145				MOV	#TST31,\$ESCAPE	::ESCAPE TO TEST 31 ON ERROR
1146				SELDRV		:SELECT DRIVE FOR TESTING
1147	004754	000004		REWIND		:GO TO "BOT"
1148	004756	012767	005032 174206	WRITE	,WLNK1	:WRITE ONE BLOCK OF DATA
1149	004764	104422		WFG		:WRITE A FILE GAP
1150	004766	104421				
1151	004770	104413	017022			
1152	004774	104412				

C04

TA11 MOTION TEST
DZTADC.NEW 24-MAY-76 00:00MAINDEC-11-DZTAD-C
T30MACY11 27(1006) 23-SEP-76 10:33 PAGE 25
TEST BSBG THRU A FILE GAP

SEQ 0041

1153	004776	104413	017026	WRITE	,WLNK2	:WRITE TWO BLOCKS OF DATA
1154	005002	104413	017032	WRITE	,WLNK3	
1155	005006	104421		REWIND		:GO TO "BOT"
1156	005010	104417		SFFG		:SPACE OVER 1ST FILE
1157	005012	104420		SFBG		:SPACE OVER 1ST BLOCK OF 2ND FILE
1158	005014	104416		BSBG		:BACK OVER 1 BLOCK
1159	005016	104416		BSBG		:BACK OVER 1 BLOCK
1160	005020	104414	016766	READ	,RLNK1	:READ
1161	005024	104423		BLKCMP		:COMPARE THE READ AND WRITE BUFFERS
1162	005026	016766		RLNK1		:LINK TO THE READ BUFFER
1163	005030	017022		WLNK1		:LINK TO THE WRITE BUFFER
1164				:*****		
1165				:TEST 31 TEST BSFG AFTER BSBG		
1166				:*****		
1167	005032	000004		TST31: SCOPE		
1168	005034	012767	005104 174130	MOV	#TST32,\$ESCAPE	::ESCAPE TO TEST 32 ON ERROR
1169	005042	104422		SELDIV		:SELECT DRIVE FOR TESTING
1170	005044	104421		REWIND		:GO TO "BOT"
1171	005046	104413	017022	WRITE	,WLNK1	:WRITE ONE BLOCK OF DATA
1172	005052	104412		WFG		:WRITE A FILE GAP
1173	005054	104413	017026	WRITE	,WLNK2	:WRITE A BLOCK OF DATA
1174	005060	104421		REWIND		:GO TO "BOT"
1175	005062	104417		SFFG		:SPACE OVER FILE
1176	005064	104420		SFBG		:SPACE OVER 1ST BLOCK OF 2ND FILE
1177	005066	104416		BSBG		:BACK OVER BLOCK
1178	005070	104415		BSFG		:BACK OVER FILE
1179	005072	104414	016766	READ	,RLNK1	:READ 1ST BLOCK ON TAPE
1180	005076	104423		BLKCMP		:COMPARE THE READ AND WRITE BUFFERS
1181	005100	016766		RLNK1		:LINK TO THE READ BUFFER
1182	005102	017022		WLNK1		:LINK TO THE WRITE BUFFER
1183				:*****		
1184				:TEST 32 TEST BSBG AFTER BSFG		
1185				:*****		
1186	005104	000004		TST32: SCOPE		
1187	005106	012767	005150 174056	MOV	#TST33,\$ESCAPE	::ESCAPE TO TEST 33 ON ERROR
1188	005114	104422		SELDIV		:SELECT DRIVE FOR TESTING
1189	005116	104421		REWIND		:GO TO "CLEAR LEADER"
1190	005120	104413	017022	WRITE	,WLNK1	:WRITE BLOCK
1191	005124	104412		WFG		:WRITE A FILE GAP
1192	005126	104413	017026	WRITE	,WLNK2	:FOLLOW WITH ANOTHER BLOCK
1193	005132	104415		BSFG		:BACK UP 1 FILE
1194	005134	104416		BSBG		:BACK UP 1 BLOCK
1195	005136	104414	016766	READ	,RLNK1	:READ 1ST BLOCK ON TAPE
1196	005142	104423		BLKCMP		:COMPARE THE READ AND WRITE BUFFERS
1197	005144	016766		RLNK1		:LINK TO THE READ BUFFER
1198	005146	017022		WLNK1		:LINK TO THE WRITE BUFFER
1199				:*****		
1200				:TEST 33 TEST BSFG AFTER SFFG AND SFFG AFTER READ		
1201				:*****		
1202	005150	000004		TST33: SCOPE		
1203	005152	012767	005232 174012	MOV	#TST34,\$ESCAPE	::ESCAPE TO TEST 34 ON ERROR
1204	005160	104422		SELDIV		:SELECT DRIVE FOR TESTING
1205	005162	104421		REWIND		:GO TO "BOT"
1206	005164	104413	017022	WRITE	,WLNK1	:WRITE A BLOCK OF DATA
1207	005170	104412		WFG		:WRITE A FILE GAP
1208	005172	104413	017026	WRITE	,WLNK2	:WRITE A BLOCK OF DATA

D04

TAII MOTION TEST
DZTADC.NEW 24-MAY-76 00:00 T33

MAINDEC-11-DZTAD-C
00:00 T33

MACY11 27(1006) 23-SEP-76 10:33 PAGE 26
TEST BSFG AFTER SFFG AND SFFG AFTER READ

SEQ 0042

1209	005176	104421		REWIND		:GO TO "BOT"
1210	005200	104417		SFFG		:SPACE OVER 1ST FILE
1211	005202	104415		BSFG		:BACK TO BEGINNING OF 1ST FILE
1212	005204	104414	016766	READ	,RLNK1	:READ FIRST DATA BLOCK
1213	005210	104423		BLKCOMP		:COMPARE THE READ AND WRITE BUFFERS
1214	005212	016766		RLNK1		:LINK TO THE READ BUFFER
1215	005214	017022		WLNK1		:LINK TO THE WRITE BUFFER
1216	005216	104417		SFFG		:SPACE TO BEGINNING ON 2ND FILE
1217	005220	104414	016772	READ	,RLNK2	:READ 1ST BLOCK OF 2ND FILE
1218	005224	104423		BLKCOMP		:COMPARE THE READ AND WRITE BUFFERS
1219	005226	016772		RLNK2		:LINK TO THE READ BUFFER
1220	005230	017026		WLNK2		:LINK TO THE WRITE BUFFER
1221				:*****		
1222				:TEST 34 TEST SFFG AFTER SFFG		
1223				:*****		
1224	005232	000004		TST34: SCOPE		
1225	005234	012767	005310 173730	MOV	#TST35,\$ESCAPE	::ESCAPE TO TEST 35 ON ERROR
1226	005242	104422		SELDIV		:SELECT DRIVE FOR TESTING
1227	005244	104421		REWIND		:GO TO BOT
1228	005246	104413	017022	WRITE	,WLNK1	:WRITE A BLOCK
1229	005252	104412		WFG		:WRITE FILE GAP
1230	005254	104413	017026	WRITE	,WLNK2	:WRITE A BLOCK
1231	005260	104412		WFG		:WRITE A FILE GAP
1232	005262	104413	017032	WRITE	,WLNK3	:WRITE A BLOCK
1233	005266	104412		WFG		:WRITE A FILE GAP
1234	005270	104421		REWIND		:GO TO "BOT"
1235	005272	104417		SFFG		:SPACE OVER 1ST FILE
1236	005274	104417		SFFG		:SPACE OVER 2ND FILE
1237	005276	104414	016776	READ	,RLNK3	:READ BLOCK OF 3RD FILE
1238	005302	104423		BLKCOMP		:COMPARE THE READ AND WRITE BUFFERS
1239	005304	016776		RLNK3		:LINK TO THE READ BUFFER
1240	005306	017032		WLNK3		:LINK TO THE WRITE BUFFER
1241				:*****		
1242				:TEST 35 TEST SFFG AFTER SFBG		
1243				:*****		
1244	005310	000004		TST35: SCOPE		
1245	005312	012767	005364 173652	MOV	#TST36,\$ESCAPE	::ESCAPE TO TEST 36 ON ERROR
1246	005320	104422		SELDIV		:SELECT DRIVE FOR TESTING
1247	005322	104421		REWIND		:GO TO "BOT"
1248	005324	104413	017022	WRITE	,WLNK1	:WRITE A BLOCK
1249	005330	104413	017026	WRITE	,WLNK2	:WRITE A BLOCK
1250	005334	104412		WFG		:WRITE A GAP
1251	005336	104413	017032	WRITE	,WLNK3	:WRITE A BLOCK
1252	005342	104412		WFG		:WRITE A GAP
1253	005344	104421		REWIND		:GO TO "BOT"
1254	005346	104420		SFBG		:SPACE OVER THE 1ST BLOCK(AND 1ST FILE)
1255	005350	104417		SFFG		:SPACE TO BEGINNING OF 2ND FILE
1256	005352	104414	016776	READ	,RLNK3	:READ 1ST BLOCK OF 2ND FILE
1257	005356	104423		BLKCOMP		:COMPARE THE READ AND WRITE BUFFERS
1258	005360	016776		RLNK3		:LINK TO THE READ BUFFER
1259	005362	017032		WLNK3		:LINK TO THE WRITE BUFFER
1260				:*****		
1261				:TEST 36 TEST WFG AFTER SFBG		
1262				:*****		
1263	005364	000004		TST36: SCOPE		
1264	005366	012767	005436 173576	MOV	#TST37,\$ESCAPE	::ESCAPE TO TEST 37 ON ERROR

E04

TA11 MOTION TEST
DZTADC.NEW 24-MAY-76 00:00 T36

MACY11 27(1006) 23-SEP-76 10:33 PAGE 27
TEST WFG AFTER SFBG

SEQ 0043

1265	005374	104422		SELDRV		;SELECT DRIVE FOR TESTING
1266	005376	104421		REWIND		
1267	005400	104413	017022	WRITE	,WLNK1	;WRITE 2 BLOCKS
1268	005404	104413	017026	WRITE	,WLNK2	
1269	005410	104421		REWIND		;GO TO "BOT"
1270	005412	104420		SFBG		;GET OVER THE 1ST BLOCK
1271	005414	104412		WFG		;WRITE A GAP OVER 2ND BLOCK
1272	005416	104413	017032	WRITE	,WLNK3	;WRITE A BLOCK
1273	005422	104415		BSFG		;BACK OVER THE BLOCK
1274	005424	104414	016776	READ	,RLNK3	;AND READ IT
1275	005430	104423		BLKCOMP		;COMPARE THE READ AND WRITE BUFFERS
1276	005432	016776		RLNK3		;LINK TO THE READ BUFFER
1277	005434	017032		WLNK3		;LINK TO THE WRITE BUFFER
1278						
1279						
1280						
1281	005436	000004		TEST 37		TEST WRITE AFTER SFBG
1282	005440	012767	005510 173524	TEST 37:		
1283	005446	104422		MOV	#TST40,\$ESCAPE	;;ESCAPE TO TEST 40 ON ERROR
1284	005450	104421		SELDRV		;SELECT DRIVE FOR TESTING
1285	005452	104413	017022	REWIND		;GO TO BOT
1286	005456	104413	017026	WRITE	,WLNK1	;WRITE 2 BLOCK
1287	005462	104421		WRITE	,WLNK2	
1288	005464	104420		REWIND		;GO TO BOT
1289	005466	104413	017032	SFBG		;SPACE OVER 1ST BLOCK
1290	005472	104415		WRITE	,WLNK3	;WRITE OVER 2ND BLOCK
1291	005474	104420		BSFG		;GO TO BEGINNING OF FILE
1292	005476	104414	016776	SFBG		;SPACE OVER 1ST BLOCK
1293	005502	104423		READ	,RLNK3	;READ THE NEW 2ND BLOCK
1294	005504	016776		BLKCOMP		;COMPARE THE READ AND WRITE BUFFERS
1295	005506	017032		RLNK3		;LINK TO THE READ BUFFER
1296				WLNK3		;LINK TO THE WRITE BUFFER
1297						
1298						
1299	005510	000004		TEST 40		3 ONE BLOCK FILES---MULTI SPACING & TEST SFFG AFTER BSFG
1300	005512	012767	005600 173452	TEST 40:		
1301	005520	104422		MOV	#TST41,\$ESCAPE	;;ESCAPE TO TEST 41 ON ERROR
1302	005522	104421		SELDRV		;SELECT DRIVE FOR TESTING
1303	005524	104413	017022	REWIND		;GO TO "BOT"
1304	005530	104412		WRITE	,WLNK1	;WRITE A BLOCK
1305	005532	104413	017026	WFG		;FILE GAP
1306	005536	104412		WRITE	,WLNK2	;BLOCK
1307	005540	104413	017032	WFG		;FILE GAP
1308	005544	104412		WRITE	,WLNK3	;BLOCK
1309	005546	104421		WFG		;FILE GAP
1310	005550	104417		REWIND		;GO TO "BOT"
1311	005552	104417		SFFG		;SPACE OVER 1ST FILE
1312	005554	104417		SFFG		;SPACE OVER 2ND FILE
1313	005556	104415		SFFG		;SPACE OVER 3RD FILE
1314	005560	104415		BSFG		;BACK OVER 3RD FILE
1315	005562	104415		BSFG		;BACK OVER 2ND FILE
1316	005564	104417		BSFG		;BACK OVER 1ST FILE
1317	005566	104414	016772	SFFG		;GO TO BEGINNING OF 2ND FILE
1318	005572	104423		READ	,RLNK2	;READ THE BLOCK
1319	005574	016772		BLKCOMP		;COMPARE THE READ AND WRITE BUFFERS
1320	005576	017026		RLNK2		;LINK TO THE READ BUFFER
				WLNK2		;LINK TO THE WRITE BUFFER

F04

TA11 MOTION TEST
DZTADC.NEWMAINDEC-11-DZTAD-C
24-MAY-76 00:00 T41MACY11 27(1006) 23-SEP-76 10:33 PAGE 28
TWO BLOCK FILES---MULTI SPACING FUNCTIONS

SEQ 0044

```

1321
1322
1323
1324 005600 000004
1325 005602 012767 005706 173362
1326 005610 104422
1327 005612 104421
1328 005614 104413 017022
1329 005620 104413 017026
1330 005624 104412
1331 005626 104413 017032
1332 005632 104413 017036
1333 005636 104412
1334 005640 104413 017042
1335 005644 104413 017046
1336 005650 104412
1337 005652 104421
1338 005654 104417
1339 005656 104417
1340 005660 104414 017006
1341 005664 104423
1342 005666 017006
1343 005670 017042
1344 005672 104415
1345 005674 104414 017006
1346 005700 104423
1347 005702 017006
1348 005704 017042
1349
1350
1351
1352 005706 000004
1353 005710 012767 006056 173254
1354 005716 104422
1355 005720 104421
1356 005722 104413 017022
1357 005726 104413 017026
1358 005732 104413 017032
1359 005736 104412
1360 005740 104413 017036
1361 005744 104413 017042
1362 005750 104412
1363 005752 104421
1364 005754 104417
1365 005756 104417
1366 005760 104415
1367 005762 104414 017002
1368 005766 104423
1369 005770 017002
1370 005772 017036
1371 005774 104415
1372 005776 104415
1373 006000 104414 016766
1374 006004 104423
1375 006006 016766
1376 006010 017022

*****
*TEST 41 TWO BLOCK FILES---MULTI SPACING FUNCTIONS
*****
TST41: SCOPE
MOV #TST42,$ESCAPE ;;ESCAPE TO TEST 42 ON ERROR
SELDRV ;SELECT DRIVE FOR TESTING
REWIND ;GO TO "BOT"
WRITE ,WLNK1 ;WRITE A 2 BLOCK FILE
WRITE ,WLNK2
WFG
WRITE ,WLNK3 ;WRITE A 2 BLOCK FILE
WRITE ,WLNK4
WFG
WRITE ,WLNK5 ;WRITE A 2 BLOCK FILE
WRITE ,WLNK6
WFG
REWIND ;GO TO "BOT"
SFFG ;SPACE OVER 1ST FILE
SFFG ;SPACE OVER 2ND FILE
READ ,RLNK5 ;READ 1ST BLOCK OF 3RD FILE
BLKCMP ;COMPARE THE READ AND WRITE BUFFERS
RLNK5 ;LINK TO THE READ BUFFER
WLNK5 ;LINK TO THE WRITE BUFFER
BSFG ;BACK TO BEGINNING OF 3RD FILE
READ ,RLNK5 ;READ 1ST BLOCK OF 3RD FILE
BLKCMP ;COMPARE THE READ AND WRITE BUFFERS
RLNK5 ;LINK TO THE READ BUFFER
WLNK5 ;LINK TO THE WRITE BUFFER
*****
*TEST 42 MULTI SPACING & TEST SFBG AFTER READ
*****
TST42: SCOPE
MOV #TST42,$ESCAPE ;;ESCAPE TO TEST 43 ON ERROR
SELDRV ;SELECT DRIVE FOR TESTING
REWIND ;GO TO "BOT"
WRITE ,WLNK1 ;WRITE A 3 BLOCK FILE
WRITE ,WLNK2
WRITE ,WLNK3
WFG
WRITE ,WLNK4 ;WRITE A 2 BLOCK FILE
WRITE ,WLNK5
WFG
REWIND ;GO TO "BOT"
SFFG ;SPACE OVER THE 1ST FILE
SFFG ;SPACE OVER THE 2ND FILE
BSFG ;BACK TO BEGINNING OF 2 FILE
READ ,RLNK4 ;READ 1ST BLOCK OF 2ND FILE
BLKCMP ;COMPARE THE READ AND WRITE BUFFERS
RLNK4 ;LINK TO THE READ BUFFER
WLNK4 ;LINK TO THE WRITE BUFFER
BSFG ;BACK TO BEGINNING OF 2ND FILE
BSFG ;BACK OVER 1ST FILE
READ ,RLNK1 ;READ 1ND BLOCK OF 1ST FILE
BLKCMP ;COMPARE THE READ AND WRITE BUFFERS
RLNK1 ;LINK TO THE READ BUFFER
WLNK1 ;LINK TO THE WRITE BUFFER

```


G04

 TAIL MOTION TEST
 DZTADC.NEW 24-MAY-76 00:00 MAINDEC-11-DZTAD-C
 T42

 MACY11 27(1006) 23-SEP-76 10:33 PAGE 29
 MULTI SPACING & TEST SFBG AFTER READ

SEQ 0045

1377	006012	104415		BSFG			GO TO BEGINNING OF 1ST FILE
1378	006014	104414	016766	READ	,RLNK1		READ 1ST BLOCK
1379	006020	104423		BLKCMP			COMPARE THE READ AND WRITE BUFFERS
1380	006022	016766		RLNK1			LINK TO THE READ BUFFER
1381	006024	017022		WLNK1			LINK TO THE WRITE BUFFER
1382	006026	104420		SFBG			GET OVER 2ND BLOCK
1383	006030	104414	016776	READ	,RLNK3		READ 3RD BLOCK
1384	006034	104423		BLKCMP			COMPARE THE READ AND WRITE BUFFERS
1385	006036	016776		RLNK3			LINK TO THE READ BUFFER
1386	006040	017032		WLNK3			LINK TO THE WRITE BUFFER
1387	006042	104417		SFBG			GO TO BEGINNING OF 2ND FILE
1388	006044	104414	017002	READ	,RLNK4		READ 1ST BLOCK OF 2ND FILE
1389	006050	104423		BLKCMP			COMPARE THE READ AND WRITE BUFFERS
1390	006052	017002		RLNK4			LINK TO THE READ BUFFER
1391	006054	017036		WLNK4			LINK TO THE WRITE BUFFER
1392				*****			
1393				*TEST 43 TWO BLOCK FILES--MULTI SPACING & TEST SFBG AFTER BSFG			
1394				*****			
1395	006056	000004		TST43: SCOPE			
1396	006060	012767	006176 173104	MOV	#TST44,\$ESCAPE	::ESCAPE TO TEST 44 ON ERROR	
1397	006066	104422		SELDIV		SELECT DRIVE FOR TESTING	
1398	006070	104421		REWIND		GO TO BOT	
1399	006072	104413	017022	WRITE	,WLNK1	WRITE A 2 BLOCK FILE	
1400	006076	104413	017026	WRITE	,WLNK2		
1401	006102	104412		WFG			
1402	006104	104413	017032	WRITE	,WLNK3	WRITE A 2 BLOCK FILE	
1403	006110	104413	017036	WRITE	,WLNK4		
1404	006114	104412		WFG			
1405	006116	104413	017042	WRITE	,WLNK5	WRITE A 2 BLOCK FILE	
1406	006122	104413	017046	WRITE	,WLNK6		
1407	006126	104412		WFG			
1408	006130	104421		REWIND		BOT	
1409	006132	104417		SFBG		SPACE OVER 3 FILES	
1410	006134	104417		SFBG			
1411	006136	104417		SFBG			
1412	006140	104415		BSFG		BACK OVER 2 FILES	
1413	006142	104415		BSFG			
1414	006144	104416		BSFG			
1415	006146	104417		SFBG		BACK TO 2ND BLOCK OF 1ST FILE	
1416	006150	104414	016776	READ	,RLNK3	GO TO BEGINNING OF 2ND FILE	
1417	006154	104423		BLKCMP		READ 1ST RECORD OF 2ND FILE	
1418	006156	016776		RLNK3		COMPARE THE READ AND WRITE BUFFERS	
1419	006160	017032		WLNK3		LINK TO THE READ BUFFER	
1420	006162	104417		SFBG		LINK TO THE WRITE BUFFER	
1421	006164	104414	017006	READ	,RLNK5	GO TO 3RD FILE	
1422	006170	104423		BLKCMP		READ 1ST RECORD OF 3RD FILE	
1423	006172	017006		RLNK5		COMPARE THE READ AND WRITE BUFFERS	
1424	006174	017042		WLNK5		LINK TO THE READ BUFFER	
1425				*****			
1426				*TEST 44 6 ONE BLOCK FILES---MULTI SPACING			
1427				*****			
1428	006176	000004		TST44: SCOPE			
1429	006200	012767	006366 172764	MOV	#SEOP,\$ESCAPE	::ESCAPE TO SEOP ON ERROR	
1430	006206	104422		SELDIV		SELECT DRIVE FOR TESTING	
1431	006210	104421		REWIND		GO TO BOT	
1432	006212	104413	017022	WRITE	,WLNK1	WRITE A 1 RECORD FILE	

H04

 TAIL MOTION TEST
 DZTADC.NEW 24-MAY-76 00:00 T44

 MACY11 27(1006) 23-SEP-76 10:33 PAGE 30
 6 ONE BLOCK FILES---MULTI SPACING

SEQ 0046

1433	006216	104412		WFG		
1434	006220	104413	017026	WRITE	,WLNK2	;WRITE A 1 BLOCK FILE
1435	006224	104412		WFG		
1436	006226	104413	017032	WRITE	,WLNK3	;WRITE A 1 BLOCK FILE
1437	006232	104412		WFG		
1438	006234	104413	017036	WRITE	,WLNK4	;WRITE A 1 BLOCK FILE
1439	006240	104412		WFG		
1440	006242	104413	017042	WRITE	,WLNK5	;WRITE A 1 BLOCK FILE
1441	006246	104412		WFG		
1442	006250	104413	017046	WRITE	,WLNK6	;WRITE A 1 BLOCK FILE
1443	006254	104412		WFG		
1444	006256	104421		REWIND		;BOT
1445	006260	104417		SFFG		;SPACE OVER 1ST FILE
1446	006262	104417		SFFG		;SPACE OVER 2ND FILE
1447	006264	104417		SFFG		;SPACE OVER 3RD FILE
1448	006266	104417		SFFG		;SPACE OVER 4TH FILE
1449	006270	104417		SFFG		;SPACE OVER 5TH FILE
1450	006272	104417		SFFG		;SPACE OVER 6TH FILE
1451	006274	104415		BSFG		;NOW BACK UP AND READ IT
1452	006276	104414	017012	READ	,RLNK6	
1453	006302	104423		BLKCMP		;COMPARE THE READ AND WRITE BUFFERS
1454	006304	017012		RLNK6		;LINK TO THE READ BUFFER
1455	006306	017046		WLNK6		;LINK TO THE WRITE BUFFER
1456	006310	104415		BSFG		;BACK UP TO 5TH FILE
1457	006312	104415		BSFG		
1458	006314	104414	017006	READ	,RLNK5	;AND READ IT
1459	006320	104423		BLKCMP		;COMPARE THE READ AND WRITE BUFFERS
1460	006322	017006		RLNK5		;LINK TO THE READ BUFFER
1461	006324	017042		WLNK5		;LINK TO THE WRITE BUFFER
1462	006326	104415		BSFG		;BACK UP TO 3RD FILE
1463	006330	104415		BSFG		
1464	006332	104415		BSFG		
1465	006334	104414	016776	READ	,RLNK3	;AND READ IT
1466	006340	104423		BLKCMP		;COMPARE THE READ AND WRITE BUFFERS
1467	006342	016776		RLNK3		;LINK TO THE READ BUFFER
1468	006344	017032		WLNK3		;LINK TO THE WRITE BUFFER
1469	006346	104415		BSFG		;BACK UP TO 1ST FILE
1470	006350	104415		BSFG		
1471	006352	104415		BSFG		
1472	006354	104414	016766	READ	,RLNK1	;AND READ IT
1473	006360	104423		BLKCMP		;COMPARE THE READ AND WRITE BUFFERS
1474	006362	016766		RLNK1		;LINK TO THE READ BUFFER
1475	006364	017022		WLNK1		;LINK TO THE WRITE BUFFER


```

1476 .SBTTL END OF PASS ROUTINE
1477
1478 *****
1479 **INCREMENT THE PASS NUMBER ($PASS)
1480 **TYPE "END PASS"
1481 **IF THERES A MONITOR GO TO IT
1482 **IF THERE ISN'T JUMP TO START
1483 **IF IT IS DESIRED TO HAVE A BELL INDICATE THE "END OF PASS" LOCATION
1484 **$SENDMG CAN BE CHANGED TO 7.
1485
1486 $EOP:
1487 SCOPE
1488 CLR $STNM ;; ZERO THE TEST NUMBER
1489 CLR $TIMES ;; ZERO THE NUMBER OF ITERATIONS
1490 INC $PASS ;; INCREMENT THE PASS NUMBER
1491 BIC #100000,$PASS ;; DON'T ALLOW A NEG. NUMBER
1492 DEC (PC)+ ;; LOOP?
1493 $EOPCT: .WORD 1
1494 BGT $DOAGN ;; YES
1495 MOV (PC)+,$(PC)+ ;; RESTORE COUNTER
1496 $ENDCT: .WORD 1
1497 $EOPCT
1498 TYPE $SENDMG ;; TYPE "END PASS"
1499 $GET42: MOV $42,R0 ;; GET MONITOR ADDRESS
1500 BEQ $DOAGN ;; BRANCH IF NO MONITOR
1501 RESET ;; CLEAR THE WORLD
1502 $ENDAD: JSR PC,(R0) ;; GO TO MONITOR
1503 NOP ;; SAVE ROOM
1504 NOP ;; FOR
1505 NOP ;; ACT11
1506 $DOAGN:
1507 JMP $(PC)+ ;; RETURN
1508 $RTNAD: .WORD START
1509 $ENULL: .BYTE -1,-1,0 ;; NULL CHARACTER STRING
1510 $ENDMG: .ASCIZ <15><12>/END PASS/
1511
006366 000004 172506
006366 005067 172570
006370 005067 172474
006374 005267 100000 172466
006400 042767
006404 005327
006412 000001
006414 003015
006416 012737
006420 000001
006422 006414
006424 104401 006461
006426 013700 000042
006432 001405
006436 000005
006440 004710
006442 000240
006444 000240
006446 000240
006450 000240
006452 000137
006452 002326
006454 377 377 000
006456 015 042412 042116
006461 050040 051501 000123

```



```

1512      ;*****
1513      ;
1514      ;SBTTL          ROUTINE TO EXAMINE DRIVE(S) FOR AVAILABILITY
1515
1516      ;CALL:
1517      ;               MOV      #DRVKEY,RO
1518      ;               JSR      PC,3#EXAM          ;R1 IS DESTROYED
1519      ;               NORMAL RETURN
1520      ;               ERROR RETURN
1521
1522      006474 013701 001220      EXAM:  MOV      3#TACSL,R1          ;PICKUP THE "CONTROL & STATUS" REG. ADR.
1523      006500 005011              CLR      (R1)          ;DRIVE="A", FUNCTION="WFG"
1524      006502 122710 000101      CMPB     #'A,(R0)          ;EXAMINE DRIVE "A"?
1525      006506 001402              BEQ      1$              ;BR IF YES
1526      006510 052711 000400      BIS      #UNIT,(R1)          ;SELECT DRIVE "B"
1527      006514 032711 000040      1$:  BIT      #READY,(R1)      ;WAIT ON READY
1528      006520 001775              BEQ      1$
1529      006522 005711              TST      (R1)          ;ANY ERROR?
1530      006524 100024              BPL      4$              ;BR IF NO
1531      006526 032711 001000      BIT      #OFFLINE,(R1)      ;ERROR DUE TO "OFF LINE"?
1532      006532 001017              BNE      3$              ;BR IF YES
1533      006534 032711 010000      BIT      #WRTLOCK,(R1)      ;ERROR DUE TO "WRITE LOCK"?
1534      006540 001411              BEQ      2$              ;BR IF NO
1535      006542 122777 000201 172370      CMPB     #BIT07!BIT00,3$SWR      ;"READONLY" SELECTED? (RD1PAS)
1536      006550 001412              BEQ      4$              ;BR IF YES
1537      006552 122777 000203 172360      CMPB     #BIT07!BIT01!BIT00,3$SWR      ;(RD2PAS)?
1538      006560 001406              BEQ      4$              ;BR IF YES
1539      006562 000403              BR       3$              ;TAKE THE ERROR EXIT
1540      006564 032711 020000      2$:  BIT      #LEADER,(R1)      ;ERROR DUE TO "CLEAR LEADER"?
1541      006570 001002              BNE      4$              ;BR IF YES
1542      006572 062716 000002      3$:  ADD      #2,(SP)          ;TAKE ERROR RETURN
1543      006576 000207      4$:  RTS      PC              ;RETURN

```


.SBTTL SCOPE HANDLER ROUTINE

```
*****
*THIS ROUTINE CONTROLS THE LOOPING OF SUBTESTS. IT WILL INCREMENT
*AND LOAD THE TEST NUMBER($STNM) INTO THE DISPLAY REG.(DISPLAY<7:0>)
*AND LOAD THE ERROR FLAG ($ERFLG) INTO DISPLAY<15:08>
*THE SWITCH OPTIONS PROVIDED BY THIS ROUTINE ARE:
*SW14=1      LOOP ON TEST
*SW11=1      INHIBIT ITERATIONS
*SW09=1      LOOP ON ERROR
*CALL
*           SCOPE           ;;SCOPE=IOT
```

SSCOPE:

```
CKSWR           ;;TEST  FOR CHANGE IN SOFT-SWR
TSTB            ;;TTY FLAG UP?
BEQ             BR IF NO
RDCHR           GO GET ONE CHARACTER FROM TTY
CMPB            IS IT A CONTROL/C ?
BNE             BR IF NO
SETLOOP        GO GET "TEST" AND "SCOPE LOOP" ADDRESSES
100$: TST      (PC)+
LOOPKEY:        0
BNE            $OVER      BR IF YES
1$: BIT        #BIT14,$SWR ;;LOOP ON PRESENT TEST?
BNE            $OVER      YES IF SW14=1
*****START OF CODE FOR THE XOR TESTER*****
$XTSTR: BR      6$
MOV            3$ERRVEC, -(SP)
MOV            5$, 3$ERRVEC
TST            3$177060
MOV            (SP)+, 3$ERRVEC
BR             $SVLAD
5$: CMP        (SP)+, (SP)+
MOV            (SP)+, 3$ERRVEC
BR             7$
6$: *****END OF CODE FOR THE XOR TESTER*****
2$: TSTB       $ERFLG      ;;HAS AN ERROR OCCURRED?
BEQ            3$          BR IF NO
CMPB           $ERMAX, $ERFLG
BHI            3$          MAX. ERRORS FOR THIS TEST OCCURRED?
BIT            #BIT09, $SWR
BEQ            4$          BR IF NO
MOV            $LPERR, $LPADR
BR             $OVER      BR IF NO
4$: CLRB       $ERFLG      ;;SET LOOP ADDRESS TO LAST SCOPE
CLR            $TIMES
BR             1$          ;;ZERO THE ERROR FLAG
3$: BIT        #BIT11, $SWR ;;CLEAR THE NUMBER OF ITERATIONS TO MAKE
BNE            1$          ;;ESCAPE TO THE NEXT TEST
TST            $PASS      INHIBIT ITERATIONS
BEQ            1$          IF FIRST PASS OF PROGRAM
INC            $ICNT      INCREMENT ITERATION COUNT
CMP            $TIMES, $ICNT
BGE            $OVER      CHECK THE NUMBER OF ITERATIONS MADE
                        ;;BR IF MORE ITERATION REQUIRED
```

```
1544
1545
1546
1547
1548
1549
1550
1551
1552
1553
1554
1555
1556
1557 006600
1558 006600 104406
1559 006602 105777 172336
1560 006606 001405
1561 006610 104407
1562 006612 122627 000003
1563 006616 001001
1564 006620 104424
1565 006622 005727
1566 006624 000000
1567 006626 001105
1568 006630 032777 040000 172302
1569 006636 001101
1570
1571 006640 000416
1572
1573 006642 013746 000004
1574 006646 012737 006666 000004
1575 006654 005737 177060
1576 006660 012637 000004
1577 006664 000453
1578 006666 022626
1579 006670 012637 000004
1580 006674 000413
1581 006676
1582 006676 105767 172201
1583 006702 001421
1584 006704 126767 172205 172171
1585 006712 101015
1586 006714 032777 001000 172216
1587 006722 001404
1588 006724 016767 172160 172154
1589 006732 000443
1590 006734 105067 172143
1591 006740 005067 172224
1592 006744 000415
1593 006746 032777 004000 172164
1594 006754 001011
1595 006756 005767 172116
1596 006762 001406
1597 006764 005267 172114
1598 006770 026767 172174 172106
1599 006776 002021
```

L04

TA11 MOTION TEST MAINDEC-11-DZTAD-C MACY11 27(1006) 23-SEP-76 10:33 PAGE 34
 DZTADC.NEW 24-MAY-76 00:00 SCOPE HANDLER ROUTINE

SEQ 0050

1600	007000	012767	000001	172076	1S:	MOV	#1,\$ICNT	::REINITIALIZE THE ITERATION COUNTER
1601	007006	016767	172204	172154		MOV	\$MXCNT,\$TIMES	::SET NUMBER OF ITERATIONS TO DO
1602	007014	105267	172062		\$SVLAD:	INCB	\$TSTNM	::COUNT TEST NUMBERS
1603	007020	011667	172062			MOV	(SP),\$LPADR	::SAVE SCOPE LOOP ADDRESS
1604	007024	011667	172060			MOV	(SP),\$LPERR	::SAVE ERROR LOOP ADDRESS
1605	007030	005067	172136			CLR	\$ESCAPE	::CLEAR THE ESCAPE FROM ERROR ADDRESS
1606	007034	112767	000001	172053		MOVB	#1,\$ERMAX	::ONLY ALLOW ONE(1) ERROR ON NEXT TEST
1607	007042	016777	172034	172072	\$OVER:	MOV	\$TSTNM,\$DISPLAY	::DISPLAY TEST NUMBER
1608	007050	016716	172032			MOV	\$LPADR,(SP)	::FUDGE RETURN ADDRESS
1609	007054	000002				RTI		::FIXES PS

.SBTTL ERROR HANDLER ROUTINE

```

*****
*THIS ROUTINE WILL INCREMENT THE ERROR FLAG AND THE ERROR COUNT,
*SAVE THE ERROR ITEM NUMBER AND THE ADDRESS OF THE ERROR CALL
*AND GO TO TYPERR ON ERROR
*THE SWITCH OPTIONS PROVIDED BY THIS ROUTINE ARE:
*SW15=1      HALT ON ERROR
*SW13=1      INHIBIT ERROR TYPEOUTS
*SW10=1      BELL ON ERROR
*SW09=1      LOOP ON ERROR
*CALL
*          ERROR   N          ;;ERROR=EMT AND N=ERROR ITEM NUMBER

```

```

$ERROR:  CKSWR          ;;TEST FOR CHANGE IN SOFT-SWR
          MOV          @TACS,@#$REG0      ;SAVE THE TAIL STATUS
          MOV          @TADB,@#$REG1      ;SAVE THE DATA BUFFER
          TSTB         @TACS              ;IF "XFER REQ" IS UP KNOCK IT DOWN
          BPL          100$
          MOVB         #ILBS,@TACS
          BR           101$
100$:    BIT           #READY,@TACS        ;IS READY SET?
          BNE          101$              ;BR IF YES
          RESET
          MOV          @#$REG0,@TACS      ;REGAIN CONTROL
          CMP          (SP)+,(SP)+       ;RESTORE THE TACS
          TST          ASKKEY            ;POP THE STACK
          BGE          102$              ;MANUAL LOOPING?
          MOVB         (SP), $ESCAPE     ;BR IF NO
          MOV          -4(SP), (SP)      ;SETUP FOR MANUAL ESCAPE
          INCB         $ERFLG            ;GET PC+2 OF ERROR
          BEQ          7$                ;SET THE ERROR FLAG
          MOV          $TSTNM,@DISPLAY   ;DON'T LET THE FLAG GO TO ZERO
          BIT          #BIT10,@SWR      ;DISPLAY TEST NUMBER AND ERROR FLAG
          BEQ          1$                ;BELL ON ERROR?
          TYPE         $BELL             ;NO - SKIP
          INC          $ERTTL            ;RING BELL
          MOV          (SP), $ERRPC      ;COUNT THE NUMBER OF ERRORS
          SUB          #2,$ERRPC         ;GET ADDRESS OF ERROR INSTRUCTION
          MOVB         @ $ERRPC,$ITEMB   ;STRIP AND SAVE THE ERROR ITEM CODE
          BIT          #BIT13,@SWR      ;SKIP TYPEOUT IF SET
          BNE          20$              ;SKIP TYPEOUTS
          JSR          PC,TYPERR        ;GO TO USER ERROR ROUTINE
          TYPE         , $CRLF
20$:     TST          @SWR
2$:      BPL          3$
          HALT
          CKSWR
3$:      BIT          #BIT09,@SWR
          BEQ          4$
          MOV          $LPERR,(SP)
          TST          $ESCAPE
          BEQ          5$
          MOV          $ESCAPE,(SP)
          ;;HALT ON ERROR
          ;;SKIP IF CONTINUE
          ;;HALT ON ERROR!
          ;;TEST FOR CHANGE IN SOFT-SWR
          ;;LOOP ON ERROR SWITCH SET?
          ;;BR IF NO
          ;;FUDGE RETURN FOR LOOPING
          ;;CHECK FOR AN ESCAPE ADDRESS
          ;;BR IF NONE
          ;;FUDGE RETURN ADDRESS FOR ESCAPE

```

```

1610
1611
1612
1613
1614
1615
1616
1617
1618
1619
1619
1620
1621
1622
1623
1624 007056
1625 007056 104406
1626 007060 011437 001162
1627 007064 011537 001164
1628 007070 105714
1629 007072 100003
1630 007074 112714 000020
1631 007100 000406
1632 007102 032714 000040
1633 007106 001003
1634 007110 000005
1635 007112 013714 001162
1636 007116 022626
1637 007120 005767 172122
1638 007124 002002
1639 007126 011667 172040
1640 007132 016616 177774
1641 007136 105267 171741
1642 007142 001775
1643 007144 016777 171732 171770
1644 007152 032777 002000 171760
1645 007160 001402
1646 007162 104401 001174
1647 007166 005267 171720
1648 007172 011667 171720
1649 007176 162767 000002 171712
1650 007204 117767 171706 171702
1651 007212 032777 020000 171720
1652 007220 001004
1653 007222 004767 000060
1654 007226 104401 001201
1655 007232
1656 007232 005777 171702
1657 007236 100002
1658 007240 000000
1659 007242 104406
1660 007244 032777 001000 171666
1661 007252 001402
1662 007254 016716 171630
1663 007260 005767 171706
1664 007264 001402
1665 007266 016716 171700

```

N04

TA11 MOTION TEST MAINDEC-11-DZTAD-C MACY11 27(1006) 23-SEP-76 10:33 PAGE 36
 DZTADC.NEW 24-MAY-76 00:00 ERROR HANDLER ROUTINE

SEQ 0052

1666	007272			5\$:			
1667	007272	022737	006442	000042	CMP	#SENDAD, 2#42	::ACT-11 AUTO-ACCEPT?
1668	007300	001001			BNE	6\$	::BRANCH IF NO
1669	007302	000000			HALT		::YES
1670	007304			6\$:			
1671	007304	000002			RTI		::RETURN


```

1672 007306 104401 001201
1673 007312 104401
1674 007314 000000
1675 007316 104401 001201
1676 007322 162737 000002 001204
1677 007330 010046
1678 007332 113700 001114
1679 007336 005300
1680 007340 006300
1681 007342 006300
1682 007344 006300
1683 007346 062700 001270
1684 007352 012067 000002
1685 007356 104401
1686 007360 000000
1687 007362 012067 000004
1688 007366 001402
1689 007370 104401
1690 007372 000000
1691 007374 012000
1692 007376 001004
1693 007400 012600
1694 007402 104401 001201
1695 007406 000207
1696 007410
1697 007410 013046
1700 007412 104402
1701 007414 005710
1702 007416 001770
1703 007420 104401 007426
1704 007424 000771
1705 007426 020040 000
1706 007432
1707

```

 THIS ROUTINE WILL TYPEOUT THE ERROR MESSAGES
 TYPERR: TYPE ,SCLF :TYPE A CARRIAGE RETURN & LINE FEED
 :TYPE THE FUNCTION BEING PERFORMED
 MFUNC: 0 :MESSAGE POINTER GOES HERE
 :TYPE ,SCLF
 SUB #2,3 SAVPC :ADJUST THE MAIN FLOW PC
 MOV RO, -(SP) :SAVE RO
 MOVB 3,4 ITEMB,RO :PICKUP THE ITEM INDEX
 DEC RO :ADJUST THE INDEX
 ASL RO :SO IT WILL WORK FOR
 ASL RO :THE ERROR TABLE
 ASL RO
 ADD #ERRTB,RO :FORM THE TABLE POINTER
 MOV (RO)+,1\$:PICKUP "ERROR MESSAGE" POINTER
 TYPE :TYPE "ERROR MESSAGE"
 1\$: 0 : "ERROR MESSAGE POINTER" GOES HERE
 MOV (RO)+,2\$:PICKUP "DATA HEADER" POINTER
 BEQ 3\$:IF "0" DON'T TYPE
 TYPE :TYPE "DATA HEADER"
 2\$: 0 : "DATA HEADER" POINTER GOES HERE
 3\$: MOV (RO)+,RO :PICKUP "DATA POINTER"
 BNE 5\$:IF THERE IS DATA TO TYPE GO DO IT
 4\$: MOV (SP)+,RO :RESTORE RO
 TYPE ,SCLF :TYPE A CARRAGE RETURN&LINE FEED
 RTS PC :RETURN
 5\$: MOV 3(RO)+, -(SP) :SAVE 3(RO)+ FOR TYPEOUT
 :TYPE DATA
 :GO TYPE--OCTAL ASCII(ALL DIGITS)
 TST (RO) :TERMINATOR?
 BEQ 4\$:BR IF YES
 TYPE 2\$:TYPE 2 SPACES
 BR 5\$:LOOP
 6\$: .ASCIZ / :ASCII STRING OF 2 SPACES
 .EVEN


```

1708      ::*****
1709
1710      .SBTTL      ROUTINE TO ASK THE OPERATOR WHAT DRIVE(S) TO TEST
1711
1712      :CALL
1713      :          JSR      PC,2*ASKDRV
1714      :          RETURN
1715      :          ;NOTE: R0 AND R1 ARE DESTROYED
1716
1716      007432      104401      015146      ASKDRV: TYPE      MSGDRV      ;<CRLF>"DRIVE(S)? "
1717      007436      005067      171576      CLR      DRVKEY
1718      007442      104410      RDLIN
1719      007444      012600      MOV      (SP)+,R0      ;GO GET A DRIVE
1720      007446      105710      TSTB      3R0      ;SETUP TO CHECK FOR VALID DRIVE(S)
1721      007450      001425      BEQ      NOTLGL      ;WAS A DRIVE SELECTED?
1722      007452      012701      001240      MOV      2*DRVKEY,R1      ;BR IF NO
1723      007456      122710      000101      LOOP:  CMPB      #'A,3R0      ;WAS DRIVE "A" SELECTED?
1724      007462      001002      BNE      NOTA      ;BR IF NO
1725      007464      112021      MOVB      (R0)+,(R1)+      ;SET KEY FOR DRIVE "A"
1726      007466      000411      BR      NEXT
1727      007470      122710      000102      NOTA:  CMPB      #'B,3R0      ;WAS DRIVE "B" SELECTED?
1728      007474      001002      BNE      NOTB      ;BR IF NO
1729      007476      112021      MOVB      (R0)+,(R1)+      ;SET KEY FOR DRIVE "B"
1730      007500      000404      BR      NEXT
1731      007502      122710      000054      NOTB:  CMPB      #54,3R0      ;WAS A COMMA TYPED?
1732      007506      001006      BNE      NOTLGL      ;BR IF NO
1733      007510      105720      TSTB      (R0)+      ;DUMP THE COMMA
1734      007512      105710      NEXT:  TSTB      3R0      ;TERMINATOR?
1735      007514      001406      BEQ      EXIT      ;BR IF YES
1736      007516      022701      001242      CMP      2*DRVKEY+2,R1      ;TWO DRIVES SELECTED?
1737      007522      101355      BHI      LOOP      ;BR IF NO
1738      007524      104401      001200      NOTLGL: TYPE      $QUES      ;ILLEGAL INPUT DETECTED
1739      007530      000740      BR      ASKDRV      ;GO TRY AGAIN
1740      007532      005767      171502      EXIT:  TST      DRVKEY      ;ANY DRIVE SELECTED?
1741      007536      001772      BEQ      NOTLGL      ;BR IF NO
1742      007540      000207      RTS      PC
1743
1744      ::*****
1745
1746      :CALL
1747      :          JSR      PC,2*ASKADR
1748
1748      007542      010046      ASKADR: MOV      R0,-(SP)      ;SAVE R0
1749      007544      104401      015162      1$:  TYPE      ,MSGASK      ;"TACS?"
1750      007550      104411      RDOCT
1751      007552      012600      MOV      (SP)+,R0      ;GET VALUE
1752      007554      001423      BEQ      3$      ;PICK UP THE OCTAL NUMBER
1753      007556      020027      160000      CMP      R0,#160000      ;IF "0" USE OLD VALUES
1754      007562      103770      BLO      1$      ;MAKE SURE IT IS A BUS ADDRESS
1755      007564      010037      001220      MOV      R0,2*TACSL      ;SAVE TOE TACS
1756      007570      062700      000002      ADD      #2,R0      ;STEP TO TADB ADDRESS
1757      007574      010037      001224      MOV      R0,2*TADBL      ;AND SAVE IT
1758      007600      013737      001220      001222      MOV      2*TACSL,2*TACSH      ;SET UP TACS UPPER
1759      007606      005237      001222      INC      2*TACSH      ;BYTE POINTER
1760
1761      007612      013737      001224      001226      MOV      2*TADBL,2*TADBH      ;SET UP TADB UPPER
1762      007620      005237      001226      INC      2*TADBH      ;BYTE POINTER
1763      007624      104401      015172      3$:  TYPE      ,MSGVEC      ;"VECTOR?"

```


TA11 MOTION TEST
DZTADC.NEW

24-MAY-76 00:00

MAINDEC-11-DZTAD-C

MACY11 27(1006) 23-SEP-76 10:33 PAGE 39
ROUTINE TO INPUT CSR,DBR, AND VECTOR ADDRESS AND PRIORITY

SEQ 0055

1764	007630	104411		RDOCT		
1765	007632	012600		MOV	(SP)+,RO	
1766	007634	001411		BEQ	5\$	
1767	007636	020027	001000	CMP	RO,#1000	;MAKE SURE ADDRESS IS IN VECTOR AREA
1768	007642	103370		BHIS	3\$	
1769	007644	010037	001230	MOV	RO,3#TAVEC	;SAVE AS VECTOR ADDRESS
1770	007650	062700	000002	ADD	#2,RO	
1771	007654	010037	001232	MOV	RO,3#TAVEC+2	
1772	007660	104401	015202	5\$: TYPE	,MSGPRI	;ASK FOR PRIORITY
1773	007664	104411		RDOCT		
1774	007666	012600		MOV	(SP)+,RO	
1775	007670	001413		BEQ	6\$	;IF "0" USE OLD VALUE
1776	007672	020027	000007	CMP	RO,#7	;MAKE SURE ITS VALID
1777	007676	101370		BHI	5\$	
1778	007700	000300		SWAB	RO	;PUT INTO HIGH BYTE
1779	007702	006200		ASR	RO	;AND SHIFT
1780	007704	006200		ASR	RO	;INTO PROPER
1781	007706	006200		ASR	RO	;POSITION
1782	007710	042700	177437	BIC	#1C(340),RO	;SAVE ONLY PRIORITY BITS
1783	007714	010037	001234	MOV	RO,3#TAPRIO	;STORE IT AWAY
1784	007720	104401	015214	6\$: TYPE	MTACS	TACS="
1785	007724	016746	171270	MOV	TACSL,-(SP)	::SAVE TACSL FOR TYPEOUT
1786	007730	104402		TYP0C		::GO TYPE--OCTAL ASCII(ALL DIGITS)
1787	007732	104401	015223	TYPE	MTADB	TADB="
1788	007736	016746	171262	MOV	TADBL,-(SP)	::SAVE TADBL FOR TYPEOUT
1789	007742	104402		TYP0C		::GO TYPE--OCTAL ASCII(ALL DIGITS)
1790	007744	104401	015232	TYPE	MTAVEC	VECTOR="
1791	007750	016746	171254	MOV	TAVEC,-(SP)	::SAVE TAVEC FOR TYPEOUT
1792	007754	104402		TYP0C		::GO TYPE--OCTAL ASCII(ALL DIGITS)
1793	007756	104401	015243	TYPE	MTAPRI	PRIORITY="
1794	007762	016746	171246	MOV	TAPRIO,-(SP)	::SAVE TAPRIO FOR TYPEOUT
1795	007766	104402		TYP0C		::GO TYPE--OCTAL ASCII(ALL DIGITS)
1796	007770	104401	015256	TYPE	,MSGOK	"OK?"
1797	007774	104407		RDCHR		;GO READ ONE CHARACTER
1798	007776	012600		MOV	(SP)+,RO	;GET IT
1799	010000	022700	000015	CMP	#15,RO	;IS IT "CR"?
1800	010004	001406		BEQ	7\$	;BRANCH IF YES
1801	010006	022700	000131	CMP	#'Y,RO	;IS IT "Y"?
1802	010012	001403		BEQ	7\$	;IT WAS
1803	010014	104401	001200	TYPE	,SQUES	TYPE "?"
1804	010020	000651		BR	1\$	;AND LET HIM CORRECT THEM
1805	010022	104401	015264	7\$: TYPE	,MYES	;TYPE OUT "YES"
1806	010026	012600		MOV	(SP)+,RO	;RESTORE RO
1807	010030	000207		RTS	PC	;AND RETURN

THIS ROUTINE WILL CHANGE AN .ASCII STRING TO AN OCTAL NUMBER

REGISTER R1 MUST BE INITIALIZED BEFORE ENTERING

CALL:

JSR R3,3#A20CT
POINTER

ERROR RETURN

;CALL THE ROUTINE
;ADDRESS OF THE FIRST CHARACTER IN THE STRING
;STRING MUST BE TERMINATED BY A BYTE OF ALL 0'S
;RETURN HERE IF ILLEGAL CHARACTER

TAIL MOTION TEST MAINDEC-11-DZTAD-C
DZTADC.NEW 24-MAY-76 00:00

SEQ 0056

ADDRESS	HEX	ASCII	OPERATION	COMMENT
1820			:	:OR STRING IS TO LONG
1821			:	:OCTAL NUMBER IS IN R1
1822				
1823	010032	010046	A20CT: MOV R0, -(SP)	:SAVE R0
1824	010034	012300	MOV (R3)+, R0	:PICK UP THE POINTER
1825	010036	012767	MOV #6, 2\$	:MAX. NUMBER OF CHAR.'S ALLOWED
1826	010044	105710	1\$: TSTB (R0)	:TERMINATOR?
1827	010046	001420	BEQ 3\$	:BR IF YES
1828	010050	005327	DEC (PC)+	:COUNT THIS CHARACTER
1829	010052	000000	2\$: 0	
1830	010054	002416	BLT 4\$	:BR IF TO MANY CHARACTERS IN THE STRING
1831	010056	121027	000060 CMPB (R0), #'0	:CHECK THIS CHAR. IS BETWEEN
1832	010062	002413	BLT 4\$	:0 AND 7
1833	010064	121027	000067 CMPB (R0), #'7	
1834	010070	003010	BGT 4\$	
1835	010072	142710	177770 BICB #'C7, (R0)	:STRIP AWAY THE ASCII CODE
1836	010076	006301	ASL R1	:POSITION THE FOR THIS CHAR
1837	010100	006301	ASL R1	
1838	010102	006301	ASL R1	
1839	010104	152001	BISB (R0)+, R1	:SET IN THIS DIGIT
1840	010106	000756	BR 1\$	:GO GET THE NEXT CHARACTER
1841	010110	005723	3\$: TST (R3)+	:INCREMENT FOR NORMAL RETURN
1842	010112	012600	4\$: MOV (SP)+, R0	:RESTORE R0
1843	010114	000203	RTS R3	:RETURN TO USER

F05

TA11 MOTION TEST MAINDEC-11-DZTAD-C
DZTADC.NEW 24-MAY-76 00:00

MACY11 27(1006) 23-SEP-76 10:33 PAGE 41
ROUTINE TO GET "TEST" AND "SCOPE LOOP" ADDRESSES FROM THE TTY

SEQ 0057

```

1844
1845
1846
1847 010116 104401 015326
1848 010116 104410
1849 010122 104410
1850 010124 012667 000022
1851 010130 105777 000016
1852 010134 001442
1853 010136 022626
1854 010140 010167 000110
1855 010144 005001
1856 010146 004337 010032
1857 010152 000000
1858 010154 000760
1859 010156 005721
1860 010160 010116
1861 010162 104401 015341
1862 010166 104410
1863 010170 012667 000014
1864 010174 105777 000010
1865 010200 001410
1866 010202 005001
1867 010204 004337 010032
1868 010210 000000
1869 010212 000763
1870 010214 010137 001106
1871 010220 000402
1872 010222 011637 001106
1873 010226 011637 001110
1874 010232 012737 177777 006624
1875 010240 000402
1876 010242 005037 006624
1877 010246 016701 000002
1878 010252 000002
1879 010254 000000

;*****
;THIS ROUTINE WILL INPUT THE ADDRESSES OF "TEST" AND "SCOPE LOOP" FROM THE TTY

T.SETLOOP:
    TYPE      ,MTEST
    RDLIN
    MOV      (SP)+,11$
    TSTB     011$
    BEQ      3$
    CMP      (SP)+,(SP)+
    MOV      R1,SAVR1
    CLR      R1
    JSR      R3,0#A20CT
11$:    D
    BR       T.SETLOOP
    TST      (R1)+
    MOV      R1,(SP)
5$:    TYPE      ,MLOOP
    RDLIN
    MOV      (SP)+,12$
    TSTB     012$
    BEQ      1$
    CLR      R1
    JSR      R3,0#A20CT
12$:    D
    BR       5$
    MOV      R1,0#$LPADR
    BR       2$
1$:    MOV      (SP),0#$LPADR
2$:    MOV      (SP),0#$LPERR
    MOV      #-1,0#LOOPKEY
    BR       4$
3$:    CLR      0#LOOPKEY
4$:    MOV      SAVR1,R1
    RTI
SAVR1:  D

;TYPE "TEST PC?"
;GET AN ASCII STRING
;GET FIRST ADDRESS OF INPUT BUFFER
;FIRST CHAR. A "CR" ?
;BR IF YES---GO CLEAR THE LOOP KEY
;POP POP
;SAVE R1
;SET PARTIAL TO ZERO
;GO CHANGE ASCII TO OCTAL
;POINTER TO STRING
;ERROR RETURN--GO TRY AGAIN
;JUST INCREMENTING
;PUT "TEST PC"; ON THE STACK
;TYPE "LOOP PC?"
;GET A STRING OF ASCII
;GET FIRST ADDRESS OF INPUT BUFFER
;JUST A "CR" ?
;BR IF YES
;CLEAR PARTIAL
;CHANGE TO OCTAL

;SAVE SCOPE LOOP ADDRESS
;GO SET LOOP KEY
;SAVE SCOPE LOOP ADDRESS
;ERROR LOOP SAME AS THE TEST ADDRESS
;SET THE LOOP KEY

;CLEAR THE LOOP KEY
;RESTORE R1
;BYE BYE
;HOLD R1 HERE

```

G05

TA11 MOTION TEST
DZTADC.NEW 24-MAY-76 00:00

MAINDEC-11-DZTAD-C

MACY11 27(1006) 23-SEP-76 10:33 PAGE 42
ROUTINE TO SELECT THE DRIVE TO BE TESTED

SEQ 0058

```

1880
1881
1882
1883
1884 010256 032777 000200 170654 T.SELDRV:
1885 010256 001041 BIT #SW07, @SWR ;LOCK ON CURRENT DRIVE?
1886 010264 010146 BNE 5$ ;BR IF YES
1887 010266 010146 MOV R1, -(SP) ;SAVE R1 ON THE STACK
1888 010270 016701 170742 MOV DRIVE, R1 ;GET THE DRIVE
1889 010274 111137 001250 MOV B (R1), @#CURDRV ;SAVE CURRENT DRIVE
1890 010300 121127 000101 CMP B (R1), #'A ;IS IT DRIVE "A"?
1891 010304 001406 BEQ 1$ ;BR IF YES
1892 010306 052714 000400 BIS #UNIT, @TACS ;SELECT DRIVE "B"
1893 010312 012737 001262 001256 MOV #RWDB, @#RWDFLG ;SET REWIND POINTER TO "B"
1894 010320 000405 BR 2$
1895 010322 042714 000400 1$: BIC #UNIT, @TACS ;SELECT DRIVE "A"
1896 010326 012737 001260 001256 MOV #RWDA, @#RWDFLG ;SET REWIND POINTER TO "A"
1897 010334 005201 2$: INC R1 ;INCREMENT TO THE NEXT DRIVE
1898 010336 105711 TST B (R1) ;OUT OF DRIVES TO TEST?
1899 010340 001002 BNE 3$ ;BR IF NO
1900 010342 012701 001240 MOV #DRVKEY, R1 ;RESET TO THE FIRST DRIVE
1901 010346 010167 170664 3$: MOV R1, DRIVE ;SAVE FOR NEXT TIME
1902 010352 032777 001000 170560 BIT #SW09, @SWR ;IS SWITCH 9=1?
1903 010360 001002 BNE 4$ ;BR IF YES
1904 010362 105037 001103 CLRB @#SERFLG ;CLEAR THE ERROR INDICATOR
1905 010366 012601 4$: MOV (SP)+, R1 ;RESTORE R1
1906 010370 000002 5$: RTI

```


H05

TA11 MOTION TEST
DZTADC.NEW 24-MAY-76 00:00

MAINDEC-11-DZTAD-C

MACY11 27(1006) 23-SEP-76 10:33 PAGE 43
ROUTINE TO COMPARE THE READ AND WRITE BUFFERS

SEQ 0059

```

1907
1908
1909 010372
1910 010372 011600
1911 010374 010037 001204
1912 010400 012737 015531 007314
1913 010406 012003
1914 010410 012301
1915 010412 011303
1916 010414 013002
1917 010416 005067 000034
1918 010422 010016
1919 010424 005037 001206
1920 010430 032777 000010 170502
1921 010436 001401
1922 010440 000002
1923 010442 005303
1924 010444 002001
1925 010446 000002
1926 010450 005237 001206
1927 010454 122162 000000
1928 010460 001006
1929 010462 005267 177770
1930 010466 042767 177774 177762
1931 010474 000762
1932 010476 114137 001126
1933 010502 066702 177750
1934 010506 111237 001124
1935 010512 104010
1936
1937
1938 010514 011637 001204
1939 010520 012737 015512 007314
1940 010526 004037 012234
1941 010532 000101
1942 010534 005037 177776
1943 010540 004037 011726
1944 010544 104003
1945 010546 104004
1946 010550 004037 012024
1947 010554 104002
1948 010556 104002
1949 010560 000002
1950 010562 104002

::*****
:THIS ROUTINE WILL COMPARE THE READ AND WRITE BUFFERS
:BLKCMP:
MOV (SP),R0 ;GET THE PC OF THE CALL
MOV R0,#SAVPC ;SAVE THE PC OF THE CALL
MOV #MBUFCK,#MFUNC ;POINTER FOR .ASCII MESSAGE
MOV (R0)+,R3 ;PICKUP THE READ LINK ADDRESS
MOV (R3)+,R1 ;PICKUP THE READ BUFFER ADDRESS
MOV (R3),R3 ;PICKUP THE READ BUFFER SIZE
MOV #R0+,R2 ;PICKUP THE WRITE BUFFER ADDRESS
CLR 2$+2 ;ZERO THE OFFSET
MOV R0,(SP) ;SET THE RETURN
CLR #BYTNUM ;ZERO THE BYTE NUMBER
BIT #SW03,#SWR ;COMPARE DATA BUFFERS?
BEQ 1$ ;BR IF YES
RTI ;NO---GO BACK
1$: DEC R3 ;CHECK FOR THE TERMINATION
BGE 4$ ;BR IF MORE BYTES TO COMPARE
RTI ;RETURN
4$: INC #BYTNUM ;COUNT THIS BYTE
2$: CMPB (R1)+,0(R2) ;COMPARE READ AND WRITE BUFFER
BNE 3$ ;BR IF NOT EQUAL
INC 2$+2 ;INCREMENT WRITE BUFFER INDEX
BIC #C3,2$+2 ;DON'T LET IT GET BIGGER THAN 3
BR 1$
3$: MOVB -(R1),#SBDDAT ;GET THE RECEIVED BYTE
ADD 2$+2,R2 ;FORM ADDRESS
MOVB (R2),#SGDDAT ;GET THE EXPECTED BYTE
ERROR 10 ;RECEIVED AND EXPECTED DATA NOT EQUAL
::*****
:THIS ROUTINE WILL WRITE A FILE GAP ON THE TAPE
:WFG: MOV (SP),#SAVPC ;SAVE THE PC OF THE CALL
MOV #MXWFG,#MFUNC ;SET MESSAGE POINTER FOR "WFG"
1$: JSR R0,#DO.CMD ;GO START A "WFG"
XWFG!INT.EN!GO
CLR #PS ;ALLOW INTERRUPTS
JSR R0,#WAITFLAG ;GO WAIT FOR A FLAG
ERROR 3 ;NO FLAGS OCCURRED
ERROR 4 ;NO INTERRUPT
JSR R0,#FLAGS ;NORMAL RETURN---FIND OUT WHAT FLAG
ERROR 2 ;RETURN HERE IF "XFER REQ" AND "READY" =0
ERROR 2 ;RETURN HERE IF "XFER REQ" =1
RTI ;RETURN HERE IF "READY"=1 AND "ERROR" =0
ERROR 2 ;"READY"=1 AND "ERROR"=1

```

TA11 MOTION TEST MAINDEC-11-DZTAD-C
DZTADC.NEW 24-MAY-76 00:00

MACY11 27(1006) 23-SEP-76 10:33 PAGE 44
ROUTINE TO "WRITE" A RECORD ON THE TAPE

SEQ 0060

```

1951
1952
1953 010564 011637 001204
1954 010570 012737 015363 007314
1955 010576 017602 000000
1956 010602 062716 000002
1957 010606 012201
1958 010610 005037 012232
1959 010614 005067 000066
1960 010620 012202
1961 010622 003001
1962 010624 104011
1963 010626 004037 012234 2$:
1964 010632 000103 XWRITE!INT.EN!GO
1965 010634 005037 177776 3$:
1966 010640 004037 011726 JSR RO,0#WAITFLAG
1967 010644 104003 ERROR 3
1968 010646 104004 ERROR 4
1969 010650 005037 001254 CLR 0#WAITKEY
1970 010654 012737 000340 177776 MOV #340,0#PS
1971 010662 004037 012024 JSR RO,0#FLAGS
1972 010666 104002 ERROR 2
1973 010670 000402 BR 4$
1974 010672 104005 ERROR 5
1975 010674 104002 ERROR 2
1976 010676 005302 4$: DEC R2
1977 010700 002417 BLT 5$
1978 010702 005000 CLR RO
1979 010704 156100 000000 6$: BISB 0(R1),RO
1980 010710 110015 MOVB RO,0#ADB
1981 010712 004737 012112 JSR PC,0#DO.CRC
1982 010716 005267 177764 INC 6$+2
1983 010722 042767 177774 177756 BIC #1C3,6$+2
1984 010730 012777 012350 170272 MOV #SERV,0#AVEC
1985 010736 000736 BR 3$
1986 010740 013761 012232 000004 5$: MOV 0#CRC.WD,4(R1)
1987 010746 004037 012234 JSR RO,0#DO.CMD
1988 010752 000122 XWRITE!INT.EN!ILBS
1989 010754 005037 177776 CLR 0#PS
1990 010760 004037 011726 JSR RO,0#WAITFLAG
1991 010764 104003 ERROR 3
1992 010766 104004 ERROR 4
1993 010770 004037 012024 JSR RO,0#FLAGS
1994 010774 104002 ERROR 2
1995 010776 104002 ERROR 2
1996 011000 000002 RTI
1997 011002 104002 ERROR 2

```

THIS ROUTINE WILL WRITE A RECORD ON THE TAPE

↑.WRIT: MOV (SP),0#SAVPC ;SAVE PC OF CALL
MOV #MXWRIT,0#MFUNC ;POINTER FOR .ASCIZ OF "WRITE"
MOV 0(SP),R2 ;GET THE WRITE LINK POINTER
ADD #2,(SP) ;ADJUST THE RETURN
MOV (R2)+,R1 ;GET FIRST ADDRESS OF WRITE BUFFER
CLR 0#CRC.WD ;CLEAR THE CRC WORD
CLR 6\$+2 ;ZERO THE INDEX
MOV (R2)+,R2 ;GET NUMBER OF BYTE TO WRITE
BGT 2\$;CHECK FOR LEGAL NUMBER
ERROR 11 ;BAD NUMBER OF BYTES
JSR RO,0#DO.CMD ;GO START A "WRITE"

3\$: CLR 0#PS ;ALLOW INTERRUPTS
JSR RO,0#WAITFLAG ;WAIT FOR A FLAG
ERROR 3 ;NO FLAGS OCCURRED
ERROR 4 ;INTERRUPT FAILED
CLR 0#WAITKEY ;CLEAR THE WAIT FOR INTERRUPT KEY
MOV #340,0#PS ;LOCK OUT THE WORLD
JSR RO,0#FLAGS ;WAIT FLAG OCCURRED?
ERROR 2 ;"READY"=0 & "XFER REQ" =0
BR 4\$;"XFER REQ"=1
ERROR 5 ;"READY"=1 & "ERROR"=0
ERROR 2 ;"READY"=1 & "ERROR"=1
DEC R2 ;NEED TO WRITE A BYTE ON TAPE
BLT 5\$;BR IF NO
CLR RO

6\$: BISB 0(R1),RO ;PICKUP BYTE FOR TRANSFER
MOVB RO,0#ADB ;TRANSFER A BYTE TO THE TA11
JSR PC,0#DO.CRC ;CALCULATE CRC FOR THIS BYTE
INC 6\$+2 ;INCREASE THE INDEX
BIC #1C3,6\$+2 ;KEEP IT LESS THAN 4
MOV #SERV,0#AVEC ;SETUP TO SERVICE THE INTERRUPT
BR 3\$;LOOP
MOV 0#CRC.WD,4(R1) ;SAVE THE CRC WORD
JSR RO,0#DO.CMD ;START A "ILBS"

5\$: CLR 0#PS ;ALLOW INTERRUPTS
JSR RO,0#WAITFLAG ;WAIT FOR A FLAG
ERROR 3 ;NO FLAG
ERROR 4 ;NO INTERRUPT
JSR RO,0#FLAGS ;WHAT FLAG IS SET?
ERROR 2 ;NONE ("XFER REQ"=0 & "READY"=0)
ERROR 2 ;"XFER REQ"
RTI ;"READY"
ERROR 2 ;"READY" & "ERROR"

TA11 MOTION TEST
DZTADC.NEW 24-MAY-76 00:00

MAINDEC-11-DZTAD-C

MACY11 27(1006) 23-SEP-76 10:33 PAGE 45
ROUTINE TO "READ" A RECORD

SEQ 0061

```

1998
1999
2000 011004 011637 001204
2001 011010 012737 015371 007314
2002 011016 017602 000000
2003 011022 062716 000002
2004 011026 010200
2005 011030 016000 000034
2006 011034 016067 000004 170146
2007 011042 012201
2008 011044 012202
2009 011046 003001
2010 011050 104011
2011 011052 005037 012232
2012 011056 004037 012234
2013 011062 000105
2014 011064 005037 177776
2015 011070 105011
2016 011072 004037 011726
2017 011076 104003
2018 011100 104004
2019 011102 005037 001254
2020 011106 012737 000340 177776
2021 011114 004037 012024
2022 011120 104002
2023 011122 000402
2024 011124 104005
2025 011126 000413
2026 011130 005302
2027 011132 002431
2028 011134 005000
2029 011136 151500
2030 011140 110021
2031 011142 004737 012112
2032 011146 012777 012350 170054
2033 011154 000743
2034 011156 010237 001166
2035 011162 001414
2036 011164 013746 001172
2037 011170 005037 001172
2038 011174 016646 000004
2039 011200 016646 000004
2040 011204 104013
2041 011206 012637 001172
2042 011212 000002
2043 011214 104002
2044 011216 013737 012232 001212
2045 011224 004037 012234
2046 011230 000124
2047 011232 111537 001214
2048 011236 005037 177776
2049 011242 004037 011726
2050 011246 104003
2051 011250 104004
2052 011252 111567 167737
2053 011256 004037 012024

```

THIS ROUTINE WILL READ A RECORD FROM THE TAPE
T.READ: MOV (SP),J#SAVPC ;SAVE THE PC OF THE CALL
MOV #MXREAD,J#MFUNC ;SET POINTER OF .ASCIZ MESSAGE
MOV J(SP),R2 ;GET THE READ LINK POINTER
ADD #2,(SP) ;ADJUST FOR RETURN
MOV R2,R0 ;PICKUP AND SAVE THE "CRC" THAT
MOV WLNK1-RLNK1(R0),R0 ; WAS CALCULATED ON THE WRITE
MOV 4(R0),RCRC0 ; DATA FOR THIS BLOCK
MOV (R2)+,R1 ;GET ADDRESS OF BUFFER
MOV (R2)+,R2 ;GET NUMBER OF BYTES
BGT 2\$;MAKE SURE NUMBER IS LEGAL
ERROR 11 ;ILLEGAL NUMBER OF BYTES
CLR J#CRC.WD ;CLEAR CRC WORD
JSR R0,J#DO.CMD ;GO START A "READ"
XREAD!INT.EN!GO
3\$: CLR J#PS ;ALLOW INTERRUPTS
CLRB (R1) ;SET TERMINATOR IF LAST BYTE
JSR R0,J#WAITFLAG ;WAIT FOR A FLAG
ERROR 3 ;NONE OCCURRED
ERROR 4 ;NO INTERRUPT
CLR J#WAITKEY ;CLEAR THE WAIT ON INTERRUPT KEY
MOV #340,J#PS ;LOCK OUT ALL INTERRUPTS
JSR R0,J#FLAGS ;WHAT FLAG OCCURRED
ERROR 2 ;?UNKNOWN FLAG
BR 4\$;"XFER REQ"
ERROR 5 ;"READY"
BR 8\$;"READY" WITH "ERROR"
4\$: DEC R2 ;READ A BYTE?
BLT 5\$;BR IF NO
CLR R0
BISB JADB,R0 ;READ ONE BYTE
MOVB R0,(R1)+ ;AND PUT IT IN THE BUFFER
JSR PC,J#DO.CRC ;CALCULATE CRC FOR THIS BYTE
MOV #SERV,J#TAVEC ;SETUP TO SERVICE THE INTERRUPT
BR 3\$;LOOP
8\$: MOV R2,J#STMP0 ;GET NUMBER OF BYTES NOT READ
BEQ 9\$;SKIP OUT IF 0
MOV J#\$ESCAPE,-(SP) ;SAVE THE ESCAPE ADDRESS
CLR J#\$ESCAPE ;NO ESCAPE DESIRED
MOV 4(SP),-(SP) ;COPY THE STACK
MOV 4(SP),-(SP)
ERROR 13 ;BLOCK READ SHORT
MOV (SP)+,J#\$ESCAPE ;RESTORE ESCAPE ADDRESS
RTI ;LEAVE
9\$: ERROR 2 ;ERROR OCCURRED DURING READ
5\$: MOV J#CRC.WD,J#RCRC1 ;SAVE THE CALCULATED CRC
JSR R0,J#DO.CMD ;DO AN ILBS
XREAD!INT.EN!ILBS
MOVB JADB,J#RCRC2 ;SAVE 1ST BYTE OF CRC FROM TA11
CLR J#PS ;ALLOW INTERRUPTS
JSR R0,J#WAITFLAG ;WAIT ON FLAG
ERROR 3 ;NONE OCCURRED
ERROR 4 ;INTERRUPT FAILED
MOVB JADB,RCRC2+1 ;SAVE 2ND BYTE OF CRC FROM TA11
JSR R0,J#FLAGS ;CHECK FLAGS

TAII MOTION TEST MAINDEC-11-DZTAD-C
DZTADC.NEW 24-MAY-76 00:00

MACY11 27(1006) 23-SEP-76 10:33 PAGE 46
ROUTINE TO "READ" A RECORD

2054	011262	104002	
2055	011264	104002	
2056	011266	000002	
2057	011270	032714	037000
2058	011274	001401	
2059	011276	104002	
2060	011300	032777	000020
2061	011306	001001	
2062	011310	104012	
2063	011312	000002	
2064			

```

ERROR      2      ;UNKNOWN FLAG
ERROR      2      ;"XFER REQ"
RTI         ;"READY"
BIT         #LEADER!WRTLOCK!FGAP!TIMERR!OFFLINE,!TACS
BEQ         6$     ;BR IF CRC ERROR
ERROR      2      ;ERROR OCCURRED
BIT         #SW04,!SWR      ;IGNORE CRC ERRORS?
BNE         7$     ;BR IF YES
ERROR      12     ;"READY" AND "CRC ERROR"
RTI         ;IT WAS A CRC ERROR AND WE ARE IGNORING THEM
            ;      SO RETURN TO CALLER

```



```

2065      ::*****
2066      ::THIS ROUTINE WILL BACK SPACE ONE FILE GAP
2067      011314 011637 001204      007314      1$:  MOV      (SP),A#SAVPC      ;SAVE PC OF CALL
2068      011320 012737 015443      MOV      #MXBSFG,A#MFUNC      ;SAVE POINTER OF .ASCIZ NAME
2069      011326 004037 012234      JSR      RO,A#DO.CMD      ;START A "BSFG"
2070      011332 000107      XBSFG!INT.EN!GO
2071      011334 005037 177776      CLR      A#PS      ;ALLOW INTERRUPTS
2072      011340 004037 011726      JSR      RO,A#WAITFLAG      ;WAIT ON FLAG
2073      011344 104003      ERROR      3      ;NO FLAG
2074      011346 104004      ERROR      4      ;FLAG DIDN'T INTERRUPT
2075      011350 004037 012024      JSR      RO,A#FLAGS      ;FIND OUT WHAT FLAG OCCURRED
2076      011354 104002      ERROR      2      ;UNKNOWN
2077      011356 104002      ERROR      2      ;"XFER REQ"
2078      011360 000401      BR      2$      ;"READY"
2079      011362 104002      ERROR      2      ;"READY" & "ERROR"
2080      011364 032714 004000      2$:  BIT      #FGAP,A#TACS      ;IN A FILE GAP?
2081      011370 001401      BEQ      3$      ;BR IN NO
2082      011372 000002      RTI      ;GO BACK
2083      011374 104006      3$:  ERROR      6      ;"BSFG" DIDN'T STOP IN A FILE GAP
2084
2085      ::*****
2086      ::THIS ROUTINE WILL BACK SPACE ONE BLOCK GAP
2087      011376 011637 001204      007314      1$:  MOV      (SP),A#SAVPC      ;SAVE THE PC OF THE CALL
2088      011402 012737 015467      MOV      #MXBSBG,A#MFUNC      ;MESSAGE POINTER
2089      011410 004037 012234      JSR      RO,A#DO.CMD      ;START A "BSBG"
2090      011414 000111      XBSBG!INT.EN!GO
2091      011416 005037 177776      CLR      A#PS      ;ALLOW INTERRUPTS
2092      011422 004037 011726      JSR      RO,A#WAITFLAG      ;WAIT FOR A FLAG
2093      011426 104003      ERROR      3      ;NO FLAG OCCURRED
2094      011430 104004      ERROR      4      ;FLAG DIDN'T INTERRUPT
2095      011432 004037 012024      JSR      RO,A#FLAGS      ;WHAT FLAG OCCURRED
2096      011436 104002      ERROR      2      ;UNKNOWN
2097      011440 104002      ERROR      2      ;"XFER REQ"
2098      011442 000002      RTI      ;"READY"-- GO BACK TO USER
2099      011444 104002      ERROR      2      ;"READY" & "ERROR"
2100

```

M05

TA11 MOTION TEST MAINDEC-11-DZTAD-C
DZTADC.NEW 24-MAY-76 00:00

MACY11 27(1006) 23-SEP-76 10:33 PAGE 48
ROUTINE TO "SPACE-FORWARD-FILE-GAP"

SEQ 0064

```

2101      ;:*****
2102      ;THIS ROUTINE WILL SPACE FORWARD FOR ONE FILE GAP
2103      T.SFFG: MOV      (SP),A#SAVPC      ;SAVE PC OF CALL
2104      MOV      #MXSFFG,A#MFUNC      ;POINTER TO .ASCIZ MESSAGE
2105      1$: JSR      RD,A#DO.CMD      ;START A "SFFG"
2106      XSFFG!INT.EN!GO
2107      CLR      A#PS      ;ALLOW INTERRUPTS
2108      JSR      RD,A#WAITFLAG      ;WAIT FOR A FLAG
2109      ERROR    3      ;FLAG FAILED TO OCCUR
2110      ERROR    4      ;FLAG DIDN'T INTERRUPT
2111      JSR      RD,A#FLAGS      ;WHAT FLAG OCCURRED
2112      ERROR    2      ;UNKNOWN
2113      ERROR    2      ;"XFER REQ"
2114      BR       2$      ;"READY"
2115      ERROR    2      ;"READY" AND "ERROR"
2116      2$: BIT      #FGAP,A#TACS      ;DID "SFFG" STOP IN A "FILE GAP"?
2117      BEQ      3$      ;BR IF NO
2118      RTI      ;RETURN TO CALLER
2119      3$: ERROR    6      ;"SFFG" DIDN'T STOP IN A FILE GAP
2120
2121      ;:*****
2122      ;THIS ROUTINE WILL SPACE FORWARD ONE BLOCK GAP
2123      T.SFBG: MOV      (SP),A#SAVPC      ;SAVE PC OF CALL
2124      MOV      #MXSFBG,A#MFUNC      ;POINTER TO .ASCIZ MESSAGE
2125      1$: JSR      RD,A#DO.CMD      ;START A "SFBG"
2126      XSFBG!INT.EN!GO
2127      CLR      A#PS      ;ALLOW INTERRUPTS
2128      JSR      RD,A#WAITFLAG      ;WAIT FOR A FLAG
2129      ERROR    3      ;NO FLAG
2130      ERROR    4      ;NO INTERRUPT
2131      JSR      RD,A#FLAGS      ;FIND OUT WHAT FLAG
2132      ERROR    2      ;UNKNOWN
2133      ERROR    2      ;"XFER REQ"
2134      RTI      ;"READY"
2135      TST      A#ASKKEY      ;RUNNING UNDER MANUAL LOOPING?
2136      BGE      2$      ;BR IF NO
2137      RTI      ;RETURN
2138      2$: ERROR    2      ;"READY" AND "ERROR"
2139

```


N05

TA11 MOTION TEST
DZTADC.NEW 24-MAY-76 00:00

MAINDEC-11-DZTAD-C

MACY11 27(1006) 23-SEP-76 10:33 PAGE 49
ROUTINE TO REWIND THE TAPE

SEQ 0065

```

2140
2141
2142 011610 011637 001204
2143 011614 012737 015354 007314
2144 011622 004037 012024
2145 011626 104001
2146 011630 104001
2147 011632 000411
2148 011634 005777 167416
2149 011640 001406
2150 011642 032714 020000
2151 011646 001403
2152 011650 005077 167402
2153 011654 104001
2154 011656 004037 012234
2155 011662 000117
2156 011664 005037 177776
2157 011670 004037 011726
2158 011674 104003
2159 011676 104004
2160 011700 004037 012024
2161 011704 104002
2162 011706 104002
2163 011710 000401
2164 011712 104002
2165 011714 032714 020000
2166 011720 001401
2167 011722 000002
2168 011724 104007

;*****
;THIS ROUTINE WILL REWIND THE TAPE
;T.RWND: MOV (SP),@#SAVPC ;SAVE PC OF CALL
;MOV #MXRWND,@#MFUNC ;POINTER TO .ASCIZ MESSAGE
;JSR RO,@#FLAGS ;CHECK THE FLAGS
;ERROR 1 ;NO FLAGS
;ERROR 1 ;"XFER REQ"
;BR 1$ ;NORMAL RETURN
;TST @RWDFLG ;WAS LAST FUNCTION A REWIND?
;BEQ 1$ ;BR IF YES
;BIT #LEADER,@TACS ;ON CLEAR LEADER?
;BEQ 1$ ;BR IF NO
;CLR @RWDFLG ;ALLOW THIS ERROR ONLY ONE TIME
;ERROR 1 ;ON CLEAR LEADER BUT SHOULDN'T BE
;JSR RO,@#DO.CMD ;GO START A "REWIND"
;XRWND:INT.EN!GO
;CLR @#PS
;JSR RO,@#WAITFLAG ;ALLOW INTERRUPTS
;ERROR 3 ;WAIT ON FLAG
;ERROR 4 ;NO FLAG OCCURRED
;JSR RO,@#FLAGS ;NO INTERRUPT
;ERROR 2 ;WHAT FLAG OCCURRED
;ERROR 2 ;UNKNOWN
;BR 2$ ;"XFER REQ"
;ERROR 2 ;"READY"
;BIT #LEADER,@TACS ;"READY" & "ERROR"
;BEQ 3$ ;ON "CLEAR LEADER"?
;RTI ;BR IF NO
;ERROR 7 ;YES---RETURN
; ;"REWIND" FAILED TO GO TO "CLEAR LEADER"

```


[illegible]

011726	005067	000064	
011726	016767	000064	000060
011732	032777	000040	167172
011740	001403		
011746	012767	177777	167276
011750	032714	000240	
011756	001007		
011762	005267	000026	
011764	001372		
011770	005367	000022	
011772	100367		
011776	000405		
012000	005720		
012002	005767	167244	
012004	001401		
012010	005720		
012012	000200		
012014	000000		
012016	000000		
012020	000000		
012022	000000		

```

WAITFLAG:      CLR      LOCNT
                MOV      MAXCNT, HICNT
                BIT      #SWDS, $SWR
                BEQ      1$
                MOV      #-1, WAITKEY
1$:             BIT      #TR:REQ!READY, $TACS
                BNE      2$
                INC      LOCNT
                BNE      1$
                DEC      BPL
                BPL      1$
                BR       3$
2$:             TST      (RD)+
                TST      WAITKEY
                BEQ      3$
                TST      (RD)+
3$:             RTS      RD
LOCNT:          DD      0
HICNT:          DD      0
MAXCNT:         DD      0

```

```

;ZERO THE LOW COUNTER
;SET THE HIGH COUNTER TO MAX.COUNT
;RUNNING WITH INTERRUPTS?
;BR IF YES
;SET THE INTERRUPT OCCURRED FLAG
;TEST FOR FLAGS
;BR IF A FLAG IS UP
;COUNT THE LOW COUNTER
;LOOP IF NOT ZERO
;COUNT THE HIGH COUNTER
;LOOP IF NOT NEG.
;LEAVE
;INCREMENT THE RETURN ADDRESS
;LOOK AT THE WAIT ON INT. KEY
;BR IF NOT SET
;INCREMENT THE RETURN ADDRESS
;RETURN TO USER

```


0199
0200
0201
0202
0203
0204
0205
0206
0207
0208
0209
0210
0211
0212
0213
0214
0215
0216
0217
0218
0219
0220
0221
0222
0223
0224
0225
0226
0227
0228
0229
0230
0231

012024 03271 000240
012030 001427
012032 005720
012034 105714
012036 100424
012040 005720
012042 032777 000100 167070
012050 001414
012052 010046
012054 010146
012056 013700 001264
012062 013701 001266
012066 062700 000001
012072 005501
012074 001374
012076 012601
012100 012600
012102 005714
012104 100001
012106 005720
012110 000200

THIS ROUTINE IS USED TO DETERMINE WHAT FLAG IS SET

CALL:

JSR RO, 3#FLAGS
RETURN FOR UNKNOWN FLAG
RETURN FOR "TRANSFER REQUEST"
RETURN FOR "READY"
RETURN FOR "READY" & "ERROR"

FLAGS: BIT #TR.REQ!READY, 3TACS ;CHECK FOR "XFER REQ" OR "READY"
BEQ LVFLAG ;BR IF NO FLAGS
TST (RO)+ ;INCREMENT RETURN ADDRESS
TSTB 3TACS ;CHECK FOR "XFER REQ"
BMI LVFLAG ;BR IF SET
TST (RO)+ ;INCREMENT RETURN ADDRESS
BIT #SW06, 3SWR ;STALL?
BEQ 3\$;BR IF NO
MOV RO, -(SP) ;PUSH RO ON STACK
MOV R1, -(SP) ;PUSH R1 ON STACK
MOV 3#STALL, RO ;GET STALL TIME
MOV 3#STALL+2, R1
1\$: ADD #1, RO ;STALL
ADC R1
BNE 1\$;LOOP
MOV (SP)+, R1 ;POP STACK INTO R1
MOV (SP)+, RO ;POP STACK INTO RO
3\$: TST 3TACS ;IS "ERROR" SET
BPL LVFLAG ;BR IN NO "ERROR"
TST (RO)+ ;INCREMENT RETURN ADDRESS
LVFLAG: RTS RO

2232
2233
2234
2235
2236
2237
2238
2239
2240
2241
2242
2243
2244
2245
2246
2247
2248
2249
2250
2251
2252
2253
2254
2255
2256
2257
2258
2259
2260
2261
2262
2263
2264
2265
2266
2267
2268
2269
2270
2271
2272

012112
012112 010046
012114 010146
012116 010246
012120 010346
012122 010446
012124 012767 000010 000054
012132 016703 000074
012136 005001
012140 010302
012142 042702 177776
012146 006000
012150 006101
012152 010104
012154 040204
012156 040102
012160 050402
012162 006002
012164 103006
012166 012701 040002
012172 010104
012174 040304
012176 040103
012200 050403
012202 006003
012204 005327
012206 000000
012210 003352
012212 010367 000014
012216 012604
012220 012603
012222 012602
012224 012601
012226 012600
012230 000207
012232 000000

THIS ROUTINE WILL CALCULATE THE CRC
CALL:

JSR PC, @DO.CRC ;RO=1 BYTE OF DATA

DO.CRC:

MOV R0, -(SP) ;: PUSH R0 ON STACK
MOV R1, -(SP) ;: PUSH R1 ON STACK
MOV R2, -(SP) ;: PUSH R2 ON STACK
MOV R3, -(SP) ;: PUSH R3 ON STACK
MOV R4, -(SP) ;: PUSH R4 ON STACK
MOV #8, 3\$;: MAKE EIGHT ITERATIONS
MOV CRC.WD, R3 ;: PICKUP THE CRC WORD
1\$: CLR R1
MOV R3, R2 ;: GET THE PARTIAL CRC WORD
BIC #1<BIT00, R2 ;: STRIP AWAY EVERYTHING BUT BIT00
ROR R0 ;: PULL OFF THIS BIT
ROL R1 ;: AND SETUP TO XOR IT
MOV R1, R4 ;: FORM THE XOR OF "R1" AND "R2"
BIC R2, R4
BIC R1, R2
BIS R4, R2 ;: RESULTS TO "R2"
ROR R2
BCC 2\$
MOV #BIT14!BIT01, R1 ;: FORM THE XOR OF "R1" AND "R3"
MOV R1, R4
BIC R3, R4
BIC R1, R3
BIS R4, R3 ;: RESULTS TO "R3"
ROR R3
DEC (PC)+
3\$: O
BGT 1\$;: BR IF MORE BITS TO DO
MOV R3, CRC.WD
MOV (SP)+, R4 ;: POP STACK INTO R4
MOV (SP)+, R3 ;: POP STACK INTO R3
MOV (SP)+, R2 ;: POP STACK INTO R2
MOV (SP)+, R1 ;: POP STACK INTO R1
MOV (SP)+, R0 ;: POP STACK INTO R0
RTS PC
CRC.WD: C ;: CRC WORD

TA11 MOTION TEST
DZTADC.NEW 24-MAY-76 00:00

MAINDEC-11-DZTAD-C

MACY11 27(1006) 23-SEP-76 10:33 PAGE 53
ROUTINE TO PERFORM THE REQUESTED FUNCTION

SEQ 0069

2273
2274
2275
2276
2277
2278
2279
2280
2281
2282
2283
2284
2285
2286
2287
2288
2289
2290
2291
2292
2293
2294
2295
2296
2297
2298
2299
2300
2301
2302
2303
2304
2305
2306
2307
2308
2309
2310
2311
2312
2313
2314
2315
2316

012234 010146
012236 012737 000340 177776
012244 012001
012246 042701 177640
012252 032777 000040 166660
012260 001405
012262 042701 000100
012266 012777 012366 166734
012274 005067 166754
012300 032701 000100
012304 001403
012306 012777 012350 166714
012314 110114
012316 032701 000001
012322 001403
012324 032714 000040
012330 001375
012332 005101
012334 042701 177761
012340 010177 166712
012344 012601
012346 000200

THIS ROUTINE WILL LOAD THE LOW BYTE OF TACS WITH THE DESIRED FUNCTION

CALL:

JSR RO, @DO, CMD
DESIRED FUNCTIN
RETURN

DO.CMD: MOV R1, -(SP) ;SAVE R1
MOV #340, @#PS ;LOCK OUT ALL INERRUPTS
MOV (R0)+, R1 ;GET THE COMMAND
BIC #1<INT.EN!ILBS!FUNCTION!GO>, R1 ;STRIP AWAY JUNK
BIT #SW05, @SWR ;WITH OR WITHOUT INTERRUPTS
BEQ 4\$;BR IF WITH
BIC #INT.EN, R1 ;CLEAR INTERRUPT ENABLE IF SET
MOV #BADINT, @TAVEC ;SETUP TO CATCH BAD INTERRUPT
CLR WAITKEY ;ZERO THE WAIT ON INTERRUPT KEY
BIT #INT.EN, R1 ;RUNNING WITH INTERRUPTS?
BEQ 3\$;BR IF NO
MOV #SERV, @TAVEC ;SETUP TO SERVICE THE INTERRUPT
MOVB R1, @TACS ;LOAD THE COMMAND
BIT #GO, R1 ;IS GO BIT ON A "1"?
BEQ 2\$;BR IF GO =0
BIT #READY, @TACS ;WAIT FOR "READY" TO CLEAR
BNE 1\$
COM R1 ;SET OR CLEAR REWIND FLAG
BIC #1<FUNCTION>, R1 ;EXTRACT FUNCTION BITS
MOV R1, @RWDFLG ;AND SAVE AS THE REWIND FLAG
MOV (SP)+, R1 ;GET R1 BACK
RTS RO ;RETURN TO USER

THIS ROUTINE WILL SERVICE ALL LEGAL INTERRUPTS

SERV: MOV #-1, @WAITKEY ;SET THE WAIT KEY
MOV #BADINT, @TAVEC ;SET TO CATCH ILLEGAL INTERRUPTS
RTI

THIS ROUTINE WILL CATCH ILLEGAL INTERRUPTS FROM THE TA11

BADINT: MOV (SP), @SAVPC ;SAVE WHERE WE WERE AT
ERROR 14 ;ILLEGAL INTERRUPT OCCURRED

F06

TA11 MOTION TEST MAINDEC-11-DZTAD-C
DZTADC.NEW 24-MAY-76 00:00

MACY11 27(1006) 23-SEP-76 10:33 PAGE 54
***** MANUAL ADJUSTMENT ROUTINES *****

SEQ 0070

2317
2318
2319
2320

:THE FOLLOWING ROUTINES CAN BE USED TO MAKE ADJUSTMENTS TO THE TUBO
:NOTE: *** BEFORE USING ANY OF THE ROUTINES LOAD AND START AT 214 ***

TA11 MOTION TEST MAINDEC-11-DZTAD-C
DZTADC.NEW 24-MAY-76 00:00

MACY11 27(1006) 23-SEP-76 10:33 PAGE 55
***** MANUAL ADJUSTMENT ROUTINES *****

SEQ 0071

2321
2322
2323
2324
2325
2326
2327
2328
2329
2330
2331
2332
2333
2334
2335
2336
2337
2338
2339
2340
2341
2342
2343
2344
2345
2346
2347
2348
2349
2350
2351
2352
2353
2354
2355
2356
2357
2358
2359
2360
2361

;; ** NOTE: IF USING SOFTWARE SWITCH REGISTER THE
PROGRAM MUST HAVE BEEN STARTED AT A
NORMAL STARTING ADDRESS FIRST SO THAT
THE AUTO-SIZING ROUTINE CAN SET UP ADDRESS
176 AS THE SOFTWARE SWITCH REGISTER
PLACE VALUE INTO 176

;;*****

;;*****

WRITE FILE GAPS FROM "BOT" TO "EOT"

START AT 220

THIS ROUTINE CAN BE USED TO ADJUST THE "WRITE GAP MONO" AND
THE "WRITE DELAY MONO".

;;*****

WFGSUB:	MOV	#STACK, SP	;	KEEP THE STACK OUT OF THE WAY
	MOV	2#TACSL, TACS	;	SETUP THE TA11 STATUS AND
	MOV	2#TADBL, TADB	;	DATA BUFFER REGISTERS
	RESET		;	RESET THE WORLD
	MOV	#WFGSUB, 2#\$LPERR	;	SETUP THE LOOP ON ERROR ADDRESS
	SELDRV		;	SELECT THE DRIVE
	MOVB	#XRWND!GO, 2TACS	;	SEND TAPE TO "BOT"
100\$:	BIT	#READY, 2TACS	;	WAIT ON READY
	BEQ	100\$		
1\$:	WFG		;	WRITE A FILE GAP
	BIT	#LEADER, 2TACS	;	AT "CLEAR LEADER"?
	BEQ	1\$	;	BR IF NO
	HALT		;	STOP IF YES
	BR	WFGSUB	;	LOOP ON CONT.

;;*****

WRITE CONTINUOUS BLOCKS OF DATA

START AT 224

THE PROGRAM WILL HALT THREE(3) TIMES

AFTER EACH HALT SET THE SWR AND PRESS CONTINUE

HALT 1 --- SWR<7:0> = NUMBER OF BYTES PER BLOCK

HALT 2 --- SWR<7:0> = PATTERN DESIRED

HALT 3 --- SWR<15:0> = OPERATIONAL SWITCH SETTINGS

THIS ROUTINE CAN BE USED TO ADJUST THE "GAP TIME MONO"

H06

TA11 MOTION TEST MAINDEC-11-DZTAD-C
DZTADC.NEW 24-MAY-76 00:00

MACY11 27(1006) 23-SEP-76 10:33 PAGE 56
WRITE CONTINUOUS BLOCKS OF DATA

SEQ 0072

```

2362 ; THIS ROUTINE SUPPORTS THE S/W SWITCH REG FUNCTIONS
2363 ;*****
2364 WRTSUB: JSR PC, @#SETBUF ; GET BLOCK SIZE AND PATTERN
2365 WLOOP: MOV #STACK, SP ; KEEP THE STACK OUT OF THE WAY
2366 MOV @#TACSL, TACS ; SETUP THE TA11 STATUS AND
2367 MOV @#TADBL, TADB ; DATA BUFFER REGISTERS
2368 RESET ; RESET THE WORLD
2369 MOV #WLOOP, @#SLPERR ; SETUP THE LOOP ON ERROR ADDRESS
2370 SELDRV ; SELECT THE DRIVE
2371 MOVB #XRWD!GO, @TACS ; SEND TAPE TO "BOT"
2372 100$: BIT #READY, @TACS ; WAIT ON READY
2373 BEQ 100$
2374 1$: WRITE , WLNKX ; WRITE A BLOCK
2375 BIT #LEADER, @TACS ; AT "CLEAR LEADER"?
2376 BEQ 1$ ; BR IF NO
2377 HALT ; STOP IF "EOT"
2378 BR WLOOP ; LOOP IF CONT.
2379
2380 ;*****
2381 ; READ AND COMPARE CONTINUOUS BLOCKS OF DATA
2382 ; START AT 230
2383 ; THIS ROUTINE CAN BE USED TO ADJUST THE "SIGNAL MONO"
2384 ; AND THE "THRESHOLD POT".
2385 ;*****
2386 RDSUB: MOV #STACK, SP ; KEEP THE STACK OUT OF THE WAY
2387 MOV @#TACSL, TACS ; SETUP THE TA11 STATUS AND
2388 MOV @#TADBL, TADB ; DATA BUFFER REGISTERS
2389 RESET ; RESET THE WORLD
2390 MOV #RDSUB, @#SLPERR ; SETUP THE LOOP ON ERROR ADDRESS
2391 SELDRV ; SELECT THE DRIVE
2392 MOVB #XRWD!GO, @TACS ; SEND TAPE TO "BOT"
2393 100$: BIT #READY, @TACS ; WAIT ON READY
2394 BEQ 100$
2395 1$: READ , RLNKX ; READ A BLOCK
2396 MOV #STACK, SP ; INIT. IN CASE OF ERROR
2397 BIT #LEADER, @TACS ; AT "CLEAR LEADER"?
2398 BNE RDSUB ; BR IF YES---LOOP
2399 BLKCMP ; COMPARE THE READ AND WRITE BUFFERS
2400 RLNKX ; LINK TO THE READ BUFFER
2401 WLNKX ; LINK TO THE WRITE BUFFER
2402 BR 1$
2403
2404 ;*****
2405 ; WRITE A FILE GAP AND A BLOCK OF DATA FROM BOT TO EOT
2406 ; START AT 234
2407 ; THE PROGRAM WILL HALT THREE(3) TIMES
2408 ; AFTER EACH HALT SET THE SWR AND PRESS CONTINUE
2409 ; HALT 1 --- SWR<7:0> = NUMBER OF BYTES PER BLOCK
2410 ; HALT 2 --- SWR<7:0> = PATTERN DESIRED
2411 ; HALT 3 --- SWR<15:0> = OPERATIONAL SWITCH SETTINGS
2412 ; THIS ROUTINE CAN BE USED TO ADJUST THE "WRITE GAP MONO"
2413 ; AND THE "GAP TIME MONO".
2414
2415

```


TA11 MOTION TEST MAINDEC-11-DZTAD-C
DZTADC:NEW 24-MAY-76 00:00

MACY11 27(1006) 23-SEP-76 10:33 PAGE 57
WRITE A FILE GAP AND A BLOCK OF DATA FROM BOT TO EOT

SEQ 0073

```

2416
2417
2418 012620 004737 013116
2419 012624 012706 001100
2420 012630 013704 001220
2421 012634 013705 001224
2422 012640 000005
2423 012642 012737 012624 001110
2424 012650 104422
2425 012652 112714 000017
2426 012656 032714 000040
2427 012662 001775
2428 012664 104412
2429 012666 032714 020000
2430 012672 001005
2431 012674 104413 017052
2432 012700 032714 020000
2433 012704 001767
2434 012706 000000
2435 012710 000745
2436
2437
2438
2439
2440
2441
2442
2443
2444
2445 012712 012706 001100
2446 012716 013704 001220
2447 012722 013705 001224
2448 012726 000005
2449 012730 012737 012712 001110
2450 012736 104422
2451 012740 112714 000017
2452 012744 032714 000040
2453 012750 001775
2454 012752 104414 017016
2455 012756 012706 001100
2456 012762 032714 020000
2457 012766 001351
2458 012770 104423
2459 012772 017016
2460 012774 017052
2461 012776 104420
2462 013000 032714 020000
2463 013004 001342
2464 013006 000761
2465
2466
2467
2468
2469
2470
2471

; THIS ROUTINE SUPPORTS THE S/W SWITCH REG FUNCTIONS
;*****
WGPBLK: JSR PC, @SETBUF ; GET BLOCK SIZE AND PATTERN
WGBLOP: MOV #STACK, SP ; KEEP THE STACK OUT OF THE WAY
        MOV @TACSL, TACS ; SETUP THE TA11 STATUS AND
        MOV @TADBL, TADB ; DATA BUFFER REGISTERS
        RESET ; RESET THE WORLD
        MOV #WGBLOP, @SLPERR ; SETUP THE LOOP ON ERROR ADDRESS
        SELDRV ; SELECT THE DRIVE
        MOVB #XRWIND!GO, @TACS ; SEND TAPE TO "BOT"
100$: BIT #READY, @TACS ; WAIT ON READY
      BEQ 100$
1$: WFG ; WRITE A FILE GAP
    BIT #LEADER, @TACS ; AT "CLEAR LEADER"?
    BNE 2$ ; BR IF YES
    WRITE , WLNKX ; WRITE A BLOCK
    BIT #LEADER, @TACS ; AT "CLEAR LEADER"?
    BEQ 1$ ; BR IF NO
2$: HALT ; STOP AT "EOT"
    BR WGBLOP ; START OVER ON CONT.

;*****
; READ AND COMPARE A BLOCK OF DATA AND A FILE GAP
; START AT 240
; THIS ROUTINE IS USED AFTER "WRITE A BLOCK AND A FILE GAP" ROUTINE
; IT CAN BE USED TO ADJUST THE "SIGNAL MONO", THE THRESHOLD POT"
; AND THE "TAPE BLANK MONO".
;*****
RGPBLK: MOV #STACK, SP ; KEEP THE STACK OUT OF THE WAY
        MOV @TACSL, TACS ; SETUP THE TA11 STATUS AND
        MOV @TADBL, TADB ; DATA BUFFER REGISTERS
        RESET ; RESET THE WORLD
        MOV #RGPBLK, @SLPERR ; SETUP THE LOOP ON ERROR ADDRESS
        SELDRV ; SELECT THE DRIVE
        MOVB #XRWIND!GO, @TACS ; SEND TAPE TO "BOT"
100$: BIT #READY, @TACS ; WAIT ON READY
      BEQ 100$
1$: READ , RLNKX ; READ A BLOCK OF DATA
    MOV #STACK, SP ; KEEP OUT OF THE WAY
    BIT #LEADER, @TACS ; AT "CLEAR LEADER"?
    BNE RGPBLK ; BR IF YES
    BLKCMP ; COMPARE THE READ AND WRITE BUFFERS
    RLNKX ; LINK TO THE READ BUFFER
    WLNKX ; LINK TO THE WRITE BUFFER
    SFBG ; GET INTO FILE GAP
    BIT #LEADER, @TACS ; AT "CLEAR LEADER"?
    BNE RGPBLK ; BR IF YES
    BR 1$ ; LOOP

;*****
; SPACE FORWARD FILE GAP FROM "BOT" TO "EOT"
; START AT 244
; THIS ROUTINE CAN BE USED AFTER "WRITE FILE GAP" FOR LOW SPEED
; SPACE FOWARD (TAPE BLANK MONO CAN BE ADJUSTED), OR AFTER READ OR

```

J06

TA11 MOTION TEST MAINDEC-11-DZTAD-C
DZTADC.NEW 24-MAY-76 00:00

MACY11 27(1006) 23-SEP-76 10:33 PAGE 58
SPACE FORWARD FILE GAP FROM "BOT" TO "EOT"

SEQ 0074

```

2472                                     ;WRITE A FILE GAP AND A BLOCK OF DATA FOR HIGH SPEED SPACE FORWARD
2473                                     ; (SIGNAL MONO CAN BE CHECKED).
2474                                     ;*****
2475 013010 012706 001100 $FFGSB: MOV #STACK,SP ;KEEP THE STACK OUT OF THE WAY
2476 013014 013704 001220 MOV @#TACSL,TACS ;SETUP THE TA11 STATUS AND
2477 013020 013705 001224 MOV @#TADBL,TADB ;DATA BUFFER REGISTERS
2478 013024 000005 RESET ;RESET THE WORLD
2479 013026 012737 013010 001110 MOV #SFFGSB,@#$LPERR ;SETUP THE LOOP ON ERROR ADDRESS
2480 013034 104422 SELDRV ;SELECT THE DRIVE
2481 013036 112714 000017 MOV #XRWND!GO,@TACS ;SEND TAPE TO "BOT"
2482 013042 032714 000040 100$: BIT #READY,@TACS ;WAIT ON READY
2483 013046 001775 BEQ 100$
2484 013050 104417 1$: SFFG ;SPACE INTO A FILE GAP
2485 013052 032714 020000 BIT #LEADER,@TACS ;AT "CLEAR LEADER"?
2486 013056 001774 BEQ 1$ ;BF IF NO
2487 013060 000000 HALT ;STOP AT "EOT"
2488 013062 000752 BR SFFGSB ;LOOP ON CONT.
2489
2490
2491                                     ;*****
2492                                     ; BACK SPACE FILE GAP
2493                                     ; START AT 250
2494                                     ; THIS ROUTINE CAN BE USED TO ADJUST OR CHECK THE "SIGNAL MONO".
2495                                     ;*****
2496 013064 000005 013064 001110 BSFGSB: RESET ;RESET THE WORLD
2497 013066 012737 166072 MOV #BSFGSB,@#$LPERR ;LOOP ON ERROR ADDRESS
2498 013074 005067 CLR $ESCAPE ;DON'T ESCAPE ON ERROR
2499 013100 104422 SELDRV ;SELECT THE DRIVE
2500 013102 104415 1$: BSFG ;BACK SPACE A FILE GAP
2501 013104 032714 020000 BIT #LEADER,@TACS ;AT "CLEAR LEADER"?
2502 013110 001774 BEQ 1$ ;BR IF NO
2503 013112 000000 HALT ;STOP AT BOT
2504 013114 000753 BR BSFGSB ;START OVER ON CONT.
2505
2506
2507                                     ;*****
2508                                     ; SETUP READ AND WRITE LINKS FOR SUBROUTINES
2509 013116 005000 SETBUF: CLR RO
2510 013120 000000 HALT ;OPERATOR PUTS BYTE COUNT IN THE SWR

```


K06

TA11 MOTION TEST MAINDEC-11-DZTAD-C
DZTADC.NEW 24-MAY-76 00:00

MACY11 27(1006) 23-SEP-76 10:33 PAGE 59
SETUP READ AND WRITE LINKS FOR SUBROUTINES

SEQ 0075

2511	013122	022767	000176	166010				
2512	013130	001001						
2513	013132	104405						
2514	013134				20\$:			
2515	013134	157700	166000					
2516	013140	001006						
2517	013142	105777	165773					
2518	013146	001402						
2519	013150	012700	000376					
2520	013154	005200			1\$:			
2521	013156	010037	017020		2\$:			
2522	013162	010037	017054					
2523	013166	000000						

CMP	#SWREG,SWR			:USING S/W SWITCH REG?
BNE	20\$			:NO- GET OUT
GTSWR				:GET VALUE
				:CONTINUE
BISB	2SWR,RO		:PICKUP	THE BYTE COUNT
BNE	2\$			:BR IF NON-ZERO
TSTB	2SWR+1			:CHECK IF GREATER THAN 377
BEQ	1\$			:BR IF NO
MOV	#376,RO			:SET FOR MAX ALLOWED
INC	RO			:MAKE IT 377 OR 1
MOV	RO,2#RLNKX+2			:SETUP READ LINK
MOV	RO,2#WLNKX+2			:SETUP WRITE LINK
HALT				:SET PATTERN INTO THE SWR

L06

TA11 MOTION TEST
DZTADC.NEW 24-MAY-76 00:00

MAINDEC-11-DZTAD-C

MACY11 27(1006) 23-SEP-76 10:33 PAGE 60
SETUP READ AND WRITE LINKS FOR SUBROUTINES

SEQ 0076

2524 013170 022767 000176 165742
2525 013176 001001
2526 013200 104405
2527 013202
2528 013202 017700 165732
2529 013206 012701 017122
2530 013212 110021
2531 013214 110021
2532 013216 110021
2533 013220 110021
2534 013222 000000

21\$:

CMP #SWREG,SWR
BNE 21\$
GTSWR

MOV @SWR,RO
MOV #WBUF,X,R1
MOVB RO,(R1)+
MOVB RO,(R1)+
MOVB RO,(R1)+
MOVB RO,(R1)+
HALT

; USING S/W SWITCH REG?
; NO- GET OUT
; GET VALUE
; CONTINUE
; PICKUP THE PATTERN
; PICKUP FIRST ADDRESS OF BUFFER
; FILL THE BUFFER

; SET OPERATIONAL SWITCHES

M06

TA11 MOTION TEST
DZTADC.NEW 24-MAY-76 00:00

MAINDEC-11-DZTAD-C

MACY11 27(1006) 23-SEP-76 10:33 PAGE 61
SETUP READ AND WRITE LINKS FOR SUBROUTINES

SEQ 0077

2535 013224 022767 000176 165706
2536 013232 001001
2537 013234 104405
2538 013236
2539 013236 000207

22\$:

CMP #SWREG,SWR
BNE 22\$
GTSWR
RTS PC

;USING S/W SWITCH REG?
;NO- GET OUT
;GET VALUE
;CONTINUE
;RETURN

2540
2541
2542
2543
2544
2545
2546
2547
2548
2549
2550
2551
2552
2553
2554
2555
2556
2557
2558
2559
2560
2561
2562
2563
2564
2565
2566
2567
2568
2569
2570
2571
2572
2573
2574
2575
2576
2577
2578
2579
2580
2581
2582
2583
2584
2585
2586
2587
2588
2589
2590
2591
2592
2593
2594
2595

.SBTTL TYPE ROUTINE

```

*****
*ROUTINE TO TYPE ASCIZ MESSAGE. MESSAGE MUST TERMINATE WITH A 0 BYTE.
*THE ROUTINE WILL INSERT A NUMBER OF NULL CHARACTERS AFTER A LINE FEED.
*NOTE1: $NULL CONTAINS THE CHARACTER TO BE USED AS THE FILLER CHARACTER.
*NOTE2: $FILLC CONTAINS THE NUMBER OF FILLER CHARACTERS REQUIRED.
*NOTE3: $FILLC CONTAINS THE CHARACTER TO FILL AFTER.

```

```

*CALL:
*1) USING A TRAP INSTRUCTION
*   TYPE ,MESADR      ;;MESADR IS FIRST ADDRESS OF AN ASCIZ STRING
*OR
*   TYPE
*   MESADR

```

```

$TYPE:  TSTB    $TPFLG      ;; IS THERE A TERMINAL?
        BPL     1$          ;; BR IF YES
        HALT    3$          ;; HALT HERE IF NO TERMINAL
        BR      LEAVE
1$:      MOV     RD, -(SP)    ;; SAVE RD
        MOV     @2(SP), RD   ;; GET ADDRESS OF ASCIZ STRING
2$:      MOVB    (RD)+, -(SP) ;; PUSH CHARACTER TO BE TYPED ONTO STACK
        BNE     4$          ;; BR IF IT ISN'T THE TERMINATOR
        TST     (SP)+        ;; IF TERMINATOR POP IT OFF THE STACK
60$:     MOV     (SP)+, RD    ;; RESTORE RD
3$:      ADD     #2, (SP)     ;; ADJUST RETURN PC
        RTI
4$:      CMPB    #HT, (SP)    ;; BRANCH IF <HT>
        BEQ     8$
        CMPB    #CRLF, (SP)  ;; BRANCH IF NOT <CRLF>
        BNE     5$
        TST     (SP)+        ;; POP <CR><LF> EQUIV
        TYPE    A CR AND LF
        CLRB    $CHARCNT    ;; CLEAR CHARACTER COUNT
        BR      2$          ;; GET NEXT CHARACTER
5$:      JSR     PC, $TYPEC   ;; GO TYPE THIS CHARACTER
6$:      CMPB    $FILLC, (SP)+ ;; IS IT TIME FOR FILLER CHARS.?
        BNE     2$          ;; IF NO GO GET NEXT CHAR.
        MOV     $NULL, -(SP) ;; GET # OF FILLER CHARS. NEEDED
        AND     THE NULL CHAR.
        DOES A NULL NEED TO BE TYPED?
        BR IF NO--GO POP THE NULL OFF OF STACK
        GO TYPE A NULL
        DO NOT COUNT AS A COUNT
        LOOP
7$:      DECB    1(SP)
        BLT     6$
        JSR     PC, $TYPEC   ;; GO TYPE THIS CHARACTER
        DECB    $CHARCNT
        BR      7$

```

;HORIZONTAL TAB PROCESSOR

```

8$:      MOVB    #' , (SP)   ;; REPLACE TAB WITH SPACE
9$:      JSR     PC, $TYPEC   ;; TYPE A SPACE
        BITB    #7, $CHARCNT ;; BRANCH IF NOT AT
        BNE     9$          ;; TAB STOP
        TST     (SP)+        ;; POP SPACE OFF STACK

```

000052

TYPE ROUTINE

```

013406 000724 BR 25 ::GET NEXT CHARACTER
013410 105777 165534 $TYPEC: TSTB 3$TPS ::WAIT UNTIL PRINTER IS READY
013414 100375 BPL $TYPEC
013416 116677 000002 165526 MOVB 2(SP),3$TPB ::LOAD CHAR TO BE TYPED INTO DATA REG.
013424 122766 000015 000002 CMPB #CR,2(SP) ::IS CHARACTER A CARRIAGE RETURN?
013432 001003 BNE 1$ ::BRANCH IF NO
013434 105067 000014 CLRB $CHARCNT ::YES--CLEAR CHARACTER COUNT
013440 000406 BR $TYPEX ::EXIT
013442 122766 000012 000002 1$: CMPB #LF,2(SP) ::IS CHARACTER A LINE FEED?
013450 001402 BEQ $TYPEX ::BRANCH IF YES
013452 105227 INCB (PC)+ ::COUNT THE CHARACTER
013454 000000 $CHARCNT: .WORD 0 ::CHARACTER COUNT STORAGE
013456 000207 $TYPEX: RTS PC

.SBTTL TTY INPUT ROUTINE

*****
:ENABL LSB

*****
:SOFTWARE SWITCH REGISTER CHANGE ROUTINE.
:ROUTINE IS ENTERED FROM THE TRAP HANDLER, AND WILL
:SERVICE THE TEST FOR CHANGE IN SOFTWARE SWITCH REGISTER TRAP CALL
:WHEN OPERATING IN TTY FLAG MODE.
013460 022767 000176 165452 $CKSWR: CMP #SWREG,SWR ::IS THE SOFT-SWR SELECTED?
013466 001074 BNE 15$ ::BRANCH IF NO
013470 105777 165450 TSTB 3$TKS ::CHAR THERE?
013474 100071 BPL 15$ ::IF NO, DON'T WAIT AROUND
013476 117746 165444 MOVB 3$TKB,-(SP) ::SAVE THE CHAR
013502 042716 177600 BIC #C177,(SP) ::STRIP-OFF THE ASCII
013506 022726 000007 CMP #7,(SP)+ ::IS IT A CONTROL G?
013512 001062 BNE 15$ ::NO, RETURN TO USER
013514 126727 165414 000001 CMPB $AUTOB,#1 ::ARE WE RUNNING IN AUTO-MODE?
013522 001456 BEQ 15$ ::BRANCH IF YES

013524 104401 014205 $GTSWR: TYPE , $CNTLG ::ECHO THE CONTROL-G (!G)
013530 104401 014212 TYPE , $MSWR ::TYPE CURRENT CONTENTS
013534 016746 164436 MOV SWREG,-(SF) ::SAVE SWREG FOR TYPEOUT
013540 104402 TYPE , $MNEW ::GO TYPE--OCTAL ASCII(ALL DIGITS)
013542 104401 014223 TYPE , $MNEW ::PROMPT FOR NEW SWR
013546 005046 19$: CLR -(SP) ::CLEAR COUNTER
013550 005046 7$: CLR -(SP) ::THE NEW SWR
013552 105777 165366 TSTB 3$TKS ::CHAR THERE?
013556 100375 BPL 7$ ::IF NOT TRY AGAIN

013560 117746 165362 MOVB 3$TKB,-(SP) ::PICK UP CHAR
013564 042716 177600 BIC #C177,(SP) ::MAKE IT 7-BIT ASCII

013570 021627 000025 9$: CMP (SP),#25 ::IS IT A CONTROL-U?
013574 001005 BNE 10$ ::BRANCH IF NOT
013576 104401 014200 TYPE , $CNTLU ::YES, ECHO CONTROL-U (!U)
013602 062706 000006 20$: ADD #6,SP ::IGNORE PREVIOUS INPUT
013606 000757 BR 19$ ::LET'S TRY IT AGAIN

```



```

2652 013610 021627 000015 10$: CMP (SP),#15 :: IS IT A <CR>?
2653 013614 001022 BNE 16$ :: BRANCH IF NO
2654 013616 005766 000004 TST 4(SP) :: YES, IS IT THE FIRST CHAR?
2655 013622 001403 BEQ 11$ :: BRANCH IF YES
2656 013624 016677 000002 165306 MOV 2(SP),@SWR :: SAVE NEW SWR
2657 013632 062706 000006 11$: ADD #6,SP :: CLEAR UP STACK
2658 013636 104401 001201 14$: TYPE $CRLF :: ECHO <CR> AND <LF>
2659 013642 126727 165267 000001 CMPB $INTAG,#1 :: RE-ENABLE TTY KBD INTERRUPTS?
2660 013650 001003 BNE 15$ :: BRANCH IF NOT
2661 013652 012777 000100 165264 MOV #100,@STKS :: RE-ENABLE TTY KBD INTERRUPTS
2662 013660 000002 15$: RTI :: RETURN
2663 013662 004767 177522 16$: JSR PC,$TYPEC :: ECHO CHAR
2664 013666 021627 000060 CMP (SP),#60 :: CHAR < 0?
2665 013672 002420 BLT 18$ :: BRANCH IF YES
2666 013674 021627 000067 CMP (SP),#67 :: CHAR > 7?
2667 013700 003015 BGT 18$ :: BRANCH IF YES
2668 013702 042726 000060 BIC #60,(SP)+ :: STRIP-OFF ASCII
2669 013706 005766 000002 TST 2(SP) :: IS THIS THE FIRST CHAR
2670 013712 001403 BEQ 17$ :: BRANCH IF YES
2671 013714 006316 ASL (SP) :: NO, SHIFT PRESENT
2672 013716 006316 ASL (SP) :: CHAR OVER TO MAKE
2673 013720 006316 ASL (SP) :: ROOM FOR NEW ONE.
2674 013722 005266 000002 17$: INC 2(SP) :: KEEP COUNT OF CHAR
2675 013726 056616 177776 BIS -2(SP),(SP) :: SET IN NEW CHAR
2676 013732 000707 BR 7$ :: GET THE NEXT ONE
2677 013734 104401 001200 18$: TYPE $QUES :: TYPE ?<CR><LF>
2678 013740 000720 BR 20$ :: SIMULATE CONTROL-U
2679 .DSABL LSB

```

THIS ROUTINE WILL INPUT A SINGLE CHARACTER FROM THE TTY

*CALL:

* RDCHR
* RETURN HERE
*

INPUT A SINGLE CHARACTER FROM THE TTY
CHARACTER IS ON THE STACK
WITH PARITY BIT STRIPPED OFF

```

2691 013742 011646 $RDCHR: MOV (SP),-(SP) :: PUSH DOWN THE PC
2692 013744 016666 000004 000002 MOV 4(SP),2(SP) :: SAVE THE PS
2693 013752 105777 165166 1$: TSTB @STKS :: WAIT FOR
2694 013756 100375 BPL 1$ :: A CHARACTER
2695 013760 117766 165162 000004 MOVB @STKB,4(SP) :: READ THE TTY
2696 013766 042766 177600 000004 BIC #1C(177),4(SP) :: GET RID OF JUNK IF ANY
2697 013774 026627 000004 000023 CMP 4(SP),#23 :: IS IT A CONTROL-S?
2698 014002 001013 BNE 3$ :: BRANCH IF NO
2699 014004 105777 165134 2$: TSTB @STKS :: WAIT FOR A CHARACTER
2700 014010 100375 BPL 2$ :: LOOP UNTIL ITS THERE
2701 014012 117746 165130 MOVB @STKB,-(SP) :: GET CHARACTER
2702 014016 042716 177600 BIC #1C177,(SP) :: MAKE IT 7-BIT ASCII
2703 014022 022627 000021 CMP (SP)+,#21 :: IS IT A CONTROL-Q?
2704 014026 001366 BNE 2$ :: IF NOT DISCARD IT
2705 014030 000750 BR 1$ :: YES, RESUME
2706 014032 026627 000004 000140 3$: CMP 4(SP),#140 :: IS IT UPPER CASE?
2707 014040 002407 BLT 4$ :: BRANCH IF YES

```


007

TA11 MOTION TEST
DZTADC.NEW 24-MAY-76 00:00
MAINDEC-11-DZTAD-C
MACY11 27(1006) 23-SEP-76 10:33 PAGE 65
TTY INPUT ROUTINE

SEQ 0081

```

2708 014042 026627 000004 000175      CMP      4(SP),#175      ;; IS IT A SPECIAL CHAR?
2709 014050 003003      BGT      4$              ;; BRANCH IF YES
2710 014052 042766 000040 000004      BIC      #40,4(SP)      ;; MAKE IT UPPER CASE
2711 014060 000002      RTI              ;; GO BACK TO USER
2712      4$:
2713      *****
2714      *THIS ROUTINE WILL INPUT A STRING FROM THE TTY
2715      *CALL:
2716      *      RDLIN
2717      *      RETURN HERE
2718      *      INPUT A STRING FROM THE TTY
2719      *      ADDRESS OF FIRST CHARACTER WILL BE ON THE STACK
2720      *      TERMINATOR WILL BE A BYTE OF ALL 0'S
2721 014062 010346      $RDLIN: MOV      R3,-(SP)      ;; SAVE R3
2722 014064 012703 014170 1$: MOV      #TTYIN,R3      ;; GET ADDRESS
2723 014070 022703 014200 2$: CMP      #TTYIN+8.,R3      ;; BUFFER FULL?
2724 014074 101405      BLOS      4$              ;; BR IF YES
2725 014076 104407      RDCHR      (SP)+,(R3)      ;; GO READ ONE CHARACTER FROM THE TTY
2726 014100 112613      MOV      (SP)+,(R3)      ;; GET CHARACTER
2727 014102 122713 000177 10$: CMP      #177,(R3)      ;; IS IT A RUBOUT
2728 014106 001003      BNE      3$              ;; SKIP IF NOT
2729 014110 104401 001200 4$: TYPE      ,SQUES      ;; TYPE A '?'
2730 014114 000763      BR      1$              ;; CLEAR THE BUFFER AND LOOP
2731 014116 111367 000044 3$: MOV      (R3),9$      ;; ECHO THE CHARACTER
2732 014122 104401 014166      TYPE      ,9$
2733 014126 122723 000015      CMP      #15,(R3)+      ;; CHECK FOR RETURN
2734 014132 001356      BNE      2$              ;; LOOP IF NOT RETURN
2735 014134 105063 177777      CLRB      -1(R3)      ;; CLEAR RETURN (THE 15)
2736 014140 104401 001202      TYPE      ,SLF      ;; TYPE A LINE FEED
2737 014144 012603      MOV      (SP)+,R3      ;; RESTORE R3
2738 014146 011646      MOV      (SP),-(SP)      ;; ADJUST THE STACK AND PUT ADDRESS OF THE
2739 014150 016666 000004 000002      MOV      4(SP),2(SP)      ;; FIRST ASCII CHARACTER ON IT
2740 014156 012766 014170 000004      MOV      #TTYIN,4(SP)
2741 014164 000002      RTI              ;; RETURN
2742 014166 000      9$: .BYTE      0      ;; STORAGE FOR ASCII CHAR. TO TYPE
2743 014167 000      .BYTE      0      ;; TERMINATOR
2744 014170 000010      $TTYIN: .BLKB      8.      ;; RESERVE 9 BYTES FOR TTY INPUT
2745 014200 052536 005015 000      $CNTLU: .ASCIZ /↑U/<15><12>      ;; CONTROL "U"
2746 014205 136 006507 000012      $CNTLG: .ASCIZ /↑G/<15><12>      ;; CONTROL "G"
2747 014212 005015 053523 020122      $MSWR: .ASCIZ <15><12>/SWR = /
2748 014220 020075 000
2749 014223 040 047040 053505      $MNEW: .ASCIZ / NEW = /
2750 014230 036440 000040      .SBTTL READ AN OCTAL NUMBER FROM THE TTY
2751      *****
2752      *THIS ROUTINE WILL READ AN OCTAL (ASCII) NUMBER FROM THE TTY AND
2753      *CHANGE IT TO BINARY.
2754      *CALL:
2755      *      RDOCT
2756      *      RETURN HERE
2757      *      READ AN OCTAL NUMBER
2758      *      LOW ORDER BITS ARE ON TOP OF THE STACK
2759      *      HIGH ORDER BITS ARE IN $HIOCT
2760 014234 011646      $RDOCT: MOV      (SP),-(SP)      ;; PROVIDE SPACE FOR THE
2761 014236 016666 000004 000002      MOV      4(SP),2(SP)      ;; INPUT NUMBER
2762 014244 010046      MOV      R0,-(SP)      ;; PUSH R0 ON STACK
2763 014246 010146      MOV      R1,-(SP)      ;; PUSH R1 ON STACK
2764 014250 010246      MOV      R2,-(SP)      ;; PUSH R2 ON STACK

```

E07

TA11 MOTION TEST
DZTADC.NEW

24-MAY-76 00:00

MAINDEC-11-DZTAD-C

MACY1: 27(1006) 23-SEP-76 10:33 PAGE 66
READ AN OCTAL NUMBER FROM THE TTY

SEQ 0082

2764	014252	104410	1\$:	RD LIN	:: READ AN ASCII LINE
2765	014254	012600		MOV (SP)+, R0	:: GET ADDRESS OF 1ST CHARACTER
2766	014256	005001		CLR R1	:: CLEAR DATA WORD
2767	014260	005002		CLR R2	
2768	014262	112046	2\$:	MOVB (R0)+, -(SP)	:: PICKUP THIS CHARACTER
2769	014264	001412		BEQ 3\$	:: IF ZERO GET OUT
2770	014266	006301		ASL R1	:: *2
2771	014270	006102		ROL R2	
2772	014272	006301		ASL R1	:: *4
2773	014274	006102		ROL R2	
2774	014276	006301		ASL R1	:: *8
2775	014300	006102		ROL R2	
2776	014302	042716	177770	BIC #1C7, (SP)	:: STRIP THE ASCII JUNK
2777	014306	062601		ADD (SP)+, R1	:: ADD IN THIS DIGIT
2778	014310	000764		BR 2\$	:: LOOP
2779	014312	005726	3\$:	TST (SP)+	:: CLEAN TERMINATOR FROM STACK
2780	014314	010166	000012	MOV R1, 12(SP)	:: SAVE THE RESULT
2781	014320	010267	000010	MOV R2, \$HIOCT	
2782	014324	012602		MOV (SP)+, R2	:: POP STACK INTO R2
2783	014326	012601		MOV (SP)+, R1	:: POP STACK INTO R1
2784	014330	012600		MOV (SP)+, R0	:: POP STACK INTO R0
2785	014332	000002		RTI	:: RETURN
2786	014334	000000		\$HIOCT: .WORD 0	:: HIGH ORDER BITS GO HERE

F07

TA11 MOTION TEST
DZTADC.NEWMAINDEC-11-DZTAD-C
24-MAY-76 00:00MACY11 27(1006) 23-SEP-76 10:33 PAGE 67
BINARY TO OCTAL (ASCII) AND TYPE

SEQ 0083

.SBTTL BINARY TO OCTAL (ASCII) AND TYPE

```

*****
*THIS ROUTINE IS USED TO CHANGE A 16-BIT BINARY NUMBER TO A 6-DIGIT
*OCTAL (ASCII) NUMBER AND TYPE IT.
*$TYPOS---ENTER HERE TO SETUP SUPPRESS ZEROS AND NUMBER OF DIGITS TO TYPE
*CALL:

```

```

*      MOV      NUM, -(SP)      ;;NUMBER TO BE TYPED
*      TYPOS      ;;CALL FOR TYPEOUT
*      .BYTE      N      ;;N=1 TO 6 FOR NUMBER OF DIGITS TO TYPE
*      .BYTE      M      ;;M=1 OR 0
*                               ;;1=TYPE LEADING ZEROS
*                               ;;0=SUPPRESS LEADING ZEROS

```

```

*$TYPON---ENTER HERE TO TYPE OUT WITH THE SAME PARAMETERS AS THE LAST
*$TYPOS OR $TYPOC

```

```

*CALL:
*      MOV      NUM, -(SP)      ;;NUMBER TO BE TYPED
*      TYPON      ;;CALL FOR TYPEOUT

```

```

*$TYPOC---ENTER HERE FOR TYPEOUT OF A 16 BIT NUMBER

```

```

*CALL:
*      MOV      NUM, -(SP)      ;;NUMBER TO BE TYPED
*      TYPOC      ;;CALL FOR TYPEOUT

```

2812	014336	017646	000000		\$TYPOS: MOV	2(SP), -(SP)	;; PICKUP THE MODE
2813	014342	116667	000001	000211	MOVB	1(SP), \$OFILL	;; LOAD ZERO FILL SWITCH
2814	014350	112667	000207		MOVB	(SP)+, \$OMODE+1	;; NUMBER OF DIGITS TO TYPE
2815	014354	062716	000002		ADD	#2, (SP)	;; ADJUST RETURN ADDRESS
2816	014360	000406			BR	\$TYPON	
2817	014362	112767	000001	000171	\$TYPOC: MOVB	#1, \$OFILL	;; SET THE ZERO FILL SWITCH
2818	014370	112767	000006	000165	MOVB	#6, \$OMODE+1	;; SET FOR SIX(6) DIGITS
2819	014376	112767	000005	000154	\$TYPON: MOVB	#5, \$OCNT	;; SET THE ITERATION COUNT
2820	014404	010346			MOV	R3, -(SP)	;; SAVE R3
2821	014406	010446			MOV	R4, -(SP)	;; SAVE R4
2822	014410	010546			MOV	R5, -(SP)	;; SAVE R5
2823	014412	116704	000145		MOVB	\$OMODE+1, R4	;; GET THE NUMBER OF DIGITS TO TYPE
2824	014416	005404			NEG	R4	
2825	014420	062704	000006		ADD	#6, R4	;; SUBTRACT IT FOR MAX. ALLOWED
2826	014424	110467	000132		MOVB	R4, \$OMODE	;; SAVE IT FOR USE
2827	014430	116704	000125		MOVB	\$OFILL, R4	;; GET THE ZERO FILL SWITCH
2828	014434	016605	000012		MOV	12(SP), R5	;; PICKUP THE INPUT NUMBER
2829	014440	005003			CLR	R3	;; CLEAR THE OUTPUT WORD
2830	014442	006105		1\$:	ROL	R5	;; ROTATE MSB INTO "C"
2831	014444	000404			BR	3\$	;; GO DO MSB
2832	014446	006105		2\$:	ROL	R5	;; FORM THIS DIGIT
2833	014450	006105			ROL	R5	
2834	014452	006105			ROL	R5	
2835	014454	010503			MOV	R5, R3	
2836	014456	006103		3\$:	ROL	R3	;; GET LSB OF THIS DIGIT
2837	014460	105367	000076		DECB	\$OMODE	;; TYPE THIS DIGIT?
2838	014464	100016			BPL	7\$	;; BR IF NO
2839	014466	042703	177770		BIC	#177770, R3	;; GET RID OF JUNK
2840	014472	001002			BNE	4\$	;; TEST FOR 0
2841	014474	005704			TST	R4	;; SUPPRESS THIS 0?
2842	014476	001403			BEQ	5\$	;; BR IF YES

G07

TA11 MOTION TEST MAINDEC-11-DZTAD-C MACY11 27(1006) 23-SEP-76 10:33 PAGE 68
 DZTADC.NEW 24-MAY-76 00:00 BINARY TO OCTAL (ASCII) AND TYPE

SEQ 0084

2843	014500	005204		4\$:	INC	R4	:: DON'T SUPPRESS ANYMORE 0'S
2844	014502	052703	000060		BIS	#'0,R3	:: MAKE THIS DIGIT ASCII
2845	014506	052703	000040	5\$:	BIS	#',R3	:: MAKE ASCII IF NOT ALREADY
2846	014512	110367	000040		MOVB	R3,8\$	:: SAVE FOR TYPING
2847	014516	104401	014556		TYPE	8\$	:: GO TYPE THIS DIGIT
2848	014522	105367	000032	7\$:	DECB	\$OCNT	:: COUNT BY 1
2849	014526	003347			BGT	2\$	:: BR IF MORE TO DO
2850	014530	002402			BLT	6\$	:: BR IF DONE
2851	014532	005204			INC	R4	:: INSURE LAST DIGIT ISN'T A BLANK
2852	014534	000744			BR	2\$	:: GO DO THE LAST DIGIT
2853	014536	012605		6\$:	MOV	(SP)+,R5	:: RESTORE R5
2854	014540	012604			MOV	(SP)+,R4	:: RESTORE R4
2855	014542	012603			MOV	(SP)+,R3	:: RESTORE R3
2856	014544	016666	000002 000004		MOV	2(SP),4(SP)	:: SET THE STACK FOR RETURNING
2857	014552	012616			MOV	(SP)+,(SP)	
2858	014554	000002			RTI		:: RETURN
2859	014556	000		8\$:	.BYTE	0	:: STORAGE FOR ASCII DIGIT
2860	014557	000			.BYTE	0	:: TERMINATOR FOR TYPE ROUTINE
2861	014560	000		\$OCNT:	.BYTE	0	:: OCTAL DIGIT COUNTER
2862	014561	000		\$OFILL:	.BYTE	0	:: ZERO FILL SWITCH
2863	014562	000000		\$CMODE:	.WORD	0	:: NUMBER OF DIGITS TO TYPE

.SBTTL TRAP DECODER

```

*****
*THIS ROUTINE WILL PICKUP THE LOWER BYTE OF THE "TRAP" INSTRUCTION
*AND USE IT TO INDEX THROUGH THE TRAP TABLE FOR THE STARTING ADDRESS
*OF THE DESIRED ROUTINE. THEN USING THE ADDRESS OBTAINED IT WILL
*GO TO THAT ROUTINE.

```

```

$TRAP:  MOV    RO, -(SP)          ;;SAVE RO
        MOV    2(SP), RO        ;;GET TRAP ADDRESS
        TST    -(RO)            ;;BACKUP BY 2
        MOVB   (RO), RO         ;;GET RIGHT BYTE OF TRAP
        ASL    RO               ;;POSITION FOR INDEXING
        MOV    $TRPAD(RO), RO    ;;INDEX TO TABLE
        RTS    RO               ;;GO TO ROUTINE

```

```

;;THIS IS USE TO HANDLE THE "GETPRI" MACRO

```

```

$TRAP2: MOV    (SP), -(SP)        ;;MOVE THE PC DOWN
        MOV    4(SP), 2(SP)      ;;MOVE THE PSW DOWN
        RTI                      ;;RESTORE THE PSW

```

.SBTTL TRAP TABLE

```

*THIS TABLE CONTAINS THE STARTING ADDRESSES OF THE ROUTINES CALLED
*BY THE "TRAP" INSTRUCTION.

```

	ROUTINE		
\$TRPAD:	.WORD	\$TRAP2	
	\$TYPE	;;CALL=TYPE	TRAP+1(104401) TTY TYPEOUT ROUTINE
	\$TYPOC	;;CALL=TYPOC	TRAP+2(104402) TYPE OCTAL NUMBER (WITH LEADING ZEROS)
	\$TYPOS	;;CALL=TYPOS	TRAP+3(104403) TYPE OCTAL NUMBER (NO LEADING ZEROS)
	\$TYPON	;;CALL=TYPON	TRAP+4(104404) TYPE OCTAL NUMBER (AS PER LAST CALL)
	\$GTSWR	;;CALL=GTSWR	TRAP+5(104405) GET SOFT-SWR SETTING
	\$CKSWR	;;CALL=CKSWR	TRAP+6(104406) TEST FOR CHANGE IN SOFT-SWR
	\$RDCHR	;;CALL=RDCHR	TRAP+7(104407) TTY TYPEIN CHARACTER ROUTINE
	\$RDLIN	;;CALL=RDLIN	TRAP+10(104410) TTY TYPEIN STRING ROUTINE
	\$RDOCT	;;CALL=RDOCT	TRAP+11(104411) READ AN OCTAL NUMBER FROM TTY
	T.WFG	;;CALL=WFG	TRAP+12(104412) ROUTINE TO WRITE A FILE GAP
	T.WRIT	;;CALL=WRITE	TRAP+13(104413) ROUTINE TO WRITE A BLOCK OF DATA
	T.READ	;;CALL=READ	TRAP+14(104414) ROUTINE TO READ A BLOCK OF DATA
	T.BSFG	;;CALL=BSFG	TRAP+15(104415) ROUTINE TO BACK SPACE A FILE GAP
	T.BSBG	;;CALL=BSBG	TRAP+16(104416) ROUTINE TO BACK SPACE A BLOCK GAP
	T.SFFG	;;CALL=SFFG	TRAP+17(104417) ROUTINE TO SPACE FWD A FILE GAP
	T.SFBG	;;CALL=SFBG	TRAP+20(104420) ROUTINE TO SPACE FWD A BLOCK GAP
	T.RWND	;;CALL=REWIND	TRAP+21(104421) ROUTINE TO REWIND THE TAPE
	T.SELDRV	;;CALL=SELDRV	TRAP+22(104422) ROUTINE TO SELECT THE NEXT DRIVE
	T.BLKCMP	;;CALL=BLKCMP	TRAP+23(104423) COMPARE READ AND WRITE BUFFERS
	T.SETLOOP	;;CALL=SETLOOP	TRAP+24(104424) SETUP TO LOOP AS PER THE TTY

.SBTTL POWER DOWN AND UP ROUTINES

;*****

:POWER DOWN ROUTINE

```

$PWRDN: MOV    $SILLUP, $PWRVEC    ;;SET FOR FAST UP
        MOV    #340, $PWRVEC+2    ;;PRIO:7
        MOV    R0, -(SP)           ;;PUSH R0 ON STACK
        MOV    R1, -(SP)           ;;PUSH R1 ON STACK
        MOV    R2, -(SP)           ;;PUSH R2 ON STACK
        MOV    R3, -(SP)           ;;PUSH R3 ON STACK
        MOV    R4, -(SP)           ;;PUSH R4 ON STACK
        MOV    R5, -(SP)           ;;PUSH R5 ON STACK
        MOV    $SWR, -(SP)         ;;PUSH $SWR ON STACK
        MOV    SP, $SAVR6          ;;SAVE SP
        MOV    $PWRUP, $PWRVEC    ;;SET UP VECTOR
        HALT
        BR      -2                ;;HANG UP

```

;*****

:POWER UP ROUTINE

```

$PWRUP: MOV    $SILLUP, $PWRVEC    ;;SET FOR FAST DOWN
        MOV    $SAVR6, SP          ;;GET SP
        CLR    $SAVR6             ;;WAIT LOOP FOR THE TTY
        1$: INC    $SAVR6          ;;WAIT FOR THE INC
        BNE    1$                 ;;OF WORD
        MOV    (SP)+, $SWR         ;;POP STACK INTO $SWR
        MOV    (SP)+, R5           ;;POP STACK INTO R5
        MOV    (SP)+, R4           ;;POP STACK INTO R4
        MOV    (SP)+, R3           ;;POP STACK INTO R3
        MOV    (SP)+, R2           ;;POP STACK INTO R2
        MOV    (SP)+, R1           ;;POP STACK INTO R1
        MOV    (SP)+, R0           ;;POP STACK INTO R0
        MOV    $PWRDN, $PWRVEC    ;;SET UP THE POWER DOWN VECTOR
        MOV    #340, $PWRVEC+2    ;;PRIO:7
        TYPE    $POWER             ;;REPORT THE POWER FAILURE
        $PWRMG: .WORD $POWER       ;;POWER FAIL MESSAGE POINTER
        MOV    (PC)+, (SP)         ;;RESTART AT PWRST
        $PWRAD: .WORD PWRST        ;;RESTART ADDRESS
        RTI
        $ILLUP: HALT
        BR      -2                ;;THE POWER UP SEQUENCE WAS STARTED
        $SAVR6: 0                  ;;BEFORE THE POWER DOWN WAS COMPLETE
        $POWER: .ASCIZ <15><12>"POWER" ;;PUT THE SP HERE
        .EVEN

```


J07

TA11 MOTION TEST
DZTADC.NEW 24-MAY-76 00:00

MAINDEC-11-DZTAD-C

MACY11 27(1006) 23-SEP-76 10:33 PAGE 71
DATA TABLE POINTERS FOR ERRORS

SEQ 0087

```
2962 ;*****
2963
2964 ;DATA POINTERS
2965
2966 015054 001204 001116 001162 DT1: .WORD SAVPC,$ERRPC,$REG0,$REG1,0
2967 015062 001164 000000
2968
2969 015066 001204 001116 001162 DT2: .WORD SAVPC,$ERRPC,$REG0,$GDDAT,$BDDAT,BYTNUM,0
2970 015074 001124 001126 001206
2971 015102 000000
2972
2973 015104 001204 001116 001162 DT3: .WORD SAVPC,$ERRPC,$REG0,RCRC0,RCRC1,RCRC2,0
2974 015112 001210 001212 001214
2975 015120 000000
2976
2977 015122 001204 001116 001162 DT4: .WORD SAVPC,$ERRPC,$REG0,$REG1,$TMP0,0
2978 015130 001164 001166 000000
2979
2980 015136 001204 001116 001162 DT5: .WORD SAVPC,$ERRPC,$REG0,0
2981 015144 000000
2982
```

K07

TA11 MOTION TEST
DZTADC.NEW 24-MAY-76 00:00

MAINDEC-11-DZTAD-C

MACY11 27(1006) 23-SEP-76 10:33 PAGE 72
ASCII MESSAGES

SEQ 0088

```

2983      ;*****
2984
2985      ;MESSAGES
2986
2987      015146 042200 044522 042526 MSGDRV: .ASCIZ <CRLF>"DRIVE(S)? "
2988      015154 051450 037451 000040
2989      015162 005015 040524 051503 MSGASK: .ASCIZ <15><12>/TACS?/
2990      015170 000077
2991      015172 042526 052103 051117 MSGVEC: .ASCIZ /VECTOR?/
2992      015200 000077
2993      015202 051120 047511 044522 MSGPRI: .ASCIZ /PRIORITY?/
2994      015210 054524 000077
2995      015214 052040 041501 036523 MTACS: .ASCIZ / TACS=/
2996      015222 000
2997      015223 040 040524 041104 MTADB: .ASCIZ / TADB=/
2998      015230 000075
2999      015232 053040 041505 047524 MTAVEC: .ASCIZ / VECTOR=/
3000      015240 036522 000
3001      015243 040 051120 047511 MTAPRI: .ASCIZ / PRIORITY=/
3002      015250 044522 054524 000075
3003      015256 005015 045517 000077 MSGOK: .ASCIZ <15><12>/OK?/
3004      015264 042531 100123 000 MYES: .ASCIZ /YES/<CRLF>
3005      015271 015 052012 051505 MTSTDRV: .ASCIZ <15><12>/TESTING DRIVE /
3006      015276 044524 043516 042040
3007      015304 044522 042526 000040
3008      015312 040440 042116 042040 MANDRV: .ASCIZ / AND DRIVE /
3009      015320 044522 042526 000040
3010      015326 005015 042524 052123 MTEST: .ASCIZ <15><12>/TEST PC?/
3011      015334 050040 037503 000
3012      015341 015 046012 047517 MLOOP: .ASCIZ <15><12>/LOOP PC?/
3013      015346 020120 041520 000077
3014      015354 042522 044527 042116 MXRWND: .ASCIZ /REWIND/
3015      015362 000
3016      015363 127 044522 042524 MXWRIT: .ASCIZ /WRITE/
3017      015370 000
3018      015371 122 040505 000104 MXREAD: .ASCIZ /READ/
3019      015376 050123 041501 026505 MXSFFG: .ASCIZ /SPACE-FWD-FILE-GAP/
3020      015404 053506 026504 044506
3021      015412 042514 043455 050101
3022      015420 000
3023      015421 123 040520 042503 MXSFBG: .ASCIZ /SPACE-FWD-BLK-GAP/
3024      015426 043055 042127 041055
3025      015434 045514 043455 050101
3026      015442 000
3027      015443 102 041501 026513 MXBSFG: .ASCIZ /BACK-SPACE-FILE-GAP/
3028      015450 050123 041501 026505
3029      015456 044506 042514 043455
3030      015464 050101 000
3031      015467 102 041501 026513 MXBSBG: .ASCIZ /BACK-SPACE-BLK-GAP/
3032      015474 050123 041501 026505
3033      015502 046102 026513 040507
3034      015510 000120
3035      015512 051127 052111 026505 MXWFG: .ASCIZ /WRITE-FILE-GAP/
3036      015520 044506 042514 043455
3037      015526 050101 000
3038      015531 102 043125 042506 MBUFCK: .ASCIZ /BUFFER COMPARE/

```


L07

TA11 MOTION TEST
DZTADC.NEW 24-MAY-76 00:00

MAINDEC-11-DZTAD-C

MACY11 27(1006) 23-SEP-76 10:33 PAGE 73
ASCII MESSAGES

SEQ 0089

3039	015536	020122	047503	050115		
3040	015544	051101	000105			
3041	015550	046511	051120	050117	EM1:	.ASCIZ /IMPROPER FLAG SETTING/
3042	015556	051105	043040	040514		
3043	015564	020107	042523	052124		
3044	015572	047111	000107			
3045	015576	046511	051120	050117	EM2:	.ASCIZ /IMPROPER FLAG OCCURRED/
3046	015604	051105	043040	040514		
3047	015612	020107	041517	052503		
3048	015620	051122	042105	000		
3049	015625	115	051511	042523	EM3:	.ASCIZ /MISSED A FLAG/
3050	015632	020104	020101	046106		
3051	015640	043501	000			
3052	015643	111	052116	051105	EM4:	.ASCIZ /INTERRUPT FAILED/
3053	015650	052522	052120	043040		
3054	015656	044501	042514	000104		
3055	015664	051120	046505	052101	EM5:	.ASCIZ /PREMATURE READY OCCURRED/
3056	015672	051125	020105	042522		
3057	015700	042101	020131	041517		
3058	015706	052503	051122	042105		
3059	015714	000				
3060	015715	104	042111	023516	EM6:	.ASCIZ /DIDN'T STOP IN A FILE GAP/
3061	015722	020124	052123	050117		
3062	015730	044440	020116	020101		
3063	015736	044506	042514	043440		
3064	015744	050101	000			
3065	015747	104	042111	023516	EM7:	.ASCIZ /DIDN'T STOP ON CLEAR LEADER/
3066	015754	020124	052123	050117		
3067	015762	047440	020116	046103		
3068	015770	040505	020122	042514		
3069	015776	042101	051105	000		
3070	016003	102	042101	042040	EM10:	.ASCIZ /BAD DATA READ/
3071	016010	052101	020101	042522		
3072	016016	042101	000			
3073	016021	111	046114	043505	EM11:	.ASCIZ /ILLEGAL BUFFER SIZE/
3074	016026	046101	041040	043125		
3075	016034	042506	020122	044523		
3076	016042	042532	000			
3077	016045	103	041522	042440	EM12:	.ASCIZ /CRC ERROR/
3078	016052	051122	051117	000		
3079	016057	123	047510	052122	EM13:	.ASCIZ /SHORT RECORD/
3080	016064	051040	041505	051117		
3081	016072	000104				
3082	016074	040502	020104	047111	EM14:	.ASCIZ /BAD INTERRUPT/
3083	016102	042524	051122	050125		
3084	016110	000124				
3085	016112	005015			DH1:	.ASCII <15><12>
3086	016114	042524	052123	020040		.ASCII /TEST ERROR/<15><12>
3087	016122	020040	051105	047522		
3088	016130	006522	012			
3089	016133	120	020103	020040		.ASCIZ /PC PC TACS TADB/<15><12>
3090	016140	020040	050040	020103		
3091	016146	020040	020040	052040		
3092	016154	041501	020123	020040		
3093	016162	052040	042101	006502		
3094	016170	000012				

M07

TA11 MOTION TEST
DZTADC.NEW

24-MAY-76

MAINDEC-11-DZTAD-C
00:00MACY11 27(1006)
ASCII MESSAGES

23-SEP-76 10:33 PAGE 74

SEQ 0090

3095	016172	005015		
3096	016174	042524	052123	020040
3097	016202	020040	051105	047522
3098	016210	020122	020040	020040
3099	016216	020040	020040	020040
3100	016224	054105	052120	042047
3101	016232	020040	041522	023526
3102	016240	020104	020040	054502
3103	016246	042524	005015	
3104	016252	041520	020040	020040
3105	016260	020040	041520	020040
3106	016266	020040	020040	040524
3107	016274	051503	020040	020040
3108	016302	040504	040524	020040
3109	016310	020040	040504	040524
3110	016316	020040	020040	052516
3111	016324	041115	051105	005015
3112	016332	000		
3113	016333	015	012	
3114	016335	124	051505	020124
3115	016342	020040	042440	051122
3116	016350	051117	020040	020040
3117	016356	020040	020040	020040
3118	016364	053440	044522	042524
3119	016372	020040	051040	040505
3120	016400	020104	020040	052040
3121	016406	033125	006460	012
3122	016413	120	020103	020040
3123	016420	020040	050040	020103
3124	016426	020040	020040	052040
3125	016434	041501	020123	020040
3126	016442	041440	041522	020040
3127	016450	020040	041440	041522
3128	016456	020040	020040	041440
3129	016464	041522	005015	000
3130	016471	015	012	
3131	016473	124	051505	020124
3132	016500	020040	042440	051122
3133	016506	051117	020040	020040
3134	016514	020040	020040	020040
3135	016522	020040	020040	020040
3136	016530	020040	041040	052131
3137	016536	051505	005015	
3138	016542	041520	020040	020040
3139	016550	020040	041520	020040
3140	016556	020040	020040	040524
3141	016564	051503	020040	020040
3142	016572	040524	041104	020040
3143	016600	020040	042514	052106
3144	016606	005015	000	
3145	016611	015	012	
3146	016613	124	051505	020124
3147	016620	020040	042440	051122
3148	016626	051117	005015	
3149	016632	041520	020040	020040
3150	016640	020040	041520	020040

DH2:

.ASCII <15><12>
.ASCII /TEST

ERROR

EXPT'D RCV'D BYTE/<15><12>

.ASCIZ /PC

PC

TACS

DATA

DATA

NUMBER/<15><12>

DH3:

.ASCII <15><12>
.ASCII /TEST

ERROR

WRITE

READ

TU60/<15><12>

.ASCIZ /PC

PC

TACS

CRC

CRC

CRC/<15><12>

DH4:

.ASCII <15><12>
.ASCII /TEST

ERROR

BYTES/<15><12>

.ASCIZ /PC

PC

TACS

TADB

LEFT/<15><12>

DH5:

.ASCII <15><12>
.ASCII /TEST

ERROR/<15><12>

.ASCIZ /PC

PC

TACS/<15><12>

NO7

TA11 MOTION TEST
DZTADC.NEW 24-MAY-76

MAINDEC-11-DZTAD-C
00:00

MACY11 27(1006)
ASCII MESSAGES

23-SEP-76 10:33 PAGE 75

SEQ 0091

3151	016646	020040	020040	040524		
3152	016654	051503	005015	000		
3153		016662				
3154	016662	001116	001220	000000	DT201:	.EVEN .WORD \$ERRPC,TACSL,0
3155						
3156	016670	001116	000000		DT202:	.WORD \$ERRPC,0
3157						
3158	016674	040524	030461	043040	EM201:	.ASCIZ "TA11 FAILED TO RESPOND"
3159	016702	044501	042514	020104		
3160	016710	047524	051040	051505		
3161	016716	047520	042116	000		
3162	016723	116	020117	051104	EM202:	.ASCIZ "NO DRIVE AVAILABLE"
3163	016730	053111	020105	053101		
3164	016736	044501	040514	046102		
3165	016744	000105				
3166	016746	041520	020040	020040	DH201:	.ASCIZ /PC TACS/
3167	016754	020040	040524	051503		
3168	016762	000				
3169	016763	120	000103		DH202:	.ASCIZ /PC/ .EVEN
3170						

;BUFFER LINKS
;THESE LINKS POINT TO THE STARTING ADDRESS OF THE BUFFERS
;AND ALSO INDICATE HOW MANY BYTES TO READ OR WRITE

016766	017130	000200
016772	017332	000200
016776	017534	000010
017002	017736	000040
017006	020140	000020
017012	020342	000100
017016	017130	000000
017022	017056	000200
017026	017064	000200
017032	017072	000010
017036	017100	000040
017042	017106	000020
017046	017114	000100
017052	017122	000000

RLNK1:	RBUF1,128.
RLNK2:	RBUF2,128.
RLNK3:	RBUF3,8.
RLNK4:	RBUF4,32.
RLNK5:	RBUF5,16.
RLNK6:	RBUF6,64.
RLNKX:	RBUF1,0
WLNK1:	WBUF1,128.
WLNK2:	WBUF2,128.
WLNK3:	WBUF3,8.
WLNK4:	WBUF4,32.
WLNK5:	WBUF5,16.
WLNK6:	WBUF6,64.
WLNKX:	WBUF1,0

;BUFFERS

017056	000002
017056	314
017057	063
017060	314
017061	063
017062	000000

WBUF1:		
	.BYTE	↑B11001100
	.BYTE	↑B00110011
	.BYTE	↑B11001100
	.BYTE	↑B00110011
WCRC1:	.WORD	0

;CRC STORAGE FOR WBUF1

017064	000002
017064	377
017065	000
017066	377
017067	000
017070	000000

WBUF2:		
	.BYTE	↑B11111111
	.BYTE	↑B00000000
	.BYTE	↑B11111111
	.BYTE	↑B00000000
WCRC2:	.WORD	0

;CRC STORAGE FOR WBUF2

017072	000002
017072	231
017073	146
017074	231
017075	146
017076	000000

WBUF3:		
	.BYTE	↑B10011001
	.BYTE	↑B01100110
	.BYTE	↑B10011001
	.BYTE	↑B01100110
WCRC3:	.WORD	0

;CRC STORAGE FOR WBUF3

017100	000002
017100	360
017101	017
017102	360
017103	017
017104	000000

WBUF4:		
	.BYTE	↑B11110000
	.BYTE	↑B00001111
	.BYTE	↑B11110000
	.BYTE	↑B00001111
WCRC4:	.WORD	0

;CRC STORAGE FOR WBUF4

017106	000002
017106	252
017107	125
017110	252
017111	125
017112	000000

WBUF5:		
	.BYTE	↑B10101010
	.BYTE	↑B01010101
	.BYTE	↑B10101010
	.BYTE	↑B01010101
WCRC5:	.WORD	0

;CRC STORAGE FOR WBUF5

TA11 MOTION TEST
DZTADC.NEW 24-MAY-76 00:00

MAINDEC-11-DZTAD-C

MACY11 27(1006) 23-SEP-76 10:33 PAGE 77
READ AND WRITE BUFFERS AND LINKS

SEQ 0093

017114 000001
017115 377
017116 377
017117 000
017118 000
017120 000000
017122 000004
017123 000
017124 000
017125 000
017126 000000
017130 000202
017332 000202
017534 000202

WBUF6:

.BYTE 1811111111
.BYTE 1811111111
.BYTE 1800000000
.BYTE 1800000000

WCRC6: .WORD 0

;CRC STORAGE FOR WBUF6

WBUFx:

.BYTE 0
.BYTE 0000
.BYTE 0000
.BYTE 0000

WCRCx: .WORD 0

RBUF1: .BLKB 130.
RBUF2: .BLKB 130.
RBUF3: .BLKB 130.

D08

TA11 MOTION TEST
DZTADC.NEW 24-MAY-76 00:00

MACY11 27(1006) 23-SEP-76 10:33 PAGE 78
READ AND WRITE BUFFERS AND LINKS

SEQ 0094

017736 000202
020140 000202
020342 000202
020544
000001

RBUF4: .BLKB 130.
RBUF5: .BLKB 130.
RBUF6: .BLKB 130.
LASTADDRESS=
.END

TAIL MOTION TEST
DZTADC.NEW 24

MAINDEC-11-DZTAD-C

MACY11 27(1006)

23-SEP-76

10:33

PAGE 80

DZTADC.NEW 24-MAY-76 00:00

CROSS REFERENCE TABLE -- USER SYMBOLS

[illegible]

F08

TA11 MOTION TEST

MAINDEC-11-DZTAD-C

MACY11 27(1006)

23-SEP-76 10:33 PAGE 81

DZTADC.NEW 24-MAY-76 00:00

CROSS REFERENCE TABLE -- USER SYMBOLS

SEQ 0096

CRC.WD	012232	1958*	1986	2011*	2044	2244	2265*	2272*										
CRLF =	000200	97*	511	2571	2610	2987	3004											
CURDRV	001250	313*	632*	634	641*	544*	646	699	1889*									
DDISP =	177570	93*	275	482														
DH1	016112	338	345	352	359	366	373	390	3085*									
DH2	016172	387	3095*															
DH201	016746	423	3166*															
DH202	016763	428	3169*															
DH3	016333	401	3113*															
DH4	016471	408	3130*															
DH5	016611	394	415	3145*														
DISPLA	001142	275*	482*	490*	1607*	1643*												
DISPRE	000174	231*	490															
DO.CMD	012234	1940	1963	1987	2012	2045	2069	2090	2105	2126	2154	2281*						
DO.CRC	012112	1981	2031	2237*														
DRIVE	001236	309*	629*	1888	1901*													
DRVKEY	001240	310*	311	443*	585	598	619	637	643	681	1717*	1722	1736	1740				
		1900																
DRVPT	001244	311*	599*	619*	628	647*												
DSWR =	177570	92*	274	481														
DT1	015054	340	347	354	361	368	375	392	2966*									
DT2	015066	389	2969*															
DT201	016662	424	3154*															
DT202	016670	429	3156*															
DT3	015104	403	2973*															
DT4	015122	410	2977*															
DT5	015136	396	417	2980*														
EMTVEC=	000030	191*	465*	466*														
EM1	015550	337	3041*															
EM10	016003	386	3070*															
EM11	016021	393	3073*															
EM12	016045	400	3077*															
EM13	016057	407	3079*															
EM14	016074	414	3082*															
EM2	015576	344	3045*															
EM201	016674	422	3158*															
EM202	016723	427	3162*															
EM3	015625	351	3049*															
EM4	015643	358	3052*															
EM5	015664	365	3055*															
EM6	015715	372	3060*															
EM7	015747	379	3065*															
ERROR =	100000	199*																
ERRVEC=	000004	174*	479	480*	491*	547*	553*	1573	1574*	1576*	1579*							
EXAM	006474	586	590	595	1522*													
EXIT	007532	1735	1740*															
FATAL	002716	656	675	694*														
FGAP =	004000	203*	2057	2080	2116													
FLAGS	012024	1946	1971	1993	2021	2053	2075	2096	2111	2132	2144	2160	2211*					
FUNCTI=	000016	215*	2284	2299														
FUNCO =	000002	213*	215															
FUNC1 =	000004	212*	215															
FUNC2 =	000010	211*	215															
GNS =	***** U	230	510	697	2895	2896	2897	2898	2900	2902	2903	2904	2905	2906				
		2907	2908	2909	2910	2911	2912	2913	2914	2915	2916							
GO =	000001	214*	669	1941	1964	2013	2070	2091	2106	2127	2155	2284	2294	2343				

TA11 MOTION TEST MAINDEC-11-DZTAD-C MACY11 27(1006) 23-SEP-76 10:33 PAGE 82
DZTADC.NEW 24-MAY-76 00:00 CROSS REFERENCE TABLE -- USER SYMBOLS

	2371	2393	2425	2451	2481										
GTSWR = 104405	505	2513	2526	2537	2900#										
HERE 002052	496	498	509	511#											
HICNT 012020	662	2180*	2188*	2197#											
HT = 000011	84#	2569	2610												
ILBS = 000020	210#	1630	1988	2046	2284										
INT.EN= 000100	208#	669	1941	1964	1988	2013	2046	2070	2091	2106	2127	2155	2284		
IOTVEC= 000020	2287	2290													
ITEMS2 001430	179#	463*	464*												
LASTAD= 020544	420#														
LEADER= 020000	3250#														
	201#	1540	2057	2150	2165	2347	2375	2398	2429	2432	2456	2462	2485		
LF = 000012	2501														
LOCNT 012016	35#	2604	2610												
LOOP 007456	2179*	2186*	2196#												
LOOPKE 006624	1723#	1737													
LVFLAG 012110	626*	1566#	1874*	1876*											
MANDRV 015312	2212	2215	2229	2231#											
MANUAL 002274	645	3008#													
MAXCNT 012022	605	613#													
MBUFCK 015531	658*	662*	663*	664*	665*	691*	2180	2198#							
MFUNC 007314	1912	3038#													
MLOOP 015341	1676#	1912*	1939*	1954*	2001*	2068*	2089*	2104*	2125*	2143*					
MSGASK 015162	1861	3012#													
MSGDRV 015146	1749	2989#													
MSGID 002024	1716	2987#													
MSGOK 015256	499	511#	618												
MSGPRI 015202	1796	3003#													
MSGVEC 015172	1772	2993#													
MTACS 015214	1763	2991#													
MTADB 015223	1784	2995#													
MTAPRI 015243	1787	2997#													
MTAVEC 015232	1793	3001#													
MTEST 015326	1790	2999#													
MTSTDR 015271	1848	3010#													
MXBSBG 015467	633	3005#													
MXBSFG 015443	2089	3031#													
MXREAD 015371	2068	3027#													
MXRWND 015354	2001	3018#													
MXSFBG 015421	2143	3014#													
MXSFFG 015376	2125	3023#													
MXWFG 015512	2104	3019#													
MXWRIT 015363	1939	3035#													
MYES 015264	1954	3016#													
NEXT 007512	1805	3004#													
NOTA 007470	1726	1730	1734#												
NOTB 007502	1724	1727#													
NOTLGL 007524	1728	1731#													
OFFLIN= 001000	1721	1732	1738#	1741											
OK 002306	205#	1531	2057												
PIRQ = 177772	614	616#													
PIRQVE= 000240	91#														
PRO = 000000	185#														
PR1 = 000040	108#														
PR2 = 000100	109#														
	110#														

TA11 MOTION TEST MAINDEC-11-DZTAD-C MACY11 27(1006) 23-SEP-76 10:33 PAGE 83
DZTADC.NEW 24-MAY-76 00:00 CROSS REFERENCE TABLE -- USER SYMBOLS

[illegible]

1854*	1877	1879#										
655	683	710	723	736	749	767	785	800	819	834	849	864
880	895	945	990	1004	1018	1038	1058	1074	1091	1110	1130	1149
1169	1188	1204	1226	1246	1265	1283	1301	1326	1354	1397	1430	2342
2370	2392	2424	2450	2480	2499	2914#						
1984	2032	2292	2307#									
2364	2418	2509#										
1564	2916#											
789	854	870	885	1099	1119	1120	1137	1157	1176	1254	1270	1288
1291	1382	2461	2912#									
1064	1080	1098	1118	1136	1156	1175	1210	1216	1235	1236	1255	1310
1311	1312	1316	1338	1339	1364	1365	1387	1409	1410	1411	1415	1420
1445	1446	1447	1448	1449	1450	2484	2911#					
243	2475#	2479	2488									
79#	461	2337	2365	2387	2397	2419	2445	2455	2475			
318#	678*	679*	2221	2222								
617	620#	1508										
90#												
274#	459	481*	483	489*	503	635	1535	1537	1569	1586	1593	1644
1651	1656	1660	1885	1902	1920	2060	2181	2217	2285	2511	2515	2517
2524	2528	2535	2620	2657*	2929	2942*						
232#	489	503	2511	2524	2535	2620	2633					
143#												
133#	143											
132#	142											
131#	141											
130#	140	1920										
129#	139	2060										
128#	138	2181	2285									
127#	137	2217										
126#	136	635	1885									
125#	135											
124#	134	1902										
142#												
123#												
122#												
121#												
120#												
119#												
118#												
141#												
140#												
139#												
138#												
137#												
136#												
135#												
134#												
304#	1758*	1759*										
303#	549	623	1522	1755*	1758	1785	2338	2366	2388	2420	2446	2476
3154												
306#	1761*	1762*										
305#	624	1757*	1761									

[illegible]

TA11 MOTION TEST MAINDEC-11-DZTAD-C MACY11 27(1006) 23-SEP-76 10:33 PAGE 86
DZTADC.NEW 24-MAY-76 00:03 CROSS REFERENCE TABLE -- USER SYMBOLS

TAII MOTION TEST
DZTADC.NEW 24

MAINDEC-11-DZTAD-C
00:03 CR

MACY11 27(1006)

23-SEP-76

10:33 PAGE 86

DZTADC.NEW 24-MAY-76 00:03

CROSS REFERENCE TABLE -- USER SYMBOLS

SEQ 0101

UNIT	WAITFL	WAITKE	WBUF1	WBUF2	WBUF3	WBUF4	WBUF5	WBUF6	WCRC1	WCRC2	WCRC3	WCRC4	WCRC5	WCRC6	WFG	WFGSUB	WGBL	WGPBLK	WLNK1	WLNK2	WLNK3	WLNK4	WLNK5	WLNK6	WLOOP	WRITE	WRTLOC	WRTSUB																								
010256	010116	011530	011446	010514	010564	000400	011726	001254	017122	017056	017064	017072	017100	017106	017114	017126	017062	017070	017076	017104	017112	017120	104412	012374	012624	012620	017052	017022	017026	017032	017036	017042	017046	012454	104413	010000	012450															
1884#	1847#	2124#	2103#	1938#	1953#	206#	1943	314#	2529	3183	3184	3185	3186	3187	3188	3240#	3198#	3205#	3212#	3219#	3226#	3233#	661	238	2419#	241	2374	712	794	952	1093	1248	808	924	1104	1320	910	1289	917	967	1335	2365#	712	852	967	1095	1190	1285	1357	1438	202#	239
2914	1858	2912	2911	2906	2907	1526	1966	196#	3189	3193#	3200#	3207#	3214#	3221#	3228#								687	2337#	2423	2418#	2402	717	802	980	1112	1267	813	929	1114	1329	915	1295	922	972	1406	2369	725	866	974	1096	1192	1286	1358	1440	1533	2364#
2916						1892	1990	2019#	3235#													993	2341	2435		2431	725	807	992	1132	1285	822	939	1134	1357	974	1307	960	1334	1442	2378	738	867	992	1112	1206	1289	1360	1442	2057		
						1895	2016	2183#														1007	2350		2460	730	821	998	1143	1303	828	953	1153	1400	984	1331	965	1343	1455	751	882	1006	1114	1208	1303	1361	2374					
						2049	2192															1026			2522*	738	836	1006	1151	1328	837	958	1173	1434	1096	1358	1116	1348	3188#	769	883	1020	1115	1228	1305	1399	2431					
						2072	2289*															1046			3189#	743	843	1012	1163	1356	852	1027	1192	1434	1115	1386	1124	1361		787	897	1027	1116	1230	1307	1400	2907#					
						2093	2307*															1061				751	851	1020	1171	1376	858	1032	1192	1434	1154	1402	1332	1405		802	898	1040	1132	1232	1328	1402						
						2108																1077				756	866	1025	1182	1381	867	1047	1220	1419	1232	1402	1360	1424		808	910	1047	1134	1248	1329	1403						
						2129																1094				761	882	1040	1190	1399	874	1052	1230	1436	1240	1419	1370	1440		821	917	1060	1151	1249	1331	1405						
						2157																1113				769	897	1045	1198	1432	893	1062	1249	1468	1251																	

TA11 MOTION TEST MAINDEC-11-DZTAD-C MACY11 27(1006) 23-SEP-76 10:33 PAGE 87
DZTADC.NEW 24-MAY-76 00:00 CROSS REFERENCE TABLE -- USER SYMBOLS

[illegible]

TA11 MOTION TEST MAINDEC-11-DZTAD-C MACY11 27(1006) 23-SEP-76 10:33 PAGE 88
DZTADC.NEW 24-MAY-76 00:00 CROSS REFERENCE TABLE -- USER SYMBOLS

\$MNEW	014223	2635	2747#														
\$MSWR	014212	2632	2745#														
\$MXCNT	001216	302#	627*	639*	1601	1510											
\$NULL	001154	280#	2581	2610													
\$NWTST=	000C01	705#	718#	731#	744#	762#	780#	795#	814#	829#	844#	859#	875#	890#			
		940#	985#	999#	1013#	1033#	1053#	1069#	1086#	1105#	1125#	1144#	1164#	1183#			
		1199#	1221#	1241#	1260#	1278#	1296#	1321#	1349#	1392#	1425#						
\$OCNT	014560	2819#	2848*	2861#													
\$OMODE	014562	2814#	2818*	2823	2826*	2837*	2863#										
\$OVER	007042	1567	1569	1589	1599	1607#											
\$PASS	001100	255#	649	1490*	1491*	1509	1595	1610									
\$POWER	015044	2952	2959#														
\$PWRAD	015032	2954#															
\$PWDRN	014672	469	2921#	2949													
\$PWRMG	015026	2952#															
\$PWRUP	014744	2931	2937#														
\$QUES	001200	292#	1672	1738	1803	2610	2678	2727	2743								
\$RDCHR	013742	2691#	2903														
\$RDDEC=	***** U	2906															
\$RDLIN	014062	2719#	2904														
\$RDOCT	014234	2759#	2905														
\$RDSZ =	000010	2712#															
\$REGAD	001160	284#															
\$REGO	001162	286#	1626*	1635	2966	2969	2973	2977	2980								
\$REG1	001164	287#	1627*	2966	2977												
\$RTNAD	006454	1508#															
\$R2A =	***** U	2906															
\$SAVRE=	***** U	2906															
\$SAVR6	015042	2930*	2938	2939*	2940*	2958#											
\$SCOPE	006600	463	1557#														
\$SETUP=	000137	442#	462	463	465	467	469	471	472	" "	475	497	500	1488			
		1558	1625	1659	1667	2615	2749										
\$SS =	000000	705#															
\$STUP =	177777	442#															
\$SVLAD	007014	1577	1602#														
\$SWR =	167000	2#	12	62	63	64	65	66	67	68	289	290	291	472			
		473	475	476	709	722	735	748	766	794	799	818	833	848			
		863	879	894	944	989	1003	1017	1037	1057							

N08

TA11 MOTION TEST
DZTADC.NEW 24-MAY-76 00:00
MAINDEC-11-DZTAD-C
MACY11 27(1006) 23-SEP-76 10:33 PAGE 89
CROSS REFERENCE TABLE -- USER SYMBOLS

SEQ 0104

\$TRAP	014564	467	2872#															
\$TRAP2	014606	2883#	2894															
\$TRP =	000025	2887#	2896#	2897#	2898#	2899#	2900	2901#	2902	2903#	2904#	2905#	2906#	2907#				
		2908#	2909#	2910#	2911#	2912#	2913#	2914#	2915#	2916#	2917#							
\$TRPAD	014620	2877	2894#															
\$TSTNM	001102	256#	625*	1488*	1549	1602*	1607	1610	1643	1672								
\$TTYIN	014170	2720	2721	2738	2742#													
\$TYPBN=	***** U	2899																
\$TYPDS=	***** U	2899																
\$TYPE	013240	2557#	2887	2895														
\$TYPEC	013410	2578	2585	2592	2597#	2598	2664											
\$TYPEX	013456	2603	2605	2608#														
\$TYPOC	014362	2817#	2896															
\$TYPON	014376	2816	2819#	2898														
\$TYPOS	014336	2812#	2897															
\$XTSTR	006640	1571#																
\$SGET4=	000000	1501#																
\$OFILL	014561	2913*	2817*	2827	2862#													
\$40CAT=	***** U	1568	1653															
.	= 020544	226#	230#	253#	295	460	475	476	511#	1509	1512	1610	1672	1707#				
		2610	2613	2742#	2743	2749	2933	2957	3153#	3243#	3244#	3245#	3246#	3247#				
		3248#	3250															

. ABS. 020544 000

ERRORS DETECTED: 0
DEFAULT GLOBALS GENERATED: 0

TADC.TADC/SOL/CRF:SYM=DZTADC.NEW
RUN-TIME: 48 37 3 SECONDS
RUN-TIME RATIO: 185/90=2.0
CORE USED: 24K (47 PAGES)

Spooler runtime 14 Seconds, 63 KCS, 388 disk reads, 4 disk writes, 104 pages

***** Date 14-Oct-76 16:29:41 Monitor IPC-0 0078 (103) *****

[illegible]