

DMS11-DA

DMS11-DA DYNAMIC
CZKMEAO

AH-S886A-MC
FICHE 1 OF 1

APR 1982
COPYRIGHT © 1982
MADE IN USA



Table with multiple columns and rows of data, likely representing a dynamic system configuration or performance metrics. The data is organized in a grid format, with columns and rows of varying widths and heights, suggesting a complex data structure. The table is divided into several sections, each with its own header and data rows. The data appears to be organized into groups, with some sections having more detailed information than others. The overall layout is dense and structured, typical of a technical document or data report.



.NLIST SEQ,LOC,BIN
.REM !

IDENTIFICATION

PRODUCT CODE: AC-S884A-MC
PRODUCT NAME: CZKMEAO DMS11-DA DYNAMIC
PROGRAM DATE: SEPTEMBER 1981
MAINTAINER: CSS/NPG DIAGNOSTICS
AUTHOR: R. C. BALDWIN
R. J. COLLINS

COPYRIGHT (C) 1982 BY
DIGITAL EQUIPMENT CORPORATION,
MAYNARD, MASSACHUSETTS.
ALL RIGHTS RESERVED.

THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY
BE USED AND COPIED ONLY IN ACCORDANCE WITH THE
TERMS OF SUCH LICENSE AND WITH THE INCLUSION
OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR ANY OTHER
COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE
MADE AVAILABLE TO ANY OTHER PERSON. NO TITLE TO
AND OWNERSHIP OF THE SOFTWARE IS HEREBY TRANSFERRED.

THE INFORMATION IN THIS SOFTWARE IS SUBJECT TO CHANGE WITHOUT
NOTICE AND SHOULD NOT BE CONSTRUED AS A COMMITMENT BY DIGITAL
EQUIPMENT CORPORATION.

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

1.0 ABSTRACT

THE DMS11,-D,-A,-DA LINE UNIT DYNAMIC DIAGNOSTIC TESTS THE LINE UNIT FOR BOTH CCP & BOP MODES IN THE USYRT MAINTENANCE MODE. THIS DIAGNOSTIC EXERCISES THE LINE UNIT WITH RESIDENT FIRMWARE IN THE KMC11 GENERAL PURPOSE MICROPROCESSOR. THIS DIAGNOSTIC IS A STAND ALONE PACKAGE.

2.0 REQUIREMENTS

2.1 EQUIPMENT

PDP-11 PROCESSOR

KMC11 GENERAL PURPOSE MICROPROCESSOR

NOTE: THIS DIAGNOSTIC WILL TEST UP TO 16 KMC11
GENERAL PURPOSE MICROPROCESSOR'S, BUT THEY
MUST BE INSTALLED SEQUENTIALLY ON THE BUSS

DMS11-D LINE UNIT

NOTE: THIS DIAGNOSTIC WILL TEST UP TO 8 LINES PER
KMC11

2.2 STORAGE

THIS DIAGNOSTIC NEEDS A MINIMUM OF 16K OF MEMORY LOCATIONS

2.3 PRELIMINARY PROGRAMS

CZKMBAO KMC11-B STATIC PART1

CZKMCAO KMC11-B STATIC PART2

CZKMDAO DMS11-DA DYNAMIC

PRIOR TO EXERCISING THE DYNAMIC DIAGNOSTIC THE KMC11 MUST BE ERROR FREE. THIS DIAGNOSTIC DOES NOT TEST THE KMC11

3.0 LOADING PROCEDURE

3.1 METHOD

LOAD THE DIAGNOSTIC USING THE ABS LOADER. THIS DIAGNOSTIC IS SELF STARTING. TO RESTART THE DIAGNOSTIC LOAD ADDRESS 200 (8) AND DEPRESS START. IF THE DIAGNOSTIC IS ON MAGNETIC MEDIA, FOLLOW THE INSTRUCTIONS FOR THE MONITOR BEING USED.

4.0 STARTING PROCEDURE

4.1 STARTING ADDRESS (200)

4.2 RESTART ADDRESS

RESTART ADDRESS 200 (8)

4.3 OPERATOR ACTION

DURING THE INITIAL START UP THE DIAGNOSTIC WILL OUTPUT TO THE
TELETYPE THE TITLE OF THE DIAGNOSTIC AND ENTER THE MONITOR. DURING
A RESTART THE DIAGNOSTIC WILL NOT OUTPUT THE TITLE OF THE DIAGNOSTIC
BUT ENTER THE MONITOR DIRECTLY. AFTER ENTRY INTO THE DIAGNOSTIC
MONITOR THE OPERATOR CAN ENTER THE COMMANDS LISTED IN SECTION
4.4 OF THIS WRITEUP.

5.0 STARTING AND TEST INITIALIZATION:

LOAD AND START MEMORY ADDRESS 200 BY PRESCRIBED
METHOD AS PER PDP-11 PROCESSOR TYPE.
THE CONSOLE DEVICE WILL IDENTIFY THE DIAGNOSTIC ON THE
FIRST START UP.
THE CONSOLE MONITOR ROUTINE IS ENTERED EACH TIME ON START
UP AND OFFERS THE OPERATOR THE FOLLOWING CHOICES AS

- (1) DIAG. TEST (2) CONFIGURE
TO WHICH THE OPERATOR MUST RESPOND WITH A 1 OR 2
FOLLOWED BY A CARRIAGE RETURN.
(1) DIAG. TEST
DIAG. TEST IS ENTERED IF THE CONFIGURATION ROUTINE HAS
BEEN PREVIOUSLY ENTERED AND IF NOT WILL AUTOMATICALLY
GO INTO THE CONFIGURATION ROUTINE TO SPECIFY KMC11(S) STATUS.

BEFORE EXECUTING TESTS AN OPERATOR'S WARNING MESSAGE IS PRINTED
INFORMING OF CONFIGURATING TABLE UNITS REQUIRING EXTERNAL LOOP
CONNECTORS BE INSTALLED.

UNITS REQUIRING LOOP CONNECTORS

UNIT # 1

...ETC

- (2) CONFIGURE
WHEN THE CONFIGURE MODE IS ENTERED FOR THE FIRST TIME
THE CONFIGURE TABLE IS ENTERED FROM THE TOP AND THE FIRST
KMC11 IS SPECIFIED

DEV.ADR.= (6 OCTAL CHARACTERS) ;BUS ADDRESS OF KMC11 CSR
(1) BOP (0) CCP ;SELECT MODE
(1) ENABLE (0) DISABLE ;IF 0, UNIT WON'T BE TESTED
(WARNING) NO CRC-32 TESTING ;IF BOP WITH CRC32 & INTERNAL(***)
(1)CRC-32 (0)NO CRC (* *)
DEV.VEC.= (3 OCTAL CHARACTERS) ;VECTOR ADDRESS OF RECEIVE INTERRUPT
(1)EXTERNAL (0)INTERNAL ;SELECT LOOPBACK
INT.PRI. (4-7) ;INTERRUPT PRIORITY LEVEL
LINES(0-7)(I.E.,#, #, #) OR (A)=ALL (N)=NONE ;SELECT ENABLED LINES

(* *) IF BOP MODE WAS REQUESTED THE INQUIRY IS PROMPTED AND MUST
BE ANSWERED ACCORDING TO WHETHER THE DMS11, IS A DMS11-A, DMS11-D OR DMS11-DA.
DMS11-A & DMS11-D HAVE CRC-32 AND DMS11-DA DOES NOT.

(***)MESSAGE IS PRINTED IF BOP MODE WITH CRC-32 AND OPERATOR
REQUESTS TO RUN INTERNAL LOOP BACK MODE.
DMS11-A: SELECT LINE 0 ONLY

NOTE: IF STATUS IS NOT TO BE CHANGED ON CURRENT INQUIRY A CARRIAGE
RETURN (ONLY) MAY BE TYPED

AFTER THE INPUTTING IS COMPLETED AN '*' IS PRINTED AT WHICH POINT
THE OPERATOR HAS 3 CHOICES BY TYPING AN

- 'E' (EXIT) THE CONFIGURE ROUTINE (ENTERS CONSOLE ROUTINE OR
DIAG. TEST IF DIAG. TEST WAS THE ENTRY POINT
- 'M' (MODIFY) AN M FOLLOWED BY A #1-16 (DEC) TO SPECIFY THE DESIRED
KMC11'S STATUS TO BE MODIFIED. THE STATUS BLOCK IS PRINTED
AND THE NEW STATUS INVOLKED.
- 'A' (ALL) FROM LAST KMC11 PARAMETER BLOCK POINTED TO EITHER BY
THE INITIAL ENTRY OR THE 'MODIFY' ROUTINE THE REMAINING
BLOCKS ARE LOADED WITH THE SAME CONFIGURATION STATUS TO THE END
OF THE CONFIGURATION TABLE.

* ANY COMBINATION OF DEVICE STATUS CAN BE CONFIGURED AND RUN AS
LONG AS AT LEAST 1 KMC11 IS ENABLED IN THE TABLE

CONTROL:

TYPING A CNTR C (^C) WILL FORCE ENTRY TO THE MONITOR
ROUTINE.

PROCESSORS WITHOUT CONSOLE PANELS WILL HAVE A SOFT
SWITCH REGISTER WHICH IS INVOLKED BY THE OPERATOR TYPING
A CNTR G (^G). AFTER THE NEW SWITCH VALUE IS ENTERED THE
PROGRAM IS RESUMED AT POINT OF ^G INVOLK.

5.1 SWITCH OPTIONS:

THE SWITCH REGISTER SETTING MUST BE DONE ACCORDING TO
PROCESSOR TYPE AND HARDWARE ARCHITECT.

BIT15 = 1	:HALT ON ERROR
BIT14 = 1	:LOOP ON TEST
BIT13 = 1	:INHIBIT ERROR TYPEOUTS
BIT10 = 1	:RING BELL ON ERROR
BIT09 = 1	:LOOP ON ERROR

TEST EXECUTION + SEQUENCES:

EACH TEST MUST BE SUCCESSIVELY COMPLETED
BEFORE THE NEXT CAN BE ENTERED. ALL TESTS
ARE RUN SEQUENTIALLY ON EACH UNIT (KMC +
DMS11D) AND THE CURRENT DEVICE IS SPECIFIED
BY THE SEQUENCE OF THE CONFIGURATION TABLE
(ENABLED UNITS ONLY).

6.0 ERRORS

ALL ERRORS ARE REPORTED ON THE TTY UNLESS CONSOLE SWITCH
13 IS SET (REFER TO SECTION 5.1 OF THIS WRITEUP). ALL ERRORS
REPORTED HAVE A ENGLISH TYPE STATEMENT EXPLAINING THE ERROR
WITH OUTPUT OF CERTAIN REGISTERS OR MEMORY LOCATIONS WHICH
CAN BE USED AS AN AID.

6.1 ERROR REPORTING:
ERROR LOGGING CONTROL IS IN ACCORDANCE WITH THE SWITCH
REGISTER SETTINGS AS DESCRIBED UNDER "SWITCH OPTIONS".
ALL ERROR REPORTS ARE HEADED BY THE FOLLOWING INFORMATION

L.U. #	TEST	FAIL #
(# OF FAILING UNIT)	(# OF FAILING TEST)	(# INDICATING THE FAILURE)

THE ADDITIONAL FIELDS ARE IN ACCORDANCE TO THE TEST AND
DATA FIELDS OF CONCERN. THE FOLLOWING DESCRIBES THE POSSIBLE
DATA FIELDS.

SHBE = THE (SHOULD-BE) VALUE OF THE EXPECTED DATA OF THE
SUBTEST (FAIL #). THE SHBE VALUES ARE DESCRIBED IN
THE SECTION "TEST DESCRIPTIONS AND FAIL _# INFORMATION:"
WITH RESPECT TO THE FAIL #.

WAS = THE (WAS) FIELD IS THE VALUE RECEIVED PERTAINING
TO THE DATA OR STATUS UNDER TEST. THE WAS VALUES
ARE DESCRIBED IN THE SECTION "TEST DESCRIPTIONS AND
FAIL # INFORMATION:" WITH RESPECT TO THE FAIL #.

LINE # = THE NUMBER OF THE FAILING DMS11 MULTIPLEXED LINE
IN THE RANGE OF 0 THROUGH 7.

DEV. ADR. = FIRST CSR ADDRESS OF UNIT FAILING.

CSRO =
CSR2 =
CSR4 =
CSR6 =

THE CONTENTS OF THE FOUR CONTROL AND STATUS REGISTERS
AT TIME OF DETECTED FAILURE.

NOTE: NOT ALL CSR'S CONTENTS MAY BE PERTINENT TO THE
ENCOUNTERED ERROR, THIS IS DUE TO THE NATURE OF THE
SPECIFIC PROTOCOL. THE OPERATOR MUST BE FAMILIAR
WITH THE PROTOCOL HANDBOOK AND REGISTER USAGE WITH
RESPECT TO THE COMMAND CURRENTLY REFLECTED IN THE CSR'S.

BUFEA = BITS 0 + 1 DISPLAY ADDRESS BITS 16 + 17 RESPECTIVELY
OF THE CURRENT BUFFER ADDRESS UNDER TEST.

BUFADR = LOW ORDER 16 BITS OF BUFFER ADDRESS UNDER TEST.

T-BUFF = TRANSMIT BUFFER ADDRESS THAT SHOULD BE RETURNED
(FROM THE KMC11).

R-BUFF = RECEIVE BUFFER ADDRESS THAT SHOULD BE RETURNED
(FROM THE KMC11)

CUR-EA = BITS 0 + 1 DISPLAY ADDRESS BIT 16 + 17 RESPECTIVELY
OF THE CURRENT BUFFER ADDRESS UNDER TEST.

CURBUF = LOW ORDER 16 BITS OF BUFFER ADDRESS UNDER TEST.

7.0 TEST INFORMATION

7.1 TEST EXECUTION + SEQUENCES:

EACH TEST MUST BE SUCCESSIVELY COMPLETED BEFORE THE NEXT CAN BE ENTERED. ALL TESTS ARE RUN SEQUENTIALLY ON EACH UNIT (KMC + DMS11D) AND THE CURRENT DEVICE IS SPECIFIED BY THE SEQUENCE OF THE CONFIGURATION TABLE (ENABLED UNITS ONLY).

7.2 TEST DESCRIPTIONS AND FAIL # INFORMATION:

TEST 1

THIS TEST CHECKS FOR THE PRESENCE OF THE CURRENTLY ENABLED KMC-11 BY INITIALIZING THE FIRMWARE AND TESTING THAT THE FIRMWARE RESPONDS BY CLEARING 'BSEL2'

ERROR REPORT:

FAIL #S.

1=KMC-11 DEVICE ADDRESS DOES NOT RESPOND

2=KMC-11 FIRMWARE FAILED TO INITIALIZE
DEV.ADDR.

(BASE ADDRESS OF FAILING UNIT)

TEST 2:

THIS TEST CHECKS THAT THE READY IN BIT (BIT7-SELO) IS INITIALIZED AND CONTROLLED BY THE FIRMWARE. THE READY IN BIT IS TESTED FOR BEING INITIALLY CLEARED THEN THE REQUEST IN BIT (BIT5-SELO) IS SET AND READY IN IS TESTED FOR BEING SET BY THE FIRMWARE, THEN REQUEST IN IS CLEARED AND READY-IN TESTED FOR BEING CLEARED BY THE FIRMWARE.

ERROR REPORT:

FAIL #S

1=KMC-11 FAILED TO INITIALIZE

2=KMC-11 READY IN BIT SET WITH NOT REQUEST-IN

3=KMC-11 FIRMWARE FAILED TO SET READY-IN WHEN REQ. IN WAS SET

4=KMC-11 FIRMWARE FAILED TO CLEAR READY-IN WHEN REQ. IN WAS CLEARED

SHBE

WAS

(STATUS SELO SHOULD BE) (STATUS SELO READ AS) (* IF PERTAINING)

TEST 3:

THIS TEST CHECKS THAT THE KMC-11 FIRMWARE IS NOT REQUESTING AN OUTPUT OF ANY TYPE, BY EXAMINING BSEL2 OF THE KMC-11 CSR'S

ERROR REPORT

FAIL #S

1=FIRMWARE FAIL TO INITIALIZE

2=KMC-11 REQ AN OUTPUT AFTER BEING INITIALIZED

SHBE WAS

(STATUS BSEL2 SHOUL' BE) (STATUS BSEL2 READ AS)

TEST 4:

THIS TEST WILL DO A LINE INITIALIZE TO ALL LINES (SUCCESSIVELY)
AND TEST THAT THE READY-IN BIT (BIT7-BSEL0) SETS AFTER EACH
INITIALIZE COMMAND

ERROR REPORT:

FAIL #S

1=KMC-11 FIRMWARE FAILED TO INITIALIZE

2=CANNOT ISSUE AN INPUT COMMAND (OUTPUT REQUESTED OR RDY-IN FAILED)

3=FIRMWARE FAILED TO COMPLETE LINE INITIALIZE COMMAND.

TEST 5:

THIS TEST WILL ISSUE A "TRANSMIT BUFFER IN" TO EACH LINE
AND TEST THAT EACH COMMAND ACCEPTS.

ERROR REPORT:

FAIL #'S

1=FIRMWARE FAILED TO INITIALIZE

2=INPUT CANNOT BE ISSUED (FIRMWARE HUNG OR OUTPUT PENDING)

3=XMIT BUFFER IN FAILED TO COMPLETE

TEST 6:

THIS TEST WILL ISSUE A "RECEIVE BUFFER IN" TO EACH LINE
AND TEST THAT EACH COMMAND ACCEPTS.

ERROR REPORT

FAIL #'S

1=FIRMWARE FAILED TO INITIALIZE

2=INPUT CANNOT BE ISSUED (FIRMWARE HUNG OR OUTPUT PENDING)

3=RECEIVE BUFFER IN FAILED TO COMPLETE

TEST 7:

THIS TEST CHECKS THAT EACH LINE CAN ACCEPT AND FIELD A
TRANSMIT AND RECEIVE BUFFER. EACH LINE IS INITIALIZED,
THE MAINT. LOOP AND CLOCK IS ENABLED AND TWO RECEIVE
AND ONE TRANSMIT BUFFER ARE THEN QUEUED TO THE LINE UNDER
TEST. *A SERIES OF TESTS ARE PERFORMED TO CHECK THAT:

1) THE FIRST RECEIVE BUFFER IS RETURNED WITH AN EMA (1)
 STATUS AND THE CORRECT ENDING MEMORY ADDRESS.
 2) THE TRANSMIT BUFFER IS RETURNED WITH A 1 STATUS AND THE
 CORRECT MEMORY ADDRESS.
 THE FAIL #'S RANGE FROM 1 TO 22 (OCTAL)

ERROR REPORT:

TEST #	FAIL #	LINE #	SHBE	CSR0	CSR2	CSR4	CSR6
TSTNUM	ERNUM	CURLIN	SHBE	CSR0	CSR2	CSR4	CSR6
FAIL #'S							
1=FIRMWARE FAILED TO INITIALIZE *							
2=CURRENT LINE FAILED TO COMPLETE LINE INITIALIZE *							
3=MAINTENANCE COMMAND (START CLOCK + INTERNAL LOOP) FAILED TO COMPLETE *							
4=1ST 'BUFFER IN' COMMAND (RECEIVE BUFFER) FAILED TO COMPLETE *							
5=2ND 'BUFFER IN' COMMAND (RECEIVE BUFFER) FAILED TO COMPLETE*							
6=3RD 'BUFFER IN' COMMAND (TRANSMIT BUFFER) FAILED TO COMPLETE *							
7=BIT 5 OF SEL2 FAILED TO SET WITHIN SPECIFIED TIME (FIRST BUFFER)							
10=1ST COMMAND (SEL2) NOT PROPER RETURNED BUFFER TYPE							
11=1ST STATUS (SEL7) NOT = 1 (END COUNT REACHED)							
12=1ST RETURNED LINE NUMBER IS INCORRECT (SEL3)							
13=1ST ENDING BUFFER ADDRESS EXTENDED BITS (SEL7) INCORRECT							
14=1ST ENDING BUFFER 16-BIT ADDRESS (SEL4) INCORRECT							
15=BIT 5 OF SEL2 FAILED TO SET WITHIN SPECIFIED TIME (SECOND BUFFER)							
16=2ND COMMAND (SEL2) NOT PROPER RETURNED BUFFER TYPE							
17=2ND STATUS (SEL7) NOT = 1 (END COUNT REACHED)							
20=2ND RETURNED LINE NUMBER IS INCORRECT (SEL3)							
21=2ND ENDING BUFFER ADDRESS EXTENDED BITS (SEL7) INCORRECT							
22=2ND ENDING BUFFER 16-BIT ADDRESS (SEL4) INCORRECT							
*=IF ATTEMPTING AN INPUT FUNCTION AND AN OUTPUT IS PENDING							
THE COMMAND IS ABANDONED AND SEL2 WILL							
SHOW THE OUTPUT REQUEST, OTHERWISE IT IS ASSUMED THAT THE							
FIRMWARE IS NON-RESPONSIVE							

TEST 10:

THIS TEST CHECKS THAT A LINE OVERFLOW CONDITION WILL RETURN A
 'CONTROL OUT' STATUS INDICATING THE LINE OVERFLOW.
 EACH LINE IS TESTED FOR RETURNING THE OVERFLOW STATUS BY INITIALIZING
 THE LINE, QUEUEING A 10 CHARACTER TRANSMIT BUFFER AND A 1 CHR.
 RECEIVE BUFFER AND TESTING THAT A CONTROL OUT COMMAND INDICATING
 A LINE OVERFLOW FOR THE CURRENT LINE IS RETURNED AFTER THE REC. BUFFER IS RETURNED

ERROR REPORT:

FAIL #'S

- 1=INITIALIZATION OF THIS TEST FAILED (FIRMWARE HUNG OR OUTPUT REQ.IS SET)
- 2=RECEIVE BUFFER IN FAILED TO LOAD
- 3=TRANSMIT BUFFER IN FAILED TO LOAD
- 4=RECEIVE BUFFER NOT RETURNED
- 5=CONTROL OUT COMMAND FAILED TO RETURN
- 6=RETURNED COMMAND (SEL2) IS NOT A CONTROL OUT
- 7=SEL7 DOES NOT INDICATE A LINE OVERFLOW
- 10=RETURNED LINE # (SEL3) IS INCORRECT.

TEST 11:

THIS TEST CHECKS THAT THE RECEIVE BUFFERS ARE KILLED BY THE 'LINE RESET' COMMAND BY INITIALIZING THE CURRENT LINE, QUEUEING A SHORT RECEIVE BUFFER TO THE LINE, ISSUING A 'LINE RESET' COMMAND, STARTING THE MAINTENANCE CLOCK, QUEUEING A SECOND RECEIVE BUFFER TO THE LINE, SENDING A LONGER TRANSMIT BUFFER TO THE LINE, AND TESTING THAT A CONTROL OUT COMMAND IS RETURNED (LINE OVERFLOW OF THE CURRENT LINE) AFTER ONE RECEIVE BUFFER OUT IS RETURNED.

ERROR REPORT:

FAIL #'S

- 1=INITIALIZATION OF THIS TEST FAILED ((SHBE=0)
- 2=RECEIVE BUFFER FAILED TO LOAD (SHBE=0)
- 3=CONTROL IN COMMAND (LINE RESET) CANNOT BE ISSUED (SHBE=0)
- 4=LINE RESET COMMAND FAILED TO COMPLETE (SHBE=0)
- 5=CLOCK FAILED TO START
- 6=2ND RECEIVE BUFFER FAILED TO LOAD (SHBE=0)
- 7=TRANSMIT BUFFER IN FAILED TO LOAD (SHBE=0)
- 10=RECEIVE BUFFER OUT WAS NOT RETURNED (SHBE=0)
- 11=AN OUTPUT WAS NOT RETURNED FROM THE KMC (INDICATING A LINE OVERFLOW)(SHBE=0)
- 12=LINE OVERFLOW BIT IN SEL7 NOT SET
- 13=LINE # (SEL3) NOT SAME AS LINE UNDER TEST

TEST 12:

THIS TEST WILL CHECK THAT THE TRANSMIT BUFFERS ARE KILLED BY A LINE RESET BY INITIALIZING THE CURRENT LINE, QUEUEING A TRANSMIT BUFFER TO THE LINE, ISSUING A LINE RESET COMMAND, STARTING THE CLOCK, AND TESTING THAT NO OUTPUT IS RETURNED FROM THE KMC.

ERROR REPORT:

FAIL #'S

- 1=FIRMWARE HUNG OR REQUESTING TO DO AN OUTPUT (SHBE=0)
- 2=TRANSMIT BUFFER IN FAILED TO COMPLETE (SHBE=0)
- 3=LINE RESET COMMAND FAILED TO COMPLETE (SHBE=0)
- 4=MAINTENANCE COMMAND FAILED TO START CLOCK (SHBE=0)
- 5=AN OUTPUT COMMAND WAS RETURNED FROM THE KMC (SHBE=0)

TEST 13:

THIS TEST WILL CHECK THAT THE 'IN ABORT' COMMAND DOES NOT TERMINATE ANY OF THE TRANSMIT BUFFERS BY INITIALIZING EACH LINE, SENDING A TRANSMIT BUFFER TO THE LINE THEN ISSUING A LINE ABORT 'IN' (BIT1 SEL7=1) TO THE LINE, AND TESTING THAT THE NORMAL XMIT BUFFER OUT (EMA) IS RETURNED.

ERROR REPORT:

FAIL #'S

- 1=FIRMWARE HUNG OR AN OUTPUT REQUEST PENDING IN TEST SETUP ROUTINES
- 2=BUFFER IN FAILED TO COMPLETE
- 3=ABORT COMMAND CANNOT BE ISSUED
- 4=LINE ABORT COMMAND FAILED TO COMPLETE
- 5=NO OUTPUT REQUESTED AFTER THE ABORT COMMAND
- 6=STATUS ON XMIT BUFFER OUT IS NOT ENDING MEMORY ADDRESS (1)

TEST 14:

THIS TEST WILL CHECK THAT THE 'OUT ABORT' COMMAND DOES NOT TERMINATE ANY OF THE RECEIVE BUFFERS BY INITIALIZING EACH LINE, SENDING A RECEIVE BUFFER TO THE LINE, THEN ISSUING A LINE ABORT 'OUT' (BIT1 SEL7=1) TO THE LINE, AND TESTING THAT NO STATUS (BUFFER OUTS) IS RETURNED.

ERROR REPORT:

FAIL #'S

- 1=FIRMWARE HUNG OR AN OUTPUT REQUEST PENDING IN TEST SETUP ROUTINES
- 2=BUFFER IN FAILED TO COMPLETE
- 3=ABORT COMMAND CANNOT BE ISSUED
- 4=LINE ABORT COMMAND FAILED TO COMPLETE
- 5=OUTPUT IS BEING REQUESTED AFTER THE ABORT COMMAND

TEST 15:

THIS TEST WILL CHECK AN 'ABORT OUT' COMMAND WILL RETURN (KILL) THE BUFFER CURRENTLY QUEUED TO THE LINE UNDER TEST, BY INITIALIZING ALL LINES, SENDING A TRANSMIT BUFFER TO EACH LINE, THEN ISSUING A LINE ABORT OUT COMMAND TO EACH LINE INDIVIDUALLY, TESTING THAT ONLY THE BUFFER QUEUED TO THAT LINE IS RETURNED WITH A STATUS (SEL6) OF (-6) 'TERMINATED ON REQUEST'.

ERROR REPORT:

FAIL #'S

- 1=FIRMWARE HUNG OR OUTPUT REQ. WHEN INITIALLY SETTING UP TEST
- 2=TRANSMIT BUFFER IN FAILED TO LOAD (SHBE=0)
- 3=ABORT COMMAND CANNOT BE ISSUED (SHBE=0)
- 4=THE ABORT COMMAND FAILED TO COMPLETE (SHBE=0)
- 6=NO ABORTED BUFFER RETURNED (SHBE=0)
- 7=COMMAND OUT NOT VALID AS PER SELO (SHBE=SELO)
- 10=LINE # RETURNED NOT CORRECT (SHBE=LINE #)
- 11=STATUS RETURNED (SEL6) NOT 'TERMINATED ON REQ.' (-6) (SHBE=-6)

TEST 16:

THIS TEST WILL CHECK AN 'ABORT IN' COMMAND WILL RETURN (KILL) THE BUFFER CURRENTLY QUEUED TO THE LINE UNDER TEST BY INITIALIZING

ALL LINES, SENDING A RECEIVE BUFFER TO EACH LINE, THEN ISSUING A LINE 'ABORT IN' COMMAND TO EACH LINE INDIVIDUALLY, TESTING THAT ONLY THE BUFFER QUEUED TO THAT LINE IS RETURNED WITH A STATUS (SEL6) OF (-6) 'TERMINATED ON REQUEST'.

ERROR REPORT:

FAIL #'S

- 1=FIRMWARE HUNG OR OUTPUT REQ. WHEN INITIALLY SETTING UP TEST
- 2=RECEIVE BUFFER IN FAILED TO LOAD (SHBE=0)
- 3=ABORT COMMAND CANNOT BE ISSUED (SHBE=0)
- 4=THE ABORT COMMAND FAILED TO COMPLETE (SHBE=0)
- 5=PRIMING TRANSMIT BUFFER IN FAILED TO COMPLETE (CCP ONLY)
- 6=NO ABORTED BUFFER RETURNED (SHBE=0)
- 7=COMMAND OUT NOT VALID AS PER SEL2 (SHBE=SEL2)
- 10=LINE # RETURNED NOT CORRECT (SHBE=LINE #)
- 11=STATUS RETURNED (SEL6) NOT 'TERMINATED ON REQ.' (-6) (SHBE=-6)
- 12=PRIMING TANSMIT BUFFER FAILED TO RETURN (CCP ONLY)

TEST 17:

THIS TEST CHECKS THAT AN ALTERNATE 'TRANSMIT' BUFFER CAN BE QUEUED TO EACH LINE AND IS RETURNED WHEN A LINE ABORT COMMAND IS ISSUED. AFTER EACH LINE IS TESTED FOR RETURNING ITS BUFFERS, AN ALTERNATE BUFFER IS QUEUED TO THE LINE

ERROR REPORT:

FAIL #'S

- 1=FIRMWARE HUNG OR OUTPUT REQ. ON INITIAL TEST SETUP ROUTINES
- 2=1ST TRANSMIT BUFFER IN FAILED TO LOAD (SHBE=0)
- 3=ALTERNATE TRNASMIT BUFFER FAILED TO LOAD
- 4=ABORT COMMAND CANNOT BE ISSUED
- 5=LINE ABORT COMMAND FAILED TO COMPLETE
- 7=NO RESPONSE (OUTPUT) AFTER ABORTING BUFFER (1ST BUFFER) (SHBE=200)
- 10=OUTPUT NOT A BUFF OUT OR BIT2-SEL2 SHOWS INCORRECT BUFFER TYPE (1ST BUFF) (SHBE=200)
- 11=LINE NUMBER (SEL3) OF RETURNED BUFF INCORRECT (1ST BUFFER) (SHBE=LINE #)
- 12=RETURNED BUFFER STATUS (SEL6) NOT A -6 (1ST BUFFER) (SHBE=-6)
- 14=2ND ABORT COMMAND CANNOT BE ISSUED
- 15=2ND LINE ABORT COMMAND FAILED TO COMPLETE
- 17=NO RESPONSE (OUTPUT) AFTER ABORTING BUFFER (2ND BUFFER) (SHBE=200)
- 20=OUTPUT NOT A BUFF OUT OR BIT2-SEL2 SHOWS INCORRECT BUFFER TYPE (2ND BUFF) (SHBE=200)
- 21=LINE NUMBER (SEL3) OF RETURNED BUFF INCORRECT (2ND BUFFER) (SHBE=LINE #)
- 22=RETURNED BUFFER STATUS (SEL6) NOT A -6 (2ND BUFFER) (SHBE=-6)
- 24=LAST ALTERNATE BUFFER FAILED TO LOAD

TEST 20:

THIS TEST CHECKS THAT AN ALTERNATE 'RECEIVE' BUFFER CAN BE QUEUED TO EACH LINE AND IS RETURNED WHEN A LINE ABORT COMMAND IS ISSUED. AFTER EACH LINE IS TESTED FOR RETURNING ITS

BUFFERS, AN ALTERNATE BUFFER IS QUEUED TO THE LINE

ERROR REPORT:

FAIL #'S

- 1=FIRMWARE HUNG OR OUTPUT REQ. ON INITIAL TEST SETUP ROUTINES
- 2=1ST TRANSMIT BUFFER IN FAILED TO LOAD (SHBE=0)
- 3=ALTERNATE TRANSMIT BUFFER FAILED TO LOAD
- 4=ABORT COMMAND CANNOT BE ISSUED
- 5=LINE ABORT COMMAND FAILED TO COMPLETE
- 6=PRIMING TRANSMIT BUFFER FAILED TO LOAD (CCP ONLY)
- 7=NO RESPONSE (OUTPUT) AFTER ABORTING BUFFER (1ST BUFFER) (SHBE=200)
- 10=OUTPUT NOT A BUFF OUT OR BIT2-SEL2 SHOWS INCORRECT BUFFER TYPE (1ST BUFF) (SHBE=200)
- 11=LINE NUMBER (SEL3) OF RETURNED BUFF INCORRECT (1ST BUFFER) (SHBE=LINE #)
- 12=RETURNED BUFFER STATUS (SEL6) NOT A -6 (1ST BUFFER) (SHBE=-6)
- 13=PRIMING TRANSMIT BUFFER NOT RETURNED (CCP ONLY)
- 14=2ND ABORT COMMAND CANNOT BE ISSUED
- 15=2ND LINE ABORT COMMAND FAILED TO COMPLETE
- 16=2ND PRIMING TRANSMIT BUFFER FAILED TO LOAD (CCP ONLY)
- 17=NO RESPONSE (OUTPUT) AFTER ABORTING BUFFER (2ND BUFFER) (SHBE=200)
- 20=OUTPUT NOT A BUFF OUT OR BIT2-SEL2 SHOWS INCORRECT BUFFER TYPE (2ND BUFF) (SHBE=200)
- 21=LINE NUMBER (SEL3) OF RETURNED BUFF INCORRECT (2ND BUFFER) (SHBE=LINE #)
- 22=RETURNED BUFFER STATUS (SEL6) NOT A -6 (2ND BUFFER) (SHBE=-6)
- 23=2ND PRIMING TRANSMIT BUFFER NOT RETURNED (CCP ONLY)
- 24=LAST ALTERNATE BUFFER FAILED TO LOAD

TEST 21:

THIS TEST CHECKS THAT ALTERNATE BUFFERS CAN BE QUEUED TO EACH LINE AND ARE PROPERLY HANDLED BY THE FIRMWARE. A PROGRESSION IS DONE, TESTING THE QUEUEING OF ALTERNATE BUFFERS TO EACH LINE THE CURRENT LINE IS QUEUED 2 RECEIVE AND 2 TRANSMIT BUFFERS AND A TEST THAT THE BUFFERS FOR THE CURRENT LINE ARE RETURNED CORRECTLY.

ERROR REPORT:

FAIL #'S

- 1=FAILURE IN TEST INITIALIZATION ROUTINES (SHBE=0)
- 2=RECEIVE BUFFER FAILED TO LOAD (SHBE=0)
- 3=TRANSMIT BUFFER FAILED TO LOAD (SHBE=0)
- 4=OUTPUT REQ. NOT SET (BUFFER NOT RETURNED) (SHBE=0)
- 5=WRONG LINE NUMBER RETURNED (SHBE=CURRENT LINE #)
- 6=WRONG STATUS RETURNED (SHBE=CORRECT STATUS)
- 7=RETURNED BUFFER ADDRESS NOT CORRECT (SHBE=CORRECT ADDRESS)
- 10=THERE HAVE BEEN 4 BUFFERS OUT BUT NOT 2 RX + 2 TX (SHBE=# OF RX BUFFERS)
- 11=AFTER 4 BUFFERS ANOTHER REQ.OUT IS POSTED (SHBE=# OF RX BUFFERS SO FAR)

TEST 22:

THIS TEST CHECKS THAT A NON-EXISTANT MEMORY STATUS (-5) WILL BE RETURNED WITH A RECEIVE BUFFER OUT. THE NON-EXISTANT MEMORY LOCATION IS I/O PAGE LOCATION 10 TO WHICH THE RECEIVE BUFFER IS ADDRESSED.

ERROR REPORT:

FAIL #'S

- 1=FAILURE IN TEST INITIALIZATION ROUTINES (SHBE=0)
- 2=RECEIVE BUFFER FAILED TO LOAD
- 3=TRANSMIT BUFFER FAILED TO LOAD
- 4=NO OUTPUT RESPONSE FROM THE (NON-EXISTANT MEMORY) TRANSFER. (SHBE=204 (SEL2))
- 5=OUTPUT RESPONSE NOT FROM A RETURNED RECEIVE BUFFER (SHBE=204(SEL2))
- 6=RETURNED RECEIVE BUFFER STATUS NOT CORRECT (SHBE=-5 (NXM-STATUS))

TEST 23:

CCP MODE ONLY

THIS TEST WILL CHECK THAT THE 2 CHARACTERS ETX AND ETB
ARE DETECTED AND PROPERLY RETURN THE BUFFER OUT COMMAND STATUS.
BOTH CHARACTERS ARE TESTED INDIVIDUALLY WITH EACH PREFIXED
BY AN SOH THEN A STX

ERROR REPORT:

WAS = LOW BYTE = SOH OR STX HIGH BYTE = ETX OR ETB (FAILING COMBINATION)

FAIL#S

- 1=FAILURE IN TEST SET UP ROUTINES
- 2=RECEIVE BUFFER FAILED TO LOAD
- 3=TRANSMIT BUFFER FAILED TO LOAD
- 4=NO RESPONSE WAS RECEIVED FROM THE TERMINATION CHARACTER
- 5=RETURNED COMMAND IS NOT A RECEIVE BUFFER OUT
- 6=BUFFER RETURNED DOES NOT HAVE CORRECT ENDING MEMORY ADDRESS
- 7=SEL6 RETURNED STATUS IS NOT CORRECT

TEST 24:

CCP MODE ONLY

THIS TEST CHECKS THAT AN LRC ERROR CONDITION CAN BE DETECTED
AND RETURNS THE RECEIVE BUFFER WITH AN LRC CONDITION
TO GENERATE THE LRC ERROR A SPECIALLY FORMATTED BUFFER IS OUTPUT WITH
AN ADDITIONAL CHARACTER FOLLOWING THE 'ETX' WHICH THE RECEIVER
INTERPRETS AS THE LRC CHARACTER

ERROR REPORTS:

FAIL#S

- 1=FAILURE IN TEST INITIALIZE ROUTINE
- 2=RECEIVE BUFFER FAILED TO LOAD
- 3=TRANSMIT BUFFER FAILED TO LOAD
- 4=THERE IS NO OUTPUT RESPONSE FROM THE LRC ERROR CONDITION
- 5=OUTPUT IS NOT A BUFFER OUT COMMAND
- 6=RETURNED ENDING MEMORY ADDRESS NOT CORRECT
- 7=RETURNED STATUS DOES NOT INDICATE AN LRC ERROR

TEST 25:

CCP MODE ONLY

THIS TEST CHECKS THAT A PARITY ERROR CONDITION CAN BE DETECTED
AND RETURNS THE RECEIVE BUFFER WITH A PARITY CONDITION.
TO GENERATE THE PARITY ERROR A SPECIALLY FORMATTED BUFFER IS OUTPUT
WITH AN EVEN PARITY DATA CHARACTER FOLLOWING THE CONTROL CHARACTERS

ERROR REPORTS:

UNIT# TEST# FAIL# SHBE CSR0 CSR2 CSR4 CSR6

FAIL #S

1=FAILURE IN TEST INITIALIZATION ROUTINES

2=RECEIVE BUFFER FAILED TO LOAD

3=TRANSMIT BUFFER FAILED TO LOAD

4=THERE IS NO OUTPUT RESPONSE FROM THE PARITY ERROR CONDITION

5=OUTPUT IS NOT A BUFFER OUT COMMAND

6=RETURNED ENDING MEMORY ADDRESS NOT CORRECT

7=RETURNED STATUS DOES NOT INDICATE A PARITY ERROR

TEST 26:

THIS TEST CHECKS THAT THE CHARACTER COUNT WILL CORRECTLY COUNT
TO THE HIGHEST VALUE OF CORE AVAILABLE (ABOVE THE DIAGNOSTIC)
BY USING THE SAME BUFFER FOR RECEIVE AS IS TRANSMIT
AND TESTING THAT THE ENDING MEMORY ADDRESS RETURNED
OF BOTH THE BUFFERS (XMIT AND RECEIVE) ARE THE SAME

ERROR REPORT:

FAIL #'S

1=FAILURE IN TEST INITIALIZATION ROUTINES

2=RECEIVE BUFFER FAILED TO LOAD

3=TRANSMIT BUFFER FAILED TO LOAD

4=NO BUFFER RETURNED FROM TRANSFER

5=COMMAND RETURNED WAS NOT A BUFFER OUT

6=1ST BUFFER HAS INCORRECT ENDING MEMORY ADDRESS

7=2ND BUFFER FAILED TO RETURN

10=2ND COMMAND WAS NOT A BUFFER OUT

11=2ND BUFFER HAD INCORRECT ENDING MEMORY ADDRESS

12=BUFFERS RETURNED WERE NOT 1 TRANSMIT & 1 RECEIVE (SHBE=# REC.BUFFERS)

TEST 27:

THIS TEST WILL CHECK THE LOWER 13 (4K) BIT TRANSMIT AND RECEIVE
BUFFER ADDRESSING (BUFFER ADDRESS-IN AND OUT) BY TRANSMITTING
AND RECEIVING TWO WORD BUFFERS. THE ADDRESSES ARE SEQUENCED BY
WALKING A ONE ACROSS THE RETURNED BUFFER ADDRESS VALUE.
THE RETURNED BUFFER ADDRESSES ARE TESTED FOR BEING CORRECT.

ERROR REPORT:

FAIL #'S

- 1=FAILURE IN TEST INITIALIZATION ROUTINES
- 2=CURRENT RECEIVE BUFFER FAILED TO LOAD (SHBE + WAS =N/A)
- 3=CURRENT TRANSMIT BUFFER FAILED TO LOAD (SHBE+WAS=N/A)
- 4=NO OUTPUT RESPONSE AFTER QUEUEING BUFFERS (SHBE,WAS=N/A)
- 5=1ST RETURNED BUFFER IS NOT BUFFER OUT
- 6=1ST RETURNED BUFFER (ENDING MEMORY ADDRESS) NOT CORRECT (SHBE=CORRECT ADDRESS)
- 7=2ND BUFFER NOT RETURNED
- 10=2ND RETURNED COMMAND IS NOT BUFFER OUT
- 11=2ND RETURNED BUFFER (ENDING MEMORY ADDRESS) NOT CORRECT (SHBE=CORRECT ADDRESS)
- 12=TWO BUFFER OUT COMMANDS RETURNED BUT NOT ONE TX AND ONE RX (SHBE=#OF RX BUFFERS)
- 13=EXTRA RETURNED COMMAND AFTER TWO BUFFER OUTS

TEST 30:

THIS TEST WILL CHECK BUS ADDRESSING OF THE HIGHER ORDER BITS (BITS 14-17) OF THE QUEUED RECEIVE BUFFER AND MULTIPLE DATA TRANSFERS BY TRANSFERRING A TEN WORD BUFFER TO EACH 4K BOUNDARY (IN THE TEST RANGE) OUTPUT IS FROM 'TBUF'

* THE TEST WILL EXECUTE ONLY IF SUFFICIENT CORE EXISTS.

ERROR REPORT:

FAIL #'S

- 1=FAILURE IN TEST INITIALIZATION ROUTINES
- 2=THE RECEIVE BUFFER-IN COMMAND FAILED TO LOAD (SHBE,WAS=0)
- 3=THE TRANSMIT BUFFER-IN COMMAND FAILED TO LOAD (SHBE,WAS=0)
- 4=CURRENT TRANSFER FAILED TO INDICATE A COMPLETION (SHBE,WAS=0)
- 5=RETURNED COMMAND WAS NOT A BUFFER OUT
- 6=RETURNED (TRANSMIT) BUFFER EA-BITS INCORRECT (SHBE+WAS=EA-BIT STATUS)
- 7=RETURNED (TRANSMIT) BUFFER 16 BIT ADDRESS INCORRECT (SHBE+WAS=ADDRESS)
- 10=RETURNED (RECEIVE) BUFFER EA-BITS INCORRECT (SHBE+WAS=EA-BIT STATUS)
- 11=RETURNED (RECEIVE) BUFFER 16 BIT ADDRESS INCORRECT (SHBE+WAS=ADDRESS)
- 12=TWO RETURNED BUFFERS WERE NOT 1 TX AND 1 RX BUFFER

TEST 31:

THIS TEST WILL CHECK BUS ADDRESSING OF THE HIGHER ORDER BITS (BITS 14-17) OF THE QUEUED TRANSMIT BUFFER AND MULTIPLE DATA TRANSFERS BY TRANSFERRING A TEN WORD BUFFER FROM EACH 4K BOUNDARY (IN THE TEST RANGE) INPUT IS TO RBUF.

* THE TEST WILL EXECUTE ONLY IF SUFFICIENT CORE EXISTS.

ERROR REPORT:

FAIL #'S

- 1=FAILURE IN TEST INITIALIZATION ROUTINES
- 2=THE RECEIVE BUFFER-IN COMMAND FAIL TO LOAD (SHBE,WAS=0)
- 3=THE TRANSMIT BUFFER-IN COMMAND FAIL TO LOAD (SHBE,WAS=0)
- 4=CURRENT TRANSFER FAILED TO INDICATE A COMPLETION (SHBE,WAS=0)
- 5=RETURNED COMMAND IS NOT A BUFFER OUT
- 6=RETURNED (TRANSMIT) BUFFER EA-BITS INCORRECT (SHBE+WAS=EA-BIT STATUS)

7=RETURNED (TRANSMIT) BUFFER 16 BIT ADDRESS INCORRECT (SHBE+WAS=ADDRESS)
10=RETURNED (RECEIVE) BUFFER EA-BITS INCORRECT (SHBE+WAS=EA-BIT STATUS)
11=RETURNED (RECEIVE) BUFFER 16 BIT ADDRESS INCORRECT (SHBE+WAS=ADDRESS)
12=TWO RETURNED BUFFERS WERE NOT 1 TX AND 1 RX BUFFER

TEST 32:

THIS TEST WILL CHECK MULTIPLE DATA TRANSFERS AND BUS ADDRESSING
BY TRANSFERRING BUFFERS TO EACH INCREMENTAL 4K ABOVE THE DIAGNOSTIC
IN CCP MODE, THE FIRST 4K BLOCK (16-20K) IS WRITTEN WITH AN
INCREMENTAL PATTERN (40-175) AND TRANSFERRED TO EACH 4K ABOVE
THE DIAGNOSTIC (20K UP)
IN BOP MODE, THE FIRST 1024 BYTES OF THE FIRST 4K BLOCK (16K) IS
WRITTEN WITH AN INCREMENTAL PATTERN (0-377) AND TRANSFERRED
TO THE START OF EACH 4K ABOVE THE DIAGNOSTIC (20K UP)
TEST WILL RUN ONLY IF 20K OR MORE OF CONTIGUOUS CORE (0 UP)
EXISTS ON SYSTEM.

ERROR REPORT:

FAIL #'S

1=FAILURE IN TEST INITIALIZATION ROUTINES
2=RECEIVE BUFFER FAILED TO LOAD
3=TRANSMIT BUFFER FAILED TO LOAD
4=NO BUFFER RETURNED FROM TRANSFER
5=1ST RETURN WAS NOT A BUFFER OUT
6=TRANSMIT BUFFER RETURN STATUS INCORRECT
7=16-BIT TRANSMIT ADDRESS INCORRECT
10=EXTENDED TRANSMIT ADDRESS BITS INCORRECT
11=2ND BUFFER (RECEIVE) NOT RETURNED
12=2ND RETURN WAS NOT EXPECTED RECEIVE BUFFER
13=RECEIVE BUFFER RETURNED STATUS INCORRECT
14=16-BIT RECEIVE ADDRESS INCORRECT
15=EXTENDED RECEIVE ADDRESS BITS INCORRECT
16=RECEIVED DATA CHARACTER INCORRECT
17=2ND BUFFER (TRANSMIT) NOT RETURNED
20=2ND RETURN WAS NOT EXPECTED TRANSMIT BUFFER
21=TWO RETURNED BUFFERS NOT 1TX AND 1 RX BUFFER

TEST 33:

THIS TEST WILL CHECK THE INPUT REQUEST INTERRUPT LOGIC
AND FIRMWARE ARBITRATION BY INITIALIZING THE INPUT REQUEST
VECTOR (XX0) WITH A CORRECT RETURN ADDRESS AND THE OUTPUT REQUEST
VECTOR WITH ADDRESS TO ERROR FLAG ROUTINE.
THE INPUT REQUEST IS FIRST DONE WITH THE PROCESSOR STATUS
DROPPED TO 0 AND THE INTERRUPT ENABLE BIT NOT SET TO CHECK
THAT NO INTERRUPT OCCURS WHEN THE 'RDYIN' (BIT 7-SELO) SETS.
THEN THE INPUT REQUEST IS MADE WITH THE INTERRUPT ENABLE
BIT SET AND THE PROCESSOR STATUS AT 0 TO CHECK THAT THE INTERRUPT
OCCURS TO THE CORRECT VECTOR ADDRESS
THEN THE PROCESSOR STATUS IS SET TO LEVEL 7 THE REQUEST MADE
WITH THE INTERRUPT ENABLE BIT SET AND TEST THAT THE INTERRUPT DOES

NOT OCCUR UNTIL THE CORRECT PROCESSOR STATUS IS REACHED.

ERROR REPORT:

FAIL #'S

- 1=FAILURE IN TEST INITIALIZATION ROUTINES (WAS=N/A)
- 2='RDY-IN' FAILED TO SET (WAS=N/A)
- 3=ILLEGAL INTERRUPT OCCURED (W/PS @ 7) (WAS=VECTOR TRAPPED TO)
- 4=FIRMWARE INITIALIZE FAILED AFTER RDYIN SET (WAS=N/A)
- 5=2ND RDYIN FAILED TO SET (WAS=N/A)
- 6=INTERRUPTED TO WRONG VECTOR ADDRESS (WAS=VECTOR TRAPPED TO)
- 7=FIRMWARE FAILED TO INIT. AFTER INPUT INTERRUPT (WAS=N/A)
- 10=3RD RDY-IN FAILED TO SET (WAS=N/A)
- 11=INTERRUPT OCCURED AT LEVEL OTHER THAN SPECIFIED (WAS=PS WORD)

TEST 34:

THIS TEST WILL CHECK THAT AN OUTPUT REQUEST INTERRUPT CAN CORRECTLY BE EXECUTED TO INTERRUPT VECTOR XX4 AND TEST THE LOGIC AND FIRMWARE ARBITRATION.

A TRANSMIT AND RECEIVE BUFFER IS QUEUED TO LINE 0 WITH THE PROCESSOR STATUS AT 0 AND THE 'IEO' BIT NOT SET TEST IS DONE TO CHECK THAT NO INTERRUPT OCCURS. THE FIRMWARE IS REINITIALIZED THE BUFFERS REQUEUED WITH THE 'IEO' BIT SET AND THE PROCESSOR STATUS AT 0 A CHECK IS DONE TO TEST THAT THE INTERRUPT OCCURS TO THE OUTPUT VECTOR ADDRESS XX4.

ERROR REPORT:

FAIL #'S

- 1=FAILURE IN TEST INITIALIZATION ROUTINES (WAS=N/A)
- 2=RECEIVE BUFFER FAILED TO LOAD
- 3=TRANSMIT BUFFER FAILED TO LOAD
- 4=BUFFER FAILED TO RETURN
- 5=ILLEGAL INTERRUPT WITHOUT IEO SET (WAS=VECTOR TRAPPED TO)
- 6=2ND RECEIVE BUFFER FAILED TO LOAD
- 7=2ND XMIT BUFFER FAILED TO LOAD
- 10=INTERRUPT OCCURED TO WRONG VECTOR (WAS=VECTOR TRAPPED TO)
- 11=INTERRUPT FIALED WHEN 'RDYOUT' SET (WAS=N/A)
- 12=BUFFER FAILED TO RETURN (NO 'RDYOUT') (WAS=N/A)

CZKMEAO DMS11-DA DYNAMIC
CZKMEA.P11 20-OCT-81 17:05

MACY11 30A(1052) 21-OCT-81 08:50 ¹ ² PAGE 21

SEQ 0021

.NLIST MD

CZKMEAO DMS11-DA DYNAMIC
CZKMEA.P11 20-OCT-81 17:05

MACY11 30A(1052) 21-OCT-81 08:50 J² PAGE 22

SEQ 0022

C
C.

CZKMEAO DMS11-DA DYNAMIC
CZKMEA.P11 20-OCT-81 17:05

MACY11 30A(1052) 21-OCT-81 08:50 ^{K 2}PAGE 23

SEQ 0023

CZKMEAC DMS11-DA DYNAMIC
CZKMEA.P11 20-OCT-81 17:05 MACY11 30A(1052) 21-OCT-81 08:50 L² PAGE 24

SEQ 0024

C
C

CZKMEAO DMS11-DA DYNAMIC
CZKMEA.P11 20-OCT-81 17:05

MACY11 30A(1052) 21-OCT-81 08:50 ^{M 2}PAGE 25

SEQ 0025

CZKMEAO DMS11-DA DYNAMIC
CZKMEA.P11 20-OCT-81 17:05

MACY11 30A(1052) 21-OCT-81 08:50 N 2
PAGE 26

SEQ 0026

CZ
CZ

```

1068 .LIST SEQ,BIN,LOC
1069 .ENABL ABS,AMA
1070 .TITLE CZKMEAO DMS11-DA DYNAMIC
1071 ;*COPYRIGHT (C) 1981
1072 ;*DIGITAL EQUIPMENT CORP.
1073 ;*MAYNARD, MASS. 01754
1074 ;*
1075 ;*PROGRAM BY R.BALDWIN, R. J. COLLINS
1076 ;*
1077 ;*THIS PROGRAM WAS ASSEMBLED USING THE PDP-11 MAINDEC SYSMAC
1078 ;*PACKAGE (MAINDEC-11-DZQAC-C5), JAN, 1981.
1079 ;*
1080 $TN=1
1081 $SWR=160000 ;:HALT ON ERROR, LOOP ON TEST, INHIBIT ERROR TYP0UT
1082 $SWR=167400
1083 $TN=1
1084 $N=1
1085 $E=1
1086 POP2SP=22626
1087 POPPOP=22626 ;:POP THE STACK TWICE
1088 POPSP=005726 ;:POP THE STACK ONCE
1089 .SBTTL BASIC DEFINITIONS
1090
1091 ;*INITIAL ADDRESS OF THE STACK POINTER *** 1100 ***
1092 001100 STACK= 1100
1093 .EQUIV EMT,ERROR ;:BASIC DEFINITION OF ERROR CALL
1094 .EQUIV IOT,SCOPE ;:BASIC DEFINITION OF SCOPE CALL
1095
1096 ;*MISCELLANEOUS DEFINITIONS
1097 000011 HT= 11 ;:CODE FOR HORIZONTAL TAB
1098 000012 LF= 12 ;:CODE FOR LINE FEED
1099 000015 CR= 15 ;:CODE FOR CARRIAGE RETURN
1100 000200 CRLF= 200 ;:CODE FOR CARRIAGE RETURN-LINE FEED
1101 177776 PS= 177776 ;:PROCESSOR STATUS WORD
1102 .EQUIV PS,PSW
1103 177774 STKLMT= 177774 ;:STACK LIMIT REGISTER
1104 177772 PIRQ= 177772 ;:PROGRAM INTERRUPT REQUEST REGISTER
1105 177570 DSWR= 177570 ;:HARDWARE SWITCH REGISTER
1106 177570 DDISP= 177570 ;:HARDWARE DISPLAY REGISTER
1107
1108 ;*GENERAL PURPOSE REGISTER DEFINITIONS
1109 000000 R0= %0 ;:GENERAL REGISTER
1110 000001 R1= %1 ;:GENERAL REGISTER
1111 000002 R2= %2 ;:GENERAL REGISTER
1112 000003 R3= %3 ;:GENERAL REGISTER
1113 000004 R4= %4 ;:GENERAL REGISTER
1114 000005 R5= %5 ;:GENERAL REGISTER
1115 000006 R6= %6 ;:GENERAL REGISTER
1116 000007 R7= %7 ;:GENERAL REGISTER
1117 000006 SP= %6 ;:STACK POINTER
1118 000007 PC= %7 ;:PROGRAM COUNTER
1119
1120 ;*PRIORITY LEVEL DEFINITIONS
1121 000000 PR0= 0 ;:PRIORITY LEVEL 0
1122 000040 PR1= 40 ;:PRIORITY LEVEL 1
1123 000100 PR2= 100 ;:PRIORITY LEVEL 2

```

1124	C00140	PR3=	140	::PRIORITY LEVEL 3
1125	000200	PR4=	200	::PRIORITY LEVEL 4
1126	000240	PR5=	240	::PRIORITY LEVEL 5
1127	000300	PR6=	300	::PRIORITY LEVEL 6
1128	000340	PR7=	340	::PRIORITY LEVEL 7

1130 :*'SWITCH REGISTER' SWITCH DEFINITIONS

1131	100000	SW15=	100000
1132	040000	SW14=	40000
1133	020000	SW13=	20000
1134	010000	SW12=	10000
1135	004000	SW11=	4000
1136	002000	SW10=	2000
1137	001000	SW09=	1000
1138	000400	SW08=	400
1139	000200	SW07=	200
1140	000100	SW06=	100
1141	000040	SW05=	40
1142	000020	SW04=	20
1143	000010	SW03=	10
1144	000004	SW02=	4
1145	000002	SW01=	2
1146	000001	SW00=	1
1147		.EQUIV	SW09,SW9
1148		.EQUIV	SW08,SW8
1149		.EQUIV	SW07,SW7
1150		.EQUIV	SW06,SW6
1151		.EQUIV	SW05,SW5
1152		.EQUIV	SW04,SW4
1153		.EQUIV	SW03,SW3
1154		.EQUIV	SW02,SW2
1155		.EQUIV	SW01,SW1
1156		.EQUIV	SW00,SW0

1158 :*DATA BIT DEFINITIONS (BIT00 TO BIT15)

1159	100000	BIT15=	100000
1160	040000	BIT14=	40000
1161	020000	BIT13=	20000
1162	010000	BIT12=	10000
1163	004000	BIT11=	4000
1164	002000	BIT10=	2000
1165	001000	BIT09=	1000
1166	000400	BIT08=	400
1167	000200	BIT07=	200
1168	000100	BIT06=	100
1169	000040	BIT05=	40
1170	000020	BIT04=	20
1171	000010	BIT03=	10
1172	000004	BIT02=	4
1173	000002	BIT01=	2
1174	000001	BIT00=	1
1175		.EQUIV	BIT09,BIT9
1176		.EQUIV	BIT08,BIT8
1177		.EQUIV	BIT07,BIT7
1178		.EQUIV	BIT06,BIT6
1179		.EQUIV	BIT05,BIT5

```
1180      .EQUIV BIT04,BIT4
1181      .EQUIV BIT03,BIT3
1182      .EQUIV BIT02,BIT2
1183      .EQUIV BIT01,BIT1
1184      .EQUIV BIT00,BIT0
1185
1186      ;*BASIC "CPU" TRAP VECTOR ADDRESSES
1187      000004 ERRVEC= 4      ;;TIME OUT AND OTHER ERRORS
1188      000010 RESVEC= 10     ;;RESERVED AND ILLEGAL INSTRUCTIONS
1189      000014 TBITVEC=14    ;; "T" BIT
1190      000014 TRTVEC= 14    ;;TRACE TRAP
1191      000014 BPTVEC= 14    ;;BREAKPOINT TRAP (BPT)
1192      000020 IOTVEC= 20    ;;INPUT/OUTPUT TRAP (IOT) **SCOPE**
1193      000024 PWRVEC= 24    ;;POWER FAIL
1194      000030 EMTVEC= 30    ;;EMULATOR TRAP (EMT) **ERROR**
1195      000034 TRAPVEC=34    ;; "TRAP" TRAP
1196      000060 TKVEC= 60     ;;TTY KEYBOARD VECTOR
1197      000064 TPVEC= 64     ;;TTY PRINTER VECTOR
1198      000240 PIRQVEC=240   ;;PROGRAM INTERRUPT REQUEST VECTOR
1199      .SBTTL MEMORY MANAGEMENT DEFINITIONS
1200
1201      ;*KT11 VECTOR ADDRESS
1202
1203      000250 MMVEC= 250
1204
1205      ;*KT11 STATUS REGISTER ADDRESSES
1206
1207      177572 SR0= 177572
1208      177574 SR1= 177574
1209      177576 SR2= 177576
1210      172516 SR3= 172516
1211
1212      ;*KERNEL "I" PAGE DESCRIPTOR REGISTERS
1213
1214      172300 KIPDR0= 172300
1215      172302 KIPDR1= 172302
1216      172304 KIPDR2= 172304
1217      172306 KIPDR3= 172306
1218      172310 KIPDR4= 172310
1219      172312 KIPDR5= 172312
1220      172314 KIPDR6= 172314
1221      172316 KIPDR7= 172316
1222
1223      ;*KERNEL "I" PAGE ADDRESS REGISTERS
1224
1225      172340 KIPAR0= 172340
1226      172342 KIPAR1= 172342
1227      172344 KIPAR2= 172344
1228      172346 KIPAR3= 172346
1229      172350 KIPAR4= 172350
1230      172352 KIPAR5= 172352
1231      172354 KIPAR6= 172354
1232      172356 KIPAR7= 172356
1233
1234      .SBTTL TRAP CATCHER
1235
```

1236 000000

1237
1238
1239
1240
1241
1242
1243
1244
1245
1246
1247
1248
1249
1250
1251
1252
1253
1254
1255
1256
1257
1258
1259
1260
1261
1262
1263
1264

```

      . = 0
      ; *ALL UNUSED LOCATIONS OF THE VECTOR AREA CONTAIN
      ; *A ".+2, IOT" SEQUENCE TO CATCH AND PROCESS ILLEGAL
      ; *TRAPS AND INTERRUPTS THAT MIGHT OCCUR.
      ; *THE IOT TRAP WHICH IS TAKEN ON THE ILLEGAL TRAP/INT
      ; *TRAPS TO THE $SCOPE ROUTINE WHICH (IF THE RETURN PC IS
      ; *LESS THAN 1002) JUMPS TO THE $ERROR ROUTINE.
      ; *THE $ERROR ROUTINE WILL REPORT THE ERROR AS FOLLOWS:
      ; *      PC=YYYYYY UNEXPECTED TRAP TO XXX
      ; *AND RETURN TO THE PROGRAM AT PC=YYYYYY+2
      ; *WHERE XXX=LOCATION OF ILLEGAL TRAP
      ; *      YYYYYY=PC AT TIME OF TRAP
      ; *NOTE: IF THE PROCESSOR IS NOT AN 11/05 THE PROGRAM
      ; *      CAN BE STARTED AT ADDRESS 0 AS WELL AS ADDRESS 200.

```

1251	000000	000000		\$40CAT: HALT	::HALT
1252	000002	000737		BR	::BRANCH TO 177700 & TIME OUT (NOT ON
1253					::11/05)
1254	000004	001542		.WORD BEGIN	::VECTOR TO STARTING ADDRESS
1255	000006	000340		.WORD 340	::WITH PRIORITY LEVEL 7
1256		000174		. = 174	
1257	000174	000000		DISPREG: .WORD 0	::SOFTWARE DISPLAY REGISTER
1258	000176	000000		SWREG: .WORD 0	::SOFTWARE SWITCH REGISTER
1259				.SBTTL STARTING ADDRESS(ES)	
1260	000200	000137	001542	JMP @#BEGIN	::GO TO START OF PROGRAM
1261		000220		. = 220	
1262	000220	000240		NOP	::ALLOW INSERTION OF KMC DEBUF DUMP
1263	000222	000240		NOP	
1264	000224	000137	002076	JMP MONIT	::RETURN TO MONITOR

```
1265 .SBTTL COMMON TAGS
1266
1267 ::*****
1268 ::*THIS TABLE CONTAINS VARIOUS COMMON STORAGE LOCATIONS
1269 ::*USED IN THE PROGRAM.
1270
1271      001100      . =1100
1272 001100 000000 $CMTAG:      :: START OF COMMON TAGS
1273 001100 000000 $PASS: .WORD 0      :: CONTAINS PASS COUNT
1274 001102 000 $TSTNM: .BYTE 0      :: CONTAINS THE TEST NUMBER
1275 001103 000 $ERFLG: .BYTE 0      :: CONTAINS ERROR FLAG
1276 001104 000000 $ICNT: .WORD 0      :: CONTAINS SUBTEST ITERATION COUNT
1277 001106 000000 $LPADR: .WORD 0      :: CONTAINS SCOPE LOOP ADDRESS
1278 001110 000000 $LPERR: .WORD 0      :: CONTAINS SCOPE RETURN FOR ERRORS
1279 001112 000000 $ERTTL: .WORD 0      :: CONTAINS TOTAL ERRORS DETECTED
1280 001114 000 $ITEMB: .BYTE 0      :: CONTAINS ITEM CONTROL BYTE
1281 001115 001 $ERMAX: .BYTE 1      :: CONTAINS MAX. ERRORS PER TEST
1282 001116 000000 $ERRPC: .WORD 0      :: CONTAINS PC OF LAST ERROR INSTRUCTION
1283 001120 000000 $GDADR: .WORD 0      :: CONTAINS ADDRESS OF 'GOOD' DATA
1284 001122 000000 $BDADR: .WORD 0      :: CONTAINS ADDRESS OF 'BAD' DATA
1285 001124 000000 $GDDAT: .WORD 0      :: CONTAINS 'GOOD' DATA
1286 001126 000000 $BDDAT: .WORD 0      :: CONTAINS 'BAD' DATA
1287 001130 000000      .WORD 0      :: RESERVED--NOT TO BE USED
1288 001132 000000      .WORD 0
1289 001134 000 $AUTOB: .BYTE 0      :: AUTOMATIC MODE INDICATOR
1290 001135 000 $INTAG: .BYTE 0      :: INTERRUPT MODE INDICATOR
1291 001136 000000      .WORD 0
1292 001140 177570 $SWR: .WORD DSWR      :: ADDRESS OF SWITCH REGISTER
1293 001142 177570 $DISPLAY: .WORD DDISP      :: ADDRESS OF DISPLAY REGISTER
1294 001144 177560 $TKS: 177560      :: TTY KBD STATUS
1295 001146 177562 $TKB: 177562      :: TTY KBD BUFFER
1296 001150 177564 $TPS: 177564      :: TTY PRINTER STATUS REG. ADDRESS
1297 001152 177566 $TPB: 177566      :: TTY PRINTER BUFFER REG. ADDRESS
1298 001154 000 $NULL: .BYTE 0      :: CONTAINS NULL CHARACTER FOR FILLS
1299 001155 002 $FILLS: .BYTE 2      :: CONTAINS # OF FILLER CHARACTERS REQUIRED
1300 001156 012 $FILLC: .BYTE 12      :: INSERT FILL CHARS. AFTER A 'LINE FEED'
1301 001157 000 $TPFLG: .BYTE 0      :: 'TERMINAL AVAILABLE' FLAG (BIT<07>=0=YES)
1302 001160 000000 $TMP0: .WORD 0      :: USER DEFINED
1303 001162 000000 $TMP1: .WORD 0      :: USER DEFINED
1304 001164 000000 $TMP2: .WORD 0      :: USER DEFINED
1305 001166 000000 $TIMES: 0      :: MAX. NUMBER OF ITERATIONS
1306 001170 000000 $ESCAPE: 0      :: ESCAPE ON ERROR ADDRESS
1307 001172 177607 000377 $BELL: .ASCIIZ <207><377><377>      :: CODE FOR BELL
1308 001176 077 $QUES: .ASCII /?/      :: QUESTION MARK
1309 001177 015 $CRLF: .ASCII <15>      :: CARRIAGE RETURN
1310 001200 000012 $LF: .ASCIIZ <12>      :: LINE FEED
1311 ::*****
```

1312 .SBTTL ERROR POINTER TABLE
1313
1314 : *THIS TABLE CONTAINS THE INFORMATION FOR EACH ERROR THAT CAN OCCUR.
1315 : *THE INFORMATION IS OBTAINED BY USING THE INDEX NUMBER FOUND IN
1316 : *LOCATION \$ITEMB. THIS NUMBER INDICATES WHICH ITEM IN THE TABLE IS PERTINENT.
1317 : *NOTE1: IF \$ITEMB IS 0 THE ONLY PERTINENT DATA IS (\$ERRPC).
1318 : *NOTE2: EACH ITEM IN THE TABLE CONTAINS 4 POINTERS EXPLAINED AS FOLLOWS:
1319
1320 : * EM ::POINTS TO THE ERROR MESSAGE
1321 : * DH ::POINTS TO THE DATA HEADER
1322 : * DT ::POINTS TO THE DATA
1323 : * DF ::POINTS TO THE DATA FORMAT
1324
1325
1326 001202 \$ERRTB:
1327 :ERROR MESSAGE TABLE
1328
1329 001202 033050 034446 035562 EM01,DH1,DT1,DF1
1330 001210 036042
1331 001212 033101 034507 035574 EM02,DH2,DT2,DF1
1332 001220 036042
1333 001222 033137 034507 035574 EM03,DH2,DT2,DF1
1334 001230 036042
1335 001232 033164 034553 035610 EM04,DH3,DT3,DF1
1336 001240 036042
1337 001242 033211 034612 035622 EM05,DH4,DT4,DF1
1338 001250 036042
1339 001252 033240 034612 035622 EM06,DH4,DT4,DF1
1340 001260 036042
1341 001262 033270 034707 035644 EM07,DH5,DT5,DF1
1342 001270 036042
1343 001272 033324 034707 035644 EM10,DH5,DT5,DF1
1344 001300 036042
1345 001302 033347 034707 035644 EM11,DH5,DT5,DF1
1346 001310 036042
1347 001312 033405 034707 035644 EM12,DH5,DT5,DF1
1348 001320 036042
1349 001322 033444 034612 035622 EM13,DH4,DT4,DF1
1350 001330 036042
1351 001332 033503 034612 035622 EM14,DH4,DT4,DF1
1352 001340 036042
1353 001342 033542 034707 035644 EM15,DH5,DT5,DF1
1354 001350 036042
1355 001352 033602 034612 035622 EM16,DH4,DT4,DF1
1356 001360 036042
1357 001362 033641 034707 035644 EM17,DH5,DT5,DF1
1358 001370 036042
1359 001372 033702 034707 035644 EM20,DH5,DT5,DF1
1360 001400 036042
1361 001402 033742 034707 035644 EM21,DH5,DT5,DF1
1362 001410 036042
1363 001412 034001 034707 035644 EM22,DH5,DT5,DF1
1364 001420 036042
1365 001422 034021 035260 035750 EM23,DH11,DT11,DF1
1366 001430 036042
1367 001432 034054 034707 035644 EM24,DH5,DT5,DF1

1368	001440	036042			
1369	001442	034106	034707	035644	
1370	001450	036042			
1371	001452	034143	034707	035644	
1372	001460	036042			
1373	001462	034170	035454	036016	
1374	001470	036042			
1375	001472	034222	035164	035726	
1376	001500	036042			
1377	001502	034275	035164	035726	
1378	001510	036042			
1379	001512	034350	035260	035750	
1380	001520	036042			
1381	001522	034373	035365	035774	
1382	001530	036042			
1383	001532	034420	035365	035774	
1384	001540	036042			

1385					
1386					
1387	001542				
1388					
1389					
1390	001542	012706	001100		
1391	001546	005026			
1392	001550	022706	001140		
1393	001554	001374			
1394	001556	012706	001100		
1395					
1396	001562	012737	020550	000020	
1397	001570	012737	000340	000022	
1398	001576	012737	021034	000030	
1399	001604	012737	000340	000032	
1400	001612	012737	024256	000034	
1401	001620	012737	000340	000036	
1402	001626	013737	020462	020454	
1403	001634	005037	001166		
1404	001640	005037	001170		
1405	001644	112737	000001	001115	
1406	001652	012737	001652	001106	
1407	001660	012737	001660	001110	
1408					
1409					
1410	001666	013746	000004		
1411	001672	012737	001726	000004	
1412	001700	012737	177570	001140	
1413	001706	012737	177570	001142	
1414	001714	022777	177777	177216	
1415	001722	001012			
1416					
1417	001724	000403			
1418	001726	012716	001734		
1419	001732	000002			
1420	001734	012737	000176	001140	
1421	001742	012737	000174	001142	
1422	001750	012637	000004		
1423					

```

BEGIN:
.SBTTL INITIALIZE THE COMMON TAGS
::CLEAR THE COMMON TAGS ($CMTAG) AREA
MOV     # $CMTAG,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     # $SCOPE,@#IOTVEC ::IOT VECTOR FOR SCGPE ROUTINE
MOV     #340,@#IOTVEC+2   ::LEVEL 7
MOV     # $ERROR,@#EMTVEC ::EMT VECTOR FOR ERROR ROUTINE
MOV     #340,@#EMTVEC+2   ::LEVEL 7
MOV     # $TRAP,@#TRAPVEC ::TRAP VECTOR FOR TRAP CALLS
MOV     #340,@#TRAPVEC+2  ::LEVEL 7
MOV     #ENDCT,$EOPCT     ::SETUP END-OF-PROGRAM COUNTER
CLR     $TIMES            ::INITIALIZE NUMBER OF ITERATIONS
CLR     $ESCAPE           ::CLEAR THE ESCAPE ON ERROR ADDRESS
MOVB    #1,$SERMAX        ::ALLOW ONE ERROR PER TEST
MOV     #.,$SLPADR        ::INITIALIZE THE LOOP ADDRESS FOR SCOPE
MOV     #.,$LPERR         ::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     @#ERRVEC,-(SP)    ::SAVE ERROR VECTOR
MOV     #64$,@#ERRVEC    ::SET UP ERROR VECTOR
MOV     #DSWR,SWR        ::SETUP FOR A HARDWARE SWICH REGISTER
MOV     #DDISP,DISPLAY    ::AND A HARDWARE DISPLAY REGISTER
CMP     #-1,@SWR          ::TRY TO REFERENCE HARDWARE SWR
BNE     66$              ::BRANCH IF NO TIMEOUT TRAP OCCURRED
::AND THE HARDWARE SWR IS NOT = -1
BR      65$              ::BRANCH IF NO TIMEOUT
MOV     #65$,(SP)        ::SET UP FOR TRAP RETURN
RTI
65$:    MOV     #SWREG,SWR  ::POINT TO SOFTWARE SWR
MOV     #DISPREG,DISPLAY
66$:    MOV     (SP)+,@#ERRVEC ::RESTORE ERROR VECTOR

```

```
1424 .SBTTL TYPE PROGRAM NAME
1425 ::TYPE THE NAME OF THE PROGRAM IF FIRST PASS
1426 001754 005227 177777 INC #1 ::FIRST TIME?
1427 001760 001046 BNE 67$ ::BRANCH IF NO
1428 001762 022737 020514 000042 CMP #SENDAD,@#42 ::ACT-11?
1429 001770 001442 BEQ 67$ ::BRANCH IF YES
1430 001772 104401 002030 TYPE 68$ ::TYPE ASCIZ STRING
1431 .SBTTL GET VALUE FOR SOFTWARE SWITCH REGISTER
1432 001776 005737 000042 TST @#42 ::ARE WE RUNNING UNDER XXDP/ACT?
1433 002002 001006 BNE 69$ ::BRANCH IF YES
1434 002004 023727 001140 000176 CMP SWR,#SWREG ::SOFTWARE SWITCH REG SELECTED?
1435 002012 001005 BNE 70$ ::BRANCH IF NO
1436 002014 104406 GTSWR ::GET SOFT-SWR SETTINGS
1437 002016 000403 BR 70$
1438 002020 112737 000001 001134 69$: MOVB #1,$AUTOB ::SET AUTO-MODE INDICATOR
1439 002026 70$:
1440 002026 000423 BR 67$ ::GET OVER THE ASCIZ
1441 ::68$: .ASCIZ <CRLF>#CZKMEAO DMS11-BA DYNAMIC DIAGNOSTIC#<CRLF>
1442 002076 67$:
1443 002076 012706 001100 MONIT: MOV #STACK,SP
1444 002102 012737 000340 177776 MOV #340,PSW
1445 002110 012737 000006 000004 MOV #6,4 ;SET 4 FOR TIMEOUT
1446 002116 012737 000004 000006 MOV #4,6
1447 002124 005737 002256 TST 4$
1448 002130 001004 BNE 1$
1449 002132 004737 027452 JSR PC,GETCOR
1450 002136 005137 002256 COM 4$
1451 002142 004737 021660 1$: JSR PC,@$TKINT ;ENABLE TTY INTS.
1452 002146 104401 002154 TYPE 65$ ::TYPE ASCIZ STRING
1453 002152 000421 BR 64$ ::GET OVER THE ASCIZ
1454 ::65$: .ASCIZ <15><12>/((1)DIAG.TEST(2)CONFIGURATION /
1455 64$:
1456 002216 RDOCT
1457 002220 012600 MOV (SP)+,R0
1458 002222 022700 000001 CMP #1,R0 ;LOGIC TEST?
1459 002226 001451 BEQ LOOP
1460 002230 022700 000002 CMP #2,R0 ;CONFIG. MODE?
1461 002234 001003 BNE 3$
1462 002236 004737 024424 JSR PC,MODTAB ;MODIFY THE CONFIGURATION ROUTINE
1463 002242 000715 BR MONIT
1464 002244 3$:
1465 002244 104401 002252 TYPE 67$ ::TYPE ASCIZ STRING
1466 002250 000401 BR 66$ ::GET OVER THE ASCIZ
1467 ::67$: .ASCIZ ??
1468 66$:
1469 002254 000710 BR MONIT
1470 002256 000000 4$: 0 ;SWITCH
1471
1472 ;KMC11 DEVICE REGISTERS AND INTERRUPT BEGINOL
1473
1474
1475 002260 000001 SEL0: 1
1476 002262 000001 SEL1: 1
1477 002264 000001 SEL2: 1
1478 002266 000001 SEL3: 1
1479 002270 000001 SEL4: 1
```

CZKMEAO DMS11-DA DYNAMIC
CZKMEA.P11 20-OCT-81 17:05

J 3
MACY11 30A(1052) 21-OCT-81 08:50 PAGE 35
GET VALUE FOR SOFTWARE SWITCH REGISTER

SEQ 0035

```
1480 002272 000001 SEL5: 1
1481 002274 000001 SEL6: 1
1482 002276 000001 SEL7: 1
1483 002300 000001 VECIN: 1
1484 002302 000001 VECOUT: 1
1485 002304 000001 PRIIN: 1
1486 002306 000001 PRIOUT: 1
1487 002310 000 FSWSW: .BYTE
1488 002311 000 MODE: .BYTE ;200=BOP, 0=CCP
1489 002312 000 LINEB: .BYTE
1490 002313 000 CLINE: .BYTE
1491
1492 .EVEN
1493 002314 000000 CRC32F: 0 ;CRC32 ENABLED (0=NO 200=YES)
1494 002316 000000 CLKDAT: 0 ;MODE OF DATA LOOP (I/E) (200=INT. 0=EXT.)
1495 002320 000000 CHRADR: 0 ;HOLD CHARACTER OF RDCHR FOR ECHO
1496 002322 000000 CURLUR: 0 ;CURRENT LINE UNIT REGISTER UNDER TEST
1497 002324 000 CURDAT: .BYTE ;CURRENT DATA
1498 002325 000 CNTBIT: .BYTE ;UPPER BYTE TO CONTAIN CONTROL BIT
1499 002326 000000 CONBIT: .WORD ;CONTROL BIT
1500 002330 000000 MODMSK: 0
1501 002332 000000 TYPFLG: 0
1502 002334 000000 SAVE: .WORD
1503
1504
1505
1506 002336 000000 HADDR: 0 ;HIGH ADDRESS
1507 002340 000000 XBITS: 0 ;EXTENDED ADDRESS BITS
1508 002342 000000 CSR0: 0
1509 002344 000000 CSR2: 0
1510 002346 000000 CSR4: 0
1511 002350 000000 CSR6: 0
1512
1513
1514
1515
1516 002352 012706 001100 LOOP: MOV #STACK,SP
1517 002356 004737 030270 PC,LPARND ;DISPLAY UNITS NEEDING LOOP CONNECTORS
1518 002362 012737 026232 024360 RESTRT: MOV #KMCTAB-12,CURTAB
1519 002370 004737 026506 JSR PC,GETKMC ;GET NEXT ENABLED KMC11
1520 002374 000422 BR SETUP ;BR=NO KMCS ENABLED
1521 002376 005037 001102 I.TEST: CLR $TSTNM
1522 002402 005037 002460 CLR TSTNUM
1523 002406 005037 002330 CLR MODMSK ;SET MODE MASK
1524 002412 105737 002311 TSTB MODE ;CHECK FOR MODE
1525 002416 100403 BMI 11$ ;SKIP IF BOP MODE
1526 002420 012737 170000 002330 MOV #170000,MODMSK ;SET CCP MODE MASK
1527 002426 004737 021660 11$: JSR PC,@$TKINT ;ENABLE TTY INPUT
1528 002432 005037 177776 CLR PSW ;ENABLE INTS.
1529 002436 000137 002536 JMP STTEST
1530 002442 004737 024362 SETUP: JSR PC,SETTAB ;SETUP CONFIGURATION TABLE
1531 002446 000137 002352 JMP LOOP
1532 000004 IOT=4
1533 002452 000000 ERNUM: 0 ;INDICATES FAILURE NUMBER
1534 002454 000000 CURLIN: 0 ;CURRENT LINE # UNDER TEST
1535 002456 000000 CUNIT: 0 ;CURRENT LINE UNIT # UNDER TEST
```

```
1536 002460 000000 TSTNUM: 0 ;CURRENT TEST #
1537 002462 000000 SHBE: 0
1538 002464 000000 WAS: 0
1539 002466 000000 XOUT: 0
1540 002470 000000 ROUT: 0
1541 002472 000000 XRET: 0 ;RETURN TRANSMIT BUFFER ADDR.
1542 002474 000000 RRET: 0 ;RETURN RECEIVE BUFFER ADDR.
1543 002476 000000 CUREA: 0
1544 002500 000000 CURBUF: 0
1545 002502 000000 MAPADR: 0
1546 002504 000000 RECEA: 0 ;REC. BUFFER EA
1547 002506 000000 RECBUF: 0 ;REC. BUFFER ADR
1548 002510 000000 RETEA: 0 ;ADDR BUF. SHOULD RETURN
1549 002512 000000 RETADR: 0 ;ADDR BUF. SHOULD RETURN
1550 002514 000000 RETBUF: 0
1551 002516 000000 CHAR: 0
1552 002520 177770 NXMADR: -10 ;NON-EXISTANT MEMORY ADDR
1553
1554 002522 000000 BUFADR: 0 ;16 BIT ADDRESS OF BUFFER
1555 002524 000000 BUFEA: 0 ;EA BITS OF BUFFER (BITS 17+16 IN 15+14)
1556 002526 000000 BUFcnt: 0 ;CHARACTER COUNT OF BUFFER
1557 002530 000000 RBUFSZ: 0 ;FLAG FOR CONTROL OF REC. BUFFER
1558
1559 002532 100000 BADDR: 100000 ;BASE ADDRESS OF BUFFER
1560 002534 000000 CCTST: 0 ;CHAR. CNT UP TO 8K BYTES OVER DIAG
1561 002536 004737 030616 STTEST: JSR PC,INTPRM ;PRIME INTERRUPT VECTORS
1562 002542 004737 030020 JSR PC,LDMODE ;LOAD FIRMWARE ACCORDING TO CONFIG.
1563 002546 005037 177776 CLR PSW
1564 ;*****
1565 ;INTTST
1566 ;THIS TEST WILL CHECK FOR THE PRESENCE OF THE
1567 ;CURRENTLY ENABLED KMC-11 BY INITIALIZING THE
1568 ;FIRMWARE AND TESTING THAT THE FIRMWARE RESPONDS
1569 ;BY CLEARING BSEL2
1570 ;*****
1571 ;TEST 1
1572 ;*****
1573 002552 000004 TST1: SCOPE
1574 002554 012737 000001 002460 MOV #1,TSTNUM ;LOAD THE WD OF THIS TEST
1575 002562 004737 030244 JSR PC,SETLIN ;SETUP THE FIRST LINE
1576 002566 012737 002574 001110 MOV #1$,SLPERR ;LOAD LOOP ADDRESS (LOOP ON ERROR)
1577 002574 012737 000001 002452 1$: MOV #1,ERNUM
1578 002602 005037 002462 CLR SHBE
1579 002606 013746 000004 MOV @#4,-(SP)
1580 002612 012737 002712 000004 MOV #3$,@#4 ;TEST FOR PRESENCE OF KMC-11 #=1
1581 002620 012700 000015 MOV #15,R0 ;WATCH-DOG
1582 002624 112777 000377 177432 MOVb #377,@SEL2
1583 002632 005077 177422 CLR @SELO ;*:CLEAR RUN, IF UP
1584 002636 012777 040000 177414 MOV #40000,@SELO ;MASTER CLR.
1585 002644 012777 100000 177406 MOV #100000,@SELO ;RUN
1586 002652 012637 000004 MOV (SP)+,@#4 ;RESET LOC 4 + SP
1587 002656 005037 001160 CLR $TMP0
1588 002662 105237 001160 2$: INCB $TMP0 ;WAIT LOOP
1589 002666 001375 BNE 2$
1590 002670 105777 177370 TSTB @SEL2
1591 002674 001413 BEQ 4$ ;FIRMWARE INITIALIZED
```

```
1592 002676 005300      DEC      R0
1593 002700 001370      BNE      2$
1594 002702 012737 000002 002452  MOV      #2,ERNUM      ;TRY AGAIN
1595 002710 000403      BR       31$      ;FIRMWARE FAILED TO INITIALIZE #=2
1596 002712      3$:      ;CURRENT KMC-11 DEVICE ADDRESS FAILED TO RESPOND
1597 002712 022626      POPPOP      ;SKIP
1598 002714 012637 000004      MOV      (SP)+,a#4      ;FIX THE STACK
1599 002720      31$:
1600 002720 104001      ERROR+ 1
1601 002722 000724      BR       1$      ;LOOP a1$
1602 002724      4$:
1603      ;:*****
1604      ;INRDY
1605      ;THIS TEST CHECKS THAT THE READY IN BIT (BIT7-SELO) IS
1606      ;INITIALIZED AND CONTROLLED BY THE FIRMWARE. THE READY IN
1607      ;BIT IS TESTED FOR BEING INITIALLY CLEARED THEN
1608      ;THE REQUEST IN BIT (BIT5-SELO) IS SET AND READY IN
1609      ;IS TESTED FOR BEING SET BY THE FIRMWARE, THEN REQUEST IN
1610      ;IS CLEARED AND READY-IN TESTED FOR BEING CLEARED BY THE FIRMWARE.
1611      ;
1612      ;:*****
1613      ;TEST 2
1614      ;:*****
1615 002724 000004      TST2:  SCOPE
1616 002726 012737 000002 002460  MOV      #2,TSTNUM      ;LOAD THE WD OF THIS TEST
1617 002734 004737 030244      JSR      PC,SETLIN      ;SETUP THE FIRST LINE
1618 002740 012737 002746 001110  MOV      #1$,SLPERR      ;LOAD LOOP ADDRESS (LOOP ON ERROR)
1619 002746 012737 000001 002452  1$:  MOV      #1,ERNUM
1620 002754 005037 002462      CLR      SHBE
1621 002760 004737 030664      JSR      PC,INTKMC      ;INITIALIZE KMC-11 FIRMWARE
1622 002764 000442      BR       3$      ;ERROR RET.KMC FAILED TO INITIALIZE FAIL#=1
1623 002766 005237 002452      INC      ERNUM
1624 002772 105777 177262      TSTB     @SELO      ;RDY-IN SET?
1625 002776 100424      BMI      2$      ;B=Y REPORT ERROR RDY-IN SET #=2
1626 003000 112777 000040 177252  MOVB     #40,@SELO      ;SET REQUEST IN
1627 003006 005237 002452      INC      ERNUM      ;FAIL #3=REQ IN NOT SET
1628 003012 004737 031714      JSR      PC,SYNTIM      ;WAIT A WHILE
1629 003016 105777 177236      TSTB     @SELO      ;DID RDY-IN SET?
1630 003022 100025      BPL      4$      ;B=N REPORT ERROR RDYIN FAILED TO SET #=3
1631 003024 142777 000040 177226  BICB     #40,@SELO      ;CLEAR REQUEST IN
1632 003032 005237 002452      INC      ERNUM      ;READY IN FAIL TO CLEAR #=4
1633 003036 004737 031714      JSR      PC,SYNTIM      ;WAIT A WHILE
1634 003042 105777 177212      TSTB     @SELO      ;DID RDY-IN CLEAR?
1635 003046 100025      BPL      TST3      ;RDY-IN UNDER CONTROL OF FIRMWARE
1636 003050 117737 177204 002464  2$:  MOVB     @SELO,WAS
1637 003056 013737 002464 002462  MOV      WAS,SHBE
1638 003064 042737 000200 002462  BIC      #200,SHBE
1639 003072      3$:
1640 003072 104002      ERROR+ 2
1641 003074 000724      BR       1$
1642 003076 117737 177156 002464  4$:  MOVB     @SELO,WAS
1643 003104 013737 002464 002462  MOV      WAS,SHBE
1644 003112 052737 000200 002462  BIS      #200,SHBE
1645 003120 000764      BR       3$
1646
1647      ;:*****
```

```

1648                                     ;NOOUT
1649                                     ;THIS TEST CHECKS THAT THE KMC-11 FIRMWARE IS NOT
1650                                     ;REQUESTING AN OUTPUT OF ANY TYPE, BY EXAMINING BYTE
1651                                     ;2 OF THE KMC-11 CSR'S
1652                                     ;:*****
1653                                     ;TEST 3
1654                                     ;:*****
1655 003122 000004 TST3: SCOPE
1656 003124 012737 000003 002460 MOV #3,TSTNUM ;LOAD THE WD OF THIS TEST
1657 003132 004737 030244 JSR PC,SETLIN ;SETUP THE FIRST LINE
1658 003136 012737 003144 001110 MOV #1$,SLPERR ;LOAD LOOP ADDRESS (LOOP ON ERROR)
1659 003144 012737 000001 002452 1$: MOV #1,ERNUM
1660 003152 005037 002462 CLR SHBE
1661 003156 004737 030664 JSR PC,INTKMC ;INITIALIZE KMC-11 FIRMWARE
1662 003162 000412 BR 3$ ;B=FIRMWARE FAILED TO INITIALIZE #=1
1663 003164 005237 002452 INC ERNUM ;FAIL #2=FIRMWARE REQ. AN OUTPUT
1664 003170 132777 000200 177066 BITB #200,@SEL2 ;OUTPUT REQUEST POSTED?
1665 003176 001001 BNE 2$ ;B=YES FIRMWARE REQUEST AN OUTPUT #=2
1666 003200 000405 BR TST4 ;:OK GO ON
1667 003202 117737 177056 002464 2$: MOVB @SEL2,WAS
1668 003210 3$: ;FIRMWARE FAILED TO INITIALIZE
1669 003210 104003 ERROR+ 3
1670 003212 000754 BR 1$
1671
1672                                     ;:*****
1673                                     ;LININT
1674                                     ;THIS TEST WILL DO A LINE INITIALIZE TO ALL LINES (SUCCESSFULLY)
1675                                     ;AND TEST THAT THE READY-IN BIT SETS AFTER EACH INITIALIZE
1676                                     ;COMMAND
1677
1678                                     ;:*****
1679
1680                                     ;TEST 4
1681                                     ;:*****
1682 003214 000004 TST4: SCOPE
1683 003216 012737 000004 002460 MOV #4,TSTNUM ;LOAD THE WD OF THIS TEST
1684 003224 004737 030244 JSR PC,SETLIN ;SETUP THE FIRST LINE
1685 003230 012737 003236 001110 MOV #1$,SLPERR ;LOAD LOOP ADDRESS (LOOP ON ERROR)
1686 003236 012737 000001 002452 1$: MOV #1,ERNUM ;INIT FAIL #'S
1687 003244 005037 002462 CLR SHBE
1688 003250 004737 030664 JSR PC,INTKMC ;INITIALIZE KMC-11 FIRMWARE
1689 003254 000433 BR 4$ ;REPORT ERROR - FIRMWARE HUNG #=1
1690 003256 2$: ;TEST THAT A 'REQUEST-IN' CAN BE DONE
1691 003256 012737 000002 002452 MOV #2,ERNUM ;FAIL #2=CAN NOT ISSUE AN INPUT COMMAND
1692 003264 004737 031020 JSR PC,REQNI ;CAN A INITIALIZE IN BE EXECUTED?
1693 003270 000425 BR 4$ ;REPORT ERROR RDY-IN FAILED OR OUTPUT REQUEST
1694 ;LOAD THE LINE-INIT COMMAND
1695 003272 113777 002454 176766 MOVB CURLIN,@SEL3 ;LOAD LINE #
1696 003300 012777 000400 176766 MOV #400,@SEL6 ;ENABLE LINE
1697 003306 005077 176756 CLR @SEL4 ;NOT USED
1698 ;CLEAR THE REQUEST-IN BIT TO NOTIFY FIRMWARE THAT COMMAND IS READY
1699 003312 042777 000040 176740 BIC #40,@SEL0 ;CLEAR REQUEST IN
1700 ;NOW TEST THAT THE FIRMWARE CLEARS READY-IN
1701 003320 005237 002452 INC ERNUM ;FIRMWARE FAILED TO COMPLETE LINE INIT COMMAND #=3
1702 003324 005037 001160 CLR $TMP0
1703 003330 105777 176724 3$: TSTB @SEL0 ;RDY-IN CLEAR?

```

CZKMEAO DMS11-DA DYNAMIC
CZKMEA.P11 20-OCT-81 17:05

MACY11 30A(1052) 21-OCT-81 08:50 PAGE 39
GET VALUE FOR SOFTWARE SWITCH REGISTER

SEQ 0039

```
1704 003334 100005          BPL      5$          ;B=Y
1705 003336 005237 001160    INC      $TMP0
1706 003342 001372          BNE      3$
1707 003344          4$:
1708 003344 104004          ERROR+  4
1709 003346 000733          BR       1$          ;LOOP @ 1$
1710
1711 003350          5$:
1712 003350 004737 031650    JSR      PC,GETLIN      ;GET THE NEXT LINE NUMBER
1713 003354 000401          BR       TST5          ;:DONE ALL LINES
1714 003356 000737          BR       2$          ;TRY NEXT LINE
1715
1716
1717 ;:*****
1718 ;IOBUF IN
1719 ;THIS TEST WILL ISSUE A 'TRANSMIT BUFFER' IN TO EACH LINE
1720 ;AND TEST THAT EACH COMMAND ACCEPTS.
1721 ; BLANK=XMIT BUFFERS IN
1722 ; NON-BLANK=REC BUFFERS IN
1723 ;:*****
1724
1725 ;TEST 5
1726 ;:*****
1727 003360 000004          TST5:  SCOPE
1728 003362 012737 000005 002460    MOV      #5,TSTNUM      ;LOAD THE WD OF THIS TEST
1729 003370 004737 030244          JSR      PC,SETLIN      ;SETUP THE FIRST LINE
1730 003374 012737 003402 001110    MOV      #1$,SLPERR      ;LOAD LOOP ADDRESS (LOOP ON ERROR)
1731 003402 012737 000001 002452  1$:  MOV      #1,ERNUM
1732 003410 005037 002462          CLR      SHBE
1733 003414 004737 030664          JSR      PC,INTKMC      ;INITIALIZE KMC-11 FIRMWARE
1734 003420 000430          BR       4$          ;FIRMWARE FAILED TO INITIALIZE #=1
1735
1736 003422 012737 000002 002452  2$:  MOV      #2,ERNUM
1737 003430 004737 030760          JSR      PC,REQNT      ;CAN A TRANSMIT BUFF. IN BE EXECUTED?
1738 003434 000422          BR       4$          ;CANNOT ISSUE A REQ.IN #=2
1739 ;LOAD THE BUFFER IN TO EACH LINE
1740 003436 005237 002452          INC      ERNUM
1741 003442 012737 010002 002526    MOV      #10002,BUFCNT      ;GET LENGTH OF BUFFER
1742 003450 013777 002526 176616    MOV      BUFCNT,@SEL6      ;LOAD LENGTH
1743 003456 012777 052260 176604    MOV      #TBUF,@SEL4      ;ADDRESS OF BUFFER
1744 003464 113777 002454 176574    MOV      CURLIN,@SEL3      ;LOAD LINE #
1745 ;TEST THAT FIRMWARE ACCEPTS COMMAND
1746 003472 004737 031772          JSR      PC,INPDUN      ;TEST THAT INPUT COMPLETED
1747 003476 000401          BR       4$          ;COMMAND FAIL TO COMPLETE #=3
1748 003500 000404          BR       5$          ;OK GO ON
1749 003502          4$:
1750 003502 004737 031470          JSR      PC,GETCSR      ;GET CONTENTS OF THE CSRS
1751 003506 104005          ERROR+  5
1752 003510 000744          BR       2$
1753
1754 003512          5$:
1755 003512 004737 031650          JSR      PC,GETLIN      ;GET THE NEXT LINE NUMBER
1756 003516 000401          BR       TST6          ;:ALL BUFFERS TESTED
1757 003520 000740          BR       2$
1758
1759 ;:*****
```

```
1760      ;IOBUF IN
1761      ;THIS TEST WILL ISSUE A 'RECEIVE BUFFER IN' TO EACH LINE
1762      ;AND TEST THAT EACH COMMAND ACCEPTS.
1763      ;X BLANK=XMIT BUFFERS IN
1764      ;X NON-BLANK=REC BUFFERS IN
1765      ;:*****
1766
1767      ;TEST 6
1768      ;:*****
1769      TST6: SCOPE
1770      MOV #6,TSTNUM      ;LOAD THE WD OF THIS TEST
1771      JSR PC,SETLIN      ;SETUP THE FIRST LINE
1772      MOV #1$,SLPERR     ;LOAD LOOP ADDRESS (LOOP ON ERROR)
1773      MOV #1,ERNUM
1774      CLR SHBE
1775      JSR PC,INTKMC      ;INITIALIZE KMC-11 FIRMWARE
1776      BR 4$             ;FIRMWARE FAILED TO INITIALIZE #=1
1777
1778      MOV #2,ERNUM
1779      JSR PC,REQNR      ;CAN A REC. BUFFER-IN BE EXECUTED?
1780      BR 4$             ;CANNOT ISSUE A REQ.IN #=2
1781      ;LOAD THE BUFFER IN TO EACH LINE
1782      INC ERNUM
1783      MOV #10002,BUFCNT  ;GET LENGTH OF BUFFER
1784      MOV BUFCNT,@SEL6   ;LOAD LENGTH
1785      MOV #RBUF,@SEL4    ;RECEIVE BUFFER
1786      MOVB CURLIN,@SEL3  ;LOAD LINE #
1787      ;TEST THAT FIRMWARE ACCEPTS COMMAND
1788      JSR PC,INPDUN      ;TEST THAT INPUT COMPLETED
1789      BR 4$             ;COMMAND FAIL TO COMPLETE #=3
1790      BR 5$             ;OK GO ON
1791
1792      JSR PC,GETCSR      ;GET CONTENTS OF THE CSRS
1793      ERROR+ 6
1794      BR 2$
1795
1796      JSR PC,GETLIN      ;GET THE NEXT LINE NUMBER
1797      BR TST7            ;:ALL BUFFERS TESTED
1798      BR 2$
1799
1800      ;:*****
1801      ;XRTST
1802      ;THIS TEST CHECKS THAT EACH LINE CAN ACCEPT AND FIELD A
1803      ;TRANSMIT AND RECEIVE BUFFER. EACH LINE IS INITIALIZED,
1804      ;THE MAINT. LOOP AND CLOCK IS ENABLED AND TWO RECEIVE
1805      ;AND ONE TRANSMIT BUFFER ARE THEN QUEUED TO THE LINE UNDER
1806      ;TEST. *A SERIES OF TESTS ARE PERFORMED TO CHECK THAT:
1807      ;1) THE FIRST RECEIVE BUFFER IS RETURNED WITH AN EMA (1)
1808      ;STATUS AND THE CORRECT ENDING MEMORY ADDRESS,
1809      ;2) THE TRANSMIT BUFFER IS RETURNED WITH A 1 STATUS AND THE
1810      ;CORRECT MEMORY ADDRESS.
1811      ;THE FAIL #'S RANGE FROM 1 TO 22 (OCTAL)
1812      ;:*****
1813
1814      ;TEST 7
1815
```

```
1816
1817 003664 000004
1818 003666 012737 000007 002460
1819 003674 004737 030244
1820 003700 012737 003706 001110
1821 003706 012737 000001 002452 1$:
1822 003714 013700 002454
1823 003720 005037 002462
1824 003724 004737 030664
1825 003730 000413
1826 003732 005237 002452
1827 003736 004737 031204
1828 003742 000406
1829 003744 005237 002452
1830 003750 004737 031274
1831 003754 000401
1832 003756 000402
1833 003760 000137 004510 15$:
1834
1835 003764 012737 052660 002522 16$:
1836 003772 005037 002524
1837 003776 012737 000002 002526
1838 004004 005237 002452
1839 004010 004737 031344
1840 004014 000761
1841 004016 005237 002452
1842 004022 012737 000020 002526
1843 004030 004737 031344
1844 004034 000751
1845
1846 004036 012737 033022 002522
1847 004044 012737 000004 002526
1848 004052 005237 002452
1849 004056 004737 031354
1850 004062 000736
1851 004064 005237 002452
1852 004070 012737 000204 002462
1853 004076 005037 001160
1854 004102 105777 176156 2$:
1855 004106 100405
1856 004110 005237 001160
1857 004114 001372
1858 004116 000137 004510
1859 004122 005237 002452 3$:
1860
1861 004126 123777 002462 176130
1862 004134 001165
1863 004136 005237 002452
1864
1865 004142 012737 000001 002462
1866 004150 123777 002462 176116
1867 004156 001154
1868 004160 005237 002452
1869
1870 004164 013737 002454 002462
1871 004172 127737 176070 002454

*****
TST7: SCOPE
MOV #7,TSTNUM ;LOAD THE WD OF THIS TEST
JSR PC,SETLIN ;SETUP THE FIRST LINE
MOV #1$,SLPERR ;LOAD LOOP ADDRESS (LOOP ON ERROR)
1$: MOV #1,ERNUM ;INITIALIZE ERROR
MOV CURLIN,RO
CLR SHBE
JSR PC,INTKMC ;INITIALIZE KMC-11 FIRMWARE
BR 15$ ;FIRMWARE FAILED TO INITIALIZE #=1
INC ERNUM
JSR PC,INTLNE ;INITIALIZE LINE (# IN CURLIN)
BR 15$ ;CURRENT LINE FAILED TO INITIALIZE #=2
INC ERNUM
JSR PC,STCLK ;SET MAINT LOOP + START CLOCK
BR 15$ ;B=CLOCK NOT STARTED #=3
BR 16$ ;OK
15$: JMP 6$ ;GO TO ERR
;LOAD RECEIVE BUFFER IN COMMANDS
16$: MOV #RBUF,BUFADR ;LOAD THE BUFFER ADDRESS
CLR BUFEA ;NO EA BITS
MOV #2,BUFCNT ;SET 1ST BUFFER SHORT
INC ERNUM
JSR PC,RIN ;LOAD A RECEIVE BUFFER IN
BR 15$ ;1ST RECEIVE BUFFER FAILED TO LOAD #=4
INC ERNUM
MOV #20,BUFCNT ;SET 2ND BUFFER LONG
JSR PC,RIN ;LOAD A RECEIVE BUFFER IN
BR 15$ ;2ND RECEIVE BUFFER FAILED TO LOAD #=5
;LOAD TRANSMIT BUFFER IN COMMAND
MOV #T7BUF,BUFADR ;LOAD THE TRANSMIT BUFFER ADDRESS
MOV #4,BUFCNT ;SET TX BUFFER LENGTH TO MEDIUM
INC ERNUM
JSR PC,XIN ;LOAD A TRANSMIT BUFFER IN
BR 15$ ;TRANSMIT BUFF FAIL TO LOAD #=6
INC ERNUM
MOV #204,SHBE ;GET RECEIVE BUFFER OUT COMMAND
CLR $TMP0 ;SET UP WAIT LOOP
2$: TSTB @SEL2 ;OUTPUT REQUESTED?
BMI 3$ ;B=Y OK
INC $TMP0
BNE 2$ ;GO WAIT SOME MORE
JMP 6$ ;REQ.OUT FAILED TO SET REPORT ERROR #=7
3$: INC ERNUM
;TEST THAT COMMAND IS RECEIVE BUFFER OUT
CMPB SHBE,@SEL2 ;CHEC.. SEL2
BNE 6$ ;B=NO REPORT 1ST COMMAND BUFFER TYPE INCORRECT #=10
INC ERNUM
;TEST THAT CORRECT STATUS IS RETURNED
MOV #1,SHBE ;SET STATUS TO END COUNT REACHED
CMPB SHBE,@SEL6 ;CORRECT STATUS?
BNE 6$ ;B=NO 1ST STATUS INCORRECT #=11
INC ERNUM
;TEST THAT RETURNED LINE # IS THE CURRENT LINE
MOV CURLIN,SHBE
CMPB @SEL3,CURLIN ;SAME LINE #?
```

1872	004200	001143				BNE	6\$		;B=NO 1ST LINE NUMBER INCORRECT #=12
1873	004202	005237	002452			INC	ERNUM		
1874						;TEST THAT ENDING MEMORY ADDRESS OF RETURNED BUFFER IS CORRECT			
1875	004206	017737	176062	002462		MOV	@SEL6,SHBE		;GET EXTENDED ADDRESS
1876	004214	042737	140000	002462		BIC	#140000,SHBE		;EXTENDED BITS SHOULD BE CLEAR
1877	004222	132777	000300	176046		BITB	#300,@SEL7		;SHOULD BE EA- BITS
1878	004230	001127				BNE	6\$		;REPORT EXTENDED ADDRESS NOT CORRECT #=13
1879	004232	005237	002452			INC	ERNUM		
1880	004236	105737	002311			TSTB	MODE		;TEST MODE FOR PROPER 16-BIT ADDRESS
1881	004242	100404				BMI	30\$		
1882	004244	012737	052662	002462		MOV	#RBUF+2,SHBE		;CCP MODE
1883	004252	000403				BR	31\$		
1884	004254	012737	052664	002462	30\$:	MOV	#RBUF+4,SHBE		;BOP MODE
1885	004262	023777	002462	176000	31\$:	CMP	SHBE,@SEL4		;CORRECT 16 BIT ADDRESS?
1886	004270	001107				BNE	6\$		;B=N REPORT 1ST 16-BIT ADDRESS NOT CORRECT #=14
1887	004272	005237	002452			INC	ERNUM		
1888						;NOW CLEAR READY OUT AND TEST THAT ANOTHER BUFFER OUT COMMAND			
1889						;IS PRESENT WITHIN PRESCRIBED TIME PERIOD.			
1890	004276	105077	175762			CLRB	@SEL2		;CLEAR READY OUT
1891	004302	012737	000200	002462		MOV	#200,SHBE		;GET TRANSMIT BUFFER OUT COMMAND
1892	004310				33\$:				
1893	004310	004737	031742			JSR	PC,REQOUT		;TEST REQUEST OUT OF KMC SETS
1894	004314	000475				BR	6\$		;REPORT ERROR 2ND BUFFER NOT RETURNED #=15
1895	004316	005237	002452		5\$:	INC	ERNUM		
1896						;TEST THAT COMMAND IS TRANSMIT BUFFER OUT			
1897	004322	122777	000204	175734		CMPB	#204,@SEL2		;CHECK IF COMMAND IS 2ND RETURNING RX BUFFER
1898	004330	001005				BNE	4\$		;B=NO
1899	004332	105077	175726			CLRB	@SEL2		;DISMISS 2ND RECEIVE BUFFER
1900	004336	005337	002452			DEC	ERNUM		;RESET ERROR #
1901	004342	000762				BR	33\$		; & GO WAIT SOME MORE
1902	004344	123777	002462	175712	4\$:	CMPB	SHBE,@SEL2		;IS COMMAND CORRECT?
1903	004352	001056				BNE	6\$		;B=NO REPORT COMMAND NOT CORRECT #=16
1904	004354	005237	002452			INC	ERNUM		
1905						;TEST THAT RETURNED STATUS IS AN ENDING MEMORY ADDRESS			
1906	004360	012737	000001	002462		MOV	#1,SHBE		;GET CORRECT TERMINATION STATUS
1907	004366	123777	002462	175700	34\$:	CMPB	SHBE,@SEL6		;CORRECT STATUS?
1908	004374	001045				BNE	6\$		;B=N ERROR #=17
1909	004376	005237	002452			INC	ERNUM		
1910						;TEST THAT RETURNED LINE # IS THE CURRENT LINE			
1911	004402	013737	002454	002462		MOV	CURLIN,SHBE		
1912	004410	127737	175652	002454		CMPB	@SEL3,CURLIN		;CORRECT LINE #?
1913	004416	001034				BNE	6\$		;B=NO ERROR #=20
1914	004420	005237	002452			INC	ERNUM		
1915						;TEST THAT ENDING MEMORY ADDRESS OF RETURNED BUFFER IS CORRECT			
1916	004424	017737	175644	02462		MOV	@SEL6,SHBE		;GET EXTENDED ADDRESS
1917	004432	042737	140000	002462		BIC	#140000,SHBE		;EXTENDED BITS SHOULD BE CLEAR
1918	004440	132777	000300	175630		BITB	#300,@SEL7		;EA BITS?
1919	004446	001020				BNE	6\$		;B=YES ERROR #=21
1920	004450	005237	002452			INC	ERNUM		
1921	004454	105737	002311			TSTB	MODE		
1922	004460	100404				BMI	35\$		
1923	004462	012737	033026	002462		MOV	#T7BUF+4,SHBE		;CCP MODE TX ENDING BUFFER ADDRESS
1924	004470	000403				BR	36\$		
1925	004472	012737	033032	002462	35\$:	MOV	#T7BUF+10,SHBE		;BOP MODE TX ENDING BUFFER ADDRESS
1926	004500	023777	002462	175562	36\$:	CMP	SHBE,@SEL4		;CORRECT 16 BIT ADDRESS?
1927	004506	001405				BEQ	7\$		;B=YES

```
1928                                     ;INCORRECT 16-BIT ADDRESS #=22
1929 004510 6$: ;REPORT ERROR BY DISPLAYING CONTENTS OF THE CSRS
1930 004510 JSR PC,GETCSR ;GET CONTENTS OF THE CSRS
1931 004514 104007 ERROR+ 7
1932 004516 000137 003706 JMP 1$ ;LOOP @ 1$
1933
1934 ;TEST ALL LINES
1935 004522 7$: JSR PC,GETLIN ;GET THE NEXT LINE NUMBER
1936 004522 004737 031650 BR TST10 ;:ALL LINES TESTED
1937 004526 000402
1938 004530 000137 003706 8$: JMP 1$
1939
1940 ;:*****
1941 ;LINOVR
1942 ;THIS TEST CHECKS THAT A LINE OVERFLOW CONDITION WILL RETURN A
1943 ;"CONTROL OUT" STATUS INDICATING THE LINE OVERFLOW.
1944 ;EACH LINE IS TESTED FOR RETURNING THE OVERFLOW STATUS BY INITIALIZING
1945 ;THE LINE, QUEUEING A 10 WORD TRANSMIT BUFFER AND A 1 WORD REC. BUFF.
1946 ;AND TESTING THAT A CONTROL OUT COMMAND INDICATING A LINE
1947 ;OVERFLOW FOR THE CURRENT LINE IS RETURNED AFTER THE REC. BUFFER IS RETURNED
1948 ;:*****
1949
1950 ;TEST 10
1951 ;:*****
1952 004534 000004 TST10: SCOPE
1953 004536 012737 000010 002460 MOV #10,TSTNUM ;LOAD THE WD OF THIS TEST
1954 004544 004737 030244 JSR PC,SETLIN ;SETUP THE FIRST LINE
1955 004550 012737 004556 001110 MOV #1$,SLPERR ;LOAD LOOP ADDRESS (LOOP ON ERROR)
1956 004556 012737 000001 002452 1$: MOV #1,ERNUM
1957 004564 005037 002462 CLR SHBE
1958 004570 004737 030664 JSR PC,INTKMC ;INITIALIZE KMC-11 FIRMWARE
1959 004574 000532 BR 3$ ;FIRMWARE FAILED TO INITIALIZE #=1
1960 004576 004737 031204 JSR PC,INTLNE ;INITIALIZE LINE (# IN CURLIN)
1961 004602 000527 BR 3$ ;LINE FAILED TO INITIALIZE #=1
1962 004604 004737 031274 JSR PC,STCLK ;SET MAINT LOOP + START CLOCK
1963 004610 000524 BR 3$ ;MAINT.COMMAND NOT ISSUED #=1
1964 004612 004737 031714 JSR PC,SYNTIM ;ALLOW TIME FOR SYNC
1965 004616 012702 052260 MOV #TBUF,R2 ;INITIALIZE TRANSMIT BUFFER BEFORE 1ST USE
1966 004622 012701 000005 MOV #5,R1 ; (LENGTH FOR CCP)
1967 004626 004737 032542 JSR PC,MEMFIL
1968 004632 005037 002524 CLR BUFEA ;NO EA BITS
1969 004636 012737 052660 002522 MOV #RBUF,BUFADR
1970 004644 012737 000001 002526 MOV #1,BUFCNT ;1 CHR. REC. BUFF.
1971 004652 005237 002452 INC ERNUM
1972 004656 004737 031344 JSR PC,RIN ;LOAD A RECEIVE BUFFER IN
1973 004662 000477 BR 3$ ;REC BUF FAILED TO LOAD #=2
1974 004664 012737 052260 002522 MOV #TBUF,BUFADR
1975 004672 012737 000005 002526 MOV #5,BUFCNT ;SET LENGTH OF TRANSMIT BUFFER
1976 004700 005237 002452 INC ERNUM
1977 004704 004737 031354 JSR PC,XIN ;LOAD A TRANSMIT BUFFER IN
1978 004710 000464 BR 3$ ;TRANSMIT BUFFER FAILED TO LOAD #=3
1979 004712 012737 000204 002462 MOV #204,SHBE
1980 ;WAIT FOR THE RECEIVE BUFFER (1 WORD) TO RETURN
1981 004720 005237 002452 INC ERNUM
1982 004724 004737 031742 JSR PC,REQOUT ;TEST REQUEST OUT OF KMC SETS
1983 004730 000454 BR 3$ ;RECEIVE BUFFER NOT RETURNED #=4
```

```
1984 004732 105077 175326 CLR B @SEL2 ;DISMISS THE FIRST RETURN (REC.)
1985 ;NOW WAIT FOR A CONTROL OUT TO BE RETURNED INDICATING LINE OVERFLOW
1986 004736 005237 002452 INC ERNUM
1987 004742 012737 000205 002462 MOV #205,SHBE ;SET EXPECTED COMMAND TO CONTROL OUT
1988 004750 004737 031742 JSR PC,REQOUT ;TEST REQUEST OUT OF KMC SETS
1989 004754 000442 BR 3$ ;CONTROL OUT FAILED TO RETURN #=5
1990 004756 005237 002452 INC ERNUM
1991 004762 123777 002462 175274 CMPB SHBE,@SEL2 ;CONTROL OUT COMMAND?
1992 004770 001034 BNE 3$ ;RETURNED COMMAND IS NOT A CONTROL OUT #=6
1993 004772 005237 002452 INC ERNUM
1994 ;TEST THAT CORRECT OVERRUN STATUS FOR CURRENT MODE IS
1995 ;CORRECT. CCP MODE =BIT5 SEL7 - BOP MODE =BIT2 SEL7
1996 004776 012737 000004 002462 MOV #4,SHBE ;SEL7 OVERRUN STATUS BIT FOR BOP MODE
1997 005004 105737 002311 TSTB MODE ;BOP OR CCP?
1998 005010 100403 BMI 2$ ;B=BOP
1999 005012 012737 000040 002462 MOV #40,SHBE ;CCP MODE OVERRUN STATUS BIT
2000 005020 123777 002462 175250 2$: CMPB SHBE,@SEL7 ;LINE OVERRUN SET
2001 005026 001015 BNE 3$ ;LINE OVERRUN STATUS BIT NOT SET #=7
2002 005030 005237 002452 INC ERNUM
2003 ;IS LINE # (SEL3) SAME AS LINE UNDER TEST?
2004 005034 013737 002454 002462 MOV CURLIN,SHBE
2005 005042 127737 175220 002454 CMPB @SEL3,CURLIN
2006 005050 001004 BNE 3$ ;B=NO ERROR #=10
2007 ;
2008 005052 004737 031650 JSR PC,GETLIN ;GET THE NEXT LINE NUMBER
2009 005056 000405 BR TST11 ;ALL LINES TESTED
2010 005060 000636 BR 1$
2011
2012 ;REPORT ERROR
2013 005062 3$: JSR PC,GETCSR ;GET CONTENTS OF THE CSRS
2014 005062 004737 031470 JSR ERROR+ 10
2015 005066 104010 BR 1$
2016 005070 000632
2017
2018
2019 ;*****
2020 ;RSTRCV
2021 ;THIS TEST CHECKS THAT THE RECEIVE BUFFERS ARE KILLED BY THE 'LINE RESET'
2022 ;COMMAND BY INITIALIZING THE CURRENT LINE, QUEUEING A SHORT RECEIVE BUFFER
2023 ;TO THE LINE, ISSUING A 'LINE RESET' COMMAND, STARTING THE
2024 ;MAINTENANCE CLOCK, QUEUEING A SECOND RECEIVE BUFFER TO THE LINE,
2025 ;SENDING A LONGER TRANSMIT BUFFER TO THE LINE,
2026 ;AND TESTING THAT A CONTROL OUT COMMAND IS RETURNED (LINE OVERFLOW
2027 ;OF THE CURRENT LINE) AFTER ONE RECEIVE BUFFER OUT IS RETURNED.
2028 ;*****
2029
2030 ;TEST 11
2031 ;*****
2032 005072 000004 TST11: SCOPE
2033 005074 012737 000011 002460 MOV #11,TSTNUM ;LOAD THE WD OF THIS TEST
2034 005102 004737 030244 JSR PC,SETLIN ;SETUP THE FIRST LINE
2035 005106 012737 005114 001110 MOV #1$, $LPERR ;LOAD LOOP ADDRESS (LOOP ON ERROR)
2036 005114 012737 000001 002452 1$: MOV #1,ERNUM
2037 005122 005037 002462 CLR SHBE
2038 005126 004737 030664 JSR PC,INTKMC ;INITIALIZE KMC-11 FIRMWARE
2039 005132 000574 BR 7$ ;FIRMWARE FAILED TO INITIALIZE #=1
```

```
2040 005134 004737 031204 JSR PC,INTLNE ;INITIALIZE LINE (# IN CURLIN)
2041 005140 000571 BR 7$ ;INITIALIZATION OF LINES FAILED #=1
2042 005142 004737 031274 JSR PC,STCLK ;SET MAINT LOOP + START CLOCK
2043 005146 000566 BR 7$ ;CLOCK FAILED TO START
2044 ;QUEUE A RECEIVE BUFFER TO THE CURRENT LINE UNDER TEST
2045 005150 012737 052660 002522 MOV #RBUF,BUFADR
2046 005156 005037 002524 CLR BUFEA
2047 005162 012737 010002 002526 MOV #10002,BUFCNT ;3 CHARACTER RECEIVE BUFFER
2048 005170 043737 002330 002526 BIC MODMSK,BUFCNT
2049 005176 005237 002452 INC ERNUM
2050 005202 004737 031344 JSR PC,RIN ;LOAD A RECEIVE BUFFER IN
2051 005206 000546 BR 7$ ;RECEIVE BUFFER FAILED TO LOAD #=2
2052 ;ISSUE THE 'LINE RESET' COMMAND TO THE CURRENT LINE
2053 005210 005237 002452 INC ERNUM
2054 005214 004737 031000 JSR PC,REQNC ;CAN A CONTROL IN BE EXECUTED?
2055 005220 000541 BR 7$ ;CANNOT ISSUE COMMAND ERROR #=3
2056 005222 005237 002452 INC ERNUM
2057 005226 113777 002454 175032 MOVB CURLIN,@SEL3 ;LOAD THE LINE #
2058 005234 012777 010000 175032 MOV #10000,@SEL6 ;LINE RESET INDICATOR
2059 005242 005077 175022 CLR @SEL4 ;UNUSED
2060 005246 004737 031772 JSR PC,INPDUN ;TEST THAT INPUT COMPLETED
2061 005252 000524 BR 7$ ;LINE RESET COMMAND FAILED TO COMPLETE #=4
2062 005254 005237 002452 INC ERNUM
2063 005260 004737 031274 JSR PC,STCLK ;SET MAINT LOOP + START CLOCK
2064 005264 000517 BR 7$
2065 005266 004737 031714 JSR PC,SYNTIM ;ALLOW FOR SYNC
2066 005272 005237 002452 INC ERNUM
2067 005276 012737 052664 002522 MOV #RBUF+4,BUFADR ;LOAD NEW RECV
2068 005304 004737 031344 JSR PC,RIN ;LOAD A RECEIVE BUFFER IN
2069 005310 000505 BR 7$ ;RECV BUF FAILED TO LOAD: #=6
2070 005312 005237 002452 INC ERNUM
2071 005316 012737 010006 002526 MOV #10006,BUFCNT
2072 005324 043737 002330 002526 BIC MODMSK,BUFCNT
2073 005332 012737 052260 002522 MOV #TBUF,BUFADR
2074 005340 004737 031354 JSR PC,XIN ;LOAD A TRANSMIT BUFFER IN
2075 005344 000467 BR 7$ ;TRANSMIT BUFFER IN FAILED TO LOAD #=7
2076 005346 005237 002452 INC ERNUM
2077 005352 012737 000204 002462 MOV #204,SHBE ;EXPECT RECEIVE BUFFER OUT
2078 005360 004737 031742 JSR PC,REQOUT ;WAIT FOR BUFFER
2079 005364 000457 BR 7$ ;BUFFER NOT RETURNED: #=10
2080 005366 105077 174672 CLRB @SEL2 ;DISMISS
2081 005372 005237 002452 INC ERNUM
2082 ;TEST THAT RECEIVE BUFFER WAS RESET (KILLED) BY TESTING THAT A LINE
2083 ;OVERFLOW 'CONTROL OUT' STATUS IS NOW RECEIVED
2084 005376 012737 000205 002462 MOV #205,SHBE ;EXPECT CONTROL OUT COMMAND
2085 005404 005037 002464 CLR WAS
2086 005410 005037 001160 CLR $TMP0
2087
2088 005414 105777 174644 4$: TSTB @SEL2
2089 005420 100404 BMI 5$ ;B=OUT RDY SET
2090 005422 005237 001160 INC $TMP0
2091 005426 001372 BNE 4$ ;TRY AGAIN
2092 005430 000435 BR 7$ ;FIRMWARE DID NOT RETURN AN OUTPUT #=11
2093 005432 005237 002452 5$: INC ERNUM
2094 ;BOP-IF A CONTROL OUT SEL7 BIT2 SHOULD BE SET
2095 ;CCP-IF A CONTROL OUT SEL7 BITS SHOULD BE SET
```

```
2096 005436 012737 002000 002462      MOV      #2000,SHBE      ;SET FOR BOP
2097 005444 105737 002311      TSTB     MODE        ;BOP?
2098 005450 100403      BMI      56$        ;B=YES
2099 005452 012737 020000 002462      MOV      #20000,SHBE    ;NO, SET TO CCP
2100 005460 033777 002462 174606 56$:  BIT      SHBE,@SEL6    ;IS BIT SET?
2101 005466 001416      BEQ      7$          ;B=NO, ERROR #=12
2102 005470 005237 002452      INC      ERNUM
2103      ;WAS CORRECT LINE NUMBER RETURNED
2104 005474 013737 002454 002462      MOV      CURLIN,SHBE
2105 005502 123777 002454 174556      CMPB     CURLIN,@SEL3  ;CORRECT LINE # RETURNED
2106 005510 001005      BNE      7$          ;B=NO ERROR #=13
2107 005512 004737 031650      JSR      PC,GETLIN     ;GET THE NEXT LINE NUMBER
2108 005516      57$:
2109 005516 000407      BR       TST12        ;:ALL LINES TESTED
2110 005520 000137 005114      JMP      1$          ;DO NEXT LINE
2111
2112      ;REPORT ERROR BY DISPLAYING THE ERROR # AND CONTENTS OF CSRS
2113 005524      7$:
2114 005524 004737 031470      JSR      PC,GETCSR     ;GET CONTENTS OF THE CSRS
2115 005530 104011      ERROR+  11
2116 005532 000137 005114      JMP      1$
2117
2118      ;:*****
2119      ;RSTXMT
2120      ;THIS TEST WILL CHECK THAT THE TRANSMIT BUFFERS ARE KILLED BY A
2121      ;LINE RESET BY INITIALIZING THE CURRENT LINE, QUEUEING A TRANSMIT BUFFER TO THE
2122      ;LINE, ISSUING A LINE RESET COMMAND, STARTING THE CLOCK AND TESTING THAT NO
2123      ;OUTPUT IS RETURNED FROM THE KMC.
2124      ;:*****
2125
2126      ;TEST 12
2127      ;:*****
2128 005536 000004      TST12:  SCOPE
2129 005540 012737 000012 002460      MOV      #12,TSTNUM    ;LOAD THE WD OF THIS TEST
2130 005546 004737 030244      JSR      PC,SETLIN     ;SETUP THE FIRST LINE
2131 005552 012737 005560 001110      MOV      #1$,SLPERR    ;LOAD LOOP ADDRESS (LOOP ON ERROR)
2132 005560 005037 002462      1$:  CLR      SHBE
2133 005564 012737 000001 002452      MOV      #1,ERNUM
2134 005572 004737 030664      JSR      PC,INTKMC     ;INITIALIZE KMC-11 FIRMWARE
2135 005576 000452      BR       4$          ;FIRMWARE FAILED TO INITIALIZE #=1
2136 005600 004737 031204      JSR      PC,INTLNE     ;INITIALIZE LINE (# IN CURLIN)
2137 005604 000447      BR       4$          ;LINE INITIALIZE FAILED TO COMPLETE #=1
2138 005606 012737 052260 002522      MOV      #TBUF,BUFADR ;TRANSMIT BUFFER ADDRESS
2139 005614 005037 002524      CLR      BUFEA        ;NO EA-BITS
2140 005620 012737 000777 002526      MOV      #777,BUFCNT  ;BIG CHARACTER BUFFER
2141 005626 043737 002330 002526      BIC      MODMSK,BUFCNT
2142 005634 005237 002452      INC      ERNUM
2143 005640 004737 031354      JSR      PC,XIN        ;LOAD A TRANSMIT BUFFER IN
2144 005644 000427      BR       4$          ;BUFFER IN FAILED TO COMPLETE #=2
2145      ;ISSUE A LINE RESET COMMAND
2146 005646 005237 002452      INC      ERNUM
2147 005652 004737 031546      JSR      PC,LRSET      ;ISSUE A LINE RESET TO THE CURRENT LINE
2148 005656 000422      BR       4$          ;LINE RESET COMMAND FAILED #=3
2149 005660 005237 002452      INC      ERNUM
2150 005664 004737 031274      JSR      PC,STCLK      ;SET MAINT LOOP + START CLOCK
2151 005670 000415      BR       4$          ;STCLK COMMAND FAILED #=4
```

```
2152 005672 004737 031714      JSR      PC,SYNTIM      ;WAIT A WHILE
2153                               ;NOW TEST THAT NO OUTPUT IS RETURNED FROM THE KMC
2154 005676 005237 002452      INC      ERNUM
2155 005702 004737 031742      JSR      PC,REQOUT      ;TEST REQUEST OUT OF KMC SETS
2156 005706 000401              BR      10$      ;B=NO OK
2157 005710 000405              BR      4$      ;ERROR OUTPUT REQUESTED #=5
2158 005712              10$:      ;ARE THERE MORE LINES
2159 005712 004737 031650      JSR      PC,GETLIN      ;GET THE NEXT LINE NUMBER
2160 005716 000406              BR      TST13      ;;
2161 005720 000137 005560      JMP      1$
2162                               ;REPORT ERROR - DISPLAY THE CSR'S
2163 005724              4$:      JSR      PC,GETCSR      ;GET CONTENTS OF THE CSRS
2164 005724 004737 031470      ERROR+ 12
2165 005730 104012              BR      1$
2166 005732 000712
2167
2168      ;:*****
2169      ;NABORT
2170      ;THIS TEST WILL CHECK THAT THE 'IN ABORT' COMMAND DOES NOT
2171      ;TERMINATE ANY OF THE TRANSMIT BUFFERS BY INITIALIZING EACH
2172      ;LINE, SENDING A TRANSMIT BUFFER TO THE LINE THEN ISSUING
2173      ;A LINE ABORT 'IN' (BIT1 SEL7=1) TO THE LINE, AND TESTING
2174      ;THAT THE NORMAL TRANSMIT BUFFER OUT IS RETURNED.
2175      ;:*****
2176
2177      ;TEST 13
2178      ;:*****
2179 005734 000004      TST13:      SCOPE
2180 005736 012737 000013 002460      MOV      #13,TSTNUM      ;LOAD THE WD OF THIS TEST
2181 005744 004737 030244      JSR      PC,SETLIN      ;SETUP THE FIRST LINE
2182 005750 012737 005756 001110      MOV      #1$,SLPERR      ;LOAD LOOP ADDRESS (LOOP ON ERROR)
2183 005756 005037 002462      1$:      CLR      SHBE
2184 005762 012737 000001 002452      MOV      #1,ERNUM
2185 005770 004737 030664      JSR      PC,INTKMC      ;INITIALIZE KMC-11 FIRMWARE
2186 005774 000507      BR      6$      ;FIRMWARE FAILED TO INITIALIZE #=1
2187 005776 004737 031204      JSR      PC,INTLNE      ;INITIALIZE LINE (# IN CURLIN)
2188 006002 000504      BR      6$      ;CURRENT LINE FAILED TO INITIALIZE #=1
2189 006004 004737 031274      JSR      PC,STCLK      ;SET MAINT LOOP + START CLOCK
2190 006010 000501      BR      6$      ;CLOCK COMMAND FAILED
2191 006012 012737 052260 002522      MOV      #TBUF,BUFADR
2192 006020 005037 002524      CLR      BUFEA      ;NO EA-BITS
2193 006024 012737 000777 002526      MOV      #777,BUFCNT      ;LARGE BUFFER
2194 006032 043737 002330 002526      BIC      MODMSK,BUFCNT
2195 006040 005237 002452      INC      ERNUM
2196 006044 004737 031354      JSR      PC,XIN      ;LOAD A TRANSMIT BUFFER IN
2197 006050 000461      BR      6$      ;BUFFER IN FAILED TO COMPLETE #=2
2198      ;NOW ISSUE THE ABORT COMMAND
2199 006052 005237 002452      INC      ERNUM
2200 006056 004737 031000      JSR      PC,REQNC      ;CAN A CONTROL IN BE EXECUTED?
2201 006062 000454      BR      6$      ;REPORT ERROR, HUNG OR OUTPUT REQUESTED #=3
2202 006064 113777 002454 174174      MOV      CURLIN,@SEL3      ;LOAD LINE #
2203 006072 112777 000002 174176      MOV      #2,@SEL7      ;ABORT LINE
2204 006100 005077 174164      CLR      @SEL4      ;UNUSED
2205 006104 105077 174164      CLRB      @SEL6      ;UNUSED
2206      ;WAIT FOR COMMAND TO COMPLETE
2207 006110 005237 002452      INC      ERNUM
```

CZKMEAO DMS11-DA DYNAMIC
CZKMEA.P11 20-OCT-81 17:05

J 4
MACY11 30A(1052) 21-OCT-81 08:50 PAGE 48
GET VALUE FOR SOFTWARE SWITCH REGISTER

SEQ 0048

```
2208 006114 004737 031772      JSR    PC,INPDUN      ;TEST THAT INPUT COMPLETED
2209 006120 000435              BR     6$             ;ERROR ABORT COMMAND FAILED TO COMPLETE #=4
2210                          ;NOW WAIT FOR ANY BUFFERS TO BE RETURNED
2211 006122 005237 002452      INC     ERNUM
2212 006126 012737 000200 002462  MOV    #200,SHBE      ;ON TRANSMIT, EXPECT TX BUFFER OUT
2213 006134 012737 000002 006224  MOV    #2,7$          ;SET UP WAIT LOOP
2214 006142
2215 006142 004737 031742      JSR    PC,REQOUT      ;TEST REQUEST OUT OF KMC SETS
2216 006146 000401              BR     3$
2217 006150 000404              BR     4$
2218 006152 005337 006224      3$:   DEC     7$
2219 006156 001371              BNE     2$
2220 006160 000415              BR     6$             ;ON TRANSMIT, NO OUTPUT IS ERROR #=5
2221 006162
2222                          4$:   ;TEST FOR CORRECT TRANSMIT TERMINATION (EMA)
2223 006162 005237 002452      INC     ERNUM
2224 006166 112737 000001 002462  MOVB   #1,SHBE
2225 006174 123777 002462 174072  CMPB   SHBE,@SEL6
2226 006202 001004              BNE     6$             ;#6, WRONG TRANSMIT STATUS ON ABORT
2227                          ;TEST IF ALL LINES ARE DONE
2228 006204
2229 006204 004737 031650      5$:   JSR    PC,GETLIN      ;GET THE NEXT LINE NUMBER
2230 006210 000406              BR     TST14          ;:ALL LINES TESTED
2231 006212 000661              BR     1$
2232
2233 006214
2234 006214 004737 031470      6$:   ;REPORT ERROR
2235 006220 104013              JSR    PC,GETCSR      ;GET CONTENTS OF THE CSRS
2236 006222 000655              ERROR+ 13
2237 006224 000000              BR     1$
2238
2239                          7$:   0             ;DELAY COUNT
2240
2241                          ;NABORT
2242                          ;THIS TEST WILL CHECK THAT THE 'OUT ABORT' COMMAND DOES NOT
2243                          ;TERMINATE ANY OF THE RECEIVE BUFFERS BY INITIALIZING EACH
2244                          ;LINE, SENDING A RECEIVE BUFFER TO THE LINE, THEN ISSUING
2245                          ;A LINE ABORT 'OUT' (BIT1 SEL7=1) TO THE LINE, AND TESTING
2246                          ;THAT NO STATUS (BUFFER OUT) IS RETURNED.
2247                          ;:*****
2248
2249                          ;TEST 14
2250                          ;:*****
2251 006226 000004      TST14:  SCOPE
2252 006230 012737 000014 002460  MOV    #14,TSTNUM      ;LOAD THE WD OF THIS TEST
2253 006236 004737 030244              JSR    PC,SETLIN      ;SETUP THE FIRST LINE
2254 006242 012737 006250 001110  MOV    #1$,SLPERR      ;LOAD LOOP ADDRESS (LOOP ON ERROR)
2255 006250 005037 002462      1$:   CLR     SHBE
2256 006254 012737 000001 002452  MOV    #1,ERNUM
2257 006262 004737 030664              JSR    PC,INTKMC      ;INITIALIZE KMC-11 FIRMWARE
2258 006266 000474              BR     6$             ;FIRMWARE FAILED TO INITIALIZE #=1
2259 006270 004737 031204              JSR    PC,INTLNE      ;INITIALIZE LINE (# IN CURLIN)
2260 006274 000471              BR     6$             ;CURRENT LINE FAILED TO INITIALIZE #=1
2261 006276 004737 031274              JSR    PC,STCLK      ;SET MAINT LOOP + START CLOCK
2262 006302 000466              BR     6$             ;CLOCK COMMAND FAILED
2263 006304 012737 052660 002522  MOV    #RBUF,BUFADR
2264 006312 005037 002524              CLR     BUFEA      ;NO EA-BITS
2265 006316 012737 000777 002526  MOV    #777,BUFCNT      ;LARGE BUFFER
```

```
2264 006324 043737 002330 002526 BIC MODMSK,BUFCNT
2265 006332 005237 002452 INC ERNUM
2266 006336 004737 031344 JSR PC,RIN ;LOAD A RECEIVE BUFFER IN
2267 006342 000446 BR 6$ ;BUFFER IN FAILED TO COMPLETE #=2
2268 ;NOW ISSUE THE ABORT COMMAND
2269 006344 005237 002452 INC ERNUM
2270 006350 004737 031010 JSR PC,REQNX ;CAN A CONTROL OUT BE EXECUTED?
2271 006354 000441 BR 6$ ;REPORT ERROR, HUNG OR OUTPUT REQUESTED #=3
2272 006356 113777 002454 173702 MOVB CURLIN,@SEL3 ;LOAD LINE #
2273 006364 112777 000002 173704 MOVB #2,@SEL7 ;ABORT LINE
2274 006372 005077 173672 CLR @SEL4 ;UNUSED
2275 006376 105077 173672 CLRB @SEL6 ;UNUSED
2276 ;WAIT FOR COMMAND TO COMPLETE
2277 006402 005237 002452 INC ERNUM
2278 006406 004737 031772 JSR PC,INPDUN ;TEST THAT INPUT COMPLETED
2279 006412 000422 BR 6$ ;ERROR ABORT COMMAND FAILED TO COMPLETE #=4
2280 ;NOW WAIT FOR ANY BUFFERS TO BE RETURNED
2281 006414 005237 002452 INC ERNUM
2282 006420 012737 000002 006470 MOV #2,7$ ;SET UP WAIT LOOP
2283 006426 2$: JSR PC,REQOUT ;TEST REQUEST OUT OF KMC SETS
2284 006426 004737 031742 BR 3$
2285 006432 000401 BR 4$
2286 006434 000404 3$: DEC 7$
2287 006436 005337 006470 BNE 2$
2288 006442 001371 BR 5$
2289 006444 000401 4$: BR 6$ ;ON RECEIVE, NO OUTPUT IS EXPECTED
2290 006446 000404 5$: ;OUTPUT REQUESTED IS ERROR #=5
2291 ;TEST IF ALL LINES ARE DONE
2292 006450 5$: JSR PC,GETLIN ;GET THE NEXT LINE NUMBER
2293 006450 004737 031650 BR TST15 ;ALL LINES TESTED
2294 006454 000406 BR 1$
2295 006456 000674
2296
2297 006460 6$: ;REPORT ERROR
2298 006460 004737 031470 JSR PC,GETCSR ;GET CONTENTS OF THE CSRS
2299 006464 104014 ERROR+ 14
2300 006466 000670 BR 1$
2301 006470 000000 7$: 0 ;DELAY COUNT
2302
2303 ;:*****
2304 ;ABORT
2305 ;THIS TEST WILL CHECK AN 'ABORT OUT' COMMAND WILL RETURN (KILL)
2306 ;THE BUFFER CURRENTLY QUEUED TO THE LINE UNDER TEST, BY INITIALIZING
2307 ;ALL LINES, SENDING A TRANSMIT BUFFER TO EACH LINE, THEN ISSUING
2308 ;A LINE ABORT OUT COMMAND TO EACH LINE INDIVIDUALLY, TESTING THAT
2309 ;ONLY THE BUFFER QUEUED TO THAT LINE IS RETURNED WITH A
2310 ;STATUS (SEL6) OF (-6) 'TERMINATED ON REQUEST'.
2311 ;:*****
2312 ;TEST 15
2313 ;:*****
2314 006472 000004 TST15: SCOPE
2315 006474 012737 000015 002460 MOV #15,TSTNUM ;LOAD THE WD OF THIS TEST
2316 006502 004737 030244 JSR PC,SETLIN ;SETUP THE FIRST LINE
2317 006506 012737 006514 001110 MOV #1$,SLPERR ;LOAD LOOP ADDRESS (LOOP ON ERROR)
2318 006514 012737 000001 002452 1$: MOV #1,ERNUM
2319 006522 005037 002462 CLR SHBE
```

```
2320 006526 004737 030664 JSR PC,INTKMC ;INITIALIZE KMC-11 FIRMWARE
2321 006532 000434 BR 20$ ;FIRMWARE FAILED TO INIT #=1
2322 006534 004737 031120 JSR PC,INTALL ;INITIALIZE ALL LINES
2323 006540 000431 BR 20$ ;INITIALIZE LINES FAILED #=1
2324 006542 004737 031226 JSR PC,CLKALL ;START ALL CLOCK LINES THAT ARE ENABLED
2325 006546 000426 BR 20$ ;CLOCK COMMAND FAILED
2326 006550 005237 002452 INC ERNUM
2327 006554 004737 030244 JSR PC,SETLIN ;INIT LINE
2328 006560 2$: ;
2329 006560 012737 052260 002522 MOV #TBUF,BUFADR ;LOAD TRANSMIT BUFFER
2330 006566 012737 010777 002526 MOV #10777,BUFCNT ;LARGE BUFFER
2331
2332 006574 043737 002330 002526 BIC MODMSK,BUFCNT
2333 006602 005037 002524 CLR BUFEA
2334 006606 004737 031354 JSR PC,XIN ;LOAD A TRANSMIT BUFFER IN
2335 006612 000404 BR 20$ ;BUFFER IN FAILED TO LOAD #=2
2336 006614 004737 031650 JSR PC,GETLIN ;GET THE NEXT LINE NUMBER
2337 006620 000403 BR 27$ ;ALL LINES TESTED
2338 006622 000756 BR 2$ ;TEST NEXT LINE
2339 006624 000137 007026 20$: JMP 8$ ;BRANCH HELP TO ERROR
2340 ;ABORT EACH BUFFER
2341 006630 004737 030244 JSR PC,SETLIN
2342 006634 012737 000003 002452 27$: MOV #3,ERNUM
2343 006642 004737 031010 3$: JSR PC,REQNX ;CAN A CONTROL OUT BE EXECUTED?
2344 006646 000766 BR 20$ ;HUNG OR OUTPUT PENDING #=3
2345 006650 005237 002452 INC ERNUM
2346 006654 113777 002454 173404 MOVB CURLIN,@SEL3 ;LOAD LINE #
2347 006662 112777 000002 173406 MOVB #2,@SEL7 ;ABORT LINE
2348 006670 005077 173374 CLR @SEL4 ;UNUSED
2349 006674 105077 173374 CLRB @SEL6 ;UNUSED
2350 ;WAIT FOR ABORT COMMAND TO COMPLETE
2351 006700 004737 031772 JSR PC,INPDUN ;TEST THAT INPUT COMPLETED
2352 006704 000747 BR 20$ ;COMMAND FAILED TO COMPLETE #=4
2353 ;TEST THAT BUFFER IS RETURNED
2354 006706 005237 002452 5$: INC ERNUM
2355 006712 005237 002452 INC ERNUM ; (ACCOUNT FOR EXTRA ERROR ON RECEIVE)
2356 006716 112737 000200 002462 MOVB #200,SHBE ;COMMAND BITS=0 + OUT I/O=0
2357 006724 004737 031742 JSR PC,REQOUT ;TEST REQUEST OUT OF KMC SETS
2358 006730 000436 BR 8$ ;NO ABORTED BUFFER RETURNED #=6
2359 006732 005237 002452 7$: INC ERNUM
2360 006736 122777 000200 173320 CMPB #200,@SEL2 ;BUFFER OUT + BIT2=0
2361 006744 001030 BNE 8$ ;B=SEL2 NOT CORRECT, ERROR #=7
2362 006746 005237 002452 INC ERNUM
2363 006752 113737 002454 002462 MOVB CURLIN,SHBE
2364 006760 123777 002454 173300 CMPB CURLIN,@SEL3 ;CORRECT LINE #?
2365 006766 001017 BNE 8$ ;B=N ERROR #=10
2366 006770 005237 002452 INC ERNUM
2367 006774 112737 177772 002462 MOVB #-6,SHBE
2368 007002 122777 177772 173264 CMPB #-6,@SEL6 ;CORRECT STATUS RETURNED?
2369 007010 001006 BNE 8$ ;B=N ERROR #=11
2370 007012 105077 173246 CLRB @SEL2 ;COMMAND DONE
2371 ;TEST FOR LINES TO ABORT
2372 007016 004737 031650 JSR PC,GETLIN ;GET THE NEXT LINE NUMBER
2373 007022 000407 BR TST16 ;ALL LINES TESTED
2374 007024 000703 BR 3$
2375 007026 8$: ;REPORT ERROR
```

```
2376 007026 004737 031470      JSR    PC,GETCSR      ;GET CONTENTS OF THE CSRS
2377 007032 104015      ERROR+ 15
2378 007034 105077 173224      CLRB   @SEL2
2379 007040 000625      BR      1$

2380
2381
2382      ;*****
2383      ;ABORT
2384      ;THIS TEST WILL CHECK AN 'ABORT IN' COMMAND WILL RETURN (KILL)
2385      ;THE BUFFER CURRENTLY QUEUED TO THE LINE UNDER TEST BY INITIALIZING
2386      ;ALL LINES, SENDING A RECEIVE BUFFER TO EACH LINE, THEN ISSUING
2387      ;A LINE 'ABORT IN' COMMAND TO EACH LINE INDIVIDUALLY, TESTING THAT
2388      ;ONLY THE BUFFER QUEUED TO THAT LINE IS RETURNED WITH A
2389      ;STATUS (SEL6) OF (-6) 'TERMINATED ON REQUEST'.
2390      ;*****
2391      ;TEST 16
2392      ;*****
2393 007042 000004      TST16: SCOPE
2394 007044 012737 000016 002460      MOV    #16,TSTNUM      ;LOAD THE WD OF THIS TEST
2395 007052 004737 030244      JSR    PC,SETLIN      ;SETUP THE FIRST LINE
2396 007056 012737 007064 001110      MOV    #1$,SLPERR      ;LOAD LOOP ADDRESS (LOOP ON ERROR)
2397 007064 012737 000001 002452 1$:  MOV    #1,ERNUM
2398 007072 005037 002462      CLR     SHBE
2399 007076 004737 030664      JSR    PC,INTKMC      ;INITIALIZE KMC-11 FIRMWARE
2400 007102 000434      BR      20$      ;FIRMWARE FAILED TO INIT #=1
2401 007104 004737 031120      JSR    PC,INTALL      ;INITIALIZE ALL LINES
2402 007110 000431      BR      20$      ;INITIALIZE LINES FAILED #=1
2403 007112 004737 031226      JSR    PC,CLKALL      ;START ALL CLOCK LINES THAT ARE ENABLED
2404 007116 000426      BR      20$      ;CLOCK COMMAND FAILED
2405 007120 005237 002452      INC     ERNUM
2406 007124 004737 030244      JSR    PC,SETLIN      ;INIT LINE
2407 007130
2408 007130 012737 052660 002522 2$:  MOV    #RBUF,BUFADR      ;LOAD RECEIVE BUFFER
2409 007136 012737 010100 002526      MOV    #10100,BUFCNT
2410
2411 007144 043737 002330 002526      BIC     MODMSK,BUFCNT
2412 007152 005037 002524      CLR     BUFEA
2413 007156 004737 031344      JSR    PC,RIN
2414 007162 000404      BR      20$      ;LOAD A RECEIVE BUFFER IN
2415 007164 004737 031650      JSR    PC,GETLIN      ;BUFFER IN FAILED TO LOAD #=2
2416 007170 000403      BR      27$      ;GET THE NEXT LINE NUMBER
2417 007172 000756      BR      2$      ;ALL LINES TESTED
2418 007174 000137 007444      BR      2$      ;TEST NEXT LINE
2419      JMP     8$      ;BRANCH HELP TO ERROR
2420 007200 004737 030244      ;ABORT EACH BUFFER
2421 007204 012737 000003 002452 27$: JSR    PC,SETLIN
2422 007212 004737 031000 3$:  MOV    #3,ERNUM
2423 007216 000766      JSR    PC,REQNC      ;CAN A CONTROL IN BE EXECUTED?
2424 007220 005237 002452      BR      20$      ;HUNG OR OUTPUT PENDING #=3
2425 007224 113777 002454 173034      INC     ERNUM
2426 007232 112777 000002 173036      MOVB   CURLIN,@SEL3      ;LOAD LINE #
2427 007240 005077 173024      MOVB   #2,@SEL7      ;ABORT LINE
2428 007244 105077 173024      CLR     @SEL4      ;UNUSED
2429      CLR    @SEL6      ;UNUSED
2430      ;WAIT FOR ABORT COMMAND TO COMPLETE
2431 007250 004737 031772      JSR    PC,INPDUN
2432 007254 000747      BR      20$      ;TEST THAT INPUT COMPLETED
2433      ;COMMAND FAILED TO COMPLETE #=4
```

C7KMEA0 DMS11-DA DYNAMIC
CZKMEA.P11 20-OCT-81 17:05

MACY11 30A(1052) 21-OCT-81 08:50 PAGE 52
GET VALUE FOR SOFTWARE SWITCH REGISTER

SEQ 0052

```
2432      ;TEST THAT BUFFER IS RETURNED
2433 007256 005237 002452      5$: INC ERNUM
2434 007262 105737 002311      TSTB MODE
2435 007266 100406      BMI 6$
2436      ;ONLY IN CCP MODE, CHARACTER MUST BE WRAPPED AROUND AND
2437      ;RECEIVED IN ORDER FOR ABORT TO BE PROCESSED
2438 007270 012737 033002 002522 MOV #TERBUF, BUFADR
2439 007276 004737 031354      JSR PC, XIN      ;LOAD A TRANSMIT BUFFER IN
2440 007302 000460      BR 8$      ;PRIMING TRANSMIT BUFFER IN FAILED TO COMPLETE #=5
2441 007304
2442 007304 005237 002452      6$: INC ERNUM      ; (ACCOUNT FOR EXTRA ERROR ON RECEIVE)
2443 007310 112737 000204 002462 MOVB #204, SHBE      ;COMMAND BITS=0 + OUT I/O=1
2444 007316 004737 031742      JSR PC, REQOUT      ;TEST REQUEST OUT OF KMC SETS
2445 007322 000450      BR 8$      ;NO ABORTED BUFFER RETURNED #=6
2446 007324 005237 002452      7$: INC ERNUM
2447 007330 122777 000204 172726 CMPB #204, @SEL2      ;BUFFER OUT + BIT2=1
2448 007336 001042      BNE 8$      ;B=SEL2 NOT CORRECT, ERROR #=7
2449 007340 005237 002452      INC ERNUM
2450 007344 113737 002454 002462 MOVB CURLIN, SHBE
2451 007352 123777 002454 172706 CMPB CURLIN, @SEL3      ;CORRECT LINE #?
2452 007360 001031      BNE 8$      ;B=N ERROR #=10
2453 007362 005237 002452      INC ERNUM
2454 007366 112737 177772 002462 MOVB #-6, SHBE
2455 007374 122777 177772 172672 CMPB #-6, @SEL6      ;CORRECT STATUS RETURNED?
2456 007402 001020      BNE 8$      ;B=N ERROR #=11
2457 007404 105077 172654      CLRB @SEL2      ;COMMAND DONE
2458 007410 005237 002452      INC ERNUM
2459 007414 105737 002311      TSTB MODE
2460 007420 100405      BMI 9$
2461 007422 004737 031742      JSR PC, REQOUT      ;TEST REQUEST OUT OF KMC SETS
2462 007426 000406      BR 8$      ;PRIMING TRANSMIT BUFFER FAILED TO RETURN #=12
2463 007430 105077 172630      CLRB @SEL2      ;DISMISS RETURNING XMIT BUFFER IN CCP MODE
2464 007434
2465      9$: ;TEST FOR LINES TO ABORT
2466 007434 004737 031650      JSR PC, GETLIN      ;GET THE NEXT LINE NUMBER
2467 007440 000407      BR TST17      ;:ALL LINES TESTED
2468 007442 000660
2469 007444
2470 007444 004737 031470      8$: ;REPORT ERROR
2471 007450 104016      JSR PC, GETCSR      ;GET CONTENTS OF THE CSRS
2472 007452 105077 172606      ERROR+ 16
2473 007456 000602      CLRB @SEL2
2474      BR 1$
2475
2476      ;:*****
2477      ;ALTSTA
2478      ;THIS TEST CHECKS THAT AN ALTERNATE TRANSMIT BUFFER CAN BE QUEUED
2479      ;TO EACH LINE AND IS RETURNED WHEN A LINE ABORT COMMAND
2480      ;IS ISSUED. AFTER EACH LINE IS TESTED FOR RETURNING ITS
2481      ;BUFFERS, AN ALTERNATE BUFFER IS QUEUED TO THE LINE
2482
2483      ;:*****
2484
2485      ;TEST 17
2486      ;:*****
2487 007460 000004      TST17: SCOPE
```

```
2488 007462 012737 000017 002460      MOV      #17,TSTNUM      ;LOAD THE WD OF THIS TEST
2489 007470 004737 030244      JSR      PC,SETLIN      ;SETUP THE FIRST LINE
2490 007474 012737 007502 001110      MOV      #1$,SLPERR      ;LOAD LOOP ADDRESS (LOOP ON ERROR)
2491 007502 004737 030244      JSR      PC,SETLIN
2492 007506 012737 000001 002452      MOV      #1,ERNUM
2493 007514 005037 002462      CLR      SHBE
2494 007520 004737 030664      JSR      PC,INTKMC      ;INITIALIZE KMC-11 FIRMWARE
2495 007524 000447      BR      20$      ;UNIT FAILED TO INITIALIZE #=1
2496 007526 004737 031120      JSR      PC,INTALL      ;INITIALIZE ALL LINES
2497 007532 000444      BR      20$      ;LINE INITIALIZE FAILED #=1
2498 007534 004737 031226      JSR      PC,CLKALL      ;START ALL CLOCK LINES THAT ARE ENABLED
2499 007540 000441      BR      20$      ;CLOCK COMMAND FAILED
2500 007542 004737 030244      JSR      PC,SETLIN      ;INITIALIZE LINE #
2501 007546 005237 002452      INC      ERNUM
2502 007552 012737 017777 002526      MOV      #17777,BUFCNT      ;LARGE BUFFERS
2503 007560 043737 002330 002526      BIC      MODMSK,BUFCNT
2504 007566 005037 002524      CLR      BUFEA
2505 007572 012737 052260 002522      MOV      #TBUF,BUFADR
2506 007600      ;1 BUFFER TO EACH LINE
2507 007600 004737 031354      JSR      PC,XIN      ;LOAD A TRANSMIT BUFFER IN
2508 007604 000417      BR      20$      ;BUFFER FAILED TO LOAD #=2
2509 007606 004737 031650      JSR      PC,GETLIN      ;GET THE NEXT LINE NUMBER
2510 007612 000401      BR      25$
2511 007614 000771      BR      2$      ;DONE ALL LINES
2512 007616 004737 030244      JSR      PC,SETLIN      ;DO NEXT LINE
2513 007622 005237 002452      INC      ERNUM
2514 007626      3$:      ;LOAD ALT TO LINE
2515
2516 007626 004737 031354      JSR      PC,XIN      ;LOAD A TRANSMIT BUFFER IN
2517 007632 000404      BR      20$      ;BUFFER IN FAILED TO COMPLETE #=3
2518 007634 004737 031650      JSR      PC,GETLIN      ;GET THE NEXT LINE NUMBER
2519 007640 000403      BR      31$
2520 007642 000771      BR      3$
2521 007644 000137 010106      JMP      9$      ;FORM ERROR JUMP HELP
2522 007650 004737 030244      JSR      PC,SETLIN
2523      ;ISSUE 'LINE ABORT' TO THE CURRENT LINE
2524 007654 012737 000004 002452      MOV      #4,ERNUM
2525 007662 005037 010120      CLR      10$
2526 007666      5$:      ;SOFT SWITCH TO INDICATE WHICH BUFFER IS UNDER TEST
2527 007666 004737 031010      JSR      PC,REQNX      ;CAN A CONTROL OUT BE EXECUTED?
2528 007672 000764      BR      20$      ;OUTPUT REQ. OR FIRMWARE HUNG #=4,14
2529 007674 005237 002452      INC      ERNUM
2530 007700 113777 002454 172360      MOV      CURLIN,@SEL3
2531 007706 112777 000002 172362      MOV      #2,@SEL7
2532 007714 005077 172350      CLR      @SEL4
2533 007720 105077 172350      CLR      @SEL6
2534 007724 004737 031772      JSR      PC,INPDUN
2535 007730 000745      BR      20$
2536 007732 005237 002452      INC      ERNUM
2537 007736 005237 002452      INC      ERNUM
2538 007742 012737 000200 002462      MOV      #200,SHBE
2539 007750 004737 031742      JSR      PC,REQOUT
2540 007754 000454      BR      9$
2541 007756 005237 002452      INC      ERNUM
2542      ;TEST THAT OUTPUT IS A BUFFER OUT + REFLECTS CORRECT I/O BIT
2543 007762 123777 002462 172274      CMP      SHBE,@SEL2      ;CORRECT STATUS?
```

```
2544 007770 001046      BNE 9$ ;B=WRONG STATUS IN SEL2 ERROR #=10,20
2545 007772 005237 002452 INC ERNUM
2546 ;TEST THAT OUTPUT BUFFER HAS CORRECT LINE #
2547 007776 113737 002454 002462 MOVB CURLIN,SHBE
2548 010004 123777 002454 172254 CMPB CURLIN,@SEL3 ;CORRECT LINE #?
2549 010012 001035      BNE 9$ ;B=N WRONG LINE # RETURNED ERROR #=11,21
2550 010014 005237 002452 INC ERNUM
2551 ;TEST THAT STATUS (SEL6) IS A 'TERMINATED ON REQUEST' STATUS
2552 010020 112737 177772 002462 MOVB #-6,SHBE
2553 010026 122777 177772 172240 CMPB #-6,@SEL6 ;CORRECT STATUS
2554 010034 001024      BNE 9$ ;B=WRONG STATUS - ERROR #=12,22
2555 010036 005237 002452 INC ERNUM
2556 010042 105077 172216 CLRB @SEL2 ;COMMAND DONE
2557 010046 005237 002452 INC ERNUM ;(ALLOW FOR EXTRA ERROR IN RX TEST)
2558 010052 005137 010120 COM 10$ ;1ST BUFFER OR 2ND?
2559 010056 001303      BNE 5$ ;B=1ST, DO 2ND
2560 010060 005237 002452 INC ERNUM
2561 010064 005037 002462 CLR SHBE
2562 ;NOW LOAD ANOTHER ALTERNATE BUFFER
2563 010070 004737 031354 JSR PC,XIN ;LOAD A TRANSMIT BUFFER IN
2564 010074 000404      BR 9$ ;ALT BUFFER FAILED TO LOAD #=24
2565 010076 ;8$: ;CHECK FOR MORE LINES TO TEST
2566 010076 004737 031650 JSR PC,GETLIN ;GET THE NEXT LINE NUMBER
2567 010102 ;81$:
2568 010102 000410      BR TST20 ;:ALL LINES TESTED
2569 010104 000663      BR 4$ ;DO NEXT LINE
2570
2571 010106 ;9$: ;REPORT ERRORS
2572 010106 004737 031470 JSR PC,GETCSR ;GET CONTENTS OF THE CSRS
2573 010112 104017      ERROR+ 17
2574 010114 000137 007502 JMP 1$
2575 010120 000000 ;10$: 0 ;SWITCH
2576 010122 000000 ;15$: 0 ;DELAY COUNT
2577
2578 ;:*****
2579 ;ALTSTA
2580 ;THIS TEST CHECKS THAT AN ALTERNATE RECEIVE BUFFER CAN BE QUEUED
2581 ;TO EACH LINE AND IS RETURNED WHEN A LINE ABORT COMMAND
2582 ;IS ISSUED. AFTER EACH LINE IS TESTED FOR RETURNING ITS
2583 ;BUFFERS, AN ALTERNATE BUFFER IS QUEUED TO THE LINE
2584
2585 ;:*****
2586
2587 ;TEST 20
2588 ;:*****
2589 010124 000004      TST20: SCOPE
2590 010126 012737 000020 002460 MOV #20,TSTNUM ;LOAD THE WD OF THIS TEST
2591 010134 004737 030244 JSR PC,SETLIN ;SETUP THE FIRST LINE
2592 010140 012737 010146 001110 MOV #1$,SLPERR ;LOAD LOOP ADDRESS (LOOP ON ERROR)
2593 010146 004737 030244 ;1$: JSR PC,SETLIN
2594 010152 012737 000001 002452 MOV #1,ERNUM
2595 010160 005037 002462 CLR SHBE
2596 010164 004737 030664 JSR PC,INTKMC ;INITIALIZE KMC-11 FIRMWARE
2597 010170 000447      BR 20$ ;UNIT FAILED TO INITIALIZE #=1
2598 010172 004737 031120 JSR PC,INTALL ;INITIALIZE ALL LINES
2599 010176 000444      BR 20$ ;LINE INITIALIZE FAILED #=1
```

```
2600 010200 004737 031226 JSR PC,CLKALL ;START ALL CLOCK LINES THAT ARE ENABLED
2601 010204 000441 BR 20$ ;CLOCK COMMAND FAILED
2602 010206 004737 030244 JSR PC,SETLIN ;INITIALIZE LINE #
2603 010212 005237 002452 INC ERNUM
2604 010216 012737 017777 002526 MOV #17777,BUFCNT ;LARGE BUFFERS
2605 010224 043737 002330 002526 BIC MODMSK,BUFCNT
2606 010232 005037 002524 CLR BUFEA
2607 010236 012737 052660 002522 MOV #RBUF,BUFADR
2608 010244 2$: ;1 BUFFER TO EACH LINE
2609 010244 004737 031344 JSR PC,RIN ;LOAD A RECEIVE BUFFER IN
2610 010250 000417 BR 20$ ;BUFFER FAILED TO LOAD #=2
2611 010252 004737 031650 JSR PC,GETLIN ;GET THE NEXT LINE NUMBER
2612 010256 000401 BR 25$ ;DONE ALL LINES
2613 010260 000771 BR 2$ ;DO NEXT LINE
2614 010262 004737 030244 25$: JSR PC,SETLIN
2615 010266 005237 002452 INC ERNUM
2616 010272 3$: ;LOAD ALT TO LINE
2617
2618 010272 004737 031344 JSR PC,RIN ;LOAD A RECEIVE BUFFER IN
2619 010276 000404 BR 20$ ;BUFFER IN FAILED TO COMPLETE #=3
2620 010300 004737 031650 JSR PC,GETLIN ;GET THE NEXT LINE NUMBER
2621 010304 000403 BR 31$
2622 010306 000771 BR 3$
2623 010310 000137 010634 20$: JMP 9$ ;FORM ERROR JUMP HELP
2624 010314 004737 030244 31$: JSR PC,SETLIN
2625 ;ISSUE 'LINE ABORT' TO THE CURRENT LINE
2626 010320 012737 000004 002452 4$: MOV #4,ERNUM
2627 010326 005037 010646 CLR 10$ ;SOFT SWITCH TO INDICATE WHICH BUFFER IS UNDER TEST
2628 010332 5$:
2629 010332 004737 031000 JSR PC,REQNC ;CAN A CONTROL IN BE EXECUTED?
2630 010336 000764 BR 20$ ;OUTPUT REQ. OR FIRMWARE HUNG #=4,14
2631 010340 005237 002452 INC ERNUM
2632 010344 113777 002454 171714 MOVB CURLIN,@SEL3
2633 010352 112777 000002 171716 MOVB #2,@SEL7 ;ABORT BUFFER
2634 010360 005077 171704 CLR @SEL4 ;UNUSED
2635 010364 105077 171704 CLRB @SEL6 ;UNUSED
2636 010370 004737 031772 JSR PC,INPDUN ;TEST THAT INPUT COMPLETED
2637 010374 000745 BR 20$ ;LINE ABORT COMMAND FAILED TO COMPLETE#=5,15
2638 010376 005237 002452 INC ERNUM ; (ALLOW FOR EXTRA ERROR IN RECEIVE TEST)
2639 010402 105737 002311 TSTB MODE
2640 010406 100406 BMI 6$
2641 ;IN CCP MODE ONLY, A CHARACTER MUST BE WRAPPED AROUND AND
2642 ;RECEIVED IN ORDER FOR ABORT TO BE PROCESSED.
2643 010410 012737 033002 002522 MOV #TERBUF,BUFADR
2644 010416 004737 031354 JSR PC,XIN ;LOAD A TRANSMIT BUFFER IN
2645 010422 000504 BR 9$ ;PRIMING TRANSMIT BUFFER FAILED TO COMPLETE #=6,16
2646 010424 6$:
2647 010424 005237 002452 INC ERNUM
2648 010430 012737 000204 002462 MOV #204,SHBE
2649 010436 004737 031742 JSR PC,REQOUT ;TEST REQUEST OUT OF KMC SETS
2650 010442 000474 BR 9$ ;NO OUTPUT AFTER ABORTING BUFFERS #=7,17
2651 010444 005237 002452 65$: INC ERNUM
2652 ;TEST THAT OUTPUT IS A BUFFER OUT + REFLECTS CORRECT I/O BIT
2653 010450 123777 002462 171606 CMPB SHBE,@SEL2 ;CORRECT STATUS?
2654 010456 001066 BNE 9$ ;B=WRONG STATUS IN SEL2 ERROR #=10,20
2655 010460 005237 002452 INC ERNUM
```

```
2656 ;TEST THAT OUTPUT BUFFER HAS CORRECT LINE #
2657 010464 113737 002454 002462 MOVB CURLIN,SHBE
2658 010472 123777 002454 171566 CMPB CURLIN,@SEL3 ;CORRECT LINE #?
2659 010500 001055 BNE 9$ ;B=N WRONG LINE # RETURNED ERROR #=11,21
2660 010502 005237 002452 INC ERNUM
2661 ;TEST THAT STATUS (SEL6) IS A "TERMINATED ON REQUEST" STATUS
2662 010506 112737 177772 002462 MOVB #-6,SHBE
2663 010514 122777 177772 171552 CMPB #-6,@SEL6 ;CORRECT STATUS
2664 010522 001044 BNE 9$ ;B=WRONG STATUS - ERROR #=12,22
2665 010524 005237 002452 INC ERNUM
2666 010530 105077 171530 CLRB @SEL2 ;COMMAND DONE
2667 010534 105737 002311 TSTB MODE
2668 010540 100415 BMI 7$
2669 010542 012737 000127 010650 MOV #127,15$ ;SET UP WAIT LOOP
2670 010550 11$: JSR PC,REQOUT ;TEST REQUEST OUT OF KMC SETS
2671 010550 004737 031742 BR 12$
2672 010554 000401 BR 13$
2673 010556 000404 BR 15$
2674 010560 005337 010650 12$: DEC 15$
2675 010564 001371 BNE 11$
2676 010566 000422 BR 9$ ;PRIMING XMIT BUFFER NOT RETURNED #-13,23
2677 010570 105077 171470 13$: CLRB @SEL2 ;ACCOUNT FOR RETURNING XMIT BUFFER
2678 010574 7$:
2679 010574 005237 002452 INC ERNUM ;(ALLOW FOR EXTRA ERROR IN RX TEST)
2680 010600 005137 010646 COM 10$ ;1ST BUFFER OR 2ND?
2681 010604 001252 BNE 5$ ;B=1ST, DO 2ND
2682 010606 005237 002452 INC ERNUM
2683 010612 005037 002462 CLR SHBE
2684 ;NOW LOAD ANOTHER ALTERNATE BUFFER
2685 010616 004737 031344 JSR PC,RIN ;LOAD A RECEIVE BUFFER IN
2686 010622 000404 BR 9$ ;ALT BUFFER FAILED TO LOAD #-24
2687 010624 8$: ;CHECK FOR MORE LINES TO TEST
2688 010624 004737 031650 JSR PC,GETLIN ;GET THE NEXT LINE NUMBER
2689 010630 81$:
2690 010630 000410 BR TST21 ;:ALL LINES TESTED
2691 010632 000632 BR 4$ ;DO NEXT LINE
2692
2693 9$: ;REPORT ERRORS
2694 010634 004737 031470 JSR PC,GETCSR ;GET CONTENTS OF THE CSRS
2695 010640 104020 ERROR+ 20
2696 010642 000137 010146 JMP 1$
2697 010646 000000 10$: 0 ;SWITCH
2698 010650 000000 15$: 0 ;DELAY COUNT
2699
2700 ;:*****
2701 ;ALTBUFFS
2702 ;THIS TEST CHECKS THAT ALTERNATE BUFFERS CAN BE QUEUED TO EACH
2703 ;LINE AND ARE PROPERLY HANDLED BY THE FIRMWARE. A PROGRESSION IS
2704 ;DONE, TESTING THE QUEUEING OF ALTERNATE BUFFERS TO EACH LINE.
2705 ;THE CURRENT LINE IS QUEUED 2 RECEIVE AND 2 TRANSMIT BUFFERS
2706 ;AND A TEST THAT THE BUFFERS FOR THE CURRENT LINE ARE RETURNED CORRECTLY.
2707
2708 ;:*****
2709
2710 ;TEST 21
2711 ;:*****
```

```
2712 010652 000004 TST21: SCOPE
2713 010654 012737 000C21 002460 MOV #21,TSTNUM ;LOAD THE WD OF THIS TEST
2714 010662 004737 030244 JSR PC,SETLIN ;SETUP THE FIRST LINE
2715 010666 012737 010674 001110 MOV #1$,SLPERR ;LOAD LOOP ADDRESS (LOOP ON ERROR)
2716 010674 012737 000001 002452 1$: MOV #1,ERNUM
2717 010702 005037 002462 CLR SHBE
2718 010706 004737 030664 JSR PC,INTKMC ;INITIALIZE KMC-11 FIRMWARE
2719 010712 000452 BR 65$ ;KMC FAILED TO INITIALIZE #=1
2720 010714 004737 031120 JSR PC,INTALL ;INITIALIZE ALL LINES
2721 010720 000447 BR 65$ ;LINE INITIALIZE FAILED #=1
2722 010722 004737 031274 JSR PC,STCLK ;SET MAINT LOOP + START CLOCK
2723 010726 000444 BR 65$ ;MAINTENANCE COMMAND NOT ISSUED #=1
2724 010730 004737 031714 JSR PC,SYNTIM ;ALLOW TIME FOR SYNC
2725 010734 005237 002452 INC ERNUM
2726 010740 005037 002524 CLR BUFEA
2727 010744 012737 010020 002526 MOV #10020,BUFCNT ;RECEIVE BUFFER LONGER THAN XMIT BUFFER
2728 010752 043737 002330 002526 BIC MODMSK,BUFCNT
2729 010760 005037 011402 4$: CLR 14$ ;TIMES SWITCH
2730 010764 012737 052660 002522 MOV #RBUF,BUADR
2731 010772 5$:
2732 010772 004737 031344 JSR PC,RIN ;LOAD A RECEIVE BUFFER IN
2733 010776 000567 BR 11$ ;B=RECEIVE BUFFER FAILED TO LOAD #=2
2734 011000 005137 011402 COM 14$ ;1ST OR 2ND BUFFER IN
2735 011004 001372 BNE 5$
2736 011006 005237 002452 INC ERNUM
2737 011012 012737 052260 002522 MOV #TBUF,BUADR ;FOR COMPATABILITY WITH CCP MODE, USE BUFFER
2738 011020 012737 010005 002526 MOV #10005,BUFCNT ;STARTING WITH SOH & ENDING WITH ETX
2739 011026 043737 002330 002526 BIC MODMSK,BUFCNT
2740 011034 6$:
2741 011034 004737 031354 JSR PC,XIN ;LOAD A TRANSMIT BUFFER IN
2742 011040 000546 65$: BR 11$ ;B=TRANSMIT BUFFER FAILED TO LOAD #=3
2743 011042 005137 011402 COM 14$ ;1ST OR 2ND BUFFER IN
2744 011046 001372 BNE 6$
2745 ;NOW TEST THAT THE BUFFERS QUEUED TO THE CURRENT LINE RETURN CORRECTLY
2746 ;THERE MUST BE 2 RECEIVE AND 2 TRANSMIT
2747 011050 005037 011402 CLR 14$ ;COUNT TOTAL BUFFERS
2748 011054 005037 011400 CLR 13$ ;COUNT RECEIVE BUFFERS
2749 011060 012737 000004 002452 9$: MOV #4,ERNUM
2750 011066 005037 002462 CLR SHBE
2751 011072 004737 031742 JSR PC,REQOUT ;TEST REQUEST OUT OF KMC SETS
2752 011076 000527 BR 11$ ;NO OUTPUT REQ. ERR #=4
2753 011100 005237 002452 INC ERNUM
2754 011104 013737 002454 002462 MOV CURLIN,SHBE ;SHOW CURRENT LINE #
2755 011112 123777 002462 171146 CMPB SHBE,@SEL3 ;CORRECT LINE #?
2756 011120 001116 BNE 11$ ;B=NO ERROR #=5
2757 011122 005237 002452 INC ERNUM
2758 011126 012737 000001 002462 MOV #1,SHBE ;RETURN STATUS SHOULD BE = 1 (END COUNT REACHED)
2759 ;FOR XMIT AND 2 (EOM/ETX) FOR RECV
2760 011134 132777 000004 171122 BITB #4,@SEL2 ;CHECK FOR RECV
2761 011142 001402 BEQ 91$ ;SKIP IF XMIT
2762 011144 005237 002462 INC SHBE ;RECV=2
2763 011150 123777 002462 171116 91$: CMPB SHBE,@SEL6 ;CORRECT STATUS?
2764 011156 001077 BNE 11$ ;B=N #=6
2765 011160 005237 002452 INC ERNUM
2766 ;FIND WHICH ENDING ADDRESS SHOULD BE TESTED
2767 011164 132777 000004 171072 BITB #4,@SEL2
```

```
2768 011172 001420      BEQ      15$      ;B=TRANSMIT BUFFER
2769 011174 012737 052665 002462  MOV     #RBUF+5,SHBE ;RECEIVE BUFFER
2770 011202 105737 002311      TSTB     MODE
2771 011206 100015      BPL      18$
2772 011210 105737 002316      TSTB     CLKDAT      ;CHECK INTERNAL OR EXTERNAL
2773 011214 100415      BMI      16$
2774 011216 105737 002314      TSTB     CRC32F      ;EXTERNAL, CHECK CRC-32 ENABLED?
2775 011222 100012      BPL      16$
2776 011224 062737 000004 002462  ADD     #4,SHBE      ;YES, ADD 4 BYTES FOR APPENDED CRC
2777 011232 000406      BR       16$
2778 011234 012737 052265 002462 15$: MOV     #TBUF+5,SHBE ;TRANSMIT BUFFER
2779 011242 105737 002311 18$: TSTB     MODE      ;CHECK MODE
2780 011246 100003      BPL      19$      ;SKIP IF CCP
2781 011250 062737 000005 002462 16$: ADD     #5,SHBE      ;IF BOP, CONVERT WORD COUNT INTO BYTES
2782 011256 023777 002462 171004 19$: CMP     SHBE,@SEL4    ;CORRECT ADDRESS?
2783 011264 001034      BNE      11$      ;B=NO FRR #=7
2784 011266 132777 000004 170770  BITB     #4,@SEL2    ;WHAT BUFFER?
2785 011274 001002      BNE      10$      ;B=XMIT
2786 011276 005237 011400      INC      13$      ;1 TO THE REC.
2787 011302 005237 011402      INC      14$      ;1 TO THE TOTAL
2788 011306 005077 170752      CLR      @SEL2    ;SHOW COMMAND DONE
2789 011312 005237 002452      INC      ERNUM
2790 011316 013737 011400 002462  MOV     13$,SHBE      ;SHOW # OF REC. BUFFERS
2791 011324 022737 000004 011402  CMP     #4,14$      ;4 YET?
2792 011332 001252      BNE      9$       ;B=N TRY AGAIN
2793 011334 022737 000002 011400  CMP     #2,13$      ;ARE THERE 2 REC. + 2 XMIT?
2794 011342 001005      BNE      11$      ;REPORT ERR #=10
2795 011344 005237 002452      INC      ERNUM
2796 011350 004737 031742      JSR      PC,REQOUT    ;TEST REQUEST OUT OF KMC SETS
2797 011354 000405      BR       17$      ;OK NO MORE OUTPUT REQUESTED #11=ADDITIONAL OUT REQ.
2798 011356      11$:  ;REPORT ERROR
2799 011356 004737 031470      JSR      PC,GETCSR    ;GET CONTENTS OF THE CSRS
2800 011362 104021      ERROR+ 21
2801 011364 000137 010674      12$: JMP      1$       ;LOOP @ 1$
2802 011370      17$:
2803 011370 004737 031650      JSR      PC,GETLIN    ;GET THE NEXT LINE NUMBER
2804 011374      81$:
2805 011374 000403      BR       TST22      ;:ALL LINES TESTED
2806 011376 000772      BR       12$
2807 011400 000000      13$: 0
2808 011402 000000      14$: 0      ;# OF RECEIVE BUFFERS RETURNED
                                   ;# OF TOTAL RECEIVED + COUNT FOR BUFFERS LOADED
2809
2810
2811 ;:*****
2812 ;:NXM
2813 ;:THIS TEST CHECKS THAT A NON-EXISTANT MEMORY STATUS (-5) WILL BE
2814 ;:RETURNED WITH A RECEIVE BUFFER OUT. THE NON-EXISTANT MEMORY
2815 ;:LOCATION IS I/O PAGE LOCATION 10 TO WHICH THE RECEIVE BUFFER IS
2816 ;:ADDRESSED.
2817
2818 ;:*****
2819
2820 ;:TEST 22
2821 ;:*****
2822 011404 000004      TST22: SCOPE
2823 011406 012737 000022 002460  MOV     #22,TSTNUM    ;LOAD THE WD OF THIS TEST
```

2824	011414	004737	030244		JSR	PC,SETLIN	;SETUP THE FIRST LINE
2825	011420	012737	011426	001110	MOV	#1\$,SLPERR	;LOAD LOOP ADDRESS (LOOP ON ERROR)
2826	011426	012737	000001	002452	MOV	#1,ERNUM	
2827	011434	005037	002462		CLR	SHBE	
2828	011440	004737	030664		JSR	PC,INTKMC	;INITIALIZE KMC-11 FIRMWARE
2829	011444	000504			BR	2\$	;FIRMWARE FAILED TO INITIALIZE #=1
2830	011446	004737	031204		JSR	PC,INTLNE	;INITIALIZE LINE (# IN CURLIN)
2831	011452	000501			BR	2\$	;LINE FAILED TO INITIALIZE #=1
2832	011454	004737	031274		JSR	PC,STCLK	;SET MAINT LOOP + START CLOCK
2833	011460	000476			BR	2\$	;MAINTENANCE COMMAND NOT ISSUED #=1
2834	011462	004737	031714		JSR	PC,SYNTIM	;ALLOW TIME FOR SYNC
2835	011466	005237	002452		INC	ERNUM	
2836							;LOAD THE RECEIVE BUFFER AT I/O PAGE ADDRESS 10
2837	011472	012737	010002	002526	MOV	#10002,BUFCNT	
2838	011500	043737	002330	002526	BIC	MODMSK,BUFCNT	
2839	011506	012737	140000	002524	MOV	#140000,BUFEA	;EA-BITS
2840	011514	012737	177770	002522	MOV	#-10,BUFADR	;I/O PAGE ADDR. 10
2841	011522	004737	031344		JSR	PC,RIN	;LOAD A RECEIVE BUFFER IN
2842	011526	000453			BR	2\$	;REC.BUFFER FAILED TO LOAD #=2
2843	011530	005237	002452		INC	ERNUM	
2844							;LOAD THE TRANSMIT BUFFER
2845	011534	012737	010004	002526	MOV	#10004,BUFCNT	
2846	011542	043737	002330	002526	BIC	MODMSK,BUFCNT	
2847	011550	005037	002524		CLR	BUFEA	
2848	011554	012737	052260	002522	MOV	#TBUF,BUFADR	
2849	011562	004737	031354		JSR	PC,XIN	;LOAD A TRANSMIT BUFFER IN
2850	011566	000433			BR	2\$	;TRANSMIT BUFFER FAILED TO LOAD #=3
2851	011570	005237	002452		INC	ERNUM	
2852	011574	012737	000204	002462	MOV	#204,SHBE	
2853							;TEST FOR AN OUTPUT RESPONSE
2854	011602	004737	031742		JSR	PC,REQOUT	;TEST REQUEST OUT OF KMC SETS
2855	011606	000423			BR	2\$	;NO OUTPUT RESPONSE #=4
2856	011610	005237	002452		INC	ERNUM	
2857	011614	123777	002462	170442	CMPB	SHBE,@SEL2	
2858	011622	001015			BNE	2\$	;B=RETURN NOT A RECEIVE BUFFER #=5
2859	011624	005237	002452		INC	ERNUM	
2860	011630	112737	177773	002462	MOVB	#-5,SHBE	
2861	011636	123777	002462	170430	CMPB	SHBE,@SEL6	;IS RETURNED STATUS A NON-EXISTANT MEMORY?
2862	011644	001004			BNE	2\$	;B = ERROR #6
2863	011646	004737	031650		JSR	PC,GETLIN	;GET THE NEXT LINE NUMBER
2864	011652	000405			BR	TST23	;EXIT
2865	011654	000664			BR	1\$	;TEST NEXT LINE
2866							
2867	011656			2\$:			
2868	011656	004737	031470		JSR	PC,GETCSR	;GET CONTENTS OF THE CSRS
2869	011662	104022			ERROR+	22	
2870	011664	000660			BR	1\$	
2871							
2872							
2873							
2874							
2875							
2876							
2877							
2878							
2879							

```

;*****
;TERMXB
;CCP MODE ONLY
;THIS TEST WILL CHECK THAT THE 2 CHARACTERS ETX AND ETB
;ARE DETECTED AND PROPERLY RETURN THE BUFFER OUT COMMAND STATUS.
;BOTH CHARACTERS ARE TESTED INDIVIDUALLY WITH EACH PREFIXED
;BY AN SOH THEN A STX

```

```
2880
2881
2882
2883
2884
2885 011666 000004
2886 011670 012737 000023 002460
2887 011676 004737 030244
2888 011702 012737 011740 001110
2889 011710 105737 002311
2890 011714 100002
2891 011716 000137 012230
2892 011722 005037 001160
2893 011726 005037 001162
2894 011732 012737 033002 012256
2895 011740 012737 000005 002526
2896 011746 005037 002524
2897 011752 012737 000001 002452
2898 011760 005037 002462
2899 011764 004737 030664
2900 011770 000522
2901 011772 004737 031204
2902 011776 000517
2903 012000 004737 031274
2904 012004 000514
2905 012006 004737 031714
2906 012012 005237 002452
2907 012016 012737 052660 002522
2908 012024 004737 031344
2909 012030 000502
2910 012032 005237 002452
2911 012036 013737 012256 002522
2912 012044 005337 002526
2913 012050 004737 031354
2914 012054 000470
2915 012056 005237 002452
2916 012062 004737 031714
2917 012066 004737 031742
2918 012072 000461
2919 012074 005237 002452
2920 012100 012737 000204 002462
2921 012106 004737 031522
2922 012112 123777 002462 170144
2923 012120 001046
2924 012122 005237 002452
2925 012126 012737 052664 002462
2926 012134 023777 002462 170126
2927 012142 001035
2928
2929 012144 005237 002452
2930 012150 012737 000002 002462
2931 012156 013700 012256
2932 012162 132760 000020 000003
2933 012170 001402
2934 012172 005237 002462
2935 012176 123777 002462 170070 15$:
```

```
*****
:*****
:TEST 23
:*****
TST23: SCOPE
MOV #23,TSTNUM ;LOAD THE WD OF THIS TEST
JSR PC,SETLIN ;SETUP THE FIRST LINE
MOV #1$,SLPERR ;LOAD LOOP ADDRESS (LOOP ON ERROR)
TSTB MODE ;CCP MODE ONLY
BPL 17$
JMP 2$
17$: CLR $TMP0 ;SWITCH FOR CHARACTER SOH & STX
CLR $TMP1 ;SWITCH FOR CHARACTER ETX & ETB
MOV #TERBUF,4$ ;
1$: MOV #5,BUFCNT
CLR BUFEA
MOV #1,ERNUM
CLR SHBE
JSR PC,INTKMC ;INITIALIZE KMC-11 FIRMWARE
BR 3$ ;FIRMWARE FAILED TO INITIALIZE #=1
JSR PC,INTLNE ;INITIALIZE LINE (# IN CURLIN)
BR 3$ ;LINE FAILED TO INITIALIZE #=1
JSR PC,STCLK ;SET MAINT LOOP + START CLOCK
BR 3$ ;MAINTENANCE COMMAND NOT ISSUED #=1
JSR PC,SYNTIM ;ALLOW TIME FOR SYNC
INC ERNUM
MOV #RBUF,BUFADR ;LOAD A RECEIVE BUFFER
JSR PC,RIN ;LOAD A RECEIVE BUFFER IN
BR 3$ ;RECEIVE BUFFER FAILED TO LOAD #=2
INC ERNUM
MOV 4$,BUFADR ;CURRENT BUFFER ADDRESS
DEC BUFCNT
JSR PC,XIN ;LOAD A TRANSMIT BUFFER IN
BR 3$ ;TRANSMIT BUFFER FAILED TO LOAD #=3
INC ERNUM
JSR PC,SYNTIM ;WAIT A WHILE
JSR PC,REQOUT ;TEST REQUEST OUT OF KMC SETS
BR 3$ ;NO RESPONSE FROM THE ENDING CHARACTER #=4
INC ERNUM
MOV #204,SHBE ;IS COMMAND A BUFFER OUT?
JSR PC,RBONLY ;SKIP XMIT
CMPB SHBE,@SEL2
BNE 3$ ;B=NO COMMAND IS NOT A RECEIVE BUFFER OUT #=5
INC ERNUM
MOV #RBUF+4,SHBE ;CORRECT ENDING MEMORY ADDRESS?
CMP SHBE,@SEL4
BNE 3$ ;B=NO ENDING MEMORY ADDRESS INCORRECT #=6
;FIND THE CORRECT RETURN STATUS VALUE
INC ERNUM
MOV #2,SHBE ;LOAD ETX RETURN STATUS VALUE
MOV 4$,R0
BITB #20,3(R0) ;WAS CHAR. ETX OR ETB
BEQ 15$ ;B=ETX
INC SHBE ;MAKE RET. STATUS AN ETB (3) DETECTED
CMPB SHBE,@SEL6 ;CORRECT TERMINATION STATUS
```

```
2936 012204 001014          BNE      3$          ;B=NO #=7
2937 012206 062737 000004 012256  ADD      #4,4$          ;MOVE TO NEXT VALUE
2938 012214 022737 033022 012256  CMP      #TERBUF+20,4$ ;DONE?
2939 012222 001246          BNE      1$          ;B=NO
2940 012224 004737 031650          JSR      PC,GETLIN      ;GET NEXT LINE NUMBER
2941 012230          2$:
2942 012230 000413          BR       TST24          ;:DONE
2943 012232 000137 011722          JMP      17$          ;DO NEXT LINE
2944
2945
2946          ;REPORT ERRORS
2947 012236 017737 000014 002464 3$:  MOV      @4$,WAS ;SHOW COMBINATION FAILING
2948 012244 004737 031470          JSR      PC,GETCSR      ;GET CONTENTS OF THE CSRS
2949 012250 104023          ERROR+  23
2950 012252 000137 011740          JMP      1$
2951 012256 000000          4$:  0          ;ADDRESS OF CURRENT TRANSMIT BUFFER
2952
2953
2954          ;:*****
2955          ;LRCST
2956          ;CCP MODE ONLY
2957          ;THIS TEST CHECKS THAT AN LRC ERROR CONDITION CAN BE DETECTED
2958          ;AND RETURNS THE RECEIVE BUFFER WITH AN LRC CONDITION.
2959          ;TO GENERATE THE LRC ERROR A SPECIALLY FORMATTED BUFFER IS OUTPUT
2960          ;WITH AN ADDITIONAL CHARACTER FOLLOWING THE 'ETX' WHICH THE RECEIVER
2961          ;INTERPRETS AS THE LRC CHARACTER
2962
2963          ;:*****
2964
2965          ;TEST 24
2966          ;:*****
2967 012260 000004          TST24:  SCOPE
2968 012262 012737 000024 002460  MOV      #24,TSTNUM      ;LOAD THE WD OF THIS TEST
2969 012270 004737 030244          JSR      PC,SETLIN      ;SETUP THE FIRST LINE
2970 012274 012737 012314 001110  MOV      #1$,SLPERR      ;LOAD LOOP ADDRESS (LOOP ON ERROR)
2971 012302 105737 002311          TSTB     MODE          ;CCP MODE ONLY
2972 012306 100002          BPL      1$
2973 012310 000137 012542          JMP      2$          ;
2974 012314 005037 002462          1$:  CLR      SHBE
2975 012320 012737 000001 002452  MOV      #1,ERNUM
2976 012326 005037 002524          CLR      BUFEA
2977 012332 012737 000010 002526  MOV      #10,BUFCNT
2978 012340 004737 030664          JSR      PC,INTKMC      ;INITIALIZE KMC-11 FIRMWARE
2979 012344 000501          BR       3$          ;FIRMWARE FAILED TO INITIALIZE #=1
2980 012346 004737 031204          JSR      PC,INTLNE      ;INITIALIZE LINE (# IN CURLIN)
2981 012352 000476          BR       3$          ;LINE FAILED TO INITIALIZE #=1
2982 012354 004737 031274          JSR      PC,STCLK      ;START CLOCK
2983 012360 000473          BR       3$          ;MAINTENANCE COMMAND NOT ISSUED #=1
2984 012362 005237 002452          INC      ERNUM
2985 012366 012737 052660 002522  MOV      #RBUF,BUFADR
2986 012374 004737 031344          JSR      PC,RIN          ;LOAD A RECEIVE BUFFER IN
2987 012400 000463          BR       3$          ;RECEIVE BUFFER FAILED TO LOAD #=2
2988 012402 005237 002452          INC      ERNUM
2989 012406 012737 033026 002522  MOV      #LRCBUF,BUFADR
2990 012414 004737 031354          JSR      PC,XIN          ;LOAD A TRANSMIT BUFFER IN
2991 012420 000453          BR       3$          ;TRANSMIT BUFFER FAILED TO LOAD #=3
```

```
2992 012422 005237 002452      INC      ERNUM
2993 012426 004737 031714      JSR      PC,SYNTIM      ;WAIT A WHILE
2994 012432 004737 031742      JSR      PC,REQOUT      ;TEST REQUEST OUT OF KMC SETS
2995 012436 000444              BR        3$              ;NO RESPONSE FROM THE TRANSFER #=4
2996 012440 005237 002452      INC      ERNUM
2997 012444 012737 000204 002462 MOV      #204,SHBE      ;BUFFER OUT COMMAND?
2998 012452 004737 031522      JSR      PC,RBONLY      ;SKIP XMIT
2999 012456 123777 002462 167600 CMPB     SHBE,@SEL2
3000 012464 001031              BNE      3$              ;B=NO ERROR #=5
3001 012466 005237 002452      INC      ERNUM
3002 012472 012737 052666 002462 MOV      #RBUF+6,SHBE      ;RETURN ADDRESS
3003 012500 023777 002462 167562 CMP      SHBE,@SEL4      ;CORRECT?
3004 012506 001020              BNE      3$              ;B=N ERR #=6
3005 012510 005237 002452      INC      ERNUM
3006 012514 005037 002462      CLR      SHBE
3007 012520 112737 177774 002462 MOVB     #-4,SHBE
3008 012526 123777 002462 167540 CMPB     SHBE,@SEL6      ;STATUS = LRC ERROR
3009 012534 001005              BNE      3$              ;B=NO #=7
3010 012536 004737 031650      JSR      PC,GETLIN      ;DO NEXT LINE
3011 012542              2$:
3012 012542 000407              BR        TST25              ;:DONE
3013 012544 000137 012314      JMP      1$
3014
3015              ;REPORT ERRORS
3016 012550              3$:
3017 012550 004737 031470      JSR      PC,GETCSR      ;GET CONTENTS OF THE CSRS
3018 012554 104024              ERROR+ 24
3019 012556 000137 012314      JMP      1$
3020
3021
3022
3023      ;:*****
3024
3025      ;PARTST
3026      ;CCP MODE ONLY
3027      ;THIS TEST CHECKS THAT A PARITY ERROR CONDITION CAN BE DETECTED
3028      ;AND RETURNS THE RECEIVE BUFFER WITH A PARITY CONDITION.
3029      ;TO GENERATE THE PARITY ERROR A SPECIALLY FORMATTED BUFFER BUFFER IS OUTPUT
3030      ;WITH AN EVEN PARITY DATA CHARACTER FOLLOWING THE CONTROL CHARACTERS
3031
3032      ;:*****
3033
3034      ;TEST 25
3035      ;:*****
3036 012562 000004              TST25: SCOPE
3037 012564 012737 000025 002460 MOV      #25,TSTNUM      ;LOAD THE WD OF THIS TEST
3038 012572 004737 030244      JSR      PC,SETLIN      ;SETUP THE FIRST LINE
3039 012576 012737 012616 001110 MOV      #1$,$LPERR      ;LOAD LOOP ADDRESS (LOOP ON ERROR)
3040 012604 105737 002311      TSTB     MODE          ;CCP MODE ONLY
3041 012610 100002              BPL      1$
3042 012612 000137 013044      JMP      2$
3043 012616 005037 002462 1$:      CLR      SHBE
3044 012622 012737 000001 002452 MOV      #1,ERNUM
3045 012630 005037 002524      CLR      BUFEA
3046 012634 012737 000010 002526 MOV      #10,BUFCNT
3047 012642 004737 030664      JSR      PC,INTKMC      ;INITIALIZE KMC-11 FIRMWARE
```

```
3048 012646 000501 BR 3$ ;FIRMWARE FAILED TO INITIALIZE #=1
3049 012650 004737 031204 JSR PC,INTLNE ;INITIALIZE LINE (# IN CURLIN)
3050 012654 000476 BR 3$ ;LINE FIALED TO INITIALIZE #=1
3051 012656 004737 031274 JSR PC,STCLK ;START CLOCK
3052 012662 000473 BR 3$ ;MAINTENANCE COMMAND NOT ISSUED #=1
3053 012664 005237 002452 INC ERNUM
3054 012670 012737 052660 002522 MOV #RBUF,BUFADR
3055 012676 004737 031344 JSR PC,RIN ;LOAD A RECEIVE BUFFER IN
3056 012702 000463 BR 3$ ;RECEIVE BUFFER FAILED TO LOAD #=2
3057 012704 005237 002452 INC ERNUM
3058 012710 012737 033036 002522 MOV #PERBUF,BUFADR
3059 012716 004737 031354 JSR PC,XIN ;LOAD A TRANSMIT BUFFER IN
3060 012722 000453 BR 3$ ;TRANSMIT BUFFER FIALED TO LOAD #=3
3061 012724 005237 002452 INC ERNUM
3062 012730 012737 000204 002462 MOV #204,SHBE ;EXPECT BUFFER OUT COMMAND
3063 012736 004737 031714 JSR PC,SYNTIM ;WAIT A WHILE
3064 012742 004737 031742 JSR PC,REQOUT ;TEST REQUEST OUT OF KMC SETS
3065 012746 000441 BR 3$ ;NO RESPONSE FROM THE TRANSFER #=4
3066 012750 005237 002452 INC ERNUM
3067 012754 004737 031522 JSR PC,RBONLY ;SKIP XMIT
3068 012760 123777 002462 167276 CMPB SHBE,@SEL2 ;IS IT A BUFFER OUT COMMAND?
3069 012766 001031 BNE 3$ ;B=NO ERROR #=5
3070 012770 005237 002452 INC ERNUM
3071 012774 012737 052663 002462 MOV #RBUF+3,SHBE ;RETURN ADDRESS
3072 013002 023777 002462 167260 CMP SHBE,@SEL4 ;CORRECT?
3073 013010 001020 BNE 3$ ;B=N ERR #=6
3074 013012 005237 002452 INC ERNUM
3075 013016 005037 002462 CLR SHBE
3076 013022 112737 177775 002462 MOVB #-3,SHBE
3077 013030 123777 002462 167236 CMPB SHBE,@SEL6 ;STATUS = PARITY ERROR
3078 013036 001005 BNE 3$ ;B=NO #=7
3079 013040 004737 031650 JSR PC,GETLIN ;DO NEXT LINE
3080 013044 2$:
3081 013044 000407 BR TST26 ;:DONE
3082 013046 000137 012616 JMP 1$
3083
3084 ;REPORT ERRORS
3085 3$:
3086 013052 JSR PC,GETCSR ;GET CONTENTS OF THE CSRS
3087 013056 104025 ERROR+ 25
3088 013060 000137 012616 JMP 1$
3089
3090
3091
3092 ;*****
3093 ;CHRCNT
3094 ;THIS TEST CHECKS THAT THE CHARACTER COUNT WILL CORRECTLY COUNT
3095 ;TO THE HIGHEST VALUE OF CORE AVAILABLE (ABOVE THE DIAGNOSTIC)
3096 ;BY USING THE SAME BUFFER FOR RECEIVE AS FOR TRANSMIT
3097 ;AND TESTING THAT THE ENDING MEMORY ADDRESS RETURNED
3098 ;OF BOTH THE BUFFERS (XMIT AND RECEIVE) ARE THE SAME
3099
3100 ;*****
3101
3102 ;TEST 26
3103 ;*****
```

3104	013064	000004			TST26:	SCOPE		
3105	013066	012737	000026	002460		MOV	#26,TSTNUM	;LOAD THE WD OF THIS TEST
3106	013074	004737	030244			JSR	PC,SETLIN	;SETUP THE FIRST LINE
3107	013100	012737	013120	001110		MOV	#1\$,SLPERR	;LOAD LOOP ADDRESS (LOOP ON ERROR)
3108						;CHECK	HAT TEST CAN BE	RUN (SUFFICIENT CORE)
3109	013106	005737	002534			TST	CCTST	;IS THERE CORE AVAILABLE?
3110	013112	001572				BEQ	5\$	;B=NO
3111	013114	004737	030244			JSR	PC,SETLIN	
3112	013120	012737	000001	002452	1\$:	MOV	#1,ERNUM	
3113	013126	005037	002462			CLR	SHBE	
3114	013132	013701	002534			MOV	CCTST,R1	;FIND THE ADDRESS THAT SHOULD BE RETURNED
3115	013136	105737	002311			TSTB	MODE	
3116	013142	100403				BMI	10\$	
3117	013144	010137	013526			MOV	R1,8\$	;IN CCP, LENGTH IS ACTUAL CHAR COUNT
3118	013150	000417				BR	15\$	
3119	013152	020127	002000		10\$:	CMP	R1,#2000	;IN BOP MODE,
3120	013156	101402				BLOS	11\$	; MAX COUNT=2000 BYTES
3121	013160	012701	002000			MOV	#2000,R1	
3122	013164	010137	013526		11\$:	MOV	R1,8\$	; SAVE BUFFER SIZE
3123	013170	006237	013526			ASR	8\$	; CHANGE TO WORD COUNT
3124	013174	042737	177000	013526		BIC	#177000,8\$	; MASK TO 9 BITS
3125	013202	062737	010000	013526		ADD	#10000,8\$	; FIX BYTE COUNT
3126	013210	013702	002532		15\$:	MOV	BADDR,R2	;GET START OF TRANSMIT BUFFER
3127	013214	004737	032542			JSR	PC,MEMFIL	;AND FILL WITH ALLOWED INCREMENTING PATTERN
3128	013220	063701	002532			ADD	BADDR,R1	;SET UP ENDING BUFFER ADDRESS
3129	013224	004737	030664			JSR	PC,INTKMC	;INITIALIZE KMC-11 FIRMWARE
3130	013230	000530				BR	6\$	;FIRMWARE FAILED TO INITIALIZE #=1
3131	013232	004737	031204			JSR	PC,INTLNE	;INITIALIZE LINE (# IN CURLIN)
3132	013236	000525				BR	6\$	;LINE FAILED TO INITIALIZE #=1
3133	013240	004737	031274			JSR	PC,STCLK	;SET MAINT LOOP + START CLOCK
3134	013244	000522				BR	6\$	;MAINTENANCE COMMAND NOT ISSUED #=1
3135	013246	005237	002452			INC	ERNUM	
3136	013252	005037	002524			CLR	BUFEA	;NO EA BITS
3137	013256	013737	002532	002522		MOV	BADDR,BUFADR	;BASE ADDRESS OF BUFFERS
3138	013264	013737	013526	002526		MOV	8\$,BUFCNT	;LOAD THE LENGTH
3139	013272	004737	031344			JSR	PC,RIN	;LOAD A RECEIVE BUFFER IN
3140	013276	000505				BR	6\$	;RECEIVE BUFFER FAILED TO LOAD #=2
3141	013300	005237	002452			INC	ERNUM	
3142	013304	004737	031354			JSR	PC,XIN	;LOAD A TRANSMIT BUFFER IN
3143	013310	000500				BR	6\$	;TRANSMIT BUFFER FAILED TO LOAD #=3
3144						;WATCHDOG	THE COMPLETION OF THE TRANSFER	
3145	013312	005237	002452			INC	ERNUM	
3146	013316	005037	013524			CLR	7\$	
3147	013322	005037	013530			CLR	20\$	;CLEAR RECEIVE BUFFER COUNT
3148	013326	005037	013532			CLR	21\$	;CLEAR TOTAL BUFFER COUNT
3149	013332	005037	002462			CLR	SHBE	
3150	013336				2\$:			
3151	013336	004737	031742			JSR	PC,REQOUT	;TEST REQUEST OUT OF KMC SETS
3152	013342	000401				BR	3\$	
3153	013344	000404				BR	4\$	;A REQUEST OUT IS SET
3154	013346	005337	013524		3\$:	DEC	7\$	;COUNT DOWN THE LOOP
3155	013352	001371				BNE	2\$	
3156	013354	000456				BR	6\$	;B=NO BUFFER RETURNED FROM TRANSFER #=4,,7
3157	013356				4\$:	;TEST	THAT THE PROPER COMMAND IS RETURNED	
3158	013356	005237	002452			INC	ERNUM	
3159	013362	122777	000200	166674		CMPB	#200,ASEL2	;TRANSMIT BUFFER?

```
3160 013370 001001      BNE      12$      ;B=N
3161 013372 000406      BR       14$
3162 013374 122777 000204 166662 12$: CMPB   #204,@SEL2 ;RECEIVE BUFFER?
3163 013402 001043      BNE      6$      ;B=N COMMAND RETURNED WAS NOT BUFFER OUT #=5,10
3164 013404 005237 013530      INC     20$      ;COUNT RECEIVE BUFFER
3165 013410 005237 013532      INC     21$      ;COUNT ALL BUFFERS
3166 013414 005237 002452      INC     ERNUM
3167 013420 010137 002462      MOV     R1,SHBE
3168 013424 020177 166640      CMP     R1,@SEL4 ;CORRECT RETURNED ADDRESS?
3169 013430 001030      BNE      6$      ;B=NO ERR #=6,11 (SHBE=ADDRESS)
3170 013432 105077 166626      CLRB   @SEL2 ;COMMAND DONE
3171 013436 005237 002452      INC     ERNUM
3172 013442 005037 002462      CLR     SHBE
3173 013446 023727 013532 000002      CMP     21$,#2 ;HAVE TWO BUFFERS RETURNED?
3174 013454 002730      BLT      2$      ;B=N TO WAIT FOR NEXT
3175 013456 013737 013530 002462      MOV     20$,SHBE
3176 013464 022737 000001 013530      CMP     #1,20$ ;ONE OF TWO IS RECEIVE?
3177 013472 001004      BNE      16$      ;B=N 1XMIT & 1REC NOT RETURNED #=12
3178 013474 004737 031650      JSR     PC,GETLIN ;GET THE NEXT LINE NUMBER
3179 013500      5$:
3180 013500 000415      BR       TST27 ;:DONE
3181 013502 000606      BR       1$      ;DO NEXT LINE
3182
3183 013504 012737 000012 002452 16$: MOV     #12,ERNUM ;ERROR # 12
3184 013512      6$:
3185 013512 004737 031470      JSR     PC,GETCSR ;GET CONTENTS OF THE CSRS
3186 013516 104026      ERROR+  26
3187 013520 000137 013120      JMP     1$
3188 013524 000000      7$: 0 ;WATCH LOOP COUNTER
3189 013526 000000      8$: 0
3190 013530 000000      20$: 0 ;COUNT OF RECEIVE BUFFERS
3191 013532 000000      21$: 0 ;TOTAL BUFFER COUNT
3192
3193
3194 ;:*****
3195 ;BSEADR
3196 ;THIS TEST WILL CHECK THE LOWER 13 (4K) BIT TRANSMIT AND RECEIVE
3197 ;BUFFER ADDRESSING (BUFFER ADDRESS-IN AND OUT) BY TRANSMITTING
3198 ;AND RECEIVING TWO WORD BUFFERS. THE ADDRESSES ARE SEQUENCED BY
3199 ;WALKING A ONE ACROSS THE RETURNED BUFFER ADDRESS VALUE.
3200 ;THE RETURNED BUFFER ADDRESSES ARE TESTED FOR BEING CORRECT.
3201 ;:*****
3202
3203 ;TEST 27
3204 ;:*****
3205 013534 000004      TST27: SCOPE
3206 013536 012737 000027 002460      MOV     #27,TSTNUM ;LOAD THE WD OF THIS TEST
3207 013544 004737 030244      JSR     PC,SETLIN ;SETUP THE FIRST LINE
3208 013550 012737 013762 001110      MOV     #2$,SLPERR ;LOAD LOOP ADDRESS (LOOP ON ERROR)
3209 013556 022737 020000 002534      CMP     #20000,CCTST ;NEED 8K OVER DIAGNOSTIC TO RUN TEST
3210 013564 003402      BLE      14$      ;SKIP IF ENOUGH CORE
3211 013566 000137 014370      JMP     13$      ;OTHERWISE CONTINUE TO NEXT TEST
3212 013572 013701 002532      14$: MOV     BADDR,R1 ;FIND LAST ADDRESS OF BUFFER
3213 013576 063701 002534      ADD     CCTST,R1
3214 013602 010137 014410      MOV     R1,5$ ;HIGHEST ADDRESS
3215 013606 012704 000004      MOV     #4,R4 ;INITIALIZE ADDRESS
```

```

3216                                     :INITIALIZE THE BASE ADDRESSES
3217 013612 013702 002532      15$: MOV BADDR,R2
3218 013616 060402            ADD R4,R2
3219 013620 020237 014410      CMP R2,5$      :DONE?
3220 013624 101402            BLOS 10$         :SKIP OVER IF NO
3221 013626 000137 014364      JMP 3$         :OTHERWISE CONT. TO NEXT LINE
3222 013632 010237 002472      10$: MOV R2,XRET
3223 013636 010237 002474      MOV R2,RRET
3224 013642 005742            TST -(R2)        :-2
3225 013644 005742            TST -(R2)        :-2
3226 013646 105737 002311      TSTB MODE
3227 013652 100002            BPL 6$           :IN CCP THIS IS CHAR. COUNT
3228 013654 005742            TST -(R2)        :-2 IN BOP THIS IS WORD COUNT
3229 013656 005742            TST -(R2)        :-2
3230 013660 010237 002466      6$: MOV R2,XOUT
3231 013664 010237 002470      MOV R2,ROUT
3232 013670 005037 002530      CLR RBUFSZ      :BUFFER SIZE FLAG
3233 013674 105737 002316      TSTB CLKDAT     :INTERNAL?
3234 013700 100414            BMI 65$          :B=YES
3235 013702 105737 002314      TSTB CRC32F     :IF CRC32 IS ENABLED ADD 4 MORE CHARS.
3236 013706 100011            BPL 65$          :B=NOT ENABLED
3237 013710 012737 177777 002530      MOV #-1,RBUFSZ :SET FLAG FOR REC.BUFF HANDLING
3238 013716 062737 000006 002474      ADD #6,RRET :UP REC BUFFER BY 4 CHARS.
3239 013724 062737 000002 002470      ADD #2,ROUT :ADD 4 CHARS. FOR CRC32
3240 013732 012701 000004      65$: MOV #4,R1   :FILL BUFFER WITH ALLOWED
3241 013736 004737 032542      JSR PC,MEMFIL  :INCREMENTING PATTERN
3242 013742 005037 002524      CLR BUFEA
3243 013746 012737 010004 002526      MOV #10004,BUFCNT :SIZE OF BUFFERS
3244 013754 043737 002330 002526      BIC MODMSK,BUFCNT
3245 013762 012737 000001 002452      2$: MOV #1,ERNUM
3246 013770 005037 002462      CLR SHBE
3247 013774 005037 002464      CLR WAS
3248 014000 004737 030664      JSR PC,INTKMC   :INITIALIZE KMC-11 FIRMWARE
3249 014004 000574            BR 4$           :FIRMWARE FAILED TO INITIALIZE #=1
3250 014006 004737 031204      JSR PC,INTLNE   :INITIALIZE LINE (# IN CURLIN)
3251 014012 000571            BR 4$           :LINE FAILED TO INITIALIZE #=1
3252 014014 004737 031274      JSR PC,STCLK    :SET MAINT LOOP + START CLOCK
3253 014020 000566            BR 4$           :MAINTENANCE COMMAND NOT ISSUED #=1
3254 014022 004737 031714      JSR PC,SYNTIM   :ALLOW TIME FOR SYNC DET.
3255                                     :QUEUE THE BUFFERS
3256 014026 005237 002452      INC ERNUM
3257 014032 005737 002530      TST RBUFSZ      :BOP/W CRC32
3258 014036 100003            BPL 46$         :B=NO
3259 014040 062737 000002 002526      ADD #2,BUFCNT
3260 014046 013737 002470 002522      46$: MOV ROUT,BUFADR
3261 014054 004737 031344      JSR PC,RIN
3262 014060 000546            BR 4$           :LOAD A RECEIVE BUFFER IN
3263 014062 005237 002452      INC ERNUM       :RECEIVE BUFFER FAILED TO LOAD #2
3264 014066 005737 002530      TST RBUFSZ     :BOP/W CRC32?
3265 014072 100003            BPL 56$         :B=NO
3266 014074 162737 000002 002526      SUB #2,BUFCNT
3267 014102 013737 002466 002522      56$: MOV XOUT,BUFADR
3268 014110 004737 031354      JSR PC,XIN
3269 014114 000530            BR 4$           :LOAD A TRANSMIT BUFFER IN
3270 014116 005237 002452      INC ERNUM       :TRANSMIT BUFFER FAILED TO LOAD #=3
3271 014122 005037 014412      CLR 20$        :CLEAR RX BUFFER COUNT

```

```
3272 014126 005037 014414          CLR      21$      ;CLEAR TOTAL BUFFER COUNT
3273 014132          16$:      JSR      PC,REQOUT      ;TEST REQUEST OUT OF KMC SETS
3274 014132 004737 031742          BR      4$      ;NO BUFFER RETURNED #=4,7
3275 014136 000517          INC      ERNUM
3276 014140 005237 002452          MOV      #200,SHBE      ;EXPECT TRANSMIT BUFFER OUT
3277 014144 012737 000.00 002462      CMPB     SHBE,@SEL2
3278 014152 123777 002462 166104      BNE      30$      ;IF NOT, CHECK IF RX BUFFER OUT
3279 014160 001015          INC      ERNUM
3280 014162 005237 002452          MOV      XRET,SHBE
3281 014166 013737 002472 002462      MOV      @SEL4,WAS
3282 014174 017737 166070 002464      CMP      SHBE,WAS
3283 014202 023737 002462 002464          BNE      4$      ;IS ADDRESS RETURNED CORRECT?
3284 014210 001072          BR      31$      ;B=NO ERR #=6,11
3285 014212 000425          ;CONTINUE TO COMMON CODE
3286
3287 014214 012737 000204 002462 30$:      MOV      #204,SHBE      ;EXPECT RECEIVE BUFFER OUT
3288 014222 123777 002462 166034      CMPB     SHBE,@SEL2
3289 014230 001062          BNE      4$      ;IF NEITHER RX NOR TX BUFFER, ERROR #=5,10
3290 014232 005237 014412          INC      20$      ;COUNT RETURNED RX BUFFERS
3291 014236 005237 002452          INC      ERNUM
3292 014242 013737 002474 002462      MOV      RRET,SHBE
3293 014250 017737 166014 002464      MOV      @SEL4,WAS
3294 014256 023737 002462 002464      CMP      SHBE,WAS
3295 014264 001044          BNE      4$      ;CORRECT RETURNED ADDRESS?
3296 014266 005237 002452          31$:      INC      ERNUM      ;B=NO ERR #=6,11
3297 014272 005237 014414          INC      21$      ;COUNT ALL BUFFERS
3298 014276 005077 165762          CLR      @SEL2      ;DISMISS COMMAND
3299 014302 023727 014414 000002      CMP      21$,#2      ;HAVE TWO COMMANDS BEEN RETURNED?
3300 014310 002710          BLT      16$      ;B=NO TO WAIT FOR NEXT
3301 014312 012737 014412 002462      MOV      #20$,SHBE      ;SET RX BUFFER COUNT FOR OUTPUT
3302 014320 023727 014412 000001      CMP      20$,#1      ;ARE RETURNED BUFFERS 1 TX AND 1 RX?
3303 014326 001023          BNE      4$      ;B=NO, #=12
3304 014330 005237 002452          INC      ERNUM
3305 014334 005037 002462          CLR      SHBE
3306 014340 004737 031742          JSR      PC,REQOUT      ;TEST REQUEST OUT OF KMC SETS
3307 014344 000401          BR      7$      ;NO RESPONSE IS CORRECT
3308 014346 000413          BR      4$      ;EXTRA COMMAND IS ERROR, #=13
3309 014350 006304          7$:      ASL      R4      ;TEST FOR NEXT WALKING VALUE
3310 014352 022704 020000          CMP      #20000,R4      ;DONE?
3311 014356 002402          BLT      3$      ;IF YES,CONTINUE TO NEXT LINE
3312 014360 000137 013612          JMP      15$      ;OTHERWISE REPEAT TEST WITH NEXT ADDRESS
3313 014364          3$:
3314 014364 004737 031650          JSR      PC,GETLIN      ;GET THE NEXT LINE NUMBER
3315 014370          13$:
3316 014370 000412          BR      TST30      ;:BR IF DONE OR TEST CAN NOT BE RUN
3317 014372 000137 013572          JMP      14$      ;DO NEXT LINE
3318
3319          ;REPORT ERROR
3320 014376 004737 031470          4$:      JSR      PC,GETCSR
3321 014402 104027          ERROR+ 27
3322 014404 000137 013762          JMP      2$
3323 014410 000000          5$:      0      ;HOLD THE HIGHEST ADDRESS THAT CAN BE TESTED
3324 014412 000000          20$:     0      ;RX BUFFER COUNT
3325 014414 000000          21$:     0      ;TOTAL BUFFER COUNT
3326
3327          ;:*****
```

```

3328      ;DATADR
3329      ;THIS TEST WILL CHECK BUSS ADDRESSING OF THE HIGHER ORDER
3330      ;BITS (BITS 14-17) OF THE QUEUED RECEIVE BUFFER AND MULTIPLE DATA
3331      ;TRANSFERS BY TRANSFERRING A 10 WORD BUFFER TO EACH 4K BOUNDARY
3332      ; (IN THE TEST RANGE) OUTPUT IS FROM 'TBUF'
3333      ;* THE TEST WILL EXECUTE ONLY IF SUFFICIENT CORE EXIST.
3334      ;:*****
3335      ;
3336      ;TEST 30
3337      ;:*****
3338      TST30: SCOPE
3339      MOV #30,TSTNUM      ;LOAD THE WD OF THIS TEST
3340      JSR PC,SETLIN      ;SETUP THE FIRST LINE
3341      MOV #1$,SLPERR     ;LOAD LOOP ADDRESS (LOOP ON ERROR)
3342      ;CAN TEST BE RUN?
3343      TST CCTST
3344      BEQ 2$              ;B=NO
3345      CLR BUFEA
3346      CLR CUREA
3347      MOV BADDR,CURBUF
3348      MOV #1,ERNUM
3349      CLR WAS
3350      CLR SHBE
3351      ;MAP TO THE CURRENT BUFFER UNDER TEST (ENABLE KT11 IF PRESENT)
3352      JSR PC,MAPBFR      ;MAP BUFFER FOR RECEIVE
3353      BR 2$              ;B=NO
3354      BR 17$             ;YES
3355      JMP 77$            ;SKIP TO NEXT TEST HELPER
3356      JMP 6$            ;ERROR BRANCH HELPER
3357      16$:
3358      17$:
3359      JSR PC,INTKMC      ;INITIALIZE KMC-11 FIRMWARE
3360      BR 16$             ;FIRMWARE FAILED TO INITIALIZE #=1
3361      JSR PC,INTLNE      ;INITIALIZE LINE (# IN CURLIN)
3362      BR 16$             ;LINE FAILED TO INITIALIZE #=1
3363      JSR PC,STCLK       ;SET MAINT LOOP + START CLOCK
3364      BR 16$             ;MAINTENANCE COMMAND NOT ISSUED #=1
3365      INC ERNUM
3366      JSR PC,INTREC      ;INITIALIZE THE REC. BUFFER
3367      MOV #10012,BUFCNT  ;SET BUFFER LENGTH
3368      BIC MODMSK,BUFCNT
3369      JSR PC,RIN         ;LOAD A RECEIVE BUFFER IN
3370      BR 16$             ;BUFFER FAILED LOAD (REC.) #=2
3371      INC ERNUM
3372      MOV #TBUF,BUFADR
3373      CLR BUFEA
3374      MOV #10010,BUFCNT  ;SET BUFFER LENGTH
3375      BIC MODMSK,BUFCNT
3376      JSR PC,XIN         ;LOAD A TRANSMIT BUFFER IN
3377      BR 16$             ;BUFFER FAILED TO LOAD (TRANSMIT) #=3
3378      INC ERNUM
3379      ;WATCHDOG THE TRANSFERS COMPLETION
3380      CLR 20$
3381      CLR 21$
3382      MOV #4,ERNUM
3383      MOV #2,R0           ;SET UP WAIT LOOP
3384      18$:
3385      3$:

```

3384	014664	004737	031742		JSR	PC,REQOUT	:TEST REQUEST OUT OF KMC SETS
3385	014670	000401			BR	4\$	
3386	014672	000403			BR	5\$	:OK OUTPUT REQUESTED
3387	014674	005300		4\$:	DEC	R0	
3388	014676	001372			BNE	3\$	
3389	014700	000524			BR	6\$	:ERR #=4 TRANSFER FAILED TO COMPLETE
3390	014702	005237	002452	5\$:	INC	ERNUM	
3391						:TEST FOR RETURNED BUFFER	
3392	014706	005037	002462		CLR	SHBE	:RETURNED EA BITS SHOULD BE=0
3393	014712	122777	000200	165344	CMPB	#200,@SEL2	:CHECK FOR TRANSMIT BUFFER
3394	014720	001015			BNE	10\$	
3395	014722	012737	000006	002452	MOV	#6,ERNUM	:SET UP TRANSMIT BUFFER ERROR #'S
3396	014730	012737	052270	002514	MOV	#TBUF+10,RETBUF	:RETURNED BUFFER ADDRESS SHOULD BE
3397	014736	105737	002311		TSTB	MODE	:CHECK MODE
3398	014742	100003			BPL	8\$	:SKIP IF CCP
3399	014744	062737	000010	002514	ADD	#10,RETBUF	:IN BOP, CONVERT WORD COUNT TO BYTES
3400	014752			8\$:			
3401	014752	000417			BR	12\$	
3402	014754	122777	000204	165302	10\$:	CMPB	#204,@SEL2
3403	014762	001073			BNE	6\$	:CHECK FOR RECEIVE BUFFER
3404	014764	005237	015162		INC	20\$	:B=N COMMAND NOT TX OR RX BUFFER OUT #=5
3405	014770	012737	000010	002452	MOV	#10,ERNUM	:COUNT RECEIVE BUFFER
3406	014776	013737	002510	002462	MOV	RETEA,SHBE	:SET UP RECEIVE BUFFER ERROR #'S
3407	015004	013737	002512	002514	MOV	RETADR,RETBUF	:CURRENTLY MAPPED BUFFER EA-BITS
3408	015012	005237	015164		12\$:	INC	21\$
3409	015016	017737	165252	002464	MOV	@SEL6,WAS	:COUNT ALL BUFFERS
3410	015024	042737	037777	002464	BIC	#37777,WAS	:GET EA BITS
3411	015032	023737	002462	002464	CMP	SHBE,WAS	:SAVE ONLY THE EA-BITS
3412	015040	001044			BNE	6\$	:ARE EA BITS CORRECT?
3413	015042	005237	002452		INC	ERNUM	:B=NO ERR #=6,10
3414	015046	017737	165216	002464	MOV	@SEL4,WAS	:TEST LOWER 16 BIT ADDRESS
3415	015054	013737	002514	002462	MOV	RETBUF,SHBE	:16 BIT ADDRESS
3416	015062	023737	002462	002464	CMP	SHBE,WAS	:16 BIT RECEIVE BUFFER ADDRESS CORRECT?
3417	015070	001030			BNE	6\$	:NO ERR #=7,11
3418	015072	005237	002452		INC	ERNUM	
3419	015076	105077	165162		CLRB	@SEL2	:DISMISS COMMAND
3420	015102	023727	015164	000002	CMP	21\$,#2	:TWO COMMANDS BACK YET
3421	015110	002660			BLT	18\$	:B=NO WAIT FOR ANOTHER
3422	015112	013737	015162	002462	MOV	20\$,SHBE	:STORE # OF RECEIVE BUFFERS
3423	015120	023727	015162	000001	CMP	20\$,#1	:1 RECEIVE BUFFER?
3424	015126	001011			BNE	6\$	:B=NO TWO BUFFERS NOT 1 TX & 1 RX #=12
3425						:GO + DO NEXT BUFFER	
3426	015130	000241			CLC		
3427	015132	062737	020000	002500	ADD	#20000,CURBUF	:4K TO NEXT BUFFER
3428	015140	103002			BCC	55\$	:OVERFLOW?
3429	015142	005237	002476		INC	CUREA	:YES +1 TO EA BITS
3430	015146	000137	014464	55\$:	JMP	1\$	:RUN IT
3431							
3432						:REPORT ERROR	
3433	015152			6\$:			
3434	015152	104030			ERROR+	30	
3435	015154	000137	014464		JMP	1\$	
3436							
3437	015160			77\$:			
3438	015160	000402			BR	TST31	:GO TO NEXT TEST
3439	015162	000000		20\$:	0		:RECEIVE BUFFER COUNT

```

3440 015164 000000      21$:      0      ;TOTAL BUFFER COUNT
3441
3442      ;*****
3443      ;DATADR
3444      ;THIS TEST WILL CHECK BUSS ADDRESSING OF THE HIGHER ORDER BITS
3445      ; (BITS 14-17) OF THE QUEUED TRANSMIT BUFFER AND MULTIPLE DATA
3446      ; TRANSFERS BY TRANSFERRING A 10 WORD BUFFER FROM EACH 4K BOUNDARY
3447      ; (IN THE TEST RANGE) INPUT IS TO RBUF.
3448      ; * THE TEST WILL EXECUTE ONLY IF SUFFICIENT CORE EXIST.
3449      ;*****
3450
3451      ;TEST 31
3452      ;*****
3453      TST31: SCOPE
3454      MOV      #31,TSTNUM      ;LOAD THE WD OF THIS TEST
3455      JSR      PC,SETLIN      ;SETUP THE FIRST LINE
3456      MOV      #1$,SLPERR      ;LOAD LOOP ADDRESS (LOOP ON ERROR)
3457      ;CAN TEST BE RUN?
3458      TST      CCTST
3459      BEQ      2$      ;B=NO
3460      CLR      BUFEA
3461      CLR      CUREA
3462      MOV      BADDR,CURBUF
3463      MOV      #1,ERNUM
3464      CLR      WAS
3465      CLR      SHBE
3466      ;MAP TO THE CURRENT BUFFER UNDER TEST (ENABLE KT11 IF PRESENT)
3467      JSR      PC,MAPBUF      ;MAP BUFFER FOR TRANSMIT
3468      BR      2$      ;B=NO
3469      BR      17$      ;YES
3470      JMP      77$      ;SKIP TO NEXT TEST HELPER
3471      JMP      6$      ;ERROR BRANCH HELPER
3472      16$:
3473      17$:
3474      JSR      PC,INTKMC      ;INITIALIZE KMC-11 FIRMWARE
3475      BR      16$      ;FIRMWARE FAILED TO INITIALIZE #=1
3476      JSR      PC,INTLNE      ;INITIALIZE LINE (# IN CURLIN)
3477      BR      16$      ;LINE FAILED TO INITIALIZE #=1
3478      JSR      PC,STCLK      ;SET MAINT LOOP + START CLOCK
3479      BR      16$      ;MAINTENANCE COMMAND NOT ISSUED #=1
3480      INC      ERNUM
3481      JSR      PC,INTTRM      ;INITIALIZE THE TRANSMIT BUFFER
3482      MOV      #10012,BUFCNT      ;SET BUFFER LENGTH
3483      BIC      MODMSK,BUFCNT
3484      MOV      BUFADR,R0      ;LOAD THE RECEIVE BUFFER FIRST
3485      MOV      BUFEA,R1
3486      MOV      #RBUF,BUFADR
3487      CLR      BUFEA
3488      JSR      PC,RIN      ;LOAD A RECEIVE BUFFER IN
3489      BR      16$      ;RECEIVE BUFFER FAILED TO LOAD ERR #=2
3490      INC      ERNUM
3491      MOV      R1,BUFEA      ;RESTORE EA-BITS
3492      MOV      R0,BUFADR
3493      MOV      #10010,BUFCNT      ;SET BUFFER LENGTH
3494      BIC      MODMSK,BUFCNT
3495      JSR      PC,XIN      ;LOAD A TRANSMIT BUFFER IN
3496      BR      6$      ;TRANSMIT BUFFER FAILED TO LOAD #=3

```

```
3496 015426 005237 002452      INC      ERNUM
3497      :WATCHDOG THE TRANSFERS COMPLETION
3498 015432 005037 016000      CLR      20$      :RECEIVE BUFFER COUNT
3499 015436 005037 016002      CLR      21$      :TOTAL BUFFER COUNT
3500 015442 012737 000004 002452 18$: MOV      #4,ERNUM
3501 015450 012700 000002      MOV      #2,R0      :SET UP WAIT LOOP
3502 015454      3$:
3503 015454 004737 031742      JSR      PC,REQOUT      :TEST REQUEST OUT OF KMC SETS
3504 015460 000401      BR      4$
3505 015462 000403      BR      5$      :OK OUTPUT REQUESTED
3506 015464 005300      4$: DEC      R0
3507 015466 001372      BNE      3$
3508 015470 000537      BR      6$      :ERR #=4 TRANSFER FAILED TO COMPLETE
3509 015472 005237 002452      5$: INC      ERNUM
3510      :TEST FOR RETURNED BUFFER
3511 015476 005037 002462      CLR      SHBE      :RETURNED EA BITS SHOULD BE=0
3512 015502 122777 000200 164554  CMPB     #200,@SEL2      :CHECK FOR TRANSMIT BUFFER
3513 015510 001012      BNE      10$
3514 015512 012737 000006 002452  MOV      #6,ERNUM      :SET UP TRANSMIT BUFFER ERROR #'S
3515 015520 013737 002510 002462  MOV      RETEA,SHBE      :CURRENTLY MAPPED BUFFER
3516 015526 013737 002512 002514  MOV      RETADR,RETBUF
3517 015534 000435      BR      12$
3518 015536 122777 000204 164520 10$: CMPB     #204,@SEL2      :CHECK FOR RECEIVE BUFFER
3519 015544 001111      BNE      6$      :B=N COMMAND NOT TX OR RX BUFFER OUT #=5
3520 015546 005237 016000      INC      20$      :COUNT RECEIVE BUFFER
3521 015552 012737 000010 002452  MOV      #10,ERNUM      :SET UP RECEIVE BUFFER ERROR #'S
3522 015560 005037 002462      CLR      SHBE
3523 015564 012737 052670 002514  MOV      #RBUF+10,RETBUF :RETURNED RECEIVE BUFFER SHOULD BE
3524 015572 105737 002311      TSTB     MODE      :CHECK MODE
3525 015576 100014      BPL      9$      :SKIP IF CCP
3526 015600 062737 000010 002514  ADD      #10,RETBUF      :IN BOP, CONVERT WORD COUNT TO BYTES
3527 015606 105737 002316      TSTB     CLKDAT      :CHECK INTERNAL OR EXTERNAL
3528 015612 100406      BMI      9$
3529 015614 105737 002314      TSTB     CRC32F      :ONLY IF EXT.,CHECK IF CRC32 USED
3530 015620 100003      BPL      9$
3531 015622 062737 000004 002514  ADD      #4,RETBUF      :IF CRC32, ADD BYTES FOR APPENDED CRC
3532 015630      9$:
3533 015630 005237 016002      12$: INC      21$      :COUNT ALL BUFFERS
3534 015634 017737 164434 002464  MOV      @SEL6,WAS      :GET EA BITS
3535 015642 042737 037777 002464  BIC      #37777,WAS      :SAVE ONLY THE EA-BITS
3536 015650 023737 002462 002464  CMP      SHBE,WAS      :ARE EA BITS CORRECT?
3537 015656 001044      BNE      6$      :B=NO ERR #=6,10
3538 015660 005237 002452      INC      ERNUM      :TEST LOWER 16 BIT ADDRESS
3539 015664 017737 164400 002464  MOV      @SEL4,WAS      :16 BIT ADDRESS
3540 015672 013737 002514 002462  MOV      RETBUF,SHBE
3541 015700 023737 002462 002464  CMP      SHBE,WAS      :16 BIT RECEIVE BUFFER ADDRESS CORRECT?
3542 015706 001030      BNE      6$      :NO ERR #=7,11
3543 015710 005237 002452      INC      ERNUM
3544 015714 105077 164344      CLRB     @SEL2      :DISMISS COMMAND
3545 015720 023727 016002 000002  CMP      21$,#2      :TWO COMMANDS BACK YET
3546 015726 002645      BLT      18$      :B=NO WAIT FOR ANOTHER
3547 015730 013737 016000 002462  MOV      20$,SHBE      :STORE # OF RECEIVE BUFFERS
3548 015736 023727 016000 000001  CMP      20$,#1      :1 RECEIVE BUFFER?
3549 015744 001011      BNE      6$      :B=NO TWO BUFFERS NOT 1 TX & 1 RX #=12
3550      :GO + DO NEXT BUFFER
3551 015746 000241      CLC
```

CZKMEAO DMS11-DA DYNAMIC
CZKMEA.P11 20-OCT-81 17:05

MACY11 30A(1052) 21-OCT-81 08:50 H 6
GET VALUE FOR SOFTWARE SWITCH REGISTER

SEQ 0072

```
3552 015750 062737 020000 002500      ADD    #20000,CURBUF    ;4K TO NEXT BUFFER
3553 015756 103002          BCC    55$      ;OVERFLOW?
3554 015760 005237 002476          INC    CUREA      ;YES +1 TO EA BITS
3555 015764 000137 015234      55$:    JMP    1$      ;RUN IT
3556
3557          ;REPORT ERROR
3558 015770          6$:
3559 015770 104031          ERROR+ 31
3560 015772 000137 015234      JMP    1$
3561
3562 015776          77$:
3563 015776 000402          BR      TST32      ;GO TO NEXT TEST
3564 016000 000000      20$:    0          ;RECEIVE BUFFER COUNT
3565 016002 000000      21$:    0          ;TOTAL BUFFER COUNT
3566
3567      ;:*****
3568      ;DATTST
3569      ;THIS TEST WILL CHECK MULTIPLE DATA TRANSFERS AND BUS ADDRESSING
3570      ;BY TRANSFERRING BUFFERS TO EACH INCREMENTAL 4K ABOVE THE DIAGNOSTIC
3571      ;IN CCP MODE, THE FIRST 4K BLOCK (16-20K) IS WRITTEN WITH AN
3572      ;INCREMENTAL PATTERN (40-175) AND TRANSFERRED TO EACH 4K ABOVE
3573      ;THE DIAGNOSTIC (20K UP)
3574      ;IN BOP MODE, THE FIRST 1024 BYTES OF THE FIRST 4K BLOCK (16K) ARE
3575      ;WRITTEN WITH AN INCREMENTAL PATTERN (0-377) AND TRANSFERRED
3576      ;TO THE START OF EACH 4K ABOVE THE DIAGNOSTIC (20K UP)
3577      ;TEST WILL RUN ONLY IF 20K OR MORE OF CONTIGUOUS CORE (0 UP) EXISTS ON SYSTEM
3578
3579      ;TEST 32
3580
3581      ;:*****
3582      ;TST32: SCOPE
3583 016004 000004          MOV    #32,TSTNUM    ;LOAD THE WD OF THIS TEST
3584 016006 012737 000032 002460      JSR    PC,SETLIN    ;SETUP THE FIRST LINE
3585 016014 004737 030244          MOV    #1$,SLPERR    ;LOAD LOOP ADDRESS (LOOP ON ERROR)
3586 016020 012737 016036 001110      ;CAN TEST BE RUN?
3587 016026 022737 000020 002534      CMP    #20,CCTST    ;AT LEAST 4K OVER DIAG.
3588 016034 003041          BGT    11$      ;B=N, SKIP TO NEXT TEST
3589          ;INITIALIZE THE RECEIVED BUFFER POINTER ADDRESS
3590 016036 013737 002532 002500      1$:    MOV    BADDR,CURBUF    ;START @ BASE ADDR.
3591 016044 062737 020000 002500      ADD    #20000,CURBUF    ;UP BY 4K
3592 016052 005037 002476          CLR    CUREA      ;NO EA BITS
3593 016056 013702 002532          MOV    BADDR,R2      ;FILL TRANSMIT BUFFER WITH INCREMENTAL PATTERN
3594 016062 012701 017777          MOV    #17777,R1
3595 016066 004737 032542          JSR    PC,MEMFIL
3596 016072 012737 000001 002452      MOV    #1,ERNUM
3597 016100 005037 002464          2$:    CLR    WAS
3598 016104 005037 002462          CLR    SHBE
3599 016110 004737 030664          JSR    PC,INTKMC    ;INITIALIZE KMC-11 FIRMWARE
3600 016114 000407          BR      10$      ;F.W. NOT INITIALIZED #=1
3601 016116 004737 031204          JSR    PC,INTLNE    ;INITIALIZE LINE (# IN CURLIN)
3602 016122 000404          BR      10$      ;LINE FAILED TO INITIALIZE #=1
3603 016124 004737 031274          JSR    PC,STCLK    ;SET MAINT LOOP + START CLOCK
3604 016130 000401          BR      10$      ;MAINTENANCE COMMAND FAILED #=1
3605 016132 000406          BR      20$
3606 016134 000137 017144          10$:    JMP    8$      ;BRANCH ERROR HELPER
3607 016140 000137 017126          11$:    JMP    12$     ;SKIP TO NEXT TEST HELPER
```

3608	016144	000137	017122	27\$:	JMP	26\$		:GO TO NEXT LINE HELPER
3609	016150	005237	002452	20\$:	INC	ERNUM		
3610	016154	005037	002462		CLR	SHBE		
3611	016160	005037	002464		CLR	WAS		
3612								:MAP THE RECEIVE BUFFER AND ENABLE KT11 (IF PRESENT)
3613								:TO THE ADDRESS POINTED TO BY 'CURBUF' AND 'CUREA'
3614								:IF THE RECEIVE BUFFER REQUEST EXCEEDS CORE LIMITS RETURN IS TO PC
3615	016164	004737	032036		JSR	PC,MAPBUF		:MAP THE RECEIVE BUFFER
3616	016170	000765			BR	27\$		:CONTINUE TO NEXT LINE
3617	016172	013737	002524	017142	MOV	BUFEA,32\$		:SAVE HIGH ORDER BITS FOR LATER USE
3618	016200	012737	017777	002526	MOV	#17777,BUFCNT		:LENGTH OF RECEIVE BUFFER
3619	016206	004737	031344		JSR	PC,RIN		:LOAD A RECEIVE BUFFER IN
3620	016212	000750			BR	10\$		:B=BUFFER FAILED TO LOAD #=2
3621								:LOAD THE TRANSMIT BUFFER
3622	016214	005237	002452		INC	ERNUM		
3623	016220	005037	002524		CLR	BUFEA		
3624	016224	013737	002532	002522	MOV	BADDR,BUFADR		:TRANSMIT BUFFER ADDRESS
3625	016232	004737	031354		JSR	PC,XIN		:LOAD A TRANSMIT BUFFER IN
3626	016236	000736			BR	10\$		:B=XBUFF FAILED TO LOAD #=3
3627	016240	005237	002452		INC	ERNUM		
3628								:WATCH DOG THE I/O COMPLETION
3629	016244	005037	017136		CLR	30\$		:CLEAR RECEIVE BUFFER COUNT
3630	016250	005037	017140		CLR	31\$		:CLEAR TOTAL BUFFER COUNT
3631	016254	012737	017777	017134	22\$:	MOV	#17777,7\$	:SPIN LOOP
3632	016262			3\$:				
3633	016262	004737	031742		JSR	PC,REQOUT		:TEST REQUEST OUT OF KMC SETS
3634	016266	000401			BR	4\$		
3635	016270	000404			BR	5\$		:B=OUTPUT RESPONSE
3636	016272	005337	017134	4\$:	DEC	7\$		
3637	016276	001371			BNE	3\$		:TRY AGAIN
3638	016300	000477			BR	13\$		:NO OUTPUT RESPONSE FROM FIRMWARE ERR #=4,11,17
3639	016302	005237	002452	5\$:	INC	ERNUM		:TEST THAT RESPONSE IS A RETURNED XMIT BUFFER
3640	016306	005037	002462		CLR	SHBE		
3641	016312	117737	163746	002464	MOV	@SEL2,WAS		
3642	016320	122737	000200	002464	CMP	#200,WAS		:IS COMMAND A TRANSMIT BUFFER?
3643	016326	001066			BNE	23\$		:B=NO
3644	016330	012737	000006	002452	MOV	#6,ERNUM		:INIT. ERROR COUNT FOR TX BUFFER
3645	016336	117737	163732	002464	MOV	@SEL6,WAS		
3646	016344	112737	000001	002462	MOV	#1,SHBE		
3647	016352	023737	002462	002464	CMP	SHBE,WAS		:IS BUFFER STATUS = 1 (END COUNT REACHED)?
3648	016360	001047			BNE	13\$		:B=N ERR #=6
3649	016362	005237	002452		INC	ERNUM		
3650	016366	017737	163676	002464	MOV	@SEL4,WAS		:IS THE 16 BIT XMIT ADDRESS CORRECT?
3651	016374	012737	017777	002462	MOV	#17777,SHBE		:SET UP 16-BIT TRANSMIT ADDRESS FOR CCP
3652	016402	105737	002311		TSTB	MODE		
3653	016406	100003			BPL	33\$		:SKIP IF CCP
3654	016410	012737	001776	002462	MOV	#1776,SHBE		:SET UP 16-BIT TRANSMIT ADDRESS FOR BOP
3655	016416	063737	002532	002462	ADD	BADDR,SHBE		
3656	016424	023737	002462	002464	CMP	SHBE,WAS		
3657	016432	001022			BNE	13\$		:B=N ERR #=7
3658	016434	005237	002452		INC	ERNUM		
3659	016440	017737	163630	002464	MOV	@SEL6,WAS		:ARE THE EA BITS CORRECT?
3660	016446	013737	002464	002462	MOV	WAS,SHBE		
3661	016454	042737	140000	002462	BIC	#140000,SHBE		:THEY SHOULD BE CLEAR
3662	016462	023737	002462	002464	CMP	SHBE,WAS		
3663	016470	001003			BNE	13\$		:B=N ERR #=10

3664	016472	005237	002452		INC	ERNUM	
3665	016476	000560			BR	35\$	
3666	016500	000137	017144	13\$:	JMP	8\$	;ERROR BRANCH HELPER
3667	016504	122737	000204	002464 23\$:	CMPB	#204,WAS	;IS COMMAND RECEIVE BUFFER OUT?
3668	016512	001372			BNE	13\$	;B=OUTPUT NOT A RECEIVE BUFFER #=5,12,20
3669	016514	012737	000013	002452	MOV	#13,ERNUM	;INIT. ERROR COUNT FOR RECEIVE BUFFER
3670	016522	012737	000002	002462	MOV	#2,SHBE	;IS RETURNED STATUS AN END OF MESSAGE
3671	016530	005037	002464		CLR	WAS	
3672	016534	105737	002314		TSTB	CRC32F	;RUNNING CRC-32?
3673	016540	100003			BPL	14\$	;B=NO
3674	016542	012737	000001	002462	MOV	#1,SHBE	;SET REC. BUFFER STATUS TO ENDING MEM.AR.
3675	016550	117737	163520	002464 14\$:	MOVB	@SEL6,WAS	
3676	016556	123737	002462	002464	CMPB	SHBE,WAS	
3677	016564	001167			BNE	8\$	;B=NO #=13
3678	016566	005237	002452		INC	ERNUM	
3679	016572	012737	017777	002462	MOV	#17777,SHBE	;SET CCP ENDING BUFFER ADDRESS
3680	016600	105737	002311		TSTB	MODE	
3681	016604	100003			BPL	24\$	
3682	016606	012737	001776	002462 24\$:	MOV	#1776,SHBE	;SET BOP ENDING BUFFER ADDRESS
3683	016614	063737	002500	002462	ADD	CURBUF,SHBE	
3684	016622	017737	163442	002464	MOV	@SEL4,WAS	
3685	016630	023737	002462	002464	CMP	SHBE,WAS	
3686	016636	001142			BNE	8\$	;B=N #=14
3687	016640	005237	002452		INC	ERNUM	
3688	016644	013737	017142	002462	MOV	32\$,SHBE	;ARE THE RETURNED REC.BUF EA BITS CORRECT?
3689	016652	017737	163416	002464	MOV	@SEL6,WAS	
3690	016660	042737	037777	002464	BIC	#37777,WAS	
3691	016666	023737	002462	002464	CMP	SHBE,WAS	
3692	016674	001123			BNE	8\$	;B=N #=15
3693	016676	005237	002452		INC	ERNUM	
3694					;TEST THE DATA RETURNED IN THE RECEIVE BUFFER		
3695	016702	013701	002532		MOV	BADDR,R1	;SET TO START OF TX BUFFER
3696	016706	013702	002502		MOV	MAPADR,R2	;SET TO START OF RX BUFFER
3697	016712	005037	002464		CLR	WAS	
3698	016716	005037	002462		CLR	SHBE	
3699	016722	105737	002311		TSTB	MODE	
3700	016726	100015			BPL	25\$	
3701					;FOR BOP MODE ONLY		
3702	016730	012700	001775		MOV	#1775,R0	;SET BOP CHARACTER COUNT
3703	016734	112137	002462	40\$:	MOVB	(R1)+,SHBE	;GET PROPER CHARACTER
3704	016740	112237	002464		MOVB	(R2)+,WAS	;GET RECEIVED CHAR
3705	016744	123737	002462	002464	CMPB	SHBE,WAS	;AGREE?
3706	016752	001074			BNE	8\$	;BRANCH ON NO TO ERROR #=16
3707	016754	005300			DEC	R0	;ARE WE DONE?
3708	016756	001366			BNE	40\$	;ON NO GO DO NEXT CHAR
3709	016760	000425			BR	43\$	;WHEN COMPARE DONE CONTINUE WITH COMMON CODE
3710					;FOR CCP MODE ONLY		
3711	016762	012700	017777	25\$:	MOV	#17777,R0	;SET CCP CHARACTER COUNT
3712	016766	112137	002462		MOVB	(R1)+,SHBE	;GET STARTING SOH
3713	016772	042737	000200	002462	BIC	#200,SHBE	;STRIP OFF PARITY BIT
3714	017000	000402			BR	41\$	
3715	017002	112137	002462	42\$:	MOVB	(R1)+,SHBE	;GET PROPER CHARACTER
3716	017006	112237	002464	41\$:	MOVB	(R2)+,WAS	;GET RECEIVED CHARACTER
3717	017012	042737	000200	002464	BIC	#200,WAS	;STRIP OFF ADDED PARITY BIT
3718	017020	123737	002462	002464	CMPB	SHBE,WAS	;AGREE?
3719	017026	001046			BNE	8\$	;ON NO, BRANCH TO ERROR #=16

CZKMEAO DMS11-DA DYNAMIC
CZKMEA.P11 20-OCT-81 17:05

MACY11 30A(1052) 21-OCT-81 08:50 K 6
GET VALUE FOR SOFTWARE SWITCH REGISTER

SEQ 0075

```
3720 017030 005300      DEC      R0      : ARE WE DONE?
3721 017032 001363      BNE      42$     : ON NO, GO DO NEXT CHAR
3722                                     : FALL THROUGH TO COMMON CODE
3723 017034 005237 017136 43$: INC      30$     :COUNT RECEIVE BUFFERS
3724
3725                                     :CHECK IF BOTH BUFFERS BACK YET
3726 017040 005237 017140 35$: INC      31$     :COUNT ALL BUFFERS
3727 017044 105077 163214      CLRB     @SEL2   :DISMISS COMMAND
3728 017050 023727 017140 000002      CMP     31$,#2   :TWO COMMANDS BACK YET?
3729 017056 001402      BEQ      36$     :B=YES
3730 017060 000137 016254      JMP      22$     :WAIT FOR ANOTHER BUFFER
3731 017064 012737 000021 002452 36$: MOV     #21,ERNUM
3732 017072 023727 017136 000001      CMP     30$,#1   :IS ONE OF THE TWO BUFFERS A RX BUFFER?
3733 017100 001021      BNE      8$       :B=NO TO ERROR, #=21
3734                                     :MOVE TO NEXT BUFFER
3735 017102 062737 020000 002500      ADD     #20000,CURBUF
3736 017110 103002      BCC      6$       :
3737 017112 005237 002476      INC      CUREA
3738 017116 000137 016100 6$: JMP      2$       :TRY NEXT BUFFER
3739 017122 26$:
3740 017122 004737 031650      JSR      PC,GETLIN :GET THE NEXT LINE NUMBER
3741 017126 12$:
3742 017126 000413      BR       TST33      :;B=TEST DONE OR BYPASSED
3743 017130 000137 016036      JMP      1$       :RETURN FOR NEXT LINE
3744
3745 017134 000000 7$: 0 :COUNT LOCATION
3746 017136 000000 30$: 0 :COUNT OF RECEIVE BUFFERS
3747 017140 000000 31$: 0 :TOTAL BUFFER COUNT
3748 017142 000000 32$: 0 :HIGH-ORDER BITS SAVE LOCATION
3749
3750 017144 8$:
3751 017144 004737 031470      JSR      PC,GETCSR :GET CONTENTS OF THE CSRS
3752 017150 104032      ERROR+ 32
3753 017152 000137 016036      JMP      1$
3754
3755 ;:*****
3756 ;INTIN
3757 ;THIS TEST WILL CHECK THE INPUT REQUEST INTERRUPT LOGIC
3758 ;AND FIRMWARE ARBITRATION BY INITIALIZING THE INPUT REQUEST
3759 ;VECTOR (XX0) WITH A CORRECT RETURN ADDRESS AND THE OUTPUT REQUEST
3760 ;VECTOR WITH ADDRESS TO ERROR FLAG ROUTINE.
3761 ;THE INPUT REQUEST IS FIRST DONE WITH THE PROCESSOR STATUS
3762 ;DROPPED TO 0 AND THE INTERRUPT ENABLE BIT NOT SET TO CHECK
3763 ;THAT NO INTERRUPT OCCURS WHEN THE 'RDYIN' (BITS 7-SEL-0) SETS.
3764 ;THEN THE INPUT REQUEST IS MADE WITH THE INTERRUPT ENABLE
3765 ;BIT SET AND THE PROCESSOR STATUS AT 0 TO CHECK THAT THE INTERRUPT
3766 ;OCCURS TO THE CORRECT VECTOR ADDRESS
3767 ;THEN THE PROCESSOR STATUS IS SET TO LEVEL 7 THE REQUEST MADE
3768 ;WITH THE INTERRUPT ENABLE BIT SET AND TEST THAT THE INTERRUPT DOES
3769 ;NOT OCCUR UNTIL THE CORRECT PROCESSOR STATUS IS REACHED
3770
3771 ;:*****
3772
3773 ;TEST 33
3774 ;:*****
3775 017156 000004 TST33: SCOPE
```

3776	017160	012737	000033	002460	MOV	#33,TSTNUM	:LOAD THE WD OF THIS TEST
3777	017166	004737	030244		JSR	PC,SETLIN	:SETUP THE FIRST LINE
3778	017172	012737	017200	001110	MOV	#1\$,SLPERR	:LOAD LOOP ADDRESS (LOOP ON ERROR)
3779	017200	012737	000001	002452	MOV	#1,ERNUM	:INITIALIZE ERR #
3780	017206	005037	002464		CLR	WAS	
3781	017212	004737	030664		JSR	PC,INTKMC	:INITIALIZE KMC-11 FIRMWARE
3782	017216	000564			BR	10\$	:FIRMWARE FAILED TO INITIALIZE #=1
3783							:PRIME INTERRUPT VECTORS, DROP PROCESSOR STATUS TO 0, AND ISSUE
3784							:A REQUEST IN W/O I.E. IN SET
3785	017220	013700	002300		MOV	VECIN,RO	
3786	017224	012720	017602		MOV	#11\$, (RO)+	:PRIME INPUT VECTOR
3787	017230	012720	000340		MOV	#PR7, (RO)+	:PRIME INPUT STATUS LEVEL
3788	017234	012720	017612		MOV	#12\$, (RO)+	:PRIME OUTPUT VECTOR
3789	017240	012710	000340		MOV	#PR7, (RO)	:PRIME OUTPUT STATUS LEVEL
3790	017244	005237	002452		INC	ERNUM	
3791	017250	004737	031274		JSR	PC,STCLK	:SET MAINT LOOP + START CLOCK
3792	017254	000545			BR	10\$	:MAINTENANCE COMMAND NOT ISSUED #=1
3793	017256	142737	000340	177776	BICB	#PR7,PS	:SET P.S. TO 0
3794	017264	112777	000043	162766	MOVB	#43,@SELO	:LOAD INIT. IN REQ. W/O INT. EN
3795	017272	005037	001160		CLR	\$TMP0	
3796	017276	105777	162756		TSTB	@SELO	:IS REQUEST GRANTED?
3797	017302	100405		2\$:	BMI	3\$	:B=YES
3798	017304	005237	001160		INC	\$TMP0	:ALLOW TIME FOR RESPONSE
3799	017310	001372			BNE	2\$	
3800	017312	000526			BR	10\$	:REPORT REQUEST IN 'RDYIN' FAILED TO SET #=2
3801	017314	000240			NOP		
3802	017316	062737	000002	002452	ADD	#2,ERNUM	:NO INTERRUPT OCCURED
3803	017324	012737	000340	177776	MOV	#PR7,PS	:SET PROCESSOR BACK TO LEVEL 7
3804	017332	004737	030664		JSR	PC,INTKMC	:INITIALIZE KMC-11 FIRMWARE
3805	017336	000514			BR	10\$	:2ND. INITIALIZE OF F.W. FAILED #=4
3806	017340	004737	031274		JSR	PC,STCLK	:SET MAINT LOOP + START CLOCK
3807	017344	000511			BR	10\$	
3808	017346	005237	002452		INC	ERNUM	
3809	017352	012777	017410	162720	MOV	#6\$,@VECIN	:LOAD RET. ADDRESS FOR LEGAL INTERRUPT
3810	017360	042737	000340	177776	BIC	#PR7,PS	:DROP PROCESSOR STATUS TO 0
3811	017366	112777	000143	162664	MOVB	#143,@SELO	:INT. ENABLE, REQ.IN INIT.
3812	017374	004737	031714		JSR	PC,SYNTIM	:WAIT A WHILE
3813	017400	012737	000340	177776	MOV	#PR7,PS	:RESET PROCESSOR STATUS LEVEL
3814	017406	000470			BR	10\$	:REPORT 'RDYIN' FAILED TO SET (2ND) #=5/ INT. WRONG VEC.
3815	017410	062737	000002	002452	ADD	#2,ERNUM	
3816	017416	022626		6\$:	POP2SP		:CLEAN THE STACK
3817	017420	005726			POPSP		
3818							:NOW TEST THAT INTERRUPT OCCURS AT THE PROPER PROCESSOR STATUS LEVEL
3819	017422	012737	000340	177776	MOV	#PR7,PS	
3820	017430	004737	030664		JSR	PC,INTKMC	:INITIALIZE KMC-11 FIRMWARE
3821	017434	000455			BR	10\$	:F.W. FAILED TO INIT. #=7
3822	017436	004737	031274		JSR	PC,STCLK	:SET MAINT LOOP + START CLOCK
3823	017442	000452			BR	10\$	
3824	017444	005237	002452		INC	ERNUM	
3825	017450	012777	017540	162622	MOV	#9\$,@VECIN	:LOAD TRAP ADDRESS OF INTERRUPT
3826	017456	005037	001160		CLR	\$TMP0	
3827	017462	112777	000143	162570	MOVB	#143,@SELO	:INT. ENABLE REQ. IN.
3828	017470	105777	162564	7\$:	TSTB	@SELO	:IS REQUEST GRANTED?
3829	017474	100404			BMI	8\$	:B=YES
3830	017476	005237	001160		INC	\$TMP0	
3831	017502	001372			BNE	7\$	:NO TRY AGAIN

```

3832 017504 000431          BR      10$          ;REPORT REQUEST IN 'RDYIN' FAILED TO SET #=10
3833 017506 005237 002452 8$: INC      ERNUM1
3834 017512 162737 000040 177776 SUB     #40,FS          ;KEEP DROPPING STATUS UNTIL INT. OCCURS.
3835 017520 132737 000340 177776 BITB    #PR7,PS        ;AT LEVEL 0
3836 017526 001367          BNE     8$          ;B=NO
3837 017530 152737 000340 177776 BISE    #PR7,PS
3838 017536 000414          BR      10$          ;REPORT INT. FAILED
3839 017540 116637 000002 002464 9$: MOVB   2(SP),WAS      ;GET LEVEL INTERRUPT OCCURED AT
3840 017546 042737 000037 002464 BIC     #37,WAS
3841 017554 023737 002464 002304 CMP     WAS,PRIIN      ;WAS INT. LEVEL SAME AS SPECIFIED?
3842 017562 001002          BNE     10$          ;B=NO REPORT ERROR #=11
3843 017564 022626          POP2SP
3844 017566 000420          BR      TST34          ;;DONE
3845          ;ERROR ROUTINE
3846 017570          10$: JSR      PC,GETCSR      ;GET CONTENTS OF THE CSRS
3847 017570 004737 031470          ERROR+ 33
3848 017574 104033          JMP      1$
3849 017576 000137 017200
3850
3851          ;INTERRUPT TRAP SERVICE ROUTINE FOR ILLEGAL INTS.
3852 017602 013737 002300 002464 11$: MOV     VECIN, WAS      ;GET ADDRESS THAT INT. VECTORED TO
3853 017610 000403          BR      13$
3854 017612 013737 002302 002464 12$: MOV     VECOUT,WAS      ;GET ADDRESS THAT INT. VECTORED TO
3855 017620 022626          13$: POP2SP
3856 017622 005237 002452          INC     ERNUM
3857 017626 000760          BR      10$
3858
3859          ;FAIL #'S
3860          ;1=FAILURE IN TEST INITIALIZE ROUTINES (WAS=N/A)
3861          ;2='RDY-IN' FAILED TO SET (WAS=N/A)
3862          ;3=ILLEGAL INTERRUPT OCCURED (W/PS @ 7) (WAS=VECTOR TRAPPED TO)
3863          ;4=FIRMWARE INITIALIZE FAILED AFTER RDYIN SET (WAS=0=N/A)
3864          ;OR CLK FAILED TO START
3865          ;5=2ND. RDYIN FAILED TO SET (WAS=0=N/A)
3866          ;6=INTERRUPTED TO WRONG VECTOR ADDRESS (WAS=VECTOR ADDRESS TRAPPED TO)
3867          ;7=FIRMWARE FAILED TO INIT. AFTER INPUT INTERRUPT (WAS=0=N/A)
3868          ;10=3RD. RDY-IN FAILED TO SET (WAS=0=N/A)
3869          ;11=INTERRUPT OCCURED AT LEVEL OTHER THAN SPECIFIED (WAS=PS WORD @)
3870
3871          ;;*****
3872
3873          ;INTOUT
3874          ;THIS TEST WILL CHECK THAT AN OUTPUT REQUEST INTERRUPT CAN CORRECTLY BE
3875          ;EXECUTED TO INTERRUPT VECTOR XX4 AND TEST THE LOGIC AND
3876          ;FIRMWARE ARBITRATION.
3877          ;A TRANSMIT AND RECEIVE BUFFER IS QUEUED TO LINE 0
3878          ;WITH THE PROCESSOR STATUS AT 0 AND THE 'IEO' BIT NOT SET
3879          ;A TEST IS DONE TO CHECK THAT NO INTERRUPT OCCURS.
3880          ;THE FIRMWARE IS REINITIALIZED THE BUFFERS REQUEUED
3881          ;WITH THE 'IEO' BIT SET AND THE PROCESSOR STATUS AT 0
3882          ;A CHECK IS DONE TO TEST THAT THE INTERRUPT OCCURS TO THE
3883          ;OUTPUT VECTOR ADDRESS XX4.
3884          ;;*****
3885
3886
3887

```

```

3888
3889
3890
3891 017630 000004
3892 017632 012737 000034 002460
3893 017640 004737 030244
3894 017644 012737 017652 001110
3895 017652 005037 002464
3896 017656 012737 000001 002452
3897 017664 004737 030664
3898 017670 000565
3899
3900
3901
3902 017672 013700 002300
3903 017676 012720 020256
3904 017702 012720 000340
3905 017706 012720 020266
3906 017712 012710 000340
3907 017716 005237 002452
3908 017722 142737 000340 177776
3909 017730 012737 052660 002522
3910 017736 004737 031204
3911 017742 000540
3912 017744 004737 031274
3913 017750 000535
3914 017752 005237 002452
3915 017756 012737 000006 002526
3916 017764 004737 031344
3917 017770 000525
3918 017772 005237 002452
3919 017776 012737 000002 002526
3920 020004 004737 031354
3921 020010 000515
3922 020012 005237 002452
3923
3924 020016 005037 001162
3925 020022
3926 020022 004737 031742
3927 020026 000506
3928 020030 105077 162230
3929 020034 005137 001162
3930 020040 001370
3931
3932
3933
3934 020042 062737 000002 002452
3935 020050 005037 001162
3936 020054 012777 020224 162220
3937 020062 012737 000006 002526
3938 020070 004737 031344
3939 020074 000463
3940 020076 005237 002452
3941 020102 012737 000002 002526
3942 020110 004737 031354
3943 020114 000453

;TEST 34
*****
TST34: SCOPE
MOV #34,TSTNUM ;LOAD THE WD OF THIS TEST
JSR PC,SETLIN ;SETUP THE FIRST LINE
MOV #1$,SLPERR ;LOAD LOOP ADDRESS (LOOP ON ERROR)
1$: CLR WAS
MOV #1,ERR # ;INITIALIZE ERR #
JSR PC,INTKMC ;INITIALIZE KMC-11 FIRMWARE
BR 6$ ;FIRMWARE FAILED TO INIT. #=1
;PRIME INTERRUPT VECTORS, DROP PROCESSOR STATUS TO 0, AND
;QUEUE A RECEIVE AND TRANSMIT BUFFER TO LINE 0, CHECK
;THAT BOTH BUFFERS RETURN WITH OUT GENERATING AN INTERRUPT.
MOV VECIN,R0 ;GET VECTOR ADDRESS
MOV #7$, (R0)+ ;PRIME INPUT VECTOR
MOV #PR7, (R0)+
MOV #8$, (R0)+ ;PRIME OUTPUT VECTOR
MOV #PR7, (R0)
INC ERNUM
BICB #PR7,PS ;DROP PROCESSOR STATUS TO 0
MOV #RBUF,BUFADR ;SET BUFFER ADDRESS
JSR PC,INTLNE ;INITIALIZE LINE (# IN CURLIN)
BR 6$ ;LINE FAILED TO INITIALIZE #=1
JSR PC,STCLK ;SET MAINT LOOP + START CLOCK
BR 6$ ;MAINTENANCE COMMAND NOT ISSUED #=1
INC ERNUM
MOV #6,BUFCNT ;A LARGER REC. BUFF. FOR CRC-32
JSR PC,RIN ;LOAD A RECEIVE BUFFER IN
BR 6$ ;RECEIVE BUFFER FAILED TO LOAD #=2
INC ERNUM
MOV #2,BUFCNT ;XMIT BUFFER
JSR PC,XIN ;LOAD A TRANSMIT BUFFER IN
BR 6$ ;TRANSMIT BUFFER FAILED TO LOAD #=3
INC ERNUM
;WAIT FOR THE 2 BUFFERS TO RETURN
CLR $TMP1 ;BUFFER COUNT FLAG
2$: JSR PC,REQOUT ;TEST REQUEST OUT OF KMC SETS
BR 6$ ;BUFFER FAILED TO RETURN #=4 + 5=ILLEGAL INT W/O IEO
CLRB @SEL2 ;COMPLETE OUTPUT
COM $TMP1 ;2ND BUFFER
BNE 2$
;NOW SET THE INTERRUPT ENABLE OUT BIT PRIME OUTPUT
;VECTOR FOR LEGAL INTERRUPT QUEUE A XMIT AND RECEIVE
;BUFFER AND TEST FOR A VALID INTERRUPT.
ADD #2,ERNUM
CLR $TMP1 ;BUFFER COUNTER
MOV #5$,@VEECOUT ;PRIME OUTPUT VECTOR FOR LEGAL INT.
MOV #6,BUFCNT ;REC.BUFF. UP BY 4 BYTES FOR CRC-32
JSR PC,RIN ;LOAD A RECEIVE BUFFER IN
BR 6$ ;2ND REC. BUFFER FAILED TO LOAD. #=6
INC ERNUM
MOV #2,BUFCNT ;XMIT BUFFER
JSR PC,XIN ;LOAD A TRANSMIT BUFFER IN
BR 6$ ;2ND XMIT BUFFER FAILED TO LOAD. #=7

```

```
3944 020116 005237 002452 INC ERNUM
3945 020122 152777 000100 162134 BISB #100,ASEL2 ;ENABLE THE INTERRUPT
3946 020130 005037 001160 3$: CLR $TMP0 ;INTERRUPTS SHOULD OCCUR NOW
3947 020134 142737 000340 177776 BICB #PR7,PS ;RESET PROCESSOR STATUS TO 0
3948 020142 105777 162116 4$: TSTB ASEL2
3949 020146 100412 BMI 45$ ;OUTPUT RDY SET
3950 020150 005237 001160 INC $TMP0
3951 020154 001372 BNE 4$
3952 020156 152737 000340 177776 BISB #PR7,PS ;SET PROCESSOR --> 7
3953 020164 062737 000002 002452 ADD #2,ERNUM ;REPORT BUFFER FAILED TO RETURN #=12
3954 020172 000424 BR 6$ ;#10=INT. TO WRONG VECTOR ***
3955 020174 012737 000012 001164 45$: MOV #10,$TMP2 ;WAIT FOR POSSIBLE INT. DELAY(FIRM.W)
3956 020202 005337 001164 46$: DEC $TMP2
3957 020206 001375 BNE 46$
3958 020210 005237 002452 INC ERNUM
3959 020214 052737 000340 177776 BIS #PR7,PS
3960 020222 000410 BR 6$ ;FAIL #=11
3961 ;INT. FAILED
3962 020224 022626 5$: POP2SP ;CLEAN STACK
3963 020226 042777 000200 162030 BIC #200,ASEL2 ;COMPLETE OUTPUT
3964 020234 005137 001162 COM $TMP1
3965 020240 001421 BEQ TST35 ;:DONE
3966 020242 000732 BR 3$ ;WAIT FOR NEXT BUFFER
3967 ;ERROR ROUTINE
3968 020244 6$: JSR PC,GETCSR ;GET CONTENTS OF THE CSRS
3969 020244 004737 031470 ERROR+ 34
3970 020250 104034 JMP 1$
3971 020252 000137 017652
3972
3973 ;INTERRUPT TRAP SERVICE ROUTINE FOR ILLEGAL INTS.
3974 020256 013737 002300 002464 7$: MOV VECIN,WAS ;ADDRESS INT VECTORED TO
3975 020264 000403 BR 9$
3976 020266 013737 002302 002464 8$: MOV VECOUT,WAS
3977 020274 022626 9$: POP2SP
3978 020276 005237 002452 INC ERNUM
3979 020302 000760 BR 6$
3980
3981
3982 ;FAIL #S
3983 ;1=FAILURE IN TEST INITIALIZATION ROUTINES (WAS=N/A)
3984 ;2=RECEIVE BUFFER FAILED TO LOAD
3985 ;3=TRANSMIT BUFFER FAILED TO LOAD
3986 ;4=BUFFER FAILED TO RETURN
3987 ;5=ILLEGAL INTERRUPT WITHOUT IEO SET (WAS=VECTOR TRAPPED TO)
3988 ;6=2ND RECEIVE BUFFER FAILED TO LOAD
3989 ;7=2ND XMIT BUFFER FAILED TO LOAD
3990 ;10=INTERRUPT OCCURED TO WRONG VECTOR (WAS=VECTOR TRAPPED TO)
3991 ;11=INTERRUPT FIALED WHEN 'RDYOUT' SET (WAS=N/A)
3992 ;12=BUFFER FAILED TO RETURN (NO 'RDYOUT') (WAS=N/A)
3993
3994 ;*****
3995 020304 0000C4 TST35: SCOPE
3996 020306 005037 001102 CLR $TSTNM
3997 020312 005037 002460 CLR TSTNUM
3998 020316 005737 000042 TST @#42 ;IF A MONITOR NO UNIT PASSES REPORTED
3999 020322 001021 BNE 1$
```

CZKMEAO DMS11-DA DYNAMIC
CZKMEA.P11 20-OCT-81 17:05

MACY11 30A(1052) 21-OCT-81 08:50 PAGE 80
GET VALUE FOR SOFTWARE SWITCH REGISTER

SEQ 0080

```

4000 020324 032777 100000 160606      BIT      #BIT15,@SWR      ;INHIBIT END REPORTS?
4001 020332 001015                BNE      1$                ;BR=YES
4002 020334 104401 020400          TYPE      ,UNTDON
4003 020340 013746 002456          MOV      CUNIT,-(SP)
4004 020344 104405                TYPDS
4005 020346 105737 002311          TSTB      MODE                ;BOP OR CCP
4006 020352 100003                BPL      15$                ;B=CCP
4007 020354 104401 020416          TYPE      ,MBOP
4008 020360 000402                BR        1$
4009 020362 104401 020410          15$:    TYPE      ,MCCP
4010 020366                1$:    ;GET NEXT KMC OR REPORT AN END PASS
4011 020366 004737 026506          JSR      PC,GETKMC          ;GET NEXT ENABLED KMC11
4012 020372 000414                BR        TST36            ;:REPORT AN END PASS
4013 020374 000137 002376          JMP      LTEST
4014 020400 005015 047125 052111 UNTDON: .ASCIZ  <15><12>/UNIT /
4015 020406 000040
4016 020410 020040 041503 000120 MCCP:  .ASCIZ  /  CCP/
4017 020416 020040 047502 000120 MBOP:  .ASCIZ  /  BOP/
4018
4019                .EVEN
4020                ;:*****
4021 020424 000004                TST36:  SCOPE
4022
4023                .SBTTL  END OF PASS ROUTINE
4024
4025                ;:*****
4026                ;*INCREMENT THE PASS NUMBER ($PASS)
4027                ;*INDICATE END-OF-PROGRAM AFTER 1 PASSES THRU THE PROGRAM
4028                ;*TYPE 'END PASS #XXXXX' (WHERE XXXXX IS A DECIMAL NUMBER)
4029                ;*IF THERES A MONITOR GO TO IT
4030                ;*IF THERE ISN'T JUMP TO RESTRT
4031
4032                $EOP:
4033 020426 000004                SCOPE
4034 020430 005037 001102          CLR      $STNM              ;;ZERO THE TEST NUMBER
4035 020434 005037 001166          CLR      $TIMES            ;;ZERO THE NUMBER OF ITERATIONS
4036 020440 005237 001100          INC      $PASS              ;;INCREMENT THE PASS NUMBER
4037 020444 042737 100000 001100 BIC      #100000,$PASS    ;;DON'T ALLOW A NEG. NUMBER
4038 020452 005327                DEC      (PC)+              ;;LOOP?
4039 020454 000001                $EOPCT: .WORD 1
4040 020456 003022                BGT      $DOAGN              ;;YES
4041 020460 012737                MOV      (PC)+,@(PC)+        ;;RESTORE COUNTER
4042 020462 000001                $ENDCT: .WORD 1
4043 020464 020454                $EOPCT
4044 020466 104401 020533          TYPE      ,SENDMG          ;;TYPE 'END PASS #'
4045 020472 013746 001100          MOV      $PASS,-(SP)        ;;SAVE $PASS FOR TYPEOUT
4046 020476 104405                TYPDS
4047 020500 104401 020530          TYPE      ,SENULL          ;;GO TYPE--DECIMAL ASCII WITH SIGN
4048 020504 013700 000042          $GET42: MOV      @#42,R0      ;;TYPE A NULL CHARACTER
4049 020510 001405                BEQ      $DOAGN              ;;GET MONITOR ADDRESS
4050 020512 000005                RESET                      ;;BRANCH IF NO MONITOR
4051 020514 004710                $ENDAD: JSR      PC,(R0)      ;;CLEAR THE WORLD
4052 020516 000240                NOP                      ;;GO TO MONITOR
4053 020520 000240                NOP                      ;;SAVE ROOM
4054 020522 000240                NOP                      ;;FOR
4055 020524                $DOAGN:                ;;ACT11
```

```
4056 020524 000137      JMP      @PC)+          ;;RETURN
4057 020526 002362      $RTNAD: .WORD  RESTRT
4058 020530      377      377      000      $ENULL: .BYTE  -1,-1,0          ;;NULL CHARACTER STRING
4059 020533      015      042412  042116  $ENDMG: .ASCIIZ <15><12>/END PASS #/
4060 020540 050040 051501 020123
4061 020546 000043
4062                                     .SBTTL  SCOPE HANDLER ROUTINE
4063
4064                                     ;*****
4065                                     ;*THIS ROUTINE CONTROLS THE LOOPING OF SUBTESTS. IT WILL INCREMENT
4066                                     ;*AND LOAD THE TEST NUMBER($TSTNM) INTO THE DISPLAY REG.(DISPLAY<7:0>)
4067                                     ;*AND LOAD THE ERROR FLAG ($ERFLG) INTO DISPLAY<15:08>
4068                                     ;*THE SWITCH OPTIONS PROVIDED BY THIS ROUTINE ARE:
4069                                     ;*SW14=1      LOOP ON TEST
4070                                     ;*SW11=1      INHIBIT ITERATIONS
4071                                     ;*SW09=1      LOOP ON ERROR
4072                                     ;*SW08=1      LOOP ON TEST IN SWR<7:0>
4073                                     ;*CALL
4074                                     ;*      SCOPE          ;;SCOPE=IOT
4075
4076 020550      $SCOPE:
4077 020550 104407      CKSWR          ;;TEST FOR CHANGE IN SOFT-SWR
4078                                     ;;GO TO ERROR ROUTINE IF RETURN PC LESS THAN 1002
4079                                     ;;OTHERWISE CONTINUE
4080 020552 021627 001002      CMP      (SP),#1002          ;;UNEXPECTED TRAP OR INTERRUPT
4081 020556 101002      BHI      1$          ;;ARE TRAPPED HERE VIA IOT
4082 020560 000137 021034      JMP      $ERROR          ;;GO PROCESS UNEXPECTED TRAP
4083 020564 032777 040000 160346 1$:      BIT      #BIT14,@SWR          ;;LOOP ON PRESENT TEST?
4084 020572 001111      BNE      $OVER          ;;YES IF SW14=1
4085                                     ;####START OF CODE FOR THE XOR TESTER####
4086 020574 000416      $XTSTR: BR      6$          ;;IF RUNNING ON THE 'XOR' TESTER CHANGE
4087                                     ;;THIS INSTRUCTION TO A 'NOP' (NOP=240)
4088 020576 013746 000004      MOV      @ERRVEC,-(SP)          ;;SAVE THE CONTENTS OF THE ERROR VECTOR
4089 020602 012737 020622 000004      MOV      #5$,@ERRVEC          ;;SET FOR TIMEOUT
4090 020610 005737 177060      TST      @177060          ;;TIME OUT ON XOR?
4091 020614 012637 000004      MOV      (SP)+,@ERRVEC          ;;RESTORE THE ERROR VECTOR
4092 020620 000463      BR      $SVLAD          ;;GO TO THE NEXT TEST
4093 020622 022626      5$:      CMP      (SP)+,(SP)+          ;;CLEAR THE STACK AFTER A TIME OUT
4094 020624 012637 000004      MOV      (SP)+,@ERRVEC          ;;RESTORE THE ERROR VECTOR
4095 020630 000423      BR      7$          ;;LOOP ON THE PRESENT TEST
4096 020632      6$:;####END OF CODE FOR THE XOR TESTER####
4097 020632 032777 000400 160300      BIT      #BIT08,@SWR          ;;LOOP ON SPEC. TEST?
4098 020640 001404      BEQ      2$          ;;BR IF NO
4099 020642 127737 160272 001102      CMPB    @SWR,$TSTNM          ;;ON THE RIGHT TEST? SWR<7:0>
4100 020650 001462      BEQ      $OVER          ;;BR IF YES
4101 020652 105737 001103      2$:      TSTB    $ERFLG          ;;HAS AN ERROR OCCURRED?
4102 020656 001421      BEQ      3$          ;;BR IF NO
4103 020660 123737 001115 001103      CMPB    $ERMAX,$ERFLG          ;;MAX. ERRORS FOR THIS TEST OCCURRED?
4104 020666 101015      BHI      3$          ;;BR IF NO
4105 020670 032777 001000 160242      BIT      #BIT09,@SWR          ;;LOOP ON ERROR?
4106 020676 001404      BEQ      4$          ;;BR IF NO
4107 020700 013737 001110 001106 7$:      MOV      $LPERR,$LPADR          ;;SET LOOP ADDRESS TO LAST SCOPE
4108 020706 000443      BR      $OVER
4109 020710 105037 001103      4$:      CLRB    $ERFLG          ;;ZERO THE ERROR FLAG
4110 020714 005037 001166      CLR      $TIMES          ;;CLEAR THE NUMBER OF ITERATIONS TO MAKE
4111 020720 000415      BR      1$          ;;ESCAPE TO THE NEXT TEST
```

```
4112 020722 032777 004000 160210 3$: BIT #BIT11,@SWR ;;INHIBIT ITERATIONS?
4113 020730 001011 BNE 1$ ;;BR IF YES
4114 020732 005737 001100 TST $PASS ;;IF FIRST PASS OF PROGRAM
4115 020736 001406 BEQ 1$ ;; INHIBIT ITERATIONS
4116 020740 005237 001104 INC $ICNT ;;INCREMENT ITERATION COUNT
4117 020744 023737 001166 001104 CMP $TIMES,$ICNT ;;CHECK THE NUMBER OF ITERATIONS MADE
4118 020752 002021 BGE $OVER ;;BR IF MORE ITERATION REQUIRED
4119 020754 012737 000001 001104 1$: MOV #1,$ICNT ;;REINITIALIZE THE ITERATION COUNTER
4120 020762 013737 021032 001166 MOV $MXCNT,$TIMES ;;SET NUMBER OF ITERATIONS TO DO
4121 020770 105237 001102 $SVLAD: INCB $STNM ;;COUNT TEST NUMBERS
4122 020774 011637 001106 MOV (SP),$LPADR ;;SAVE SCOPE LOOP ADDRESS
4123 021000 011637 001110 MOV (SP),$LPERR ;;SAVE ERROR LOOP ADDRESS
4124 021004 005037 001170 CLR $ESCAPE ;;CLEAR THE ESCAPE FROM ERROR ADDRESS
4125 021010 112737 000001 001115 MOVB #1,$ERMAX ;;ONLY ALLOW ONE(1) ERROR ON NEXT TEST
4126 021016 013777 001102 160116 $OVER: MOV $STNM,@DISPLAY ;;DISPLAY TEST NUMBER
4127 021024 013716 001106 MOV $LPADR,(SP) ;;FUDGE RETURN ADDRESS
4128 021030 000002 RTI ;;FIXES PS
4129 021032 000001 $MXCNT: 1 ;;MAX. NUMBER OF ITERATIONS
4130 .SBTTL ERROR HANDLER ROUTINE
4131
4132 *****
4133 *THIS ROUTINE WILL INCREMENT THE ERROR FLAG AND THE ERROR COUNT,
4134 *SAVE THE ERROR ITEM NUMBER AND THE ADDRESS OF THE ERROR CALL
4135 *AND GO TO $ERRTYP ON ERROR
4136 *THE SWITCH OPTIONS PROVIDED BY THIS ROUTINE ARE:
4137 *$SW15=1 HALT ON ERROR
4138 *$SW13=1 INHIBIT ERROR TYPEOUTS
4139 *$SW10=1 BELL ON ERROR
4140 *$SW09=1 LOOP ON ERROR
4141 *CALL
4142 * ERROR N ;;ERROR=EMT AND N=ERROR ITEM NUMBER
4143
4144 $ERROR:
4145 021034 104407 CKSWR ;;TEST FOR CHANGE IN SOFT-SWR
4146 021036 105237 001103 7$: INCB $ERFLG ;;SET THE ERROR FLAG
4147 021042 001775 BEQ 7$ ;;DON'T LET THE FLAG GO TO ZERO
4148 021044 013777 001102 160070 MOV $STNM,@DISPLAY ;;DISPLAY TEST NUMBER AND ERROR FLAG
4149 021052 032777 002000 160060 BIT #BIT10,@SWR ;;BELL ON ERROR?
4150 021060 001402 BEQ 1$ ;;NO - SKIP
4151 021062 104401 001172 TYPE $BELL ;;RING BELL
4152 021066 005237 001112 1$: INC $ERTTL ;;COUNT THE NUMBER OF ERRORS
4153 021072 011637 001116 MOV (SP),$ERRPC ;;GET ADDRESS OF ERROR INSTRUCTION
4154 021076 162737 000002 001116 SUB #2,$ERRPC
4155 021104 117737 160006 001114 MOVB @ $ERRPC,$ITEMB ;;STRIP AND SAVE THE ERROR ITEM CODE
4156 021112 032777 020000 160020 BIT #BIT13,@SWR ;;SKIP TYPEOUT IF SET
4157 021120 001055 BNE 20$ ;;SKIP TYPEOUTS
4158 021122 021627 001002 CMP (SP),#1002 ;;IF RETURN PC LESS THAN 1002
4159 021126 101046 BHI 12$ ;;ERROR IS ILLEGAL TRAP
4160 ;;PROCESS UNEXPECTED TRAP OR INTERRUPT
4161 021130 016637 000004 001116 MOV 4(SP),$ERRPC ;;GET PC AT TIME OF FALSE TRAP
4162 021136 162737 000002 001116 SUB #2,$ERRPC ;;ADJUST PC
4163 021144 104401 021210 TYPE 10$ ;;TYPE HEADER
4164 021150 013746 001116 MOV $ERRPC,-(SP) ;;SAVE $ERRPC FOR TYPEOUT
4165 021154 104402 TYPEOC ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
4166 021156 104401 021216 TYPE 11$
4167 021162 162716 000004 SUB #4,(SP) ;;GET FALSE TRAP VECTOR ADDR
```

```
4168 021166 011637 001116      MOV      (SP), $ERRPC
4169 021172 013746 001116      MOV      $ERRPC, -(SP)      ;;SAVE $ERRPC FOR TYPEOUT
4170 021176 104402                TYP0C      ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
4171 021200 104401 001177      TYPE      $CRLF
4172 021204 022626                CMP      (SP)+, (SP)+      ;;POP FALSE TRAP VECTOR PC&ADDR
4173 021206 000422                BR        20$
4174 021210 050200 036503 000040 10$: .ASCIZ    <200>'PC= '
4175 021216 020040 047125 054105 11$: .ASCIZ    ' UNEXPECTED TRAP TO '
4176 021224 042520 052103 042105
4177 021232 052040 040522 020120
4178 021240 047524 000040
4179
4180 021244                12$: .EVEN
4181 021244 004737 021330      JSR      PC, $ERRTYP      ;;GO TO USER ERROR ROUTINE
4182 021250 104401 001177      TYPE      $CRLF
4183 021254                20$:
4184 021254 005777 157660      2$: TST      @SWR      ;;HALT ON ERROR
4185 021260 100002                BPL      3$      ;;SKIP IF CONTINUE
4186 021262 000000                HALT      ;;HALT ON ERROR!
4187 021264 104407                CKSWR      ;;TEST FOR CHANGE IN SOFT-SWR
4188 021266 032777 001000 157644 3$: BIT      #BIT09, @SWR      ;;LOOP ON ERROR SWITCH SET?
4189 021274 001402                BEQ      4$      ;;BR IF NO
4190 021276 013716 001110      MOV      $LPERR, (SP)      ;;FUDGE RETURN FOR LOOPING
4191 021302 005737 001170      4$: TST      $ESCAPE      ;;CHECK FOR AN ESCAPE ADDRESS
4192 021306 001402                BEQ      5$      ;;BR IF NONE
4193 021310 013716 001170      MOV      $ESCAPE, (SP)      ;;FUDGE R TURN ADDRESS FOR ESCAPE
4194 021314                5$:
4195 021314 022737 020514 000042      CMP      #SENDAD, @#42      ;;ACT-11 AUTO-ACCEPT?
4196 021322 001001                BNE      6$      ;;BRANCH IF NO
4197 021324 000000                HALT      ;;YES
4198 021326
4199 021326 000002      6$: RTI      ;;RETURN
4200 .SBTTL  ERROR MESSAGE TYPEOUT ROUTINE
4201
4202 *****
4203 *THIS ROUTINE USES THE 'ITEM CONTROL BYTE' ($ITEMB) TO DETERMINE WHICH
4204 *ERROR IS TO BE REPORTED. IT THEN OBTAINS, FROM THE 'ERROR TABLE' ($ERRTB),
4205 *AND REPORTS THE APPROPRIATE INFORMATION CONCERNING THE ERROR.
4206
4207 $ERRTYP:
4208 021330 104401 001177      TYPE      $CRLF      ;;'CARRIAGE RETURN' & 'LINE FEED'
4209 021334 010046                MOV      R0, -(SP)      ;;SAVE R0
4210 021336 005000                CLR      R0      ;;PICKUP THE ITEM INDEX
4211 021340 153700 001114      BISB     @#$ITEMB, R0
4212 021344 001004                BNE      1$      ;;IF ITEM NUMBER IS ZERO, JUST
4213                                ;;TYPE THE PC OF THE ERROR
4214 021346 013746 001116      MOV      $ERRPC, -(SP)      ;;SAVE $ERRPC FOR TYPEOUT
4215                                ;;ERROR ADDRESS
4216 021352 104402                TYP0C      ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
4217 021354 000445                BR        10$      ;;GET OUT
4218 021356 005300      1$: DEC      R0      ;;ADJUST THE INDEX SO THAT IT WILL
4219 021360 006300                ASL      R0      ;;WORK FOR THE ERROR TABLE
4220 021362 006300                ASL      R0
4221 021364 006300                ASL      R0
4222 021366 062700 001202      ADD      #ERRTB, R0      ;;FORM TABLE POINTER
4223 021372 012037 021402      MOV      (R0)+, 2$      ;;PICKUP 'ERROR MESSAGE' POINTER
```

```
4224 021376 001404      BEQ      3$      ;;SKIP TYPEOUT IF NO POINTER
4225 021400 104401      TYPE      ;;TYPE THE 'ERROR MESSAGE'
4226 021402 000000      2$:      .WORD    0      ;;'ERROR MESSAGE' POINTER GOES HERE
4227 021404 104401 001177      TYPE      , $CRLF      ;;'CARRIAGE RETURN' & 'LINE FEED'
4228 021410 012037 021420      3$:      MOV      (R0)+, 4$      ;;PICKUP 'DATA HEADER' POINTER
4229 021414 001404      BEQ      5$      ;;SKIP TYPEOUT IF 0
4230 021416 104401      TYPE      ;;TYPE THE 'DATA HEADER'
4231 021420 000000      4$:      .WORD    0      ;;'DATA HEADER' POINTER GOES HERE
4232 021422 104401 001177      TYPE      , $CRLF      ;;'CARRIAGE RETURN' & 'LINE FEED'
4233 021426 010146      5$:      MOV      R1, -(SP)      ;;SAVE R1
4234 021430 012001      MOV      (R0)+, R1      ;;PICKUP 'DATA TABLE' POINTER
4235 021432 001415      BEQ      9$      ;;BR IF NO DATA TO BE TYPED
4236 021434 012000      MOV      (R0)+, R0      ;;PICKUP 'DATA FORMAT' POINTER
4237 021436 105720      6$:      TSTB      (R0)+      ;;'OCTAL' OR 'DECIMAL'
4238 021440 001003      BNE      7$      ;;BR IF DECIMAL
4239 021442 013146      MOV      @ (R1)+, -(SP)      ;;SAVE @ (R1)+ FOR TYPEOUT
4240 021444 104402      TYPOC      ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
4241 021446 000402      BR      8$
4242 021450      7$:
4243 021450 013146      MOV      @ (R1)+, -(SP)      ;;SAVE @ (R1)+ FOR TYPEOUT
4244 021452 104405      TYPDS      ;;GO TYPE--DECIMAL ASCII WITH SIGN
4245 021454 005711      8$:      TST      (R1)      ;;IS THERE ANOTHER NUMBER?
4246 021456 001403      BEQ      9$      ;;BR IF NO
4247 021460 104401 021500      TYPE      , 11$      ;;TYPE TWO(2) SPACES
4248 021464 000764      BR      6$      ;;LOOP
4249
4250 021466 012601      9$:      MOV      (SP)+, R1      ;;RESTORE R1
4251 021470 012600      10$:     MOV      (SP)+, R0      ;;RESTORE R0
4252 021472 104401 001177      TYPE      , $CRLF      ;;'CARRIAGE RETURN' & 'LINE FEED'
4253 021476 000207      RTS      PC      ;;RETURN
4254 021500 020040 000      11$:     .ASCIZ    / /      ;;TWO(2) SPACES
4255 021504 021504      .EVEN
4256      .SBTTL  READ AN OCTAL NUMBER FROM THE TTY
4257
4258      ;*****
4259      ;*THIS ROUTINE WILL READ AN OCTAL (ASCII) NUMBER FROM THE TTY AND
4260      ;*CHANGE IT TO BINARY.
4261      ;*THE INPUT CHARACTERS WILL BE CHECKED TO INSURED THEY ARE LEGAL
4262      ;*OCTAL DIGITS. IF AN ILLEGAL CHARACTER IS READ A '?' WILL BE TYPED
4263      ;*FOLLOWED BY A CARRIAGE RETURN-LINE FEED. THE COMPLETE NUMBER MUST
4264      ;*THEN BE RETYPED. THE INPUT IS TERMINATED BY TYPING A CARRIAGE RETURN.
4265      ;*CALL:
4266      ;*      RDOCT      ;;READ AN OCTAL NUMBER
4267      ;*      RETURN HERE  ;;LOW ORDER BITS ARE ON TOP OF THE STACK
4268      ;*                  ;;HIGH ORDER BITS ARE IN $HIOCT
4269
4270 021504 011646 000004 000002 $RDOCT: MOV      (SP), -(SP)      ;;PROVIDE SPACE FOR THE
4271 021506 016666      MOV      4(SP), 2(SP)      ;;INPUT NUMBER
4272 021514 010046      MOV      R0, -(SP)      ;;PUSH R0 ON STACK
4273 021516 010146      MOV      R1, -(SP)      ;;PUSH R1 ON STACK
4274 021520 010246      MOV      R2, -(SP)      ;;PUSH R2 ON STACK
4275 021522 104411      1$:      RDLIN      ;;READ AN ASCII LINE
4276 021524 012600      MOV      (SP)+, R0      ;;GET ADDRESS OF 1ST CHARACTER
4277 021526 010037 021632      MOV      R0, 5$      ;;AND SAVE IT
4278 021532 005001      CLR      R1      ;;CLEAR DATA WORD
4279 021534 005002      CLR      R2
```

```
4280 021536 112046      2$:  MOVB  (R0)+,-(SP)  ;;PICKUP THIS CHARACTER
4281 021540 001420      BEQ    3$          ;;IF ZERO GET OUT
4282 021542 122716 000060  CMPB  #'0,(SP)  ;;MAKE SURE THIS CHARACTER
4283 021546 003026      BGT    4$          ;;IS AN OCTAL DIGIT
4284 021550 122716 000067  CMPB  #'7,(SP)
4285 021554 002423      BLT    4$
4286 021556 006301      ASL    R1          ;;*2
4287 021560 006102      ROL    R2
4288 021562 006301      ASL    R1          ;;*4
4289 021564 006102      ROL    R2
4290 021566 006301      ASL    R1          ;;*8
4291 021570 006102      ROL    R2
4292 021572 042716 177770  BIC    #'C7,(SP)  ;;STRIP THE ASCII JUNK
4293 021576 062601      ADD    (SP)+,R1  ;;ADD IN THIS DIGIT
4294 021600 000756      BR     2$          ;;LOOP
4295 021602 005726      3$:  TST    (SP)+  ;;CLEAN TERMINATOR FROM STACK
4296 021604 010166 000012  MOV    R1,12(SP)  ;;SAVE THE RESULT
4297 021610 010237 021642  MOV    R2,$HIOCT
4298 021614 012602      MOV    (SP)+,R2  ;;POP STACK INTO R2
4299 021616 012601      MOV    (SP)+,R1  ;;POP STACK INTO R1
4300 021620 012600      MOV    (SP)+,R0  ;;POP STACK INTO R0
4301 021622 000002      RTI          ;;RETURN
4302 021624 005726      4$:  TST    (SP)+  ;;CLEAN PARTIAL FROM STACK
4303 021626 105010      CLRB  (R0)  ;;SET A TERMINATOR
4304 021630 104401      TYPE          ;;TYPE UP THRU THE BAD CHAR.
4305 021632 000000      5$:  .WORD  0
4306 021634 104401 001176  TYPE    ,SQUES  ;;'"'"' 'CR' & 'LF'
4307 021640 000730      BR     1$          ;;TRY AGAIN
4308 021642 000000      $HIOCT: .WORD  0  ;;HIGH ORDER BITS GO HERE
4309      .SBTTL  TTY INPUT ROUTINE
4310
4311      ;*****
4312      .ENABL  LSB
4313 021644 000000      $TKCNT: .WORD  0  ;;NUMBER OF ITEMS IN QUEUE
4314 021646 000000      $TKQIN: .WORD  0  ;;INPUT POINTER
4315 021650 000000      $TKQOUT: .WORD  0  ;;OUTPUT POINTER
4316 021652 000005      $TKQSR: .BLKB  5.  ;;TTY KEYBOARD QUEUE
4317      $TKQEND=.
4318      .EVEN
4319
4320      ;*TK INITIALIZE ROUTINE
4321      ;*THIS ROUTINE WILL INITIALIZE THE TTY KEYBOARD INPUT QUEUE
4322      ;*SETUP THE INTERRUPT VECTOR AND TURN ON THE KEYBOARD INTERRUPT
4323      ;
4324      ;*CALL:
4325      ;*      JSR    PC,$TKINT
4326      ;*      RETURN
4327      ;
4328 021660 005037 021644      $TKINT: CLR    $TKCNT  ;;CLEAR COUNT OF ITEMS IN QUEUE
4329 021664 012737 021652 021646  MOV    # $TKQSR,$TKQIN  ;;MOVE THE STARTING ADDRESS OF THE
4330 021672 013737 021646 021650  MOV    $TKQIN,$TKQOUT  ;;QUEUE INTO THE INPUT & OUTPUT POINTERS.
4331 021700 012737 021730 000060  MOV    # $TKSRV,$TKVEC  ;;INITIALIZE THE KEYBOARD VECTOR
4332 021706 012737 000200 000062  MOV    #200,$TKVEC+2  ;;'BR' LEVEL 4
4333 021714 005777 157226      TST    @ $TKB  ;;CLEAR DONE FLAG
4334 021720 012777 000100 157216  MOV    #100,$TKS  ;;ENABLE TTY KEYBOARD INTERRUPT
4335 021726 000207      RTS    PC  ;;RETURN TO CALLER
```

```
4336
4337
4338      ;*TK SERVICE ROUTINE
4339      ;*THIS ROUTINE WILL SERVICE THE TTY KEYBOARD INTERRUPT
4340      ;*BY READING THE CHARACTER FROM THE INPUT BUFFER AND PUTTING
4341      ;*IT IN THE QUEUE.
4342      ;*IF THE CHARACTER IS A "CONTROL-C" (^C) $TKINT IS CALLED AND
4343      ;*UPON RETURN EXIT IS MADE TO THE "CONTROL-C" RESTART ADDRESS (MONIT)
4344      $TKSRV: MOVB    @STKB,-(SP)      ;;PICKUP THE CHARACTER
4345              BIC     #^C177,(SP)    ;;STRIP THE JUNK
4346              CMP     (SP),#$XON     ;;IS IT A RANDOM XON?           ;RAN001
4347              BNE     30$            ;;BRANCH IF NO                 ;RAN001
4348              TST     (SP)+          ;;CLEAN RANDOM XON OFF STACK   ;RAN001
4349              RTI                    ;;RETURN                       ;RAN001
4350
4351      30$:      CMP     (SP),#3        ;;IS IT A CONTROL C?
4352              BNE     1$            ;;BRANCH IF NO
4353              TYPE    $CNTLC         ;;TYPE A CONTROL-C (^C)
4354              JSR     PC,$TKINT      ;;INIT THE KEYBOARD
4355              TST     (SP)+          ;;CLEAN UP STACK
4356              JMP     MONIT          ;;CONTROL C RESTART
4357      1$:      CMP     (SP),#7        ;;IS IT A CONTROL G?
4358              BNE     2$            ;;BRANCH IF NO
4359              CMP     #SWREG,SWR     ;;IS SOFT-SWR SELECTED?
4360              BEQ     6$            ;;GO TO SWR CHANGE
4361
4362      2$:      CMP     #5,$TKCNT      ;;IS THE QUEUE FULL?
4363              BNE     3$            ;;BRANCH IF NO
4364              TYPE    $BELL          ;;RING THE TTY BELL
4365              TST     (SP)+          ;;CLEAN CHARACTER OFF OF STACK
4366              BR      5$            ;;EXIT
4367      3$:      CMP     (SP),#23       ;;IS IT A CONTROL-S?
4368              BNE     32$           ;;BRANCH IF NO
4369              CLR     @STKS         ;;DISABLE TTY KEYBOARD INTERRUPTS
4370              TST     (SP)+          ;;CLEAN CHAR OFF STACK
4371      31$:     TSTB    @STKS         ;;WAIT FOR A CHAR
4372              BPL     31$           ;;LOOP UNTIL ITS THERE
4373              MOVB    @STKB,-(SP)    ;;GET THE CHARACTER
4374              BIC     #^C177,(SP)    ;;MAKE IT 7-BIT ASCII
4375              CMP     (SP)+,#21      ;;IS IT A CONTROL-Q?
4376              BNE     31$           ;;BRANCH IF NO
4377              MOV     #100,@STKS     ;;REENABLE TTY KEYBOARD INTERRUPTS
4378              RTI                    ;;RETURN
4379
4380      32$:     INC     $TKCNT         ;;COUNT THIS CHARACTER
4381              CMP     (SP),#140      ;;IS IT UPPER CASE?
4382              BLT     4$            ;;BRANCH IF YES
4383              CMP     (SP),#175      ;;IS IT A SPECIAL CHAR?
4384              BGT     4$            ;;BRANCH IF YES
4385              BIC     #40,(SP)       ;;MAKE IT UPPER CASE
4386      4$:      MOVB    (SP)+,@$TKQIN  ;;AND PUT IT IN QUEUE
4387              INC     $TKQIN         ;;UPDATE THE POINTER
4388              CMP     $TKQIN,$$TKQEND ;;GO OFF THE END?
4389              BNE     5$            ;;BRANCH IF NO
4390              MOV     #STKQSRST,$TKQIN ;;RESET THE POINTER
4391      5$:      RTI                    ;;RETURN
```

```
4392
4393
4394
4395
4396
4397
4398 022160 022737 000176 001140 $CKSWR: CMP #SWREG,SWR ;;IS THE SOFT-SWR SELECTED
4399 022166 001124 BNE 15$ ;;EXIT IF NOT
4400 022170 105777 156750 TSTB @STKS ;;IS A CHAR WAITING?
4401 022174 100121 BPL 15$ ;;IF NOT, EXIT
4402 022176 117746 156744 MOVB @STKB,-(SP) ;;YES
4403 022202 042716 177600 BIC #^C177,(SP) ;;MAKE IT 7-BIT ASCII
4404 022206 021627 000007 CMP (SP),#7 ;;IS IT A CONTROL-G?
4405 022212 001300 BNE 2$ ;;IF NOT, PUT IT IN THE TTY QUEUE
4406 ;;AND EXIT
4407
4408
4409
4410
4411
4412 022214 123727 001134 000001 6$: CMPB $AUTOB,#1 ;;ARE WE RUNNING IN AUTO-MODE?
4413 022222 001674 BEQ 2$ ;;BRANCH IF YES
4414 022224 005726 TST (SP)+ ;;CLEAR CONTROL-G OFF STACK
4415 022226 004737 021660 JSR PC,$TKINT ;;FLUSH THE TTY INPUT QUEUE
4416 022232 005077 156706 CLR @STKS ;;DISABLE TTY KEYBOARD INTERRUPTS
4417 022236 112737 000001 001135 MOVB #1,$INTAG ;;SET INTERRUPT MODE INDICATOR
4418
4419 022244 104401 023104 $GTSWR: TYPE , $CNTLG ;;ECHO THE CONTROL-G (^G)
4420 022250 104401 023111 TYPE , $MSWR ;;TYPE CURRENT CONTENTS
4421 022254 013746 000176 MOV SWREG,-(SP) ;;SAVE SWREG FOR TYPEOUT
4422 022260 104402 TYPOC ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
4423 022262 104401 023122 TYPE , $MNEW ;;PROMPT FOR NEW SWR
4424 022266 005046 19$: CLR -(SP) ;;CLEAR COUNTER
4425 022270 005046 CLR -(SP) ;;THE NEW SWR
4426 022272 105777 156646 7$: TSTB @STKS ;;CHAR THERE?
4427 022276 100375 BPL 7$ ;;IF NOT TRY AGAIN
4428
4429 022300 117746 156642 MOVB @STKB,-(SP) ;;PICK UP CHAR
4430 022304 042716 177600 BIC #^C177,(SP) ;;MAKE IT 7-BIT ASCII
4431
4432 022310 021627 000003 CMP (SP),#3 ;;IS IT A CONTROL-C?
4433 022314 001015 BNE 9$ ;;BRANCH IF NOT
4434 022316 104401 023072 TYPE , $CNTLC ;;YES, ECHO CONTROL-C (^C)
4435 022322 062706 000006 ADD #6,SP ;;CLEAN UP STACK
4436 022326 123727 001135 000001 CMPB $INTAG,#1 ;;REENABLE TTY KEYBOARD INTERRUPTS?
4437 022334 001003 BNE 8$ ;;BRANCH IF NO
4438 022336 012777 000100 156600 MOV #100,@STKS ;;ALLOW TTY KEYBOARD INTERRUPTS
4439 022344 000137 002076 8$: JMP MONIT ;;CONTROL-C RESTART
4440
4441
4442 022350 021627 000025 9$: CMP (SP),#25 ;;IS IT A CONTROL-U?
4443 022354 001005 BNE 10$ ;;BRANCH IF NOT
4444 022356 104401 023077 TYPE , $CNTLU ;;YES, ECHO CONTROL-U (^U)
4445 022362 062706 000006 20$: ADD #6,SP ;;IGNORE PREVIOUS INPUT
4446 022366 000737 BR 19$ ;;LET'S TRY IT AGAIN
4447
```

```
4448
4449 022370 021627 000015      10$: CMP      (SP),#15      ;;IS IT A <CR>?
4450 022374 001022              BNE      16$      ;;BRANCH IF NO
4451 022376 005766 000004      TST      4(SP)      ;;YES, IS IT THE FIRST CHAR?
4452 022402 001403              BEQ      11$      ;;BRANCH IF YES
4453 022404 016677 000002 156526 MOV      2(SP),@SWR      ;;SAVE NEW SWR
4454 022412 062706 000006      11$: ADD      #6,SP      ;;CLEAR UP STACK
4455 022416 104401 001177      14$: TYPE     $CRLF      ;;ECHO <CR> AND <LF>
4456 022422 123727 001135 000001 CMPB     $INTAG,#1      ;;RE-ENABLE TTY KBD INTERRUPTS?
4457 022430 001003              BNE      15$      ;;BRANCH IF NOT
4458 022432 012777 000100 156504 MOV      #100,@$TKS      ;;RE-ENABLE TTY KBD INTERRUPTS
4459 022440 000002              15$: RTI              ;;RETURN
4460 022442 004737 023462      16$: JSR      PC,$TYPEC      ;;ECHO CHAR
4461 022446 021627 000060      CMP      (SP),#60      ;;CHAR < 0?
4462 022452 002420              BLT      18$      ;;BRANCH IF YES
4463 022454 021627 000067      CMP      (SP),#67      ;;CHAR > 7?
4464 022460 003015              BGT      18$      ;;BRANCH IF YES
4465 022462 042726 000060      BIC      #60,(SP)+      ;;STRIP-OFF ASCII
4466 022466 005766 000002      TST      2(SP)      ;;IS THIS THE FIRST CHAR
4467 022472 001403              BEQ      17$      ;;BRANCH IF YES
4468 022474 006316              ASL      (SP)      ;;NO, SHIFT PRESENT
4469 022476 006316              ASL      (SP)      ;;CHAR OVER TO MAKE
4470 022500 006316              ASL      (SP)      ;;ROOM FOR NEW ONE.
4471 022502 005266 000002      17$: INC      2(SP)      ;;KEEP COUNT OF CHAR
4472 022506 056616 177776      BIS      -2(SP),(SP)      ;;SET IN NEW CHAR
4473 022512 000667              BR       7$      ;;GET THE NEXT ONE
4474 022514 104401 001176      18$: TYPE     $QUES      ;;TYPE ?<CR><LF>
4475 022520 000720              BR       20$      ;;SIMULATE CONTROL-U
4476      .DSABL  LSB
4477
4478
4479      ;:*****
4480      ;*THIS ROUTINE WILL INPUT A SINGLE CHARACTER FROM THE TTY
4481      ;*CALL:
4482      ;*      RDCHR              ;;GET A CHARACTER FROM THE QUEUE
4483      ;*      RETURN HERE        ;;CHARACTER IS ON THE STACK
4484      ;*                        ;;WITH PARITY BIT STRIPPED OFF
4485      ;*
4486      ;*
4487 022522 011646 000004 000002 $RDCHR: MOV      (SP),-(SP)      ;;PUSH DOWN THE PC AND
4488 022524 016666 000004      MOV      4(SP),2(SP)      ;;THE PS
4489 022532 005066 000004      CLR      4(SP)      ;;GET READY FOR A CHARACTER
4490 022536 005046      CLR      -(SP)      ;;PUT NEW PS ON STACK
4491 022540 012746 022546      MOV      #64$,-(SP)      ;;PUT NEW PC ON STACK
4492 022544 000002      RTI              ;;POP NEW PC AND PS
4493 022546
4494 022546 005737 021644      64$: TST      $TKCNT      ;;WAIT ON A CHARACTER
4495 022552 001775      1$: BEQ      1$      ;;
4496 022554 005337 021644      DEC      $TKCNT      ;;DECREMENT THE COUNTER
4497 022560 117766 177064 000004 MOVB     @$TKQOUT,4(SP)      ;;GET ONE CHARACTER
4498 022566 005237 021650      INC      $TKQOUT      ;;UPDATE THE POINTER
4499 022572 023727 021650 021657 CMP      $TKQOUT,$$TKQEND      ;;DID IT GO OFF OF THE END?
4500 022600 001003      BNE      2$      ;;BRANCH IF NO
4501 022602 012737 021652 021650 MOV      #$TKQSR,$TKQOUT      ;;RESET THE POINTER
4502 022610 000002      2$: RTI              ;;RETURN
4503      ;:*****
```

```
4504      ;*THIS ROUTINE WILL INPUT A STRING FROM THE TTY
4505      ;*CALL:
4506      ;*      RDLIN      ;:INPUT A STRING FROM THE TTY
4507      ;*      RETURN HERE ;:ADDRESS OF FIRST CHARACTER WILL BE ON THE STACK
4508      ;*                  ;:TERMINATOR WILL BE A BYTE OF ALL 0'S
4509
4510 022612 010346 $RDLIN: MOV R3,-(SP) ;:SAVE R3
4511 022614 005046      CLR -(SP) ;:CLEAR THE RUBOUT KEY
4512 022616 012703 023046 1$: MOV #TTYIN,R3 ;:GET ADDRESS
4513 022622 022703 023072 2$: CMP #TTYIN+20.,R3 ;:BUFFER FULL?
4514 022626 101456      BLOS 4$ ;:BR IF YES
4515 022630 104410      RDCHR ;:GO READ ONE CHARACTER FROM THE TTY
4516 022632 112613      MOV (SP)+,(R3) ;:GET CHARACTER
4517 022634 122713 000177 10$: CMPB #177,(R3) ;:IS IT A RUBOUT
4518 022640 001022      BNE 5$ ;:BR IF NO
4519 022642 005716      TST (SP) ;:IS THIS THE FIRST RUBOUT?
4520 022644 001007      BNE 6$ ;:BR IF NO
4521 022646 112737 000134 023044 MOVB #'\\,9$ ;:TYPE A BACK SLASH
4522 022654 104401 023044      TYPE ,9$
4523 022660 012716 177777      MOV #-1,(SP) ;:SET THE RUBOUT KEY
4524 022664 005303      DEC R3 ;:BACKUP BY ONE
4525 022666 020327 023046 6$: CMP R3,#TTYIN ;:STACK EMPTY?
4526 022672 103434      BLO 4$ ;:BR IF YES
4527 022674 111337 023044 MOVB (R3),9$ ;:SETUP TO TYPEOUT THE DELETED CHAR.
4528 022700 104401 023044      TYPE ,9$ ;:GO TYPE
4529 022704 000746      BR 2$ ;:GO READ ANOTHER CHAR.
4530 022706 005716      5$: TST (SP) ;:RUBOUT KEY SET?
4531 022710 001406      BEQ 7$ ;:BR IF NO
4532 022712 112737 000134 023044 MOVB #'\\,9$ ;:TYPE A BACK SLASH
4533 022720 104401 023044      TYPE ,9$
4534 022724 005016      CLR (SP) ;:CLEAR THE RUBOUT KEY
4535 022726 122713 000025 7$: CMPB #25,(R3) ;:IS CHARACTER A CTRL U?
4536 022732 001003      BNE 8$ ;:BR IF NO
4537 022734 104401 023077      TYPE ,SCNTLU ;:TYPE A CONTROL 'U'
4538 022740 000726      BR 1$ ;:GO START OVER
4539 022742 122713 000022 8$: CMPB #22,(R3) ;:IS CHARACTER A '^R'?
4540 022746 001011      BNE 3$ ;:BRANCH IF NO
4541 022750 105013      CLRB (R3) ;:CLEAR THE CHARACTER
4542 022752 104401 001177      TYPE ,SCRLF ;:TYPE A 'CR' & 'LF'
4543 022756 104401 023046      TYPE ,TTYIN ;:TYPE THE INPUT STRING
4544 022762 000717      BR 2$ ;:GO PICKUP ANOTHER CHARACTER
4545 022764 104401 001176 4$: TYPE ,QUES ;:TYPE A '?'
4546 022770 000712      BR 1$ ;:CLEAR THE BUFFER AND LOOP
4547 022772 111337 023044 3$: MOVB (R3),9$ ;:ECHO THE CHARACTER
4548 022776 104401 023044      TYPE ,9$
4549 023002 122723 000015      CMPB #15,(R3)+ ;:CHECK FOR RETURN
4550 023006 001305      BNE 2$ ;:LOOP IF NOT RETURN
4551 023010 105063 177777      CLRB -1(R3) ;:CLEAR RETURN (THE 15)
4552 023014 104401 001200      TYPE ,SLF ;:TYPE A LINE FEED
4553 023020 005726      TST (SP)+ ;:CLEAN RUBOUT KEY FROM THE STACK
4554 023022 012603      MOV (SP)+,R3 ;:RESTORE R3
4555 023024 011646      MOV (SP),-(SP) ;:ADJUST THE STACK AND PUT ADDRESS OF THE
4556 023026 016666 000004 000002 MOV 4(SP),2(SP) ;:FIRST ASCII CHARACTER ON IT
4557 023034 012766 023046 000004 MOV #TTYIN,4(SP)
4558 023042 000002      RTI ;:RETURN
4559 023044 000      9$: .BYTE 0 ;:STORAGE FOR ASCII CHAR. TO TYPE
```

```
4560 023045 000
4561 023046 000024
4562 023072 041536 005015 000
4563 023077 136 006525 000012
4564 023104 043536 005015 000
4565 023111 015 051412 051127
4566 023116 036440 000040
4567 023122 020040 042516 020127
4568 023130 020075 000
4569 023134
4570
4571
4572
4573
4574
4575
4576
4577
4578
4579
4580
4581
4582
4583
4584 023134 011646
4585 023136 016666 000004 000002
4586 023144 010046
4587 023146 010146
4588 023150 010246
4589 023152 104411
4590 023154 012600
4591 023156 010037 023302
4592 023162 005046
4593 023164 005002
4594 023166 122710 000055
4595 023172 001001
4596 023174 112002
4597 023176 112001
4598 023200 001424
4599 023202 122701 000060
4600 023206 003032
4601 023210 122701 000071
4602 023214 002427
4603 023216 032716 170000
4604 023222 001024
4605 023224 006316
4606 023226 011646
4607 023230 006316
4608 023232 006316
4609 023234 062616
4610 023236 102416
4611 023240 162701 000060
4612 023244 060116
4613 023246 102412
4614 023250 000752
4615 023252 005702

        .BYTE 0                ;; TERMINATOR
$TTYIN: .BLKB 20                ;; RESERVE 20 BYTES FOR TTY INPUT
$CNTLC: .ASCIZ /^C/<15><12>      ;; CONTROL 'C'
$CNTLU: .ASCIZ /^U/<15><12>      ;; CONTROL 'U'
$CNTLG: .ASCIZ /^G/<15><12>      ;; CONTROL 'G'
$MSWR: .ASCIZ <15><12>/SWR = /
$MNEW: .ASCIZ / NEW = /

.EVEN
.SBTTL READ A DECIMAL NUMBER FROM THE TTY

*****
*THIS ROUTINE WILL READ A DECIMAL (ASCII) NUMBER FROM THE TTY AND
*CHANGE IT TO BINARY. IF TOO MANY CHARACTERS OR ANY ILLEGAL CHARACTERS
*ARE READ A '?' FOLLOWED BY A CARRIAGE RETURN-LINE FEED WILL BE TYPED.
*THE COMPLETE NUMBER MUST BE RETYPED. THE INPUT IS TERMINATED BY THE
*USER TYPING A CARRIAGE RETURN. THE RANGE OF THE INPUT NUMBER IS
*POSITIVE 32767 TO NEGATIVE 32768.
*CALL:
*      RDDEC                ;; READ A DECIMAL NUMBER
*      RETURN HERE          ;; NUMBER IS ON TOP OF THE STACK
*

$RDDEC: MOV (SP),-(SP)        ;; PROVIDE SPACE FOR
MOV 4(SP),2(SP)              ;; THE INPUT NUMBER
MOV R0,-(SP)                 ;; PUSH R0 ON STACK
MOV R1,-(SP)                 ;; PUSH R1 ON STACK
MOV R2,-(SP)                 ;; PUSH R2 ON STACK
1$: RDLIN                    ;; READ AN ASCII LINE
MOV (SP)+,R0                 ;; ADDRESS OF 1ST CHAR.
MOV R0,6$                    ;; SAVE INCASE OF BAD INPUT
CLR -(SP)                    ;; CLEAR DATA WORD
CLR R2                       ;; SIGN SET POSITIVE
CMPB #'-(R0)                 ;; SEE IF A MINUS SIGN WAS TYPED
BNE 2$                       ;; BR IF NO MINUS SIGN
MOVB (R0)+,R2                ;; SAVE FOR LATER USE
2$: MOVB (R0)+,R1             ;; PICKUP THIS CHARACTER
BEQ 3$                       ;; GET OUT IF ZERO
CMPB #'0,R1                  ;; MAKE SURE THIS CHARACTER
BGT 5$                       ;; IS A DIGIT BETWEEN 0 & 9
CMPB #'9,R1
BLT 5$
BIT #^C7777,(SP)              ;; DON'T LET NUMBER GET TO BIG
BNE 5$                       ;; BR IF NUMBER WOULD OVERFLOW
ASL (SP)                     ;; *2
MOV (SP),-(SP)               ;; SAVE FOR LATER
ASL (SP)                     ;; *4
ASL (SP)                     ;; *8
ADD (SP)+,(SP)               ;; *10
BVS 5$                       ;; OVERFLOW ISN'T ALLOWED
SUB #'0,R1                   ;; STRIP AWAY THE ASCII JUNK
ADD R1,(SP)                  ;; ADD IN THIS DIGIT
BVS 5$                       ;; OVERFLOW ISN'T ALLOWED
BR 2$                        ;; LOOP
3$: TST R2                   ;; CHECK IF NUMBER IS NEG
```

```
4616 023254 001401      BEQ      4$      ;;BR IF NO
4617 023256 005416      NEG      (SP)      ;;YES--NEGATE THE NUMBER
4618 023260 012666 000012 4$:      MOV      (SP)+,12(SP)  ;;SAVE THE RESULT
4619 023264 012602      MOV      (SP)+,R2      ;;POP STACK INTO R2
4620 023266 012601      MOV      (SP)+,R1      ;;POP STACK INTO R1
4621 023270 012600      MOV      (SP)+,R0      ;;POP STACK INTO R0
4622 023272 000002      RTI              ;;RETURN
4623
4624 023274 005726      5$:      TST      (SP)+      ;;CLEAN PARTIAL NUMBER FROM STACK
4625 023276 105010      CLRB      (R0)      ;;SET A TERMINATOR
4626 023300 104401      TYPE              ;;TYPE THE INPUT UP TO BAD CHAR.
4627 023302 000000      6$:      .WORD      0      ;;POINTER GOES HERE
4628 023304 104401 001176      TYPE      ,SQUES      ;;'?' 'CR' & 'LF'
4629 023310 000720      BR        1$      ;;TRY AGAIN
4630      .SBTTL      TYPE ROUTINE
4631
4632      ;*****
4633      ;*ROUTINE TO TYPE ASCIZ MESSAGE. MESSAGE MUST TERMINATE WITH A 0 BYTE.
4634      ;*THE ROUTINE WILL INSERT A NUMBER OF NULL CHARACTERS AFTER A LINE FEED.
4635      ;*NOTE1:      $NULL CONTAINS THE CHARACTER TO BE USED AS THE FILLER CHARACTER.
4636      ;*NOTE2:      $FILLS CONTAINS THE NUMBER OF FILLER CHARACTERS REQUIRED.
4637      ;*NOTE3:      $FILLC CONTAINS THE CHARACTER TO FILL AFTER.
4638      ;*
4639      ;*CALL:
4640      ;*1) USING A TRAP INSTRUCTION
4641      ;*      TYPE      ,MESADR      ;;MESADR IS FIRST ADDRESS OF AN ASCIZ STRING
4642      ;*OR
4643      ;*      TYPE
4644      ;*      MESADR
4645      ;*
4646
4647 023312 105737 001157      $TYPE:  TSTB      $TPFLG      ;;IS THERE A TERMINAL?
4648 023316 100002      BPL        1$      ;;BR IF YES
4649 023320 000000      HALT              ;;HALT HERE IF NO TERMINAL
4650 023322 000407      BR        3$      ;;LEAVE
4651 023324 010046      1$:      MOV      R0,-(SP)      ;;SAVE R0
4652 023326 017600 000002      MOV      @2(SP),R0      ;;GET ADDRESS OF ASCIZ STRING
4653 023332 112046      2$:      MOVB      (R0)+,-(SP)      ;;PUSH CHARACTER TO BE TYPED ONTO STACK
4654 023334 001005      BNE        4$      ;;BR IF IT ISN'T THE TERMINATOR
4655 023336 005726      TST      (SP)+      ;;IF TERMINATOR POP IT OFF THE STACK
4656 023340 012600      60$:      MOV      (SP)+,R0      ;;RESTORE R0
4657 023342 062716 000002      3$:      ADD      #2,(SP)      ;;ADJUST RETURN PC
4658 023346 000002      RTI              ;;RETURN
4659 023350 122716 000011      4$:      CMPB      #HT,(SP)      ;;BRANCH IF <HT>
4660 023354 001430      BEQ        8$
4661 023356 122716 000200      CMPB      #CRLF,(SP)      ;;BRANCH IF NOT <CRLF>
4662 023362 001006      BNE        5$
4663 023364 005726      TST      (SP)+      ;;POP <CR><LF> EQUIV
4664 023366 104401      TYPE              ;;TYPE A CR AND LF
4665 023370 001177      $CRLF
4666 023372 105037 023600      CLRB      $CHARCNT      ;;CLEAR CHARACTER COUNT
4667 023376 000755      BR        2$      ;;GET NEXT CHARACTER
4668 023400 004737 023462      5$:      JSR      PC,$TYPEC      ;;GO TYPE THIS CHARACTER
4669 023404 123726 001156      6$:      CMPB      $FILLC,(SP)+      ;;IS IT TIME FOR FILLER CHARS.?
4670 023410 001350      BNE        2$      ;;IF NO GO GET NEXT CHAR.
4671 023412 013746 001154      MOV      $NULL,-(SP)      ;;GET # OF FILLER CHARS. NEEDED
```

```
4672
4673 023416 105366 000001
4674 023422 002770
4675 023424 004737 023462
4676 023430 105337 023600
4677 023434 000770
4678
4679
4680
4681 023436 112716 000040
4682 023442 004737 023462
4683 023446 132737 000007 023600
4684 023454 001372
4685 023456 005726
4686 023460 000724
4687 023462
4688 023462 105777 155456
4689 023466 100022
4690 023470 017746 155452
4691 023474 042716 177600
4692 023500 122716 000023
4693 023504 001012
4694 023506
4695 023506 105777 155432
4696 023512 100375
4697 023514 117716 155426
4698 023520 042716 177600
4699 023524 122716 000021
4700 023530 001366
4701 023532
4702 023532 005726
4703 023534
4704 023534 105777 155410
4705 023540 100375
4706 023542 116677 000002 155402
4707 023550 122766 000015 000002
4708 023556 001003
4709 023560 105037 023600
4710 023564 000406
4711 023566 122766 000012 000002
4712 023574 001402
4713 023576 105227
4714 023600 000000
4715 023602 000207
4716
4717
4718
4719
4720
4721
4722
4723
4724
4725
4726
4727

7$:   DECB    1(SP)
      BLT     6$
      JSR     PC,$TYPEC
      DECB    $CHARCNT
      BR      7$

      ;HORIZONTAL TAB PROCESSOR

8$:   MOVB    #' (SP)
9$:   JSR     PC,$TYPEC
      BITB    #7,$CHARCNT
      BNE     9$
      TST     (SP)+
      BR      2$

$TYPEC:
      TSTB    @STKS
      BPL     10$
      MOV     @STKB,-(SP)
      BIC     #177600,(SP)
      CMPB    #$XOFF,(SP)
      BNE     101$

101$:
      TSTB    @STKS
      BPL     101$
      MOVB    @STKB,(SP)
      BIC     #177600,(SP)
      CMPB    #$XON,(SP)
      BNE     101$

102$:
      TST     (SP)+
      ;FIX STACK

10$:
      TSTB    @STPS
      BPL     10$
      MOVB    2(SP),@STPB
      CMPB    #CR,2(SP)
      BNE     1$
      CLRB    $CHARCNT
      BR      $TYPEX
1$:   CMPB    #LF,2(SP)
      BEQ     $TYPEX
      INCB    (PC)+
      $CHARCNT: .WORD 0
      $TYPEX: RTS    PC

      ;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

      ;CHAR IN KYBD BUFFER?
      ;BR IF NOT
      ;GET CHAR
      ;STRIP EXTRANEIOUS BITS
      ;WAS CHAR XOFF
      ;BR IF NOT

      ;WAIT FOR CHAR

      ;GET CHAR
      ;STRIP IT
      ;WAS IT XON?
      ;BR IF NOT

      ;WAIT UNTIL PRINTER IS READY

      ;LOAD CHAR TO BE TYPED INTO DATA REG.
      ;IS CHARACTER A CARRIAGE RETURN?
      ;BRANCH IF NO
      ;YES--CLEAR CHARACTER COUNT
      ;EXIT
      ;IS CHARACTER A LINE FEED?
      ;BRANCH IF YES
      ;COUNT THE CHARACTER
      ;CHARACTER COUNT STORAGE

.SBTTL CONVERT BINARY TO DECIMAL AND TYPE ROUTINE

;*****
;THIS ROUTINE IS USED TO CHANGE A 16-BIT BINARY NUMBER TO A 5-DIGIT
;SIGNED DECIMAL (ASCII) NUMBER AND TYPE IT. DEPENDING ON WHETHER THE
;NUMBER IS POSITIVE OR NEGATIVE A SPACE OR A MINUS SIGN WILL BE TYPED
;BEFORE THE FIRST DIGIT OF THE NUMBER. LEADING ZEROS WILL ALWAYS BE
;REPLACED WITH SPACES.
;CALL:
;      MOV     NUM,-(SP)
;      TYPDS
;      ;PUT THE BINARY NUMBER ON THE STACK
;      ;GO TO THE ROUTINE
```

```
4728
4729 023604
4730 023604 010046
4731 023606 010146
4732 023610 010246
4733 023612 010346
4734 023614 010546
4735 023616 012746 020200
4736 023622 016605 000020
4737 023626 100004
4738 023630 005405
4739 023632 112766 000055 000001
4740 023640 005000
4741 023642 012703 024020
4742 023646 112723 000040
4743 023652 005002
4744 023654 016001 024010
4745 023660 160105
4746 023662 002402
4747 023664 005202
4748 023666 000774
4749 023670 060105
4750 023672 005702
4751 023674 001002
4752 023676 105716
4753 023700 100407
4754 023702 106316
4755 023704 103003
4756 023706 116663 000001 177777
4757 023714 052702 000060
4758 023720 052702 000040
4759 023724 110223
4760 023726 005720
4761 023730 020027 000010
4762 023734 002746
4763 023736 003002
4764 023740 010502
4765 023742 000764
4766 023744 105726
4767 023746 100003
4768 023750 116663 177777 177776
4769 023756 105013
4770 023760 012605
4771 023762 012603
4772 023764 012602
4773 023766 012601
4774 023770 012600
4775 023772 104401 024020
4776 023776 016666 000002 000004
4777 024004 012616
4778 024006 000002
4779 024010 023420
4780 024012 001750
4781 024014 000144
4782 024016 000012
4783 024020 000004
```

STYPDS:

```
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 R5,-(SP) ;;PUSH R5 ON STACK
MOV #20200,-(SP) ;;SET BLANK SWITCH AND SIGN
MOV 20(SP),R5 ;;GET THE INPUT NUMBER
BPL 1$ ;;BR IF INPUT IS POS.
NEG R5 ;;MAKE THE BINARY NUMBER POS.
MOVB #'-,1(SP) ;;MAKE THE ASCII NUMBER NEG.
1$: CLR R0 ;;ZERO THE CONSTANTS INDEX
MOV #SDBLK,R3 ;;SETUP THE OUTPUT POINTER
MOVB #' ,(R3)+ ;;SET THE FIRST CHARACTER TO A BLANK
2$: CLR R2 ;;CLEAR THE BCD NUMBER
MOV $DTBL(R0),R1 ;;GET THE CONSTANT
3$: SUB R1,R5 ;;FORM THIS BCD DIGIT
BLT 4$ ;;BR IF DONE
INC R2 ;;INCREASE THE BCD DIGIT BY 1
BR 3$
4$: ADD R1,R5 ;;ADD BACK THE CONSTANT
TST R2 ;;CHECK IF BCD DIGIT=0
BNE 5$ ;;FALL THROUGH IF 0
TSTB (SP) ;;STILL DOING LEADING 0'S?
BMI 7$ ;;BR IF YES
5$: ASLB (SP) ;;MSD?
BCC 6$ ;;BR IF NO
MOVB 1(SP),-1(R3) ;;YES--SET THE SIGN
6$: BIS #'0,R2 ;;MAKE THE BCD DIGIT ASCII
7$: BIS #' ,R2 ;;MAKE IT A SPACE IF NOT ALREADY A DIGIT
MOVB R2,(R3)+ ;;PUT THIS CHARACTER IN THE OUTPUT BUFFER
TST (R0)+ ;;JUST INCREMENTING
CMP R0,#10 ;;CHECK THE TABLE INDEX
BLT 2$ ;;GO DO THE NEXT DIGIT
BGT 8$ ;;GO TO EXIT
MOV R5,R2 ;;GET THE LSD
BR 6$ ;;GO CHANGE TO ASCII
8$: TSTB (SP)+ ;;WAS THE LSD THE FIRST NON-ZERO?
BPL 9$ ;;BR IF NO
MOVB -1(SP),-2(R3) ;;YES--SET THE SIGN FOR TYPING
9$: CLRB (R3) ;;SET THE TERMINATOR
MOV (SP)+,R5 ;;POP STACK INTO R5
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
TYPE $SDBLK ;;NOW TYPE THE NUMBER
MOV 2(SP),4(SP) ;;ADJUST THE STACK
MOV (SP)+,(SP)
RTI ;;RETURN TO USER
```

\$DTBL: 10000.
1000.
100.
10.
1.

\$SDBLK: .BLKW 4

4784
4785
4786
4787
4788
4789
4790
4791
4792
4793
4794
4795
4796
4797
4798
4799
4800
4801
4802
4803
4804
4805
4806
4807
4808
4809
4810
4811
4812
4813
4814
4815
4816
4817
4818
4819
4820
4821
4822
4823
4824
4825
4826
4827
4828
4829
4830
4831
4832
4833
4834
4835
4836
4837
4838
4839

024030 017646 000000
024034 116637 000001
024042 112637 024255
024046 062716 000002
024052 000406
024054 112737 000001
024062 112737 000006
024070 112737 000005
024076 010346
024100 010446
024102 010546
024104 113704 024255
024110 005404
024112 062704 000006
024116 110437 024254
024122 113704 024253
024126 016605 000012
024132 005003
024134 006105
024136 000404
024140 006105
024142 006105
024144 006105
024146 010503
024150 006103
024152 105337 024254
024156 100016
024160 042703 177770
024164 001002
024166 005704
024170 001403

```
.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
*$TYPOS: MOV      @ (SP),-(SP)  ;;PICKUP THE MODE
        MOVB     1(SP), $OFILL  ;;LOAD ZERO FILL SWITCH
        MOVB     (SP)+, $OMODE+1 ;;NUMBER OF DIGITS TO TYPE
        ADD      #2, (SP)      ;;ADJUST RETURN ADDRESS
        BR       $TYPON
*$TYPOC: MOVB     #1, $OFILL    ;;SET THE ZERO FILL SWITCH
        MOVB     #6, $OMODE+1  ;;SET FOR SIX(6) DIGITS
*$TYPON: MOVB     #5, $OCNT     ;;SET THE ITERATION COUNT
        MOV      R3,-(SP)      ;;SAVE R3
        MOV      R4,-(SP)      ;;SAVE R4
        MOV      R5,-(SP)      ;;SAVE R5
        MOVB     $OMODE+1, R4  ;;GET THE NUMBER OF DIGITS TO TYPE
        NEG      R4
        ADD      #6, R4        ;;SUBTRACT IT FOR MAX. ALLOWED
        MOVB     R4, $OMODE    ;;SAVE IT FOR USE
        MOVB     $OFILL, R4    ;;GET THE ZERO FILL SWITCH
        MOV      12(SP), R5    ;;PICKUP THE INPUT NUMBER
        CLR      R3           ;;CLEAR THE OUTPUT WORD
1$:      ROL      R5           ;;ROTATE MSB INTO 'C'
        BR       3$          ;;GO DO MSB
2$:      ROL      R5           ;;FORM THIS DIGIT
        ROL      R5
        MOV      R5, R3
3$:      ROL      R3           ;;GET LSB OF THIS DIGIT
        DECB     $OMODE        ;;TYPE THIS DIGIT?
        BPL      7$           ;;BR IF NO
        BIC      #177770, R3  ;;GET RID OF JUNK
        BNE     4$           ;;TEST FOR 0
        TST      R4           ;;SUPPRESS THIS 0?
        BEQ      5$           ;;BR IF YES
        7$:      BEQ      5$
        4$:      BEQ      5$
        5$:      BEQ      5$
```

```
4840 024172 005204          4$: INC R4          ;;DON'T SUPPRESS ANYMORE 0'S
4841 024174 052703 000060    BIS #'0,R3        ;;MAKE THIS DIGIT ASCII
4842 024200 052703 000040    5$: BIS #' ,R3      ;;MAKE ASCII IF NOT ALREADY
4843 024204 110337 024250    MOVB R3,8$         ;;SAVE FOR TYPING
4844 024210 104401 024250    TYPE 8$           ;;GO TYPE THIS DIGIT
4845 024214 105337 024252    7$: DECB $OCNT      ;;COUNT BY 1
4846 024220 003347          BGT 2$             ;;BR IF MORE TO DO
4847 024222 002402          BLT 6$             ;;BR IF DONE
4848 024224 005204          INC R4             ;;INSURE LAST DIGIT ISN'T A BLANK
4849 024226 000744          BR 2$              ;;GO DO THE LAST DIGIT
4850 024230 012605          6$: MOV (SP)+,R5     ;;RESTORE R5
4851 024232 012604          MOV (SP)+,R4       ;;RESTORE R4
4852 024234 012603          MOV (SP)+,R3       ;;RESTORE R3
4853 024236 016666 000002 000004 MOV 2(SP),4(SP) ;;SET THE STACK FOR RETURNING
4854 024244 012616          MOV (SP)+,(SP)
4855 024246 000002          RTI                 ;;RETURN
4856 024250 000           8$: .BYTE 0          ;;STORAGE FOR ASCII DIGIT
4857 024251 000           .BYTE 0             ;;TERMINATOR FOR TYPE ROUTINE
4858 024252 000           $OCNT: .BYTE 0       ;;OCTAL DIGIT COUNTER
4859 024253 000           $OFILL: .BYTE 0      ;;ZERO FILL SWITCH
4860 024254 000000        $OMODE: .WORD 0      ;;NUMBER OF DIGITS TO TYPE
4861                                     .SBTTL TRAP DECODER
4862
4863                                     ;:*****
4864                                     ;*THIS ROUTINE WILL PICKUP THE LOWER BYTE OF THE 'TRAP' INSTRUCTION
4865                                     ;*AND USE IT TO INDEX THROUGH THE TRAP TABLE FOR THE STARTING ADDRESS
4866                                     ;*OF THE DESIRED ROUTINE. THEN USING THE ADDRESS OBTAINED IT WILL
4867                                     ;*GO TO THAT ROUTINE.
4868
4869 024256 010046          $TRAP: MOV R0,-(SP)    ;;SAVE R0
4870 024260 016600 000002    MOV 2(SP),R0      ;;GET TRAP ADDRESS
4871 024264 005740          TST -(R0)           ;;BACKUP BY 2
4872 024266 111000          MOVB (R0),R0        ;;GET RIGHT BYTE OF TRAP
4873 024270 022700 000030    CMP # $TERM,R0    ;;CHECK FOR OUT OF BOUNDS
4874 024274 003002          BGT .+6             ;;BR IF OK
4875 024276 000000          HALT                ;;OUT OF BOUNDS
4876 024300 000776          BR .-2              ;;HANGUP
4877 024302 006300          ASL R0               ;;POSITION FOR INDEXING
4878 024304 016000 024324    MOV $TRPAD(R0),R0 ;;INDEX TO TABLE
4879 024310 000200          RTS R0              ;;GO TO ROUTINE
4880
4881                                     ;:THIS IS USE TO HANDLE THE 'GETPRI' MACRO
4882
4883 024312 011646          $TRAP2: MOV (SP),-(SP) ;;MOVE THE PC DOWN
4884 024314 016666 000004 000002 MOV 4(SP),2(SP) ;;MOVE THE PSW DOWN
4885 024322 000002          RTI                 ;;RESTORE THE PSW
4886
4887                                     .SBTTL TRAP TABLE
4888
4889                                     ;*THIS TABLE CONTAINS THE STARTING ADDRESSES OF THE ROUTINES CALLED
4890                                     ;*BY THE 'TRAP' INSTRUCTION.
4891
4892                                     : ROUTINE
4893                                     : -----
4894                                     $TRPAD: .WORD $TRAP2
4895 024324 024312
```

4896	024326	023312	\$TYPE	::CALL=TYPE	TRAP+1(104401)	TTY TYPEOUT ROUTINE
4897	024330	024054	\$TYPOC	::CALL=TYPOC	TRAP+2(104402)	TYPE OCTAL NUMBER (WITH LEADING ZEROS)
4898	024332	024030	\$TYPOS	::CALL=TYPOS	TRAP+3(104403)	TYPE OCTAL NUMBER (NO LEADING ZEROS)
4899	024334	024070	\$TYPON	::CALL=TYPON	TRAP+4(104404)	TYPE OCTAL NUMBER (AS PER LAST CALL)
4900	024336	023604	\$TYPDS	::CALL=TYPDS	TRAP+5(104405)	TYPE DECIMAL NUMBER (WITH SIGN)
4901						
4902	024340	022250	\$GTSWR	::CALL=GTSWR	TRAP+6(104406)	GET SOFT-SWR SETTING
4903						
4904	024342	022160	\$CKSWR	::CALL=CKSWR	TRAP+7(104407)	TEST FOR CHANGE IN SOFT-SWR
4905	024344	022522	\$RDCHR	::CALL=RDCHR	TRAP+10(104410)	TTY TYPEIN CHARACTER ROUTINE
4906	024346	022612	\$RDLIN	::CALL=RDLIN	TRAP+11(104411)	TTY TYPEIN STRING ROUTINE
4907	024350	021504	\$RDOCT	::CALL=RDOCT	TRAP+12(104412)	READ AN OCTAL NUMBER FROM TTY
4908	024352	023134	\$RDDEC	::CALL=RDDEC	TRAP+13(104413)	READ A DECIMAL NUMBER FROM TTY
4909		000030				
4910			STERM=-\$TRPAD			
4911						
4912						
4913						
4914						
4915						
4916						
4917						
4918						
4919						
4920						
4921						
4922						
4923						
4924						
4925						
4926						
4927						
4928						
4929						
4930						
4931						
4932						
4933						
4934						
4935	024354	000000	KMCNT:	0		# OF KMC11'S ENABLED
4936	024356	000000	CONFIG:	0		
4937	024360	000000	CURTAB:	0		CURRENT BLOCK OPEN
4938	024362	012701	SETTAB:	MOV	#KMCTAB,R1	POINT TO PARAMETER TABLE-BLOCK 1
4939	024366	005021	1\$:	CLR	(R1)+	CLEAN THE MESS
4940	024370	020127		CMP	R1,#KMCTBE	DONE?
4941	024374	001374		BNE	1\$	BR=N
4942	024376	012701		MOV	#KMCTAB,R1	
4943	024402	010137		MOV	R1,CURTAB	
4944	024406	000137		JMP	PT	
4945						
4946	024412		NOTR:			
4947	024412	104401		TYPE	,64\$	::TYPE ASCII STRING
4948	024416	000402		BR	MODTAB	::GET OVER THE ASCII
4949				::64\$:	.ASCIIZ	/???
4950	024424		MODTAB:			
4951	024424	005737	MODTAB:	TST	CONFIG	:FIRST TIME?

```
4952 024430 001002      BNE      1$      ;BR=NO
4953 024432 000137 024362      JMP      SETTAB
4954 024436      1$:      TYPE      ,65$      ;;TYPE ASCII STRING
4955 024436 104401 024444      BR      64$      ;;GET OVER THE ASCII
4956 024442 000402      ;;65$: .ASCIIZ <15><12> /*/
4957      64$:
4958 024450      PT:      BR      PT+2      ;GET FIRST SETUP
4959 024450 000401      BR      5$
4960 024452 000570      RDCHR      (SP),CHRADR      ;ECHO
4961 024454 104410      MOV      ,CHRADR
4962 024456 011637 002320      TYPE      ,CHRADR
4963 024462 104401 002320      CMP      #115,(SP)+      ;IS IT A M?
4964 024466 022726 000115      BEQ      25$      ;BR=YES
4965 024472 001436      CMP      #101,-(SP)      ;IS IT AN A?
4966 024474 022746 000101      BNE      111$
4967 024500 001002      JMP      12$
4968 024502 000137 026174      111$: CMPB      #105,(SP)+      ;IS IT AN E?
4969 024506 122726 000105      BEQ      113$      ;B=Y GO!
4970 024512 001407      CMP      #'S,-(SP)      ;AN S?
4971 024514 022746 000123      BEQ      112$      ;B=YES
4972 024520 001402      TST      (SP)+      ;NO TRY AGAIN
4973 024522 005726      BR      NOTR
4974 024524 000732      112$: TST      (SP)+
4975 024526 005726      BR      1$      ;NEXT COMMAND
4976 024530 000404      ;FIND # OF KMC11'S ACTIVE AND STORE IN 'KMCATV'
4977      113$: CLR      KMCNT
4978 024532 005037 024354      MOV      #KMCTAB,R1
4979 024536 012701 026244      1$: TST      2(R1)
4980 024542 005761 000002      BPL      2$      ;BR=NO
4981 024546 100002      INC      KMCNT      ;YES
4982 024550 005237 024354      2$: ADD      #12,R1
4983 024554 062701 000012      CMP      R1,#KMCTBE      ;DONE?
4984 024560 020127 026504      BNE      1$      ;B=N
4985 024564 001366      RTS      PC      ;YES
4986 024566 000207
4987
4988      ;A 'M' HAS BEEN TYPED READ THE DECIMAL # FOLLOWING
4989      25$: RDDEC
4990 024570      MOV      (SP)+,R0      ;IS # +?
4991 024572 012600      MOV      R0,CUNIT      ;PRINT LINE UNIT #
4992 024574 010037 002456      MOV      R0,R1
4993 024600 010001      BLE      NOTR
4994 024602 003703      CMPB      R0,#16.      ;IS #<16.?
4995 024604 120027 000020      BGT      NOTR      ;BR=NO
4996 024610 003300      CLC
4997 024612 000241      ROL      R0      ;FORM OFFSET INTO 'KMCTAB'
4998 024614 006100      MOV      R0,R2      ; R0 = 2 X LINE NUMBER
4999 024616 010002      ROL      R2
5000 024620 006102      ROL      R2      ; R2 = 8 X LINE NUMBER
5001 024622 006102      ADD      R2,R0      ; OFFSET = 12(OCTAL) X LINE NUMBER
5002 024624 060200
5003 024626 062700 026232      ADD      #KMCTAB-12,R0
5004 024632 010037 024360      MOV      R0,CURTAB
5005 024636 104401 024644      TYPE      ,65$      ;;TYPE ASCII STRING
5006 024642 000443      BR      64$      ;;GET OVER THE ASCII
5007      ;;65$: .ASCIIZ <15><12>!(DEV.ADR.) (ENABLE) (CRC32-MODE) (EX/IN-VEC) (LINES-PRI) UN
```

```
5008 024752 64$:
5009
5010 ;ROUTINE TO DISPLAY CONTENTS OF CURRENT BLOCK
5011 024752 3$:
5012 024752 104401 024760 TYPE 67$ ::TYPE ASCIZ STRING
5013 024756 000402 BR 66$ ::GET OVER THE ASCIZ
5014 ::67$: .ASCIZ <15><12>/ /
5015 024764 66$:
5016 024764 012703 000005 MOV #5,R3
5017 024770 012046 MOV (R0)+,-(SP)
5018 024772 104403 TYPDS
5019 024774 006 .BYTE 6
5020 024775 001 .BYTE 1 ;NO SUPPRESS
5021 024776 104401 025004 TYPE 69$ ::TYPE ASCIZ STRING
5022 025002 000404 BR 68$ ::GET OVER THE ASCIZ
5023 ::69$: .ASCIZ / /
5024 025014 68$:
5025 025014 005303 DEC R3
5026 025016 001401 BEQ 45$
5027 025020 000763 BR 4$
5028 025022 013746 002456 45$: MOV CUNIT,-(SP)
5029 025026 104405 TYPDS
5030 025030 013701 024360 MOV CURTAB,R1
5031
5032 025034 5$: ;ROUTINE TO MODIFY PARAMETER BLOCK
5033
5034 025034 104401 025042 TYPE 71$ ::TYPE ASCIZ STRING
5035 025040 000407 BR 70$ ::GET OVER THE ASCIZ
5036 ::71$: .ASCIZ <15><12>/DEV.ADR.= /
5037 025060 70$:
5038 025060 004737 026070 JSR PC,8$
5039 025064 104401 025072 TYPE 73$ ::TYPE ASCIZ STRING
5040 025070 000413 BR 72$ ::GET OVER THE ASCIZ
5041 ::73$: .ASCIZ /(1)ENABLE(0)DISABLE /
5042 025120 72$:
5043 025120 004737 026104 JSR PC,10$
5044 025124 005201 INC R1 ;BYPASS UNUSED BYTE
5045 025126 104401 025134 TYPE 75$ ::TYPE ASCIZ STRING
5046 025132 000411 BR 74$ ::GET OVER THE ASCIZ
5047 ::75$: .ASCIZ <15><12>/(1)BOP (0)CCP /
5048 025156 74$:
5049 025156 004737 026104 JSR PC,10$
5050 025162 105741 TSTB -(R1) ;CHECK FOR CCP MODE
5051 025164 100023 BPL 116$ ;IF YES, SKIP NEXT ENTRY
5052 025166 005201 INC R1 ;RESTORE TABLE POINTER
5053 025170 104401 025176 TYPE 77$ ::TYPE ASCIZ STRING
5054 025174 000414 BR 76$ ::GET OVER THE ASCIZ
5055 ::77$: .ASCIZ <15><12>/(1)CRC-32 (0)NO CRC /
5056 025226 76$:
5057 025226 004737 026104 JSR PC,10$
5058 025232 000402 BR 117$ ;CONT. TO NEXT ENTRY
5059 025234 005201 116$: INL R1 ;RESTORE TABLE POINTER
5060 025236 105021 CLRB (R1)+ ;IF CCP, SKIP CRC ENTRY; BUT UPDATE POINTER
5061
5062 025240 117$:
5063 025240 104401 025246 TYPE 79$ ::TYPE ASCIZ STRING
```

```
5064 025244 000410      BR      78$      ;;GET OVER THE ASCIZ
5065      ;;79$: .ASCIZ <15><12>/DEV.VECTOR= /
5066 025266      78$:      JSR      PC,7$
5067 025266 004737 026022      TYPE      81$      ;;TYPE ASCIZ STRING
5068 025272      118$:      BR      80$      ;;GET OVER THE ASCIZ
5069 025272 104401 025300      ;;81$: .ASCIZ /(1)EXT.(0)INT. /
5070 025276 000410      80$:      JSR      PC,10$
5071      103$:      TSTB      -(R1)      ;IF INTERNAL OVERRIDE CRC32
5072 025320      BMI      104$      ;B=EXTERNAL
5073 025320 004737 026104      TSTB      -3(R1)      ;BOP MODE??
5074 025324 105741      BEQ      104$      ;B=NO
5075 025326 100432      TSTB      -2(R1)      ;CRC-32 ENABLED?
5076 025330 105761 177775      BEQ      104$      ;B=NO
5077 025334 001427      CLRB      -2(R1)      ;OVERRIDE CRC32
5078 025336 105761 177776      TYPE      83$      ;;TYPE ASCIZ STRING
5079 025342 001424      BR      82$      ;;GET OVER THE ASCIZ
5080 025344 105061 177776      ;;83$: .ASCIZ / (WARNING) NO CRC-32 TESTING/
5081 025350 104401 025356      82$:      TSTB      (R1)+      ;RESTORE R1
5082 025354 000417      104$:      TYPE      85$      ;;TYPE ASCIZ STRING
5083      BR      84$      ;;GET OVER THE ASCIZ
5084 025414      ;;85$: .ASCIZ <15><12>/INT.PRI.(4-7)= /
5085 025414 105721      84$:      RDOCT
5086 025416 104401 025424      TSTB      1(SP)      ;HIGH BYTE=0?
5087 025422 000411      BNE      137$      ;B=NO
5088      MOV      (SP)+,R0
5089 025446      BEQ      136$      ;B=NO CHANGE
5090 025446 104412      BIC      #177760,R0      ;SAVE LOW BYTE
5091 025450 105766 000001      CMPB      #10,R0      ;LEGAL #?
5092 025454 001053      BLE      138$      ;B=N
5093 025456 012600      CMPB      #3,R0
5094 025460 001411      BGE      138$      ;B=NOT LEGAL
5095 025462 042700 177760      MOV      R0,(R1)      ;STORE THE PRIORITY
5096 025466 122700 000010      136$:      TSTB      (R1)+
5097 025472 003445      TYPE      87$      ;;TYPE ASCIZ STRING
5098 025474 122700 000003      BR      86$      ;;GET OVER THE ASCIZ
5099 025500 002042      ;;87$: .ASCIZ /LINES(0-7)(I.E.,#, #, #) OR (A)ALL(N)NONE /
5100 025502 110011      86$:      JSR      PC,6$
5101 025504 105721      MOV      #-1,CONFIG
5102 025506 104401 025514      JMP      MODIAB
5103 025512 000425      137$:      TST      (SP)+      ;POP STACK
5104      138$:      TYPE      $CRLF
5105 025566      TYPE      $QUES
5106 025566 004737 025620      BR      104$
5107 025572 012737 177777 024356      6$:      RDLIN
5108 025600 000137 024424      MOV      (SP)+,R0      ;GET TTY BUFFER
5109 025604 005726      TSTB      (R0)      ;ANY CHANGE IN STATUS?
5110 025606 104401 001177      BEQ      110$      ;B=NO
5111 025612 104401 001176      MOV      #-1,(R1)      ;SET TO ALL LINES ENABLED
5112 025616 000676      CMPB      #'A,(R0)      ;TEST FOR ALL LINES
5113 025620 104411      BEQ      110$      ;B=ENABLE ALL LINES
5114 025622 012600
5115 025624 105710
5116 025626 001473
5117 025630 112711 177777
5118 025634 122710 000101
5119 025640 001466
```

5120	025642	105011		CLRB	(R1)	;SET TO NO LINES
5121	025644	122710	000116	CMPB	#'N,(R0)	;IT IT N FOR NO LINES
5122	025650	001462		BEQ	110\$	;B=YES NO LINES ENABLED
5123	025652	112037	001160	107\$:	MOVB	(R0)+,\$TMP0
5124	025656	001457		BEQ	110\$	;DONE?
5125	025660	122737	000054 001160	CMPB	#54,\$TMP0	;B=YES
5126	025666	001771		BEQ	107\$	;A COMMA?
5127	025670	123727	001160 000060	CMPB	\$TMP0,#60	;B=YES (ONLY A SEPARATOR)
5128	025676	002423		BLT	130\$	;ONLY 0-7 ARE VALID
5129	025700	123727	001160 000067	CMPB	\$TMP0,#67	;B=NOT VALID
5130	025706	003017		BGT	130\$	;LESS THAN 7?
5131	025710	142737	000360 001160	SICB	#360,\$TMP0	;B=NOT VALID
5132	025716	105237	001160	INCB	\$TMP0	;SAVE ONLY OCTAL DIGIT
5133	025722	012702	000200	MOV	#200,R2	;INIT MASKING BIT
5134	025726	105337	001160	108\$:	DECB	\$TMP0
5135	025732	001403		BEQ	109\$	;FIND BIT POSITION OF ENABLED LINE
5136	025734	006202		ASR	R2	;B=FOUND IT
5137	025736	001373		BNE	108\$	;NEXT LINE BIT
5138	025740	000402		BR	130\$	;INVALID CHAR.
5139	025742	150211		109\$:	BISB	R2,(R1)
5140	025744	000742		BR	107\$	;SET BIT FOUND (THIS LINE ENABLED)
5141	025746			130\$:		;TRY AGAIN
5142	025746	104401	025754	TYPE	89\$	::TYPE ASCII STRING
5143	025752	000417		BR	88\$	::GET OVER THE ASCII
5144				88\$:	.ASCIIZ	/<15><12>SYNTAX ERROR<15><12>/
5145	026012			88\$:		
5146	026012	000137	025620	JMP	6\$	;TRY AGAIN
5147	026016	005201		110\$:	INC	R1
5148	026020	000207		RTS	PC	;MOV TO NEXT BYTE
5149	026022	104412		7\$:	RDOCT	; & RETURN TO MAINLINE
5150	026024	012600		MOV	(SP)+,R0	
5151	026026	042700	176000	BIC	#176000,R0	;SAVE ONLY VECTOR
5152	026032	001405		BEQ	131\$	;B=NO CHANGE
5153	026034	011103		MOV	(R1),R3	;SAVE ONLY E/I BIT
5154	026036	042703	077777	BIC	#077777,R3	;STRIP OLD VECTOR
5155	026042	050300		BIS	R3,R0	;RETURN E/I BIT
5156	026044	010011		MOV	R0,(R1)	;STORE NEW DATA
5157	026046	062701	000001	131\$:	ADD	#1,R1
5158	026052	000207		RTS	PC	
5159	026054	005726		132\$:	TST	(SP)+
5160	026056	104401	001176	TYPE	,\$QUES	;POP STACK
5161	026062	104401	001177	TYPE	,\$CRLF	;HIGH BYTE NOT =0
5162	026066	000755		BR	7\$	
5163	026070	104412		8\$:	RDOCT	
5164	026072	012600		MOV	(SP)+,R0	;CHANGE STATUS?
5165	026074	001401		BEQ	9\$	;B=NO
5166	026076	010011		MOV	R0,(R1)	;LOAD DATA
5167	026100	005721		9\$:	TST	(R1)+
5168	026102	000207		RTS	PC	
5169				91\$:	RDOCT	
5170				:	MOV	(SP)+,R0
5171				:	BEQ	92\$
5172				:	MOVB	R0,(R1)
5173				92\$:	TSTB	(R1)+
5174				:	RTS	PC
5175	026104	104410		10\$:	RDCHR	

5176	026106	012600	
5177	026110	122700	000015
5178	026114	001425	
5179	026116	110037	002320
5180	026122	104401	002320
5181	026126	122700	000060
5182	026132	001413	
5183	026134	122700	000061
5184	026140	001405	
5185	026142	104401	026150
5186	026146	000401	
5187			
5188	026152		
5189	026152	000754	
5190	026154	152721	000200
5191	026160	000404	
5192	026162	142721	000200
5193	026166	000401	
5194	026170	005201	
5195	026172	000207	
5196			
5197			
5198			
5199	026174	005726	
5200	026176	013700	024360
5201	026202		
5202	026202	010001	
5203	026204	062701	000012
5204	026210	020127	026504
5205	026214	002010	
5206	026216	012021	
5207	026220	012021	
5208	026222	012021	
5209	026224	012021	
5210	026226	012021	
5211	026230	010037	024360
5212	026234	000762	
5213			
5214	026236	000137	024424
5215	026242	000000	
5216			
5217			
5218	026244		
5219			
5220	026244	000000	
5221	026246	000000	
5222	026250	000000	
5223	026252	000000	
5224	026254	000000	
5225			
5226	026256	000000	
5227	026260	000000	
5228	026262	000000	
5229	026264	000000	
5230	026266	000000	
5231			

```

MOV      (SP)+,R0
CMPB     #15,R0          ;C.R.?
BEQ      114$
MOVB     R0,CHRADR
TYPE     ,CHRADR
CMPB     #60,R0          ;0?
BEQ      11$
CMPB     #61,R0          ;1?
BEQ      105$
TYPE     ,91$            ;;TYPE ASCIZ STRING
BR       90$             ;;GET OVER THE ASCIZ
;;91$:
90$:
BR       10$
105$:    BISB     #200,(R1)+
BR       115$
11$:     BICB     #200,(R1)+
BR       115$
114$:    INC      R1
115$:    RTS      PC
;AN 'A' WAS DETECTED SET REMAINING BLOCKS WITH SUCCESSIVE
;PARAMETERS (BASED) AS LAST BLOCK OPEN 'CURTAB'.

12$:     TST      (SP)+
MOV      CURTAB,R0        ;GET CURRENT BLOCK
125$:
MOV      R0,R1
ADD      #12,R1
13$:     CMP      R1,#KMCTBE ;AT END?
BGE      14$             ;B=Y
MOV      (R0)+,(R1)+      ;GET NEW D.A.
MOV      (R0)+,(R1)+      ;LOAD ENABLE
MOV      (R0)+,(R1)+      ;LOAD MODE, CRC32 STATUS
MOV      (R0)+,(R1)+      ;LOAD VECTOR, E/I STATUS
MOV      (R0)+,(R1)+      ;SAME PRIORITY AND LINE BYTE
MOV      R0,CURTAB
BR       125$

14$:     JMP      MODTAB
15$:     0
;KMC11 PARAMETER BLOCKS

KMCTAB:
;PARAMETER BLOCK FOR KMC#01
KMCR01:  0 ;CONTROL + STATUS REGISTER FOR KMC11 #01
KMCE01:  0 ;LO BYTE=ENABLE (200=YES,0=NO);HI BYTE UNUSED
KMCV01:  0 ;LO BYTE=MODE (200=BOP,0=CCP) HI BYTE=CRC32(200=YES,0=NO)
KMCP01:  0 ;LO BYTE=INT.VECTOR ADDR.,HI=E/I(200=EXT 0=INT)
KMCS01:  0 ;LO BYTE=INT.PRI. HI BYTE=LINE BYTE
;PARAMETER BLOCK FOR KMC#02
KMCR02:  0 ;CONTROL + STATUS REGISTER FOR KMC11 #02
KMCE02:  0 ;LO BYTE=ENABLE (200=YES,0=NO);HI BYTE UNUSED
KMCV02:  0 ;LO BYTE=MODE (200=BOP,0=CCP) HI BYTE=CRC32(200=YES,0=NO)
KMCP02:  0 ;LO BYTE=INT.VECTOR ADDR.,HI=E/I(200=EXT 0=INT)
KMCS02:  0 ;LO BYTE=INT.PRI. HI BYTE=LINE BYTE
;PARAMETER BLOCK FOR KMC#03

```

5232	026270	000000	KMCR03: 0	:CONTROL + STATUS REGISTER FOR KMC11 #03
5233	026272	000000	KMCE03: 0	:LO BYTE=ENABLE (200=YES,0=NO);HI BYTE UNUSED
5234	026274	000000	KMCV03: 0	:LO BYTE=MODE (200=BOP,0=CCP) HI BYTE=CRC32(200=YES,0=NO)
5235	026276	000000	KMCP03: 0	:LO BYTE=INT.VECTOR ADDR.,HI=E/I(200=EXT 0=INT)
5236	026300	000000	KMCS03: 0	:LO BYTE=INT.PRI. HI BYTE=LINE BYTE
5237			:PARAMETER BLOCK FOR KMC#04	
5238	026302	000000	KMCR04: 0	:CONTROL + STATUS REGISTER FOR KMC11 #04
5239	026304	000000	KMCE04: 0	:LO BYTE=ENABLE (200=YES,0=NO);HI BYTE UNUSED
5240	026306	000000	KMCV04: 0	:LO BYTE=MODE (200=BOP,0=CCP) HI BYTE=CRC32(200=YES,0=NO)
5241	026310	000000	KMCP04: 0	:LO BYTE=INT.VECTOR ADDR.,HI=E/I(200=EXT 0=INT)
5242	026312	000000	KMCS04: 0	:LO BYTE=INT.PRI. HI BYTE=LINE BYTE
5243			:PARAMETER BLOCK FOR KMC#05	
5244	026314	000000	KMCR05: 0	:CONTROL + STATUS REGISTER FOR KMC11 #05
5245	026316	000000	KMCE05: 0	:LO BYTE=ENABLE (200=YES,0=NO);HI BYTE UNUSED
5246	026320	000000	KMCV05: 0	:LO BYTE=MODE (200=BOP,0=CCP) HI BYTE=CRC32(200=YES,0=NO)
5247	026322	000000	KMCP05: 0	:LO BYTE=INT.VECTOR ADDR.,HI=E/I(200=EXT 0=INT)
5248	026324	000000	KMCS05: 0	:LO BYTE=INT.PRI. HI BYTE=LINE BYTE
5249			:PARAMETER BLOCK FOR KMC#06	
5250	026326	000000	KMCR06: 0	:CONTROL + STATUS REGISTER FOR KMC11 #06
5251	026330	000000	KMCE06: 0	:LO BYTE=ENABLE (200=YES,0=NO);HI BYTE UNUSED
5252	026332	000000	KMCV06: 0	:LO BYTE=MODE (200=BOP,0=CCP) HI BYTE=CRC32(200=YES,0=NO)
5253	026334	000000	KMCP06: 0	:LO BYTE=INT.VECTOR ADDR.,HI=E/I(200=EXT 0=INT)
5254	026336	000000	KMCS06: 0	:LO BYTE=INT.PRI. HI BYTE=LINE BYTE
5255			:PARAMETER BLOCK FOR KMC#07	
5256	026340	000000	KMCR07: 0	:CONTROL + STATUS REGISTER FOR KMC11 #07
5257	026342	000000	KMCE07: 0	:LO BYTE=ENABLE (200=YES,0=NO);HI BYTE UNUSED
5258	026344	000000	KMCV07: 0	:LO BYTE=MODE (200=BOP,0=CCP) HI BYTE=CRC32(200=YES,0=NO)
5259	026346	000000	KMCP07: 0	:LO BYTE=INT.VECTOR ADDR.,HI=E/I(200=EXT 0=INT)
5260	026350	000000	KMCS07: 0	:LO BYTE=INT.PRI. HI BYTE=LINE BYTE
5261			:PARAMETER BLOCK FOR KMC#08	
5262	026352	000000	KMCR08: 0	:CONTROL + STATUS REGISTER FOR KMC11 #08
5263	026354	000000	KMCE08: 0	:LO BYTE=ENABLE (200=YES,0=NO);HI BYTE UNUSED
5264	026356	000000	KMCV08: 0	:LO BYTE=MODE (200=BOP,0=CCP) HI BYTE=CRC32(200=YES,0=NO)
5265	026360	000000	KMCP08: 0	:LO BYTE=INT.VECTOR ADDR.,HI=E/I(200=EXT 0=INT)
5266	026362	000000	KMCS08: 0	:LO BYTE=INT.PRI. HI BYTE=LINE BYTE
5267			:PARAMETER BLOCK FOR KMC#09	
5268	026364	000000	KMCR09: 0	:CONTROL + STATUS REGISTER FOR KMC11 #09
5269	026366	000000	KMCE09: 0	:LO BYTE=ENABLE (200=YES,0=NO);HI BYTE UNUSED
5270	026370	000000	KMCV09: 0	:LO BYTE=MODE (200=BOP,0=CCP) HI BYTE=CRC32(200=YES,0=NO)
5271	026372	000000	KMCP09: 0	:LO BYTE=INT.VECTOR ADDR.,HI=E/I(200=EXT 0=INT)
5272	026374	000000	KMCS09: 0	:LO BYTE=INT.PRI. HI BYTE=LINE BYTE
5273			:PARAMETER BLOCK FOR KMC#10	
5274	026376	000000	KMCR10: 0	:CONTROL + STATUS REGISTER FOR KMC11 #10
5275	026400	000000	KMCE10: 0	:LO BYTE=ENABLE (200=YES,0=NO);HI BYTE UNUSED
5276	026402	000000	KMCV10: 0	:LO BYTE=MODE (200=BOP,0=CCP) HI BYTE=CRC32(200=YES,0=NO)
5277	026404	000000	KMCP10: 0	:LO BYTE=INT.VECTOR ADDR.,HI=E/I(200=EXT 0=INT)
5278	026406	000000	KMCS10: 0	:LO BYTE=INT.PRI. HI BYTE=LINE BYTE
5279			:PARAMETER BLOCK FOR KMC#11	
5280	026410	000000	KMCR11: 0	:CONTROL + STATUS REGISTER FOR KMC11 #11
5281	026412	000000	KMCE11: 0	:LO BYTE=ENABLE (200=YES,0=NO);HI BYTE UNUSED
5282	026414	000000	KMCV11: 0	:LO BYTE=MODE (200=BOP,0=CCP) HI BYTE=CRC32(200=YES,0=NO)
5283	026416	000000	KMCP11: 0	:LO BYTE=INT.VECTOR ADDR.,HI=E/I(200=EXT 0=INT)
5284	026420	000000	KMCS11: 0	:LO BYTE=INT.PRI. HI BYTE=LINE BYTE
5285			:PARAMETER BLOCK FOR KMC#12	
5286	026422	000000	KMCR12: 0	:CONTROL + STATUS REGISTER FOR KMC11 #12
5287	026424	000000	KMCE12: 0	:LO BYTE=ENABLE (200=YES,0=NO);HI BYTE UNUSED

```
5288 026426 000000 KMCV12: 0 ;LO BYTE=MODE (200=BOP,0=CCP) HI BYTE=CRC32(200=YES,0=NO)
5289 026430 000000 KMCP12: 0 ;LO BYTE=INT.VECTOR ADDR.,HI=E/I(200=EXT 0=INT)
5290 026432 000000 KMCS12: 0 ;LO BYTE=INT.PRI. HI BYTE=LINE BYTE
5291 ;PARAMETER BLOCK FOR KMC#13
5292 026434 000000 KMCR13: 0 ;CONTROL + STATUS REGISTER FOR KMC11 #13
5293 026436 000000 KMCE13: 0 ;LO BYTE=ENABLE (200=YES,0=NO);HI BYTE UNUSED
5294 026440 000000 KMCV13: 0 ;LO BYTE=MODE (200=BOP,0=CCP) HI BYTE=CRC32(200=YES,0=NO)
5295 026442 000000 KMCP13: 0 ;LO BYTE=INT.VECTOR ADDR.,HI=E/I(200=EXT 0=INT)
5296 026444 000000 KMCS13: 0 ;LO BYTE=INT.PRI. HI BYTE=LINE BYTE
5297 ;PARAMETER BLOCK FOR KMC#14
5298 026446 000000 KMCR14: 0 ;CONTROL + STATUS REGISTER FOR KMC11 #14
5299 026450 000000 KMCE14: 0 ;LO BYTE=ENABLE (200=YES,0=NO);HI BYTE UNUSED
5300 026452 000000 KMCV14: 0 ;LO BYTE=MODE (200=BOP,0=CCP) HI BYTE=CRC32(200=YES,0=NO)
5301 026454 000000 KMCP14: 0 ;LO BYTE=INT.VECTOR ADDR.,HI=E/I(200=EXT 0=INT)
5302 026456 000000 KMCS14: 0 ;LO BYTE=INT.PRI. HI BYTE=LINE BYTE
5303 ;PARAMETER BLOCK FOR KMC#15
5304 026460 000000 KMCR15: 0 ;CONTROL + STATUS REGISTER FOR KMC11 #15
5305 026462 000000 KMCE15: 0 ;LO BYTE=ENABLE (200=YES,0=NO);HI BYTE UNUSED
5306 026464 000000 KMCV15: 0 ;LO BYTE=MODE (200=BOP,0=CCP) HI BYTE=CRC32(200=YES,0=NO)
5307 026466 000000 KMCP15: 0 ;LO BYTE=INT.VECTOR ADDR.,HI=E/I(200=EXT 0=INT)
5308 026470 000000 KMCS15: 0 ;LO BYTE=INT.PRI. HI BYTE=LINE BYTE
5309 ;PARAMETER BLOCK FOR KMC#16
5310 026472 000000 KMCR16: 0 ;CONTROL + STATUS REGISTER FOR KMC11 #16
5311 026474 000000 KMCE16: 0 ;LO BYTE=ENABLE (200=YES,0=NO);HI BYTE UNUSED
5312 026476 000000 KMCV16: 0 ;LO BYTE=MODE (200=BOP,0=CCP) HI BYTE=CRC32(200=YES,0=NO)
5313 026500 000000 KMCP16: 0 ;LO BYTE=INT.VECTOR ADDR.,HI=E/I(200=EXT 0=INT)
5314 026502 000000 KMCS16: 0 ;LO BYTE=INT.PRI. HI BYTE=LINE BYTE
5315
5316 026504 000000 KMCTBE: 0
5317
5318 026506 013701 024360 GETKMC: MOV CURTAB,R1 ;GET CURRENT TABLE POINTER
5319 026512 062701 000012 1$: ADD #12,R1
5320 026516 020127 026504 CMP R1,#KMCS16+2 ;a END OF TABLE?
5321 026522 001004 BNE 2$ ;B=NO
5322 026524 012737 026232 024360 MOV #KMCTAB-12,CURTAB
5323 026532 000207 RTS PC
5324 026534 105761 000002 2$: TSTB 2(R1) ;ENABLED?
5325 026540 100364 BPL 1$ ;B=NO
5326 026542 010137 024360 MOV R1,CURTAB ;NOW POINTING TO CURRENT KMC
5327 026546 012702 002260 MOV #SELO,R2 ;POINT TO DEVICE REGISTER POINTERS
5328 026552 012104 MOV (R1)+,R4 ;GET KMC CSRO
5329 026554 012705 000010 MOV #8,R5 ;8 REGISTERS
5330 026560 010422 25$: MOV R4,(R2)+
5331 026562 005204 INC R4
5332 026564 005305 DEC R5
5333 026566 001374 BNE 25$ ;BR=NO
5334 026570 062701 000002 ADD #2,R1 ;SKIP ENABLE ENTRY
5335 ;LOAD THE MODE
5336 026574 112137 002311 MOVB (R1)+,MODE
5337 026600 112137 002314 MOVB (R1)+,CRC32F ;LOAD CRC-32 STATUS
5338 ;LOAD VECTOR ADDRESSES
5339 026604 011112 MOV (R1),(R2) ;LOAD INPUT VECTOR
5340 026606 042722 100000 BIC #100000,(R2)+ ;NEXT WORD & CLR JUNK
5341 026612 011112 MOV (R1),(R2) ;LOAD OUTPUT VECTOR
5342 026614 062712 000004 ADD #4,(R2) ;+4=OUTPUT VECTOR ADDR.
5343 026620 042722 100000 BIC #100000,(R2)+ ;STRIP E/I BIT
```

```

5344 026624 105721          TSTB (R1)+      ;NEXT BYTE
5345 026626 012737 000200 002316 MOV #200,CLKDAT ;INT. + CLK
5346 026634 105721          TSTB (R1)+      ;INT?
5347 026636 100003          BPL 26$         ;B=YES
5348 026640 042737 000200 002316 BIC #200,CLKDAT ;MAKE IT EXT.
5349 026646          26$:
5350 026646 112112          MOVB (R1)+,(R2)   ;INPUT PRIORITY
5351 026650 106112          ROLB (R2)         ;MAKE SAME AS P.S.WORD
5352 026652 106112          ROLB (R2)
5353 026654 106112          ROLB (R2)
5354 026656 106112          ROLB (R2)
5355 026660 106112          ROLB (R2)
5356 026662 162712 000040    SUB #40,(R2)    ;INT OCCURS @ LEVEL BELOW PROC.
5357 026666 012212          MOV (R2)+,(R2)    ;OUTPUT PRIORITY
5358 026670 111137 002312    MOVB (R1),LINEB ;LOAD LINE BYTE
5359 026674 012702 000001    MOV #1,R2
5360 026700 013700 024360    MOV CURTAB,R0    ;FIND UNIT#
5361 026704 162700 026244    SUB #KMC TAB,R0
5362 026710 001405          BEQ 4$
5363 026712 005202          3$: INC R2
5364 026714 162700 000012    SUB #12,R0
5365 026720 001401          BEQ 4$
5366 026722 000773          BR 3$
5367 026724 110237 002456    4$: MOVB R2,CUNIT ;CURRENT UNIT #
5368 026730 062716 000002    ADD #2,(SP)
5369 026734 000207          RTS PC
5370          .SBTTL ROUTINE TO SIZE MEMORY
5371
5372          ;*****
5373          ;*CALL:
5374          ;* JSR PC,$SIZE
5375          ;* RETURN
5376          ;*$LSTAD WILL CONTAIN:
5377          ;* WITH KT11 OPTION -- LAST VIRTUAL ADDRESS OF THE LAST BANK
5378          ;* WITHOUT KT11 OPTION -- LAST ABSOLUTE ADDRESS OF AVAILABLE MEMORY
5379          ;*$LSTBK WILL CONTAIN THE LAST BANK AS A SAF
5380          ;*$KT11 IS THE MEMORY MANAGEMENT KEY
5381          ;*BIT07 = 0 DON'T USE MEMORY MANAGEMENT
5382          ;* MUST BE SETUP BEFORE THE CALL
5383          ;*BIT15 = 0 DON'T HAVE MEMORY MANAGEMENT OPTION
5384          ;* DETERMINED BY ROUTINE
5385
5386 026736 010046          $SIZE: MOV R0,-(SP)  ;;SAVE R0 ON THE STACK
5387 026740 010146          MOV R1,-(SP)  ;;SAVE R1 ON THE STACK
5388 026742 010246          MOV R2,-(SP)  ;;SAVE R2 ON THE STACK
5389 026744 010346          MOV R3,-(SP)  ;;SAVE R3 ON THE STACK
5390 026746 010446          MOV R4,-(SP)  ;;SAVE R4 ON THE STACK
5391 026750 013746 000114    MOV @#114,-(SP) ;;SAVE MEMORY ERROR VECTOR PS & PC
5392 026754 013746 000116    MOV @#116,-(SP)
5393 026760 012737 000116 000114 MOV #116,@#114 ;;IGNORE PARITY ERRORS WHILE SIZING
5394 026766 012737 000002 000116 MOV #RTI,@#116
5395 026774 013746 000004    MOV @#ERRVEC,-(SP) ;;SAVE PRESENT ERROR VECTOR PS & PC
5396 027000 013746 000006    MOV @#ERRVEC+2,-(SP)
5397 027004 010600          MOV SP,R0      ;;SAVE THE STACK POINTER
5398          ;;SET THE ERRVEC PS TO THE PRESENT PS
5399 027006 104400          TRAP              ;;PUSH OLD PSW AND PC ON STACK

```

BK001

```
5400 027010 012637 000006      MOV      (SP)+, @#ERRVEC+2      ;;SAVE THE PSW IN @#ERRVEC+2
5401 027014 012701 003776      MOV      #3776, R1      ;;SETUP ADDRESS
5402 027020 105727              TSTB      (PC)+      ;;USE MEMORY MANAGEMENT?
5403 027022 000200              SKT11: .WORD 200      ;;SET TO USE MEMORY MANAGEMENT
5404 027024 100145              BPL      $CORE      ;;BR IF NO
5405 027026 012737 027332 000004      MOV      #SKTNEX, @#ERRVEC      ;;SET FOR TIMEOUT
5406 027034 005737 177572              TST      @#SR0      ;;KT11 ARE YOU THERE?
5407 027040 052737 100000 027022      BIS      #100000, SKT11      ;;YES--SET KT11 KEY
5408 027046 012737 027076 000004      MOV      #100$, @#ERRVEC      ;;SET FOR TIMEOUT
5409 027054 005737 170200              TST      @#170200      ;;UNIBUS MAP ARE YOU THERE?
5410 027060 012737 176200 027116      MOV      #176200, @#$STOP      ;;YES-SET COMPARISON VALUE FOR 11/70
5411 027066 012737 000200 027114      MOV      #200, @#$MAP      ;;TURN ON MAP INDICATOR
5412 027074 000411              BR      $MAPRG      ;;GO SET UP MAP REGISTERS
5413 027076 012737 006200 027116 100$:      MOV      #6200, @#$STOP      ;;COMPARISON VALUE FOR 18 BIT MAPPING
5414 027104 022626              CMP      (SP)+, (SP)+      ;;CLEAN OFF STACK
5415 027106 005037 027114              CLR      @#$MAP      ;;MAKE SURE MAP INDICATOR TURNED OFF
5416 027112 000412              BR      $NOMAP      ;;BK001
5417 027114 000000              $MAP: .WORD 0      ;;=200 IF MAP PRESENT
5418 027116 000000              $STOP: .WORD 0      ;;FILLED WITH APPROPRIATE COMPARISON VALUE
5419 027120 012703 000037              $MAPRG: MOV      #37, R3      ;;SET UP COUNTER
5420 027124 012702 170200              MOV      #170200, R2      ;;START WITH MAPLO
5421 027130 005022              100$: CLR      (R2)+      ;;LOAD ALL MAP REGISTERS
5422 027132 012722 000074              MOV      #74, (R2)+      ;;WITH THE VALUE 17000000
5423 027136 077304              SOB      R3, 100$      ;;DO ALL 31 REGISTERS
5424 027140              $NOMAP:
5425 027140 005046              CLR      -(SP)      ;;INITIALIZE FOR 'PAR' LOADING
5426 027142 012702 172340              MOV      #KIPAR0, R2      ;;ADDRESS OF FIRST 'PAR'
5427 027146 012703 000010              MOV      #^DB, R3      ;;LOAD EIGHT 'PAR.'S' AND EIGHT 'PDR.'S'
5428 027152 012762 077406 177740 1$:      MOV      #77406, -40(R2)      ;;PDR = 4K, UP, READ/WRITE
5429 027160 011622              MOV      (SP), (R2)+      ;;LOAD 'PAR'
5430 027162 062716 000200              ADD      #200, (SP)      ;;UPDATE FOR NEXT 'PAR'
5431 027166 077307              SOB      R3, 1$      ;;LOOP UNTIL ALL EIGHT ARE LOADED
5432 027170 012742 177600              MOV      #177600, -(R2)      ;;SETUP KIPAR7 FOR I/O
5433 027174 005042              CLR      -(R2)      ;;SETUP KIPAR6 FOR TESTING
5434 027176 012737 027214 000004      MOV      #2$, @#ERRVEC      ;;CATCH TIMEOUT IF NO SR3
5435 027204 012737 000060 172516      MOV      #60, @#SR3      ;;ENABLE 22 BIT MODE AND UNIBUS MAP
5436 027212 000401              BR      3$      ;;THIS PDP-11 HAS A SR3 REGISTER
5437 027214 022626              2$: CMP      (SP)+, (SP)+      ;;CLEAN OFF THE STACK--NO SR3
5438 027216 005237 177572              3$: INC      @#SR0      ;;TURN ON MEMORY MANAGEMENT
5439 027222 012737 027270 000004      MOV      #SKTOUT, @#ERRVEC      ;;SET FOR TIME OUT
5440 027230 105737 027114              TSTB      @#$MAP      ;;IS MAP THERE?
5441 027234 100006              BPL      4$      ;;NO-SKIP
5442 027236 012737 027312 000114      MOV      #SMMOUT, @#114      ;;SET UP MEMORY ERROR VECTOR
5443 027244 013737 000006 000116      MOV      @#ERRVEC+2, @#116      ;;LOCK OUT INTERRUPTS
5444 027252 005737 143776              4$: TST      @#143776      ;;TRAP ON NON-EX-MEM
5445 027256 062712 000040              ADD      #40, (R2)      ;;MAKE A 1K STEP
5446 027262 023712 027116              CMP      @#$STOP, (R2)      ;;LAST ONE?
5447 027266 101371              BHI      4$      ;;NO--TRY IT
5448 027270 011202              SKTOUT: MOV      (R2), R2      ;;GET LAST BANK+1
5449 027272 005037 177572              CLR      @#SR0      ;;TURN OFF MEMORY MANAGEMENT
5450 027276 105737 027114              TSTB      @#$MAP      ;;IS MAP THERE?
5451 027302 100034              BPL      $SIZEX      ;;NO-SKIP
5452 027304 005037 172516              CLR      @#SR3      ;;TURN OFF MAP
5453 027310 000431              BR      $SIZEX
5454 027312 013704 177744              $SMMOUT: MOV      @#177744, R4      ;;SAVE MEMORY ERROR REGISTER
5455 027316 010437 177744              MOV      R4, @#177744      ;;CLEAR BITS IN REGISTER
```

```
5456 027322 032704 000001      BIT      #1,R4      ;;MEMORY TIMEOUT?      BK001
5457 027326 001360      BNE      $KTOUT      ;;YES-EXIT      BK001
5458 027330 000002      RTI      ;;MUST BE PARITY ERROR-IGNORE IT      BK001
5459 027332 042737 100000 027022 $KTNEX: BIC      #100000,$KT11      ;;KT11 NON-EXISTENT
5460 027340 012737 027370 000004 $CORE:  MOV      #SCROUT,@#ERRVEC      ;;SET FOR TIMEOUT
5461 027346 005002      CLR      R2      ;;SET UP BANK
5462 027350 062701 004000      1$:  ADD      #4000,R1      ;;INCREMENT BY 1K
5463 027354 062702 000040      ADD      #40,R2      ;;1K STEP
5464 027360 005711      TST      (R1)      ;;TRAP ON TIME OUT
5465 027362 022701 177776      CMP      #177776,R1      ;;LAST ONE
5466 027366 001370      BNE      1$      ;;NO--TRY AGAIN
5467 027370 162701 004000      $SCROUT: SUB      #4000,R1
5468 027374 162702 000040      $SIZE:  SUB      #40,R2      ;;DROP BACK
5469 027400 010006      MOV      R0,SP      ;;RESTORE THE STACK
5470 027402 012637 000006      MOV      (SP)+,@#ERRVEC+2      ;;RESTORE ERROR VECTOR
5471 027406 012637 000004      MOV      (SP)+,@#ERRVEC
5472 027412 012637 000116      MOV      (SP)+,@#116      ;;RESTORE MEMORY ERROR VECTOR
5473 027416 012637 000114      MOV      (SP)+,@#114
5474 027422 010137 027446      MOV      R1,$LSTAD      ;;LAST ADDRESS
5475 027426 010237 027450      MOV      R2,$LSTBK      ;;LAST BANK
5476 027432 012604      MOV      (SP)+,R4      ;;RESTORE R4      BK001
5477 027434 012603      MOV      (SP)+,R3      ;;RESTORE R3
5478 027436 012602      MOV      (SP)+,R2      ;;RESTORE R2
5479 027440 012601      MOV      (SP)+,R1      ;;RESTORE R1
5480 027442 012600      MOV      (SP)+,R0      ;;RESTORE R0
5481 027444 000207      RTS      PC
5482 027446 000000      $LSTAD: .WORD      0      ;;CONTAINS THE LAST ADDRESS
5483 027450 000000      $LSTBK: .WORD      0      ;;CONTAINS THE LAST BANK
5484
5485 027452 012737 000200 027022 GETCOR: MOV      #200,$KT11      ;USE THE MEM. MANAGEMENT
5486 027460 004737 026736      JSR      PC,$SIZE
5487 027464 005005      CLR      R5
5488 027466 013704 027446      MOV      $LSTAD,R4      ;SET UP PHYSICAL ADDRESS
5489 027472 005737 027022      TST      $KT11      ;IS THERE A M.M?
5490 027476 100024      BPL      3$      ;B=N PHY ADDR. IN $LSTAD
5491 027500 023727 027450 007540      CMP      $LSTBK,#7540      ;MORE TH' V 124K?
5492 027506 003403      BLE      1$      ;B=NO
5493 027510 012737 007540 027450      MOV      #7540,$LSTBK      ;ONLY NEED 18 BITS (ADDRESS)
5494 027516      1$:  ;MAKE THE VIRTUAL ADDRESS INTO A PHYSICAL
5495 027516 013704 027450      MOV      $LSTBK,R4
5496 027522 012703 000006      MOV      #6,R3
5497 027526 000241      2$:  CLC
5498 027530 006304      ASL      R4      ;ALIGN BITS FOR AN 18 BIT PHYSICAL ADDRESS
5499 027532 006105      ROL      R5      ;MOVE CARRY INTO R5
5500 027534 005303      DEC      R3
5501 027536 001373      BNE      2$
5502 027540 052704 003776      BIS      #3776,R4      ;ADD 1K ($LSTAD ALWAYS 3776)
5503 027544 010437 027446      MOV      R4,$LSTAD      ;MAKE SAME AS NO KT11
5504 027550      3$:  ;$LSTAD = LOW ORDER 16 BITS + R5 = BITS 16 + 17 IN 0 + 1
5505 027550 010537 002340      MOV      R5,XBITS
5506 027554 001006      BNE      35$
5507 027556 023704 002532      CMP      BADDR,R4      ;BR=BASE<PHYSICAL ADDRESS
5508 027562 103036      BHIS      6$      ;ANY CORE OVER DIAG.
5509 027564 013737 027446 002336      MOV      $LSTAD,HADDR      ;B=NO C/C=0
5510 027572
5511 027572 000241      35$:  CLC      ;ADJUST FOR SIGNED SUBTRACT
```

```
5512 027574 006004
5513 027576 013701 002532
5514 027602 160104
5515 027604 000240
5516 027606 000240
5517 027610 000240
5518
5519 027612 103001
5520 027614 005305
5521 027616 006304
5522 027620 006105
5523 027622 010537 027662
5524 027626 010437 027664
5525
5526 027632 005705
5527 027634 001006
5528 027636 022704 017777
5529 027642 003403
5530 027644 010437 002534
5531 027650 000403
5532 027652 012737 020000 002534
5533 027660 000207
5534 027662 000000
5535 027664 000000
5536
5537
5538
5539
5540
5541
5542
5543 027666 000241
5544 027670 013704 030006
5545 027674 006004
5546 027676 060437 030010
5547 027702 103011
5548 027704 062737 040000 030012
5549 027712
5550 027712 063737 030006 030014
5551 027720 103002
5552 027722 005237 030016
5553 027726 023737 030016 027662
5554 027734 003012
5555 027736 023737 030014 027664
5556 027744 003403
5557 027746 005737 030016
5558 027752 001403
5559 027754 062716 000002
5560 027760 000207
5561 027762 005037 030012
5562 027766 013737 002532 030010
5563 027774 005037 030014
5564 030000 005037 030016
5565 030004 000207
5566 030006 000000
5567 030010 000000

ROR R4
MOV BADDR,R1
SUB R1,R4
NOP
NOP
NOP

;***** ABOVE CODE MAY NEED ADJUSTMENT
;BORROW? B=N
;YES
;2X RANGE = CHR COUNT

4$: BCC 4$
DEC R5
ASL R4
ROL R5
MOV R5,CCHI
MOV R4,CCLO
;FIND C/C FOR UP TO 8K BYTES OVER THE BASE ADDRESS
TST R5
BNE 5$
CMP #17777,R4
BLE 5$
MOV R4,CCTST
BR 6$
MOV #20000,CCTST
RTS PC
CCHI: 0
CCLO: 0

;THIS ROUTINE WILL LOAD THE 'NEXT' SEQUENTIAL 18 BIT BUFFER ADDRESS
; (BITS 17 + 16) INTO 'EABITS' (BITS 15 + 14) AND (BITS 15-0) INTO
;'ADDR'. THE BUFFER ADDRESS IS DETERMINED FROM THE 'LAST' BUFFER ADDRESS
; CALLED AND THE VALUE OF THE CHARACTER COUNT (LOCATION 'CCNT').
; IF BUFFER IS GRANTED RETURN IS PC + 2 IF NOT RETURN IS PC

GETPHA: CLC
MOV CCNT,R4
ROR R4
ADD R4,ADDR
BCC 2$
ADD #40000,EABITS
;NOW FIND OUT IF ADDRESS IS LEGAL
ADD CCNT,ADINL
BCC 2$
INC ADINH
CMP ADINH,CCHI
BOT 4$
C'AP ADINL,CCLO
BLE 3$
TST ADINH
BEQ 4$
ADD #2,(SP)
RTS PC
CLR EABITS
MOV BADDR,ADDR
CLR ADINL
CLR ADINH
RTS PC

1$: ;ADD REQ CHARACTER COUNT TO ADDRESS INDEX
2$: ;DOES THIS REQUEST EXCEED THE BUFFER LENGTH?
;B=Y RET.=PC
3$: ;B=OK GRANT BUFFER REQUEST
;ARE THERE UPPER BITS?
;B=N REG.=PC
;GOOD RETURN
4$: ;REPOINT BUFFER
;LOAD BASE ADDRESS (BUFFER)
;INITIALIZE ADDRESS INDEX COUNTERS

CCNT: 0
ADDR: 0

;# OF CHARACTERS REQUESTED IN BUFFER SIZE
;ADDRESS OF REQUESTED BUFFER
```

```
5568 030012 000000      EABITS: 0      ;E.A. BITS OF REQL STED BUFFER
5569 030014 000000      ADINL: 0      ;ADDRESS INDEX LOW (KEEPS TRACK OF INDEX IN BUFFER)
5570 030016 000000      ADINH: 0      ;ADDRESS INDEX HIGH
5571
5572      ;THIS ROUTINE WILL LOAD THE DESIRED FIRMWARE (PARAMETER
5573      ;BLOCK SPECIFIED) TO THE CURRENTLY ACTIVE KMC-11
5574
5575 030020 013746 000004      LD MODE: MOV 2#4,-(SP)      ;SAVE 4 & 6
5576 030024 013746 000006      MOV 2#6,-(SP)
5577 030030 012737 030166 000004      MOV 2#4,2#4      ;TIME OUT TRAP ROUTINE
5578 030036 012777 002000 152214      MOV #2000,2#SELO
5579 030044 013737 030242 000004      MOV 6$,2#4
5580 030052 005037 030240      CLR 5$      ;INITIALIZE THE ADDRESS COUNTER
5581 030056 012705 036060      MOV #CCPFRM,R5      ;SET POINTER FOR CCP MODE
5582 030062 105737 002311      TSTB MODE      ;REQUESTING CCP?
5583 030066 100007      BPL 2$      ;B=YES CCP
5584 030070 012705 046060      MOV #BOPN32,R5      ;POINT TO FILE WO/CRC32
5585 030074 105737 002314      TSTB CRC32F      ;IS CRC32 ENABLED?
5586 030100 100002      BPL 2$      ;B=NO
5587 030102 012705 042060      MOV #BOPW32,R5      ;POINT TO FILE WITH CRC32
5588 030106      2$: ;LOAD THE DATA WORD INTO KMC11
5589 030106 013777 030240 152154      MOV 5$,2#SEL4      ;LOAD ADDRESS
5590 030114 012577 152154      MOV (R5)+,2#SEL6      ;LOAD THE DATA WORD
5591 030120 052777 020000 152132      BIS #20000,2#SELO
5592 030126 000240      NOP
5593 030130 000240      NOP
5594 030132 042777 020000 152120      BIC #20000,2#SELO
5595      ;UPDATE THE ADDRESS AND TEST FOR DONE
5596 030140 005237 030240      INC 5$      ;+1 TO ADDRESS COUNTER
5597 030144 022737 002000 030240      CMP #2000,5$
5598 030152 001355      BNE 2$
5599 030154 012637 000006      MOV (SP)+,2#6      ;RESTORE 4 & 6
5600 030160 012637 000004      MOV (SP)+,2#4
5601 030164 000207      RTS PC      ;RETURN WHEN DONE
5602 030166      4$: ;INDICATE KMC-11 NOT FOUND
5603 030166 062706 000004      ADD #4,SP      ;RESTORE STACK
5604 030172 104401 030200      TYPE 65$      ;TYPE ASCII STRING
5605 030176 000413      BR 64$      ;GET OVER THE ASCII
5606      ;:65$: .ASCIIZ <15><12>/'KMC' ADDRESS ERROR/
5607      64$:
5608 030226      MOV SELO,-(SP)      ;GET DEVICE ADDRESS
5609 030232 104402      TYPOC      ; & TYPE IT OUT
5610 030234 000137 002076      JMP MONIT      ; RETURN TO MONITOR FOR CORRECTION
5611
5612 030240 000000      5$: 0      ;ADDRESS COUNTER
5613 030242 000000      6$: 0      ;TEMP. STORE
5614 030244 113737 002312 002313      SETLIN: MOV B LINEB,CLINE      ;LOAD LINE BYTE
5615 030252 012737 177777 002454      MOV #-1,CURLIN      ;INIT LINE
5616 030260 004737 031650      JSR PC,GETLIN      ;GET FIRST LINE
5617 030264 000000      HALT      ;:NO LINES ENABLED!!!
5618 030266 000207      RTS PC      ;EXIT
5619
5620
5621      ;THIS ROUTINE WILL DISPLAY ON THE CONSOLE DEVICE THE UNITS
5622      ;REQUIREING LOOP AROUND CONNECTORS.
5623      ;
```

```
5624
5625 030270 012700 000020 LPARND: MOV #16, R0 ;CLEAR UNIT FLAGS
5626 030274 012701 030556 MOV #UNITFG, R1
5627 030300 005021 1$: CLR (R1)+
5628 030302 005300 DEC R0
5629 030304 001375 BNE 1$
5630 030306 012737 026232 024360 MOV #KMCTAB-12, CURTAB
5631 030314 004737 026506 2$: JSR PC, GETKMC ;GET NEXT ENABLED UNIT
5632 030320 000414 BR 3$ ;B=DONE DISPLAY UNITS
5633 030322 105737 002316 TSTB CLKDAT ;INTERNAL LOOP SELECTED?
5634 030326 100772 BMI 2$ ;B=YES
5635 030330 006337 002456 ASL CUNIT ;ADJUST UNIT # FOR MEMORY REF.
5636 030334 012701 030554 MOV #UNITFG-2, R1 ;INDEX INTO TABLE
5637 030340 063701 002456 ADD CUNIT, R1
5638 030344 012711 177777 MOV #-1, (R1) ;SET FLAG
5639 030350 000761 BR 2$ ;CHECK FOR NEXT UNIT
5640 ;DISPLAY UNITS REQUIRING LOOP CONNECTORS
5641 030352 005037 002332 3$: CLR TYPFLG ;CLEAR TYPE INDICATOR FLAG
5642 030356 012737 000001 002456 MOV #1, CUNIT
5643 030364 104401 030372 TYPE ,65$ ;;TYPE ASCIZ STRING
5644 030370 000421 BR 64$ ;;GET OVER THE ASCIZ
5645 ;;65$: .ASCIZ <15><12>/UNITS REQUIRING LOOP CONNECTORS/
5646 030434 64$:
5647 030434 012701 030556 MOV #UNITFG, R1 ;POINT TO UNIT TABLE
5648 030440 005721 4$: TST (R1)+ ;UNIT RUNNING EXTERNAL?
5649 030442 100017 BPL 5$ ;B=NO
5650 030444 104401 030452 TYPE ,67$ ;;TYPE ASCIZ STRING
5651 030450 000406 BR 66$ ;;GET OVER THE ASCIZ
5652 ;;67$: .ASCIZ <15><12>/ UNIT # /
5653 66$:
5654 030466 MOV CUNIT, -(SP) ;;SAVE CUNIT FOR TYPEOUT
5655 030472 104405 TYPDS ;;GO TYPE--DECIMAL ASCII WITH SIGN
5656 030474 012737 177777 002332 MOV #-1, TYPFLG ;INDICATE UNIT # HAS BEEN PRINTED
5657 030502 005237 002456 5$: INC CUNIT
5658 030506 022737 000021 002456 CMP #17., CUNIT ;DONE?
5659 030514 001351 BNE 4$ ;B=NO
5660 030516 005737 002332 TST TYPFLG ;ANNY UNITS DECLARED?
5661 030522 100407 BMI 6$ ;B=YES
5662 030524 104401 030532 TYPE ,69$ ;;TYPE ASCIZ STRING
5663 030530 000404 BR 68$ ;;GET OVER THE ASCIZ
5664 ;;69$: .ASCIZ <15><12>/ NONE/
5665 68$:
5666 030542 6$:
5667 030542 104401 030550 TYPE ,71$ ;;TYPE ASCIZ STRING
5668 030546 000402 BR 70$ ;;GET OVER THE ASCIZ
5669 ;;71$: .ASCIZ <15><12>
5670 70$:
5671 030554 000207 RTS PC
5672
5673 030556 000020 UNITFG: .BLKW 16.
```

```
5674
5675
5676 ;THIS ROUTINE WILL LOAD EACH POTENTIAL INTERRUPT VECTOR FROM
5677 ;LOCATIONS 204 TO 776 WITH A .+2/IOT TRAP SEQUENCE TO CATCH
5678 ;ANY HARDWARE INTERRUPT TO AN ILLEGAL VECTOR.
5679
```

```
5680
5681 030616 012700 000204      INTPRM: MOV    #204,R0      ;POINTER TO LOCATIONS
5682 030622 012701 000206      MOV    #206,R1      ;CONTAINS ADDRESS
5683 030626 010120      1$:    MOV    R1,(R0)+      ;+2
5684 030630 012720 000004      MOV    #IOT,(R0)+    ;IOT INSTR.
5685 030634 062701 000004      ADD    #4,R1
5686 030640 022700 001000      CMP    #1000,R0      ;DONE?
5687 030644 001370      BNE     1$                ;B=NO
5688                                ;LOAD TIME OUT TRAP VECTOR
5689 030646 012737 000006 000004 MOV    #6,#4
5690 030654 012737 000004 000006 MOV    #4,#6
5691 030662 000207      RTS     PC
```

```
5692
5693      ;THE FOLLOWING ROUTINE WILL INITIALIZE THE CURRENT KMC-11
5694      ;FIRMWARE BY ISSUING A MASTER CLEAR AND SETTING THE RUN
5695      ;BIT. A WATCH-LOOP IS EXECUTED TO ASSURE THAT THE FIRMWARE
5696      ;IS RESPONDING TO THE USER COMMANDS. IF THE FIRMWARE
5697      ;SHOULD BE HUNG OR FAILS TO RESPOND A RETURN TO PC IS
5698      ;EXECUTED, OTHERWISE PC+2
5699
```

```
5700 030664 012737 000015 030756 INTKMC: MOV    #15,3$
5701 030672 112777 000377 151364      MOV    #377,@SEL2
5702 030700 005077 151354      CLR     @SELO      ;*: CLEAR RUN, IF UP
5703 030704 012777 040000 151346      MOV    #40000,@SELO  ;SET MASTER CLEAR
5704 030712 012777 100000 151340      MOV    #100000,@SELO ;RUN
5705 030720 005037 001160      CLR     $TMP0
5706 030724 105237 001160      1$:    INCB   $TMP0      ;WAIT LOOP
5707 030730 001375      BNE     1$
5708 030732 105777 151326      TSTB   @SEL2      ;DONE?
5709 030736 001404      BEQ     2$                ;B=Y
5710 030740 005337 030756      DEC     3$
5711 030744 001367      BNE     1$
5712 030746 000207      RTS     PC
5713 030750 062716 000002      2$:    ADD    #2,(SP)
5714 030754 000207      RTS     PC
5715 030756 000000      3$:    0
```

```
5716
5717
5718      ;THIS ROUTINE TEST THAT A 'REQUEST IN' CAN BE DONE BY
5719      ;CHECKING THAT NO OUTPUT REQUESTS ARE POSTED (SEL2) AND THAT
5720      ;THE FIRMWARE RESPONDS TO A REQUEST IN BY POSTING THE READY-IN.
5721      ;IF THE REQUEST IN CAN BE DONE THE RETURN IS TO PC+2 OTHERWISE TO PC
5722
```

```
5723
5724 030760 112777 000040 151272 REQNT: MOV    #40,@SELO      ;TRANSMIT BUFFER IN REQ.
5725 030766 000417      BR      REQGO
5726 030770 112777 000044 151262 REQNR: MOV    #44,@SELO      ;REC. BUFFER IN REQUEST
5727 030776 000413      BR      REQGO
5728 031000 112777 000045 151252 REQNC: MOV    #45,@SELO      ;CONTROL IN RECEIVE REQUEST
5729 031006 000407      BR      REQGO
5730 031010 112777 000041 151242 REQNX: MOV    #41,@SELO      ;CONTROL IN TRANSMIT REQUEST
5731 031016 000403      BR      REQGO
5732 031020 112777 000043 151232 REQNI: MOV    #43,@SELO      ;INITIALIZE IN REQUEST
5733 031026 105777 151232 REQGO: TSTB   @SEL2      ;OUTPUT REQUESTED
5734 031032 100410      BMI     2$                ;B=Y
5735 031034 005037 001160      CLR     $TMP0
```

```
5736 031040 105777 151214      1$:  TSTB    @SELO      ;IS REQUEST GRANTED?
5737 031044 100404              BMI     3$          ;B=Y
5738 031046 005237 001160      INC     $TMP0      ;ALLOW TIME FOR RESPONSE
5739 031052 001372              BNE     1$          ;TRY AGAIN
5740 031054 000207              RTS     PC          ;RETURN PC
5741 031056 062716 000002      2$:  RTS     PC
5742 031062 000207              3$:  ADD     #2,(SP)
5743                                RTS     PC
5744
5745                                ;THIS ROUTINE WILL LOAD THE LINE INITIALIZE COMMAND
5746                                ;IN TO THE DEVICE REGISTERS, CURLIN=LINE # (AT THIS POINE RDY=IN
5747                                ;IS SET)
5748 031064 113777 002454 151174 INTLIN: MOVB    CURLIN,@SEL3
5749 031072 012777 000400 151174      MOV     #400,@SEL6      ;ENABLE LINE
5750 031100 005077 151164      CLR     @SEL4          ;UNUSED
5751 031104 004737 031772      JSR     PC,INPDUN      ;TEST THAT INPUT COMPLETED
5752 031110 000207              RTS     PC          ;RDY-IN FAILED TO DROP
5753 031112 062716 000002      ADD     #2,(SP)
5754 031116 000207              RTS     PC          ;COMMAND COMPLETED
5755
5756                                ;THIS ROUTINE WILL INITIALIZE ALL THE LINES STARTING AT LINE 1
5757 031120 013700 002454      INTALL: MOV     CURLIN,RO
5758 031124 113737 002313 031202      MOVB    CLINE,HOLDB      ;SAVE LINE BYTE
5759 031132 004737 030244      JSR     PC,SETLIN      ;INITIALIZE LINE # COUNTER
5760 031136              1$:
5761 031136 004737 031020      JSR     PC,REQNI      ;CAN A INITIALIZE IN BE EXECUTED?
5762 031142 000411              BR     3$          ;A REQUEST IN FAILED
5763 031144 004737 031064      JSR     PC,INTLIN      ;INITIALIZE # IN CURLIN
5764 031150 000406              BR     3$          ;LINE INITIALIZATE COMMAND FAILED TO COMPLETE
5765 031152 004737 031650      JSR     PC,GETLIN      ;GET THE NEXT LINE NUMBER
5766 031156 000401              BR     2$
5767 031160 000766              BR     1$
5768 031162 062716 000002      2$:  ADD     #2,(SP)
5769 031166 010037 002454      3$:  MOV     RO,CURLIN
5770 031172 113737 031202 002313      MOVB    HOLDB,CLINE      ;RESTORE
5771 031200 000207              RTS     PC
5772 031202 000000      HOLDB:  0
5773
5774
5775                                ;THIS ROUTINE WILL ISSUE A LINE INITIALIZE TO THE LINE
5776                                ;SPECIFIED IN 'CURLIN'
5777
5778 031204      INTLNE:
5779 031204 004737 031020      JSR     PC,REQNI      ;CAN A INITIALIZE IN BE EXECUTED?
5780 031210 000405              BR     1$
5781 031212 004737 031064      JSR     PC,INTLIN      ;INITIALIZE # IN CURLIN
5782 031216 000402              BR     1$
5783 031220 062716 000002      ADD     #2,(SP)      ;SET GOOD RETURN
5784 031224 000207      1$:  RTS     PC
5785
5786
5787                                ;THIS ROUTINE WILL START THE MAINTENCE CLOCK ON ALL ENABLED LINES
5788
5789 031226 013746 002454      CLKALL: MOV     CURLIN,-(SP)      ;SAVE CURLIN
5790 031232 004737 030244      JSR     PC,SETLIN
5791 031236      1$:
```

```
5792 031236 004737 031274      JSR    PC,STCLK      ;SET MAINT LOOP + START CLOCK
5793 031242 000411              BR      3$
5794 031244 004737 031650      JSR    PC,GETLIN     ;GET THE NEXT LINE NUMBER
5795 031250 000401              BR      2$                ;B=NO MORE LINES ENABLED
5796 031252 000771              BR      1$                ;DO NEXT LINE
5797 031254 012637 002454      2$:  MOV    (SP)+,CURLIN    ;RESTORE CURLIN
5798 031260 062716 000002      ADD     #2,(SP)        ;SET FOR A GOOD RETURN
5799 031264 000207              RTS     PC
5800 031266 012637 002454      3$:  MOV    (SP)+,CURLIN    ;RESTORE CURLIN
5801 031272 000207              RTS     PC                ;RETURN
5802
5803                          ;THIS ROUTINE WILL SET THE MAINTENANCE LOOP AND START THE
5804                          ;CLOCK. IF COMMAND FAILS TO EXECUTE A RETURN IS MADE TO
5805                          ;PC, OTHERWISE TO PC+2.
5806
5807 031274              STCLK:
5808 031274 004737 031000      JSR    PC,REQNC      ;CAN A CONTROL IN BE EXECUTED?
5809 031300 000415              BR      2$                ;CANNOT ISSUE COMMAND
5810 031302 113777 002454      MOV     CURLIN,@SEL3    ;SELECT LINE #
5811 031310 112777 000100      MOV     #100,@SEL7      ;INDICATE LOADING A MAINT. COMMAND
5812 031316 113777 002316      MOV     CLKDAT,@SEL6    ;SET INTERNAL LOOP + START CLOCK
5813 031324 005077 150740      CLR      @SEL4          ;UNUSED
5814 031330 004737 031772      JSR    PC,INPDUN     ;TEST THAT INPUT COMPLETED
5815 031334 000207              2$:  RTS     PC                ;COMMAND NOT COMPLETED
5816 031336 062716 000002      3$:  ADD     #2,(SP)
5817 031342 000207              RTS     PC
5818
5819                          ;THIS ROUTINE WILL EXECUTE EITHER A RECEIVE OR TRANSMIT BUFFER
5820                          ;IN COMMAND DEPENDING ON THE ENTRY POINT. THE BUFFER ADDRESS
5821                          ;IS OBTAINED FROM LOCATION 'BUFADR' AND 'BUFEA' AND THE CHARACTER
5822                          ;COUNT OF THE BUFFER FROM 'BUFCNT'. CURLIN = LINE #
5823
5824 031344              RIN:
5825 031344 004737 030770      JSR    PC,REQNR      ;CAN A REC. BUFFER-IN BE EXECUTED?
5826 031350 000422              BR      RPT          ;RECEIVE BUFFER FAIL TO
5827 031352 000403              BR      FRD
5828 031354              XIN:
5829 031354 004737 030760      JSR    PC,REQNT      ;CAN A TRANSMIT BUFF. IN BE EXECUTED?
5830 031360 000416              BR      RPT          ;TRANSMIT BUFFER FAILED TO LOAD
5831 031362 113777 002454      MOV     CURLIN,@SEL3    ;LOAD LINE #
5832 031370 013777 002522      MOV     BUFADR,@SEL4    ;16 BIT BUFFER ADDRESS
5833 031376 013777 002524      MOV     BUFEA,@SEL6     ;LOAD EA BITS
5834 031404 063777 002526      ADD     BUFCNT,@SEL6    ;LOAD CHARACTER COUNT OF BUFFER
5835 031412 004737 031772      JSR    PC,INPDUN     ;TEST THAT INPUT COMPLETED
5836 031416 000207              RPT:  RTS     PC
5837 031420 062716 000002      ADD     #2,(SP)
5838 031424 000207              RTS     PC
5839
5840                          ;THIS ROUTINE WILL COUNT THE # OF ONES IN THE WORD
5841                          ;CONTAINED IN LOCATION 'WAS' AND RETURN TO PC IF THE
5842                          ;# IF ODD OR PC+2 IF EVEN.
5843
5844 031426 012701 000010      PARCHK: MOV    #8.,R1      ;COUNT BITS TESTED
5845 031432 005002              CLR      R2
5846 031434 113700 002464      MOV     WAS,R0
5847 031440 100402              BMI     13$            ;BR=BIT 7 = 1
```

```
5848 031442 106300      12$:  ASLE    R0
5849 031444 100001      BPL      14$
5850 031446 005202      13$:  INC     R2          ;COUNT THE ONES
5851 031450 005301      14$:  DEC     R1          ;COUNT THE ONES
5852 031452 001373      BNE      12$          ;B=N
5853      ;ODD OR EVEN
5854 031454 032702 000001 BIT      #1,R2
5855 031460 001002      BNE      15$          ;B=ODD
5856 031462 062716 000002 ADD      #2,(SP)      ;EVEN
5857 031466 000207      15$:  RTS     PC
5858
5859      ;THIS ROUTINE WILL STORE CONTENTS OF THE CSRS FOR DISPLAYING
5860      ;IN THE ERROR REPORT ROUTINE
5861
5862 031470 017737 150564 002342 GETCSR: MOV     @SEL0,CSR0
5863 031476 017737 150562 002344      MOV     @SEL2,CSR2
5864 031504 017737 150560 002346      MOV     @SEL4,CSR4
5865 031512 017737 150556 002350      MOV     @SEL6,CSR6
5866 031520 000207      RTS     PC
5867
5868
5869 031522 127727 150536 000200 RBONLY: CMPB    @SEL2,#200      ;XMIT?
5870 031530 001005      BNE      1$          ;SKIP IF NO
5871 031532 105077 150526      CLRB    @SEL2      ;CLEAR
5872 031536 004737 031742      JSR     PC,REQOUT
5873 031542 000240      NOP
5874 031544 000207      1$:  RTS     PC
5875
5876
5877      ;THIS ROUTINE WILL ISSUE A LINE RESET COMMAND TO THE
5878      ;LINE SPECIFIED BY 'CURLIN'.
5879
5880      LRSET:
5881 031546      JSR     PC,REQNC      ;CAN A CONTROL IN BE EXECUTED?
5882 031552 000414      BR      2$          ;OUT REQ. OR HUNG
5883 031554 113777 002454 150504      MOVB   CURLIN,@SEL3      ;LD LINE #
5884 031562 112777 000020 150506      MOVB   #20,@SEL7      ;LINE RESET COMMAND
5885 031570 005077 150474      CLR      @SEL4      ;UNUSED
5886 031574 105077 150474      CLRB    @SEL6      ;UNUSED
5887 031600 004737 031772      JSR     PC,INPDUN      ;TEST THAT INPUT COMPLETED
5888 031604 000207      2$:  RTS     PC
5889 031606 062716 000002      ADD      #2,(SP)
5890 031612 000207      RTS     PC
5891
5892
5893
5894      ;TEST THAT INPUT COMMAND COMPLETD PC+2 = COMPLETED PC = NOT COMPLETED
5895 031614 105037 001160      COMIN: CLRB    $TMP0
5896 031620 105777 150434      1$:  TSTB   @SEL0      ;DID RDY IN DROP?
5897 031624 100004      BPL      2$          ;B=Y
5898 031626 105237 001160      INCB    $TMP0
5899 031632 001372      BNE      1$
5900 031634 000207      RTS     PC          ;COMMAND NOT COMPLETED
5901 031636 062716 000002      2$:  ADD      #2,(SP)
5902 031642 000207      RTS     PC          ;OK
5903
```

```
5904      :RETURN PC+2 = NEW LINE # IN CURLIN, PC=DONE
5905 031644 106337 002313      GETLNO: ASLB CLINE ;SHIFT TO NEXT LINE
5906 031650 005237 002454      GETLIN: INC  CURLIN
5907 031654 105737 002313      TSTB CLINE ;LOOK FOR LINE BIT
5908 031660 001406              BEQ  1$ ;SKIP IF NONE LEFT
5909 031662 100370              BPL  GETLNO ;CHECK AGAIN
5910 031664 106337 002313      ASLB CLINE ;SHIFT
5911 031670 062716 000002      ADD  #2,(SP) ;FOUND LINE
5912 031674 000207              RTS  PC
5913 031676 113737 002312 002313 1$: MOVB LINEB,CLINE ;INIT LINES
5914 031704 012737 177777 002454      MOV  #-1,CURLIN ;RESET
5915 031712 000207              RTS  PC ;EXIT
5916
5917
5918
5919      :THIS ROUTINE WILL ALLOW TIME FOR KMC TO DETECT SYNC
5920 031714 005037 001160      SYNTIM: CLR  $TMP0
5921 031720 005037 001162      CLR  $TMP1
5922 031724 005237 001160      1$: INC  $TMP0
5923 031730 001375              BNE  1$
5924 031732 005137 001162      COM  $TMP1
5925 031736 001372              BNE  1$
5926 031740 000207              RTS  PC
5927
5928
5929      :THIS ROUTINE WILL MONITOR THE REQUEST OUT OF SEL2
5930      :FOR SETTING WITHIN DURATION OF 1 LOOP
5931
5932 031742 005037 001160      REQOUT: CLR  $TMP0
5933 031746 105777 150312      1$: TSTB @SEL2 ;OUTPUT REQUESTED?
5934 031752 100404              BMI  2$ ;B=YES
5935 031754 005237 001160      INC  $TMP0
5936 031760 001372              BNE  1$ ;TRY AGAIN
5937 031762 000207              RTS  PC ;NO OUTPUT REQUESTED
5938 031764 062716 000002      2$: ADD  #2,(SP)
5939 031770 000207              RTS  PC ;OUTPUT IS REQUESTED
5940      :THIS ROUTINE WILL TEST THAT THE INPUT STATUS COMMAND
5941      :WILL COMPLETE (RDY-IN DROPS)
5942 031772 042777 000040 150260 INPDUN: BIC  #40,@SELO
5943 032000 005037 001160      CLR  $TMP0 ;INSTRUCT COMMAND READY
5944 032004 105777 150250      1$: TSTB @SELO ;INPUT DONE?
5945 032010 100004              BPL  2$ ;B=YES
5946 032012 005237 001160      INC  $TMP0
5947 032016 001372              BNE  1$
5948 032020 000207              RTS  PC ;COMMAND NOT ACCEPTED
5949 032022 062716 000002      2$: ADD  #2,(SP)
5950 032026 000207              RTS  PC ;COMMAND DONE
5951
5952      :IF BUFFER REQ. IS LEGAL (CORE AVAIL + WITHIN BOUNDS) MAP KT11
5953      : (IF EXISTANT) AND RETURN TO PC+2 OTHERWISE PC
5954      :ENTER AT -
5955      :
5956      : MAPBFR - IF RECEIVE BUFFER POSSIBLY NEEDING CRC BITS
5957      : MAPBUF - IF ANY OTHER BUFFER
5958 032030 012700 000001      MAPBFR: MOV  #1,R0 ;SET RECEIVE BUFFER FLAG
5959 032034 000401              BR   MAPBFC
```

```
5960 032036 005000      MAPBUF: CLR      R0          ;SET OTHER BUFFER FLAG
5961 032040 005737 027022  MAPBFC: TST      $KT11
5962 032044 100003      BPL      8$          ;B=N
5963 032046 042737 000001 177572      BIC      #1,$SR0      ;SHUT OFF KT11
5964 032054 013737 002500 002502 8$:      MOV      CURBUF,MAPADR ;USE TO READ THE DATA
5965 032062 013702 002476      MOV      CUREA,R2
5966      ;FORM RETURN ADDRESS
5967 032066 013701 002500      MOV      CURBUF,R1
5968 032072 062701 000010      ADD      #10,R1
5969 032076 103001      BCC      10$          ;B=NO CARRY
5970 032100 005202      INC      R2          ;+1 TO THE EA BITS
5971 032102 105737 002311 10$:      TSTB     MODE      ;BOP OR CCP
5972 032106 100020      BPL      1$          ;SKIP IF CCP
5973 032110 062701 000010      ADD      #10,R1      ;CHANGE WORD COUNT TO BYTES
5974 032114 103001      BCC      11$
5975 032116 005202      INC      R2
5976 032120 005700 11$:      TST      R0          ;IS THIS A RECEIVE BUFFER?
5977 032122 001412      BEQ      1$          ;SKIP IF NO
5978 032124 105737 002316      TSTB     CLKDAT      ;EXTERNAL?
5979 032130 100407      BMI      1$          ;SKIP IF INTERNAL
5980 032132 105737 002314      TSTB     CRC32F      ;CRC32 ENABLED?
5981 032136 100004      BPL      1$          ;SKIP IF NOT ENABLED
5982 032140 062701 000004      ADD      #4,R1      ;ADD ADDITIONAL CRC BYTES
5983 032144 103001      BCC      1$
5984 032146 005202      INC      R2
5985 032150 010137 002512 1$:      MOV      R1,RETADR
5986 032154 010205      MOV      R2,R5
5987 032156 005004      CLR      R4
5988 032160 000241      CLC
5989 032162 012700 000002      MOV      #2,R0
5990 032166 006205 14$:      ASR      R5
5991 032170 006004      ROR      R4
5992 032172 005300      DEC      R0
5993 032174 001374      BNE      14$
5994 032176 010437 002510      MOV      R4,RETEA
5995 032202 020237 002340      CMP      R2,XBITS      ;IS REQUESTED BUFFER LEGAL?
5996 032206 003122      BGT      7$          ;B=N DON'T ISSUE THE REQ.
5997 032210 001003      BNE      15$          ;B=THERE FOR LESS THAN
5998 032212 020137 002336      CMP      R1,HADDR      ;THEN 16 BIT ADDR MUST BE LESS OR EQ
5999 032216 101116      BHI      7$          ;IF GREATER OR NOT EQUAL = DONE
6000      ;IF A KT11 AVAILABLE MAP APPROPRIATELY
6001 032220 005037 002524 15$:      CLR      BUFEA
6002 032224 005737 027022      TST      $KT11
6003 032230 100076      BPL      66$          ;NO KT11
6004 032232 012700 000006      MOV      #6,R0
6005 032236 012701 172300      MOV      #KIPDR0,R1
6006 032242 012721 077406 2$:      MOV      #77406,(R1)+ ;LOAD PDR.
6007 032246 005300      DEC      R0
6008 032250 001374      BNE      2$
6009 032252 012700 000001      MOV      #1,R0          ;DISABLE 6
6010 032256 012721 077400 3$:      MOV      #77400,(R1)+ ;NO ACCESS
6011 032262 005300      DEC      R0
6012 032264 000074      BNE      3$
6013 032266 012701 077406      MOV      #77406,(R1) ;PG 7
6014 032272 013702 002476      MOV      CUREA,R2
6015 032276 013701 002500      MOV      CURBUF,R1
```

```
6016 032302 000241          CLC
6017 032304 012700 000012    MOV    #10.,R0          ;KT11 FORMAT
6018 032310 006301          4$: ASL    F1
6019 032312 006102          ROL    R2
6020 032314 005300          DEC    R0
6021 032316 001374          BNE    4$
6022 032320 042702 000077    BIC    #77,R2          ;ASSURE A 4K BOUND
6023 032324 012700 172340    MOV    #KIPAR0,R0        ;MAP THE PAGES
6024 032330 005001          CLR    R1
6025 032332 012704 000005    MOV    #5,R4
6026 032336 010120          5$: MOV    R1,(R0)+        ;DO PAGES 0,1,2,3,+4 (MAIN PROGRAM)
6027 032340 062701 000200    ADD    #200,R1          ;4K
6028 032344 005304          DEC    R4                ;DONE?
6029 032346 001373          BNE    5$                ;B=N
6030 032350 010210          MOV    R2,(R0)          ;MAP PAGE 5 WITH BUFFER (REC) ADDRESS
6031 032352 012737 177600 172356 MOV    #177600,#KIPAR7        ;I/O PG.
6032 032360 013737 002532 002502 MOV    BADDR,MAPADR
6033 032366 062737 020000 002502 ADD    #20000,MAPADR
6034 032374 052737 000001 177572 BIS    #1,#SRO          ;KT11 ON
6035 032402 000241          CLC                    ;ALIGN EA BITS
6036 032404 012700 000003    MOV    #3,R0
6037 032410 013701 002476    MOV    CUREA,R1
6038 032414 006001          65$: ROR    R1            ;ALIGN EA BITS FOR CSR FORMAT
6039 032416 005300          DEC    R0
6040 032420 001375          BNE    65$
6041 032422 010137 002524    MOV    R1,BUFEA          ;PHYSICAL CSR EA FORMAT (15-14)
6042 032426 012737 010010 002526 66$: MOV    #10010,BUFCNT ;LENGTH OF BUFFER
6043 032434 043737 002330 002526 BIC    MODMSK,BUFCNT
6044 032442 013737 002500 002522 MOV    CURBUF,BUFCNT ;BUFFER ADDRESS (PHYSICAL 0-15)
6045 032450 062716 000002    ADD    #2,(SP)
6046 032454 000207          7$: RTS    PC
6047
6048          ;THIS ROUTINE WILL INITIALIZE THE "CURBUF" BY CLEARING IT
6049          ;AND WRITING THE TRANSMIT BUFFER(TBUF)
6050          ;WITH A 10 CHARACTER INCREMENT PATTERN
6051
6052 032456 013700 002502    INTREC: MOV    MAPADR,R0        ;CURRENTLY MAPPED BUFFER
6053 032462 005020 1$:      CLR    (R0)+          ;CLEAR ALL 127. WORDS
6054 032464 032700 000177    BIT    #177,R0          ;DONE
6055 032470 001374          BNE    1$
6056          ;WRITE THE TRANSMIT BUFFER
6057 032472 012702 052260    MOV    #TBUF,R2          ;SET STARTING ADDRESS
6058 032476 012701 000010    MOV    #10,R1           ;SET LENGTH OF BUFFER
6059 032502 004737 032542    JSR    PC,MEFIL          ;FILL BUFFER
6060 032506 000207          RTS    PC
6061
6062          ;THIS ROUTINE WILL INITIALIZE THE "CURBUF" BY WRITING IT WITH
6063          ;A 10 CHARACTER INCREMENT PATTERN THEN CLEARING
6064          ;THE RECEIVE BUFFER (RBUF)
6065
6066 032510 013702 002502    INTTRM: MOV    MAPADR,R2        ;CURRENTLY MAPPED BUFFER
6067 032514 012701 000010    MOV    #10,R1           ;LENGTH OF BUFFER
6068 032520 004737 032542    JSR    PC,MEFIL          ;FILL BUFFER WITH INCREMENTING PATTERN
6069          ;CLEAR THE RECEIVE BUFFER
6070 032524 012700 052660    MOV    #RBUF,R0
6071 032530 005020 2$:      CLR    (R0)+
```

```
6072 032532 020027 053056      CMP      R0,#RBUF+126.  ;DONE?
6073 032536 001374      BNE      2$          ;B=N
6074 032540 000207      RTS      PC
6075
6076      ;THIS ROUTINE WILL WRITE AN INCREMENTING PATTERN IN THE
6077      ;NAMED BUFFER.  IN BOP MODE ALL PATTERNS WILL BE USED
6078      ;IN CCP MODE ONLY PATTERNS 40-175 WILL BE USED.
6079      ;ENTER WITH      R1=LENGTH OF BUFFER IN BYTES
6080      ;                  R2=STARTING ADDRESS OF BUFFER
6081
6082 032542 010146      MEMFIL: MOV      R1,-(SP)
6083 032544 010246      MOV      R2,-(SP)
6084 032546 105737 002311      TSTB     MODE
6085 032552 100412      BMI      1$
6086 032554 012737 000175 032710      MOV      #175,20$      ;CCP HIGH LIMIT=175
6087 032562 012737 000040 032712      MOV      #40,21$      ;CCP LOW LIMIT=40
6088 032570 112722 000201      MOV      #201,(R2)+      ;SET 1ST CHAR TO SOH
6089 032574 005301      DEC      R1          ;REDUCE REMAINING CHAR COUNT
6090 032576 000406      BR       2$
6091 032600 012737 000377 032710 1$: MOV      #377,20$      ;BOP HIGH LIMIT=377
6092 032606 012737 000000 032712      MOV      #0,21$      ;BOP LOW LIMIT=0
6093 032614 023737 032712 032714 2$: CMP      21$,22$      ;CURRENT VALUE BELOW ALLOWABLE RANGE?
6094 032622 101004      BHI      3$          ;B=YES
6095 032624 023737 032710 032714      CMP      20$,22$      ;CURRENT VALUE ABOVE ALLOWABLE RANGE
6096 032632 101003      BHI      4$          ;B=NO
6097 032634 013737 032712 032714 3$: MOV      21$,22$      ;SET VALUE TO LOWEST POSSIBLE
6098 032642 113722 032714 4$: MOV      22$,(R2)+      ;STORE PATTERN INTO BUFFER
6099 032646 005301      DEC      R1          ;COUNT DOWN NUMBER OF BYTES
6100 032650 003407      BLE      6$          ;B=DONE
6101 032652 105237 032714      INCB     22$      ;INCREMENT DATA PATTERN
6102 032656 023737 032710 032714      CMP      20$,22$      ;HAVE WE PASSED HIGHEST VALUE?
6103 032664 103366      BHS      4$          ;B=CONTINUE WITH THIS PATTERN
6104 032666 000762      BR       3$          ;START OVER WITH LOWEST VALUE
6105 032670 105737 002311 6$: TSTB     MODE      ;WHEN COMPLETE , CHECK MODE
6106 032674 100402      BMI      7$          ;SKIP IF BOP
6107 032676 112742 000003      MOV      #3,-(R2)      ;IF CCP REWRITE LAST BYTE AS ETX
6108 032702 012602 7$: MOV      (SP)+,R2
6109 032704 012601      MOV      (SP)+,R1
6110 032706 000207      RTS      PC
6111 032710 000000 20$: .WORD     0          ;HIGHEST ALLOWED PATTERN
6112 032712 000000 21$: .WORD     0          ;LOWEST ALLOWED PATTERN
6113 032714 000000 22$: .WORD     0          ;CURRENT DATA PATTERN
6114
6115      ;THIS ROUTINE WILL TEST THE DATA TRANSFERRED TO THE RECEIVE BUFFER
6116      ;TWO ENTRY POINTS ALLOW FOR THE CURRENT 'RECEIVE' BUFFER
6117
6118 032716 012701 052660      TSTREC: MOV      #RBUF,R1      ;TABLE 'RBUF' IS THE RECEIVE BUFFER
6119 032722 000402      BR       .+6
6120 032724 013701 002502      TSTMAP: MOV      MAPADR,R1      ;'MAPBUF' CONTAINS THE ADDRESSED RECEIVE BUFFER
6121 032730 012737 000003 002462      MOV      #3,SHBE
6122 032736 112137 002464 1$: MOV      (R1)+,WAS
6123 032742 142737 000200 002464      BICB     #200,WAS      ;ELIMINATE THE PARITY BIT
6124 032750 123737 002462 002464      CMPB     SHBE,WAS      ;DOES DATA COMPARE?
6125 032756 001010      BNE      2$          ;REPORT ERROR
6126 032760 005237 002462      INC      SHBE
6127 032764 022737 000177 002462      CMP      #177,SHBE      ;DONE?
```

6128	032772	001361				BNE	1\$		;B=NO GET NEXT VALUE
6129	032774	062716	000002			ADD	#2,(SP)		
6130	033000	000207				RTS	PC		
6131	033002	201	101	127	2\$:	TERBUF:	.BYTE	201,101,127,3	;SOH & ETX
6132	033005	003							
6133	033006	201	101	127			.BYTE	201,101,127,27	;SOH & ETB
6134	033011	027							
6135	033012	202	101	127			.BYTE	202,101,127,3	;STX & ETX
6136	033015	003							
6137	033016	202	101	127			.BYTE	202,101,127,27	;STX & ETB
6138	033021	027							
6139									
6140	033022	201	101	101	T7BUF:		.BYTE	201,101,101,3	
6141	033025	003							
6142	033026	226	202	101	LRCBUF:		.BYTE	226,202,101,127,127,127,3,376	
6143	033031	127	127	127					
6144	033034	003	376						
6145	033036	226	202	101	PERBUF:		.BYTE	226,202,101,126,127,127,3,376	
6146	033041	126	127	127					
6147	033044	003	376						
6148	033046	000000					.WORD	0	
6149									
6150									
6151	033050	044506	046522	040527	EM01:		.ASCIZ	/FIRMWARE INITIALIZE TEST/	
6152	033056	042522	044440	044516					
6153	033064	044524	046101	055111					
6154	033072	020105	042524	052123					
6155	033100	000							
6156	033101	122	040505	054504	EM02:		.ASCIZ	/READY IN BIT (SELO-BITS) TEST/	
6157	033106	044440	020116	044502					
6158	033114	020124	051450	046105					
6159	033122	026460	044502	032524					
6160	033130	020051	042524	052123					
6161	033136	000							
6162	033137	116	026517	052517	EM03:		.ASCIZ	/NO-OUTPUT READY TEST/	
6163	033144	050124	052125	051040					
6164	033152	040505	054504	052040					
6165	033160	051505	000124						
6166	033164	044514	042516	044440	EM04:		.ASCIZ	/LINE INITIALIZE TEST/	
6167	033172	044516	044524	046101					
6168	033200	055111	020105	042524					
6169	033206	052123	000						
6170	033211	122	041505	044505	EM05:		.ASCIZ	/RECEIVE BUFFER-IN TEST/	
6171	033216	042526	041040	043125					
6172	033224	042506	026522	047111					
6173	033232	052040	051505	000124					
6174	033240	051124	047101	046523	EM06:		.ASCIZ	/TRANSMIT BUFFER-IN TEST/	
6175	033246	052111	041040	043125					
6176	033254	042506	026522	047111					
6177	033262	052040	051505	000124					
6178	033270	047111	052111	040511	EM07:		.ASCIZ	!INITIAL I/O COMPLETION TEST!	
6179	033276	020114	027511	020117					
6180	033304	047503	050115	042514					
6181	033312	044524	047117	052040					
6182	033320	051505	000124						
6183	033324	044514	042516	047440	EM10:		.ASCIZ	/LINE OVERFLOW TEST/	

CZKMEAO DMS11-DA DYNAMIC
CZKMEA.P11 20-OCT-81 17:05

MACY11 30A(1052) 21-OCT-81 08:50 C 10
ROUTINE TO SIZE MEMORY PAGE 119

SEQ 0119

6184	033332	042526	043122	047514	
6185	033340	020127	042524	052123	
6186	033346	000			
6187	033347	114	047111	020105	EM11: .ASCIIZ /LINE RESET OF RECEIVE BUFFERS/
6188	033354	042522	042523	020124	
6189	033362	043117	051040	041505	
6190	033370	044505	042526	041040	
6191	033376	043125	042506	051522	
6192	033404	000			
6193	033405	114	047111	020105	EM12: .ASCIIZ /LINE RESET OF TRANSMIT BUFFERS/
6194	033412	042522	042523	020124	
6195	033420	043117	052040	040522	
6196	033426	051516	044515	020124	
6197	033434	052502	043106	051105	
6198	033442	000123			
6199	033444	040442	047502	052122	EM13: .ASCIIZ /'ABORT IN' TO TRANSMIT BUFFERS/
6200	033452	044440	021116	052040	
6201	033460	020117	051124	047101	
6202	033466	046523	052111	041040	
6203	033474	043125	042506	051522	
6204	033502	000			
6205	033503	042	041101	051117	EM14: .ASCIIZ /'ABORT OUT' TO RECEIVE BUFFERS/
6206	033510	020124	052517	021124	
6207	033516	052040	020117	042522	
6208	033524	042503	053111	020105	
6209	033532	052502	043106	051105	
6210	033540	000123			
6211	033542	040442	047502	052122	EM15: .ASCIIZ /'ABORT OUT' TO TRANSMIT BUFFERS/
6212	033550	047440	052125	020042	
6213	033556	047524	052040	040522	
6214	033564	051516	044515	020124	
6215	033572	052502	043106	051105	
6216	033600	000123			
6217	033602	040442	047502	052122	EM16: .ASCIIZ /'ABORT IN' TO TRANSMIT BUFFERS/
6218	033610	044440	021116	052040	
6219	033616	020117	051124	047101	
6220	033624	046523	052111	041040	
6221	033632	043125	042506	051522	
6222	033640	000			
6223	033641	121	042525	044525	EM17: .ASCIIZ /QUEUING OF ALT. TRANSMIT BUFFERS/
6224	033646	043516	047440	020106	
6225	033654	046101	027124	052040	
6226	033662	040522	051516	044515	
6227	033670	020124	052502	043106	
6228	033676	051105	000123		
6229	033702	052521	052505	047111	EM20: .ASCIIZ /QUEUING OF ALT. RECEIVE BUFFERS/
6230	033710	020107	043117	040440	
6231	033716	052114	020056	042522	
6232	033724	042503	053111	020105	
6233	033732	052502	043106	051105	
6234	033740	000123			
6235	033742	046101	042524	047122	EM21: .ASCIIZ /ALTERNATE BUFFER DYNAMICS TEST/
6236	033750	052101	020105	052502	
6237	033756	043106	051105	042040	
6238	033764	047131	046501	041511	
6239	033772	020123	042524	052123	

6240	034000	000			
6241	034001	116	046530	051440	EM22: .ASCIZ /NXM STATUS TEST/
6242	034006	040524	052524	020123	
6243	034014	042524	052123	000	
6244	034021	105	041124	023040	EM23: .ASCIZ /ETB & ETX TERMINATION TEST/
6245	034026	042440	054124	052040	
6246	034034	051105	044515	040516	
6247	034042	044524	047117	052040	
6248	034050	051505	000124		
6249	034054	051114	020103	051105	EM24: .ASCIZ /LRC ERROR GENERATION TEST/
6250	034062	047522	020122	042507	
6251	034070	042516	040522	044524	
6252	034076	047117	052040	051505	
6253	034104	000124			
6254	034106	040520	044522	054524	EM25: .ASCIZ /PARITY ERROR GENERATION TEST/
6255	034114	042440	051122	051117	
6256	034122	043440	047105	051105	
6257	034130	052101	047511	020116	
6258	034136	042524	052123	000	
6259	034143	103	040510	040522	EM26: .ASCIZ /CHARACTER COUNT TEST/
6260	034150	052103	051105	041440	
6261	034156	052517	052116	052040	
6262	034164	051505	000124		
6263	034170	047514	020127	031461	EM27: .ASCIZ /LOW 13 BIT TRM + REC TEST/
6264	034176	041040	052111	052040	
6265	034204	046522	025440	051040	
6266	034212	041505	052040	051505	
6267	034220	000124			
6268	034222	042522	042503	053111	EM30: .ASCIZ /RECEIVE BUFF. HIGH ORDER BITS (14-17) TEST/
6269	034230	020105	052502	043106	
6270	034236	020056	044510	044107	
6271	034244	047440	042122	051105	
6272	034252	041040	052111	020123	
6273	034260	030450	026464	033461	
6274	034266	020051	042524	052123	
6275	034274	000			
6276	034275	124	040522	051516	EM31: .ASCIZ /TRANSMIT BUFF HIGH ORDER BITS (14-17) TEST/
6277	034302	044515	020124	052502	
6278	034310	043106	044040	043511	
6279	034316	020110	051117	042504	
6280	034324	020122	044502	051524	
6281	034332	024040	032061	030455	
6282	034340	024467	052040	051505	
6283	034346	000124			
6284	034350	04051	040524	052040	EM32: .ASCIZ /DATA TRANSFER TEST/
6285	034356	04052	051516	042506	
6286	034364	020122	042524	052123	
6287	034372	000			
6288	034373	111	050116	052125	EM33: .ASCIZ /INPUT INTERRUPT TEST/
6289	034400	044440	052116	051105	
6290	034406	052522	052120	052040	
6291	034414	051505	000124		
6292	034420	052517	050124	052125	EM34: .ASCIZ /OUTPUT INTERRUPT TEST/
6293	034426	044440	052116	051105	
6294	034434	052522	052120	052040	
6295	034442	051505	000124		

Line #	Unit #	Test #	Fail #	Dev. ADR.	SHBE	WAS	CSR0	CSR2	CSR4	CSR6
6296	034446	047125	052111	021440	DH1:	.ASCIZ	/UNIT #	TEST #	FAIL #	DEV.ADR./
6297	034454	020040	042524	052123						
6298	034462	021440	020040	040506						
6299	034470	046111	021440	020040						
6300	034476	042504	027126	042101						
6301	034504	027122	000							
6302	034507	125	044516	020124	DH2:	.ASCIZ	/UNIT #	TEST #	FAIL #	SHBE WAS/
6303	034514	020043	052040	051505						
6304	034522	020124	020043	043040						
6305	034530	044501	020114	020043						
6306	034536	051440	041110	020105						
6307	034544	020040	053440	051501						
6308	034552	000								
6309	034553	125	044516	020124	DH3:	.ASCIZ	/UNIT #	TEST #	FAIL #	LINE #/
6310	034560	020043	052040	051505						
6311	034566	020124	020043	043040						
6312	034574	044501	020114	020043						
6313	034602	046040	047111	020105						
6314	034610	000043								
6315	034612	047125	052111	021440	DH4:	.ASCIZ	/UNIT #	TEST #	FAIL #	LINE # CSR0 CSR2 CSR4 CSR6/
6316	034620	020040	042524	052123						
6317	034626	021440	020040	040506						
6318	034634	046111	021440	020040						
6319	034642	044514	042516	021440						
6320	034650	020040	051503	030122						
6321	034656	020040	020040	051503						
6322	034664	031122	020040	020040						
6323	034672	051503	032122	020040						
6324	034700	020040	051503	033122						
6325	034706	000								
6326	034707	125	044516	020124	DH5:	.ASCIZ	/UNIT #	TEST #	FAIL #	LINE # SHBE CSR0 CSR2 CSR4 CSR6/
6327	034714	020043	052040	051505						
6328	034722	020124	020043	043040						
6329	034730	044501	020114	020043						
6330	034736	046040	047111	020105						
6331	034744	020043	051440	041110						
6332	034752	020105	020040	041440						
6333	034760	051123	020060	020040						
6334	034766	041440	051123	020062						
6335	034774	020040	041440	051123						
6336	035002	020064	020040	041440						
6337	035010	051123	0							

LINE #	UNIT #	TEST #	FAIL #	SHBE	CSR0	CSR2	CSR4	CSR6
6352	035130	026524	052502	043106				
6353	035136	020040	026522	052502				
6354	035144	043106	020040	044123				
6355	035152	042502	020040	020040				
6356	035160	040527	000123					
6357	035164	047125	052111	021440	DH10:	.ASCIZ	/UNIT #	TEST #
6358	035172	020040	042524	052123		FAIL #	LINE #	CUR-EA
6359	035200	021440	020040	040506		CUR-ADR	SHBE	WAS/
6360	035206	046111	021440	020040				
6361	035214	044514	042516	021440				
6362	035222	020040	052503	026522				
6363	035230	040505	020040	052503				
6364	035236	026522	042101	020122				
6365	035244	044123	042502	020040				
6366	035252	020040	040527	000123				
6367	035260	047125	052111	021440	DH11:	.ASCIZ	/UNIT #	TEST #
6368	035266	020040	042524	052123		FAIL #	SHBE	WAS
6369	035274	021440	020040	040506		CSR0	CSR2	CSR4
6370	035302	046111	021440	020040		CSR6/		
6371	035310	044123	042502	020040				
6372	035316	020040	040527	020123				
6373	035324	020040	020040	051503				
6374	035332	030122	020040	020040				
6375	035340	051503	031122	020040				
6376	035346	020040	051503	032122				
6377	035354	020040	020040	051503				
6378	035362	033122	000					
6379	035365	125	044516	020124	DH12:	.ASCIZ	/UNIT #	TEST #
6380	035372	020043	052040	051505		FAIL #	WAS	CSR0
6381	035400	020124	020043	043040		CSR2	CSR4	CSR6/
6382	035406	044501	020114	020043				
6383	035414	053440	051501	020040				
6384	035422	041440	051123	020060				
6385	035430	020040	051503	031122				
6386	035436	020040	041440	051123				
6387	035444	020064	041440	051123				
6388	035452	000066						
6389	035454	047125	052111	021440	DH13:	.ASCIZ	/UNIT #	TEST #
6390	035462	020040	042524	052123		FAIL #	LINE #	T-BUFF
6391	035470	021440	020040	040506		SHBE	CSR2	CSR4
6392	035476	046111	021440	020040		CSR6/		
6393	035504	044514	042516	021440				
6394	035512	020040	026524	052502				
6395	035520	043106	020040	044123				
6396	035526	042502	020040	020040				
6397	035534	051503	031122	020040				
6398	035542	020040	051503	032122				
6399	035550	020040	020040	051503				
6400	035556	033122	000					
6401		035562						
6402	035562	002456	002460	002452	DT1:	.EVEN		
6403	035570	002260	000000			.WORD	CUNIT,TSTNUM,ERNUM,SELO,0	
6404	035574	002456	002460	002452	DT2:	.WORD		

CZKMEAO DMS11-DA DYNAMIC
CZKMEA.P11 20-OCT-81 17:05

MACY11 30A(1052) 21-OCT-81 08:50 G 10
ROUTINE TO SIZE MEMORY PAGE 123

SEQ 0123

6408	035622	002456	002460	002452	DT4:	.WORD	CUNIT,TSTNUM,ERNUM,CURLIN,CSR0,CSR2,CSR4,CSR6,0
6409	035630	002454	002342	002344			
6410	035636	002346	002350	000000			
6411	035644	002456	002460	002452	DT5:	.WORD	CUNIT,1STNUM,ERNUM,CURLIN,SHBE,CSR0,CSR2,CSR4,CSR6,0
6412	035652	002454	002462	002342			
6413	035660	002344	002346	002350			
6414	035666	000000					
6415	035670	002456	002460	002452	DT6:	.WORD	CUNIT,TSTNUM,ERNUM,RECBUF,SHBE,WAS,0
6416	035676	002506	002462	002464			
6417	035704	000000					
6418	035706	002456	002460	002452	DT7:	.WORD	CUNIT,TSTNUM,ERNUM,XOUT,ROUT,SHBE,WAS,0
6419	035714	002466	002470	002462			
6420	035722	002464	000000				
6421	035726	002456	002460	002452	DT10:	.WORD	CUNIT,TSTNUM,ERNUM,CURLIN,CUREA,CURBUF,SHBE,WAS,0
6422	035734	002454	002476	002500			
6423	035742	002462	002464	000000			
6424	035750	002456	002460	002452	DT11:	.WORD	CUNIT,TSTNUM,ERNUM,SHBE,WAS,CSR0,CSR2,CSR4,CSR6,0
6425	035756	002462	002464	002342			
6426	035764	002344	002346	002350			
6427	035772	000000					
6428	035774	002456	002460	002452	DT12:	.WORD	CUNIT,TSTNUM,ERNUM,WAS,CSR0,CSR2,CSR4,CSR6,0
6429	036002	002464	002342	002344			
6430	036010	002346	002350	000000			
6431	036016	002456	002460	002452	DT13:	.WORD	CUNIT,TSTNUM,ERNUM,CURLIN,XOUT,SHBE,CSR2,CSR4,CSR6,0
6432	036024	002454	002466	002462			
6433	036032	002344	002346	002350			
6434	036040	000000					
6435	036042	000001	000000	000000	DF1:	.WORD	1,0,0,0,0,0,0
6436	036050	000000	000000	000000			
6437	036056	000000					
6438	036060				CCPFRM:		
6439	036060	016777	123200	060520	.WORD		016777,123200,060520,103405,100400,000423,063230,000401
6440	036066	103405	100400	000423			
6441	036074	063230	000401				
6442	036100	063233	000402	063232	.WORD		063233,000402,063232,000777,063231,010001,004002,016403
6443	036106	000777	063231	010001			
6444	036114	004002	016403				
6445	036120	016403	004003	016601	.WORD		016403,004003,016601,016602,016777,016626,016427,016403
6446	036126	016602	016777	016626			
6447	036134	016427	016403				
6448	036140	016406	016634	016625	.WORD		016406,016634,016625,016422,016436,016421,016405,016430
6449	036146	016422	016436	016421			
6450	036154	016405	016430				
6451	036160	016607	016435	016424	.WORD		016607,016435,016424,000400,061220,061222,110422,000560
6452	036166	000400	061220	061222			
6453	036174	110422	000560				
6454	036200	063236	004400	073237	.WORD		063236,004400,073237,062231,047634,102473,020661,106532
6455	036206	062231	047634	102473			
6456	036214	020661	106532				
6457	036220	102076	004420	073016	.WORD		102076,004420,073016,047634,020660,113745,113213,004420
6458	036226	047634	020660	113745			
6459	036234	113213	004420				
6460	036240	073417	103446	100453	.WORD		073417,103446,100453,004420,063016,100470,002012,000400
6461	036246	004420	063016	100470			
6462	036254	002012	000400				
6463	036260	063223	057235	023601	.WORD		063223,057235,023601,063222,061620,061620,061620,061620

CZKMEAO DMS11-DA DYNAMIC
CZKMEA.P11 20-OCT-81 17:05

MACY11 30A(1052) 21-OCT-81 08:50 H 10
ROUTINE TO SIZE MEMORY PAGE 124

SEQ 0124

6464	036266	063222	061620	061620	
6465	036274	061620	061620		
6466	036300	063721	061620	061620	.WORD 063721,061620,061620,063721,061620,063721,140632,102140
6467	036306	063721	061620	063721	
6468	036314	140632	102140		
6469	036320	010003	004003	054362	.WORD 010003,004003,054362,101544,054362,101544,054362,101461
6470	036326	101544	054362	101544	
6471	036334	054362	101461		
6472	036340	054362	101461	014412	.WORD 054362,101461,014412,077223,054362,101553,063163,101134
6473	036346	077223	054362	101553	
6474	036354	063163	101134		
6475	036360	060602	063223	000402	.WORD 060602,063223,000402,104543,070217,064214,074615,107553
6476	036366	104543	070217	064214	
6477	036374	074615	107553		
6478	036400	016575	016400	100772	.WORD 016575,016400,100772,070217,064214,014414,002562,070017
6479	036406	070217	064214	014414	
6480	036414	002562	070017		
6481	036420	062602	100461	002517	.WORD 062602,100461,002517,000414,070017,040362,101572,000410
6482	036426	000414	070017	040362	
6483	036434	101572	000410		
6484	036440	043223	104543	043223	.WORD 043223,104543,043223,000401,104572,102201,016606,060615
6485	036446	000401	104572	102201	
6486	036454	016606	060615		
6487	036460	100724	016517	014420	.WORD 100724,016517,014420,043223,104543,104543,102323,004003
6488	036466	043223	104543	104543	
6489	036474	102323	004003		
6490	036500	010006	054362	101461	.WORD 010006,054362,101461,054362,101713,054362,101717,003004
6491	036506	054362	101713	054362	
6492	036514	101717	003004		
6493	036520	054362	101704	063164	.WORD 054362,101704,063164,101220,070077,064214,060615,107553
6494	036526	101220	070077	064214	
6495	036534	060615	107553		
6496	036540	014775	077225	060615	.WORD 014775,077225,060615,103241,060602,062223,000402,063221
6497	036546	103241	060602	062223	
6498	036554	000402	063221		
6499	036560	104424	062602	104423	.WORD 104424,062602,104423,016517,054362,101650,000774,104416
6500	036566	016517	054362	101650	
6501	036574	000774	104416		
6502	036600	000403	104416	016517	.WORD 000403,104416,016517,054362,101657,000774,104416,000402
6503	036606	054362	101657	000774	
6504	036614	104416	000402		
6505	036620	104416	002606	000414	.WORD 104416,002606,000414,070017,040362,101670,043222,100624
6506	036626	070017	040362	101670	
6507	036634	043222	100624		
6508	036640	000401	043223	104572	.WORD 000401,043223,104572,023301,000776,063661,062234,060461
6509	036646	023301	000776	063661	
6510	036654	062234	060461		
6511	036660	062234	000772	063225	.WORD 062234,000772,063225,104476,070077,064234,002661,000414
6512	036666	104476	070077	064234	
6513	036674	002661	000414		
6514	036700	070017	062602	100461	.WORD 070017,062602,100461,070077,064214,002643,100722,070077
6515	036706	070077	064214	002643	
6516	036714	100722	070077		
6517	036720	064214	002652	063163	.WORD 064214,002652,063163,074615,056722,107553,103372,056222
6518	036726	074615	056722	107553	
6519	036734	103372	056222		

CZKMEAO DMS11-DA DYNAMIC
CZKMEA.P11 20-OCT-81 17:05

MACY11 30A(1052) 21-OCT-81 08:50 I 10
ROUTINE TO SIZE MEMORY PAGE 125

SEQ 0125

6520	036740	060602	062223	042226	.WORD	060602,062223,042226,056412,101362,056227,057222,123221
6521	036746	056412	101362	056227		
6522	036754	057222	123221			
6523	036760	054661	060702	061231	.WORD	054661,060702,061231,055230,060603,101752,056411,101352
6524	036766	055230	060603	101752		
6525	036774	056411	101352			
6526	037000	056411	105454	000420	.WORD	056411,105454,000420,070217,062735,120600,102355,120620
6527	037006	070217	062735	120600		
6528	037014	102355	120620			
6529	037020	106007	100461	042227	.WORD	106007,100461,042227,056413,101366,100736,043222,000404
6530	037026	056413	101366	100736		
6531	037034	043222	000404			
6532	037040	076402	100737	076602	.WORD	076402,100737,076602,000411,070017,060603,105403,056411
6533	037046	000411	070017	060603		
6534	037054	105403	056411			
6535	037060	105003	056411	105454	.WORD	105003,056411,105454,070217,000420,062735,100461,063225
6536	037066	070217	000420	062735		
6537	037074	100461	063225			
6538	037100	000500	060665	061231	.WORD	000500,060665,061231,000773,063225,104476,063225,060615
6539	037106	000773	063225	104476		
6540	037114	063225	060615			
6541	037120	107076	000403	070017	.WORD	107076,000403,070017,003001,056222,042226,056401,105044
6542	037126	003001	056222	042226		
6543	037134	056401	105044			
6544	037140	056227	057222	123221	.WORD	056227,057222,123221,054661,060702,061231,055230,120600
6545	037146	054661	060702	061231		
6546	037154	055230	120600			
6547	037160	106037	120620	106007	.WORD	106037,120620,106007,104476,042227,056413,105050,104431
6548	037166	104476	042227	056413		
6549	037174	105050	104431			
6550	037200	043222	000404	076402	.WORD	043222,000404,076402,104432,070217,014401,063225,057221
6551	037206	104432	070217	014401		
6552	037214	063225	057221			
6553	037220	057222	070217	000401	.WORD	057222,070217,000401,064334,074615,076601,076602,064214
6554	037226	064334	074615	076601		
6555	037234	076602	064214			
6556	037240	107023	120600	106071	.WORD	107023,120600,106071,120620,106007,104476,004620,070217
6557	037246	120620	106007	104476		
6558	037254	004620	070217			
6559	037260	042733	064214	043221	.WORD	042733,064214,043221,076701,016517,016400,014410,057221
6560	037266	076701	016517	016400		
6561	037274	014410	057221			
6562	037300	057222	057223	004002	.WORD	057222,057223,004002,010000,056413,043227,062407,070207
6563	037306	010000	056413	043227		
6564	037314	062407	070207			
6565	037320	076617	016604	076601	.WORD	076617,016604,076601,076602,076603,062605,100461,000776
6566	037326	076602	076603	062605		
6567	037334	100461	000776			
6568	037340	063225	104476	023301	.WORD	063225,104476,023301,000776,063661,062234,060461,062234
6569	037346	000776	063661	062234		
6570	037354	060461	062234			
6571	037360	057635	107553	104527	.WORD	057635,107553,104527,063225,023301,000776,063661,062234
6572	037366	063225	023301	000776		
6573	037374	063661	062234			
6574	037400	060461	062234	104573	.WORD	060461,062234,104573,060535,107563,070217,000500,076715
6575	037406	060535	107563	070217		

CZKMEAO DMS11-DA DYNAMIC
CZKMEA.P11 20-OCT-81 17:05

MACY11 30A(1052) 21-OCT-81 08:50 J 10
ROUTINE TO SIZE MEMORY PAGE 126

SEQ 0126

6576	037414	000500	076715		
6577	037420	002517	000440	104543	.WORD 002517,000440,104543,023301,000776,063661,062234,060461
6578	037426	023301	000776	063661	
6579	037434	062234	060461		
6580	037440	062234	100461	063225	.WORD 062234,100461,063225,000410,004002,010000,056413,043224
6581	037446	000410	004002	010000	
6582	037454	056413	043224		
6583	037460	062404	070204	076617	.WORD 062404,070204,076617,016405,076603,076602,016400,076605
6584	037466	016405	076603	076602	
6585	037474	016400	076605		
6586	037500	100461	043620	106614	.WORD 100461,043620,106614,110556,002720,004001,016621,002517
6587	037506	110556	002720	004001	
6588	037514	016621	002517		
6589	037520	004600	002517	070017	.WORD 004600,002517,070017,016600,002400,004001,002400,070017
6590	037526	016600	002400	004001	
6591	037534	002400	070017		
6592	037540	002601	000611	070317	.WORD 002601,000611,070317,064617,062231,002635,000673,062232
6593	037546	064617	062231	002635	
6594	037554	000673	062232		
6595	037560	000547	062230	060220	.WORD 000547,062230,060220,000400,002372,062230,060220,000626
6596	037566	000400	002372	062230	
6597	037574	060220	000626		
6598	037600	002232	062230	000421	.WORD 002232,062230,000421,062234,000401,002172,062230,110556
6599	037606	062234	000401	002172	
6600	037614	062230	110556		
6601	037620	123561	107214	061620	.WORD 123561,107214,061620,106272,106671,061620,106711,060521
6602	037626	106272	106671	061620	
6603	037634	106711	060521		
6604	037640	107723	110556	120400	.WORD 107723,110556,120400,061620,106704,070216,047634,043635
6605	037646	061620	106704	070216	
6606	037654	047634	043635		
6607	037660	113556	000500	062715	.WORD 113556,000500,062715,110556,047634,057635,113556,002673
6608	037666	110556	047634	057635	
6609	037674	113556	002673		
6610	037700	110556	070216	043634	.WORD 110556,070216,043634,113156,000420,062714,000412,070016
6611	037706	113156	000420	062714	
6612	037714	000412	070016		
6613	037720	136500	122520	110556	.WORD 136500,122520,110556,123141,000437,063222,000740,063261
6614	037726	123141	000437	063222	
6615	037734	000740	063261		
6616	037740	020700	060662	060701	.WORD 020700,060662,060701,062234,110556,061620,106740,070216
6617	037746	062234	110556	061620	
6618	037754	106740	070216		
6619	037760	047634	112556	043635	.WORD 047634,112556,043635,107750,064333,043635,107750,110556
6620	037766	107750	064333	043635	
6621	037774	107750	110556		
6622	040000	000423	076675	120400	.WORD 000423,076675,120400,061620,112405,016400,136500,136520
6623	040006	061620	112405	016400	
6624	040014	136500	136520		
6625	040020	123160	000700	063260	.WORD 123160,000700,063260,063120,063140,063140,063120,076520
6626	040026	063120	063140	063140	
6627	040034	063120	076520		
6628	040040	123141	000437	076561	.WORD 123141,000437,076561,111002,123160,063260,063160,076660
6629	040046	111002	123160	063260	
6630	040054	063160	076660		
6631	040060	123160	110556	123160	.WORD 123160,110556,123160,076660,110556,137160,014700,077260

CZKMEAO DMS11-DA DYNAMIC
CZKMEA.P11 20-OCT-81 17:05

MACY11 30A(1052) 21-OCT-81 08:50 K 10
ROUTINE TO SIZE MEMORY PAGE 127

SEQ 0127

6632	040066	076660	110556	137160	
6633	040074	014700	077260		
6634	040100	063120	063140	063140	.WORD 063120,063140,063140,063120,136500,136520,076520,016500
6635	040106	063120	136500	136520	
6636	040114	076520	016500		
6637	040120	016421	104770	170610	.WORD 016421,104770,170610,004002,010000,043620,111514,076560
6638	040126	004002	010000	043620	
6639	040134	111514	076560		
6640	040140	014410	043225	062405	.WORD 014410,043225,062405,070205,054620,061620,061620,061620
6641	040146	070205	054620	061620	
6642	040154	061620	061620		
6643	040160	061620	061223	057625	.WORD 061620,061223,057625,112103,113457,054411,061224,111060
6644	040166	112103	113457	054411	
6645	040174	061224	111060		
6646	040200	054411	061225	111061	.WORD 054411,061225,111061,057221,000404,060741,110462,055224
6647	040206	057221	000404	060741	
6648	040214	110462	055224		
6649	040220	055225	054620	061620	.WORD 055225,054620,061620,061620,061620,061620,061227,055226
6650	040226	061620	061620	061620	
6651	040234	061227	055226		
6652	040240	003360	120440	060660	.WORD 003360,120440,060660,063305,000600,060705,061222,060525
6653	040246	063305	000600	060705	
6654	040254	061222	060525		
6655	040260	113510	000604	110611	.WORD 113510,000604,110611,055224,055225,055226,055227,110470
6656	040266	055224	055225	055226	
6657	040274	055227	110470		
6658	040300	123620	000700	061311	.WORD 123620,000700,061311,110501,123400,061620,113120,100447
6659	040306	110501	123400	061620	
6660	040314	113120	100447		
6661	040320	000600	061300	060520	.WORD 000600,061300,060520,113526,000532,110611,000600,061231
6662	040326	113526	000532	110611	
6663	040334	000600	061231		
6664	040340	000575	110611	123400	.WORD 000575,110611,123400,061620,113163,123077,000407,063277
6665	040346	061620	113163	123077	
6666	040354	000407	063277		
6667	040360	063137	063137	063137	.WORD 063137,063137,063137,073537,062231,004600,060417,063236
6668	040366	073537	062231	004600	
6669	040374	060417	063236		
6670	040400	120400	112154	112556	.WORD 120400,112154,112556,104735,106611,104660,123000,000500
6671	040406	104735	106611	104660	
6672	040414	123000	000500		
6673	040420	061260	000423	110611	.WORD 061260,000423,110611,060600,113566,110514,060520,113571
6674	040426	060600	113566	110514	
6675	040434	060520	113571		
6676	040440	100447	000600	061231	.WORD 100447,000600,061231,000575,110611,123400,061620,113201
6677	040446	000575	110611	123400	
6678	040454	061620	113201		
6679	040460	110535	060600	103447	.WORD 110535,060600,103447,110514,123440,103447,000500,061262
6680	040466	110514	123440	103447	
6681	040474	000500	061262		
6682	040500	000423	063230	100447	.WORD 000423,063230,100447,057635,154632,000401,070016,002671
6683	040506	057635	154632	000401	
6684	040514	070016	002671		
6685	040520	000407	070016	120600	.WORD 000407,070016,120600,112222,120620,112337,022420,023002
6686	040526	112222	120620	112337	
6687	040534	022420	023002		

CZKMEAO DMS11-DA DYNAMIC
CZKMEA.P11 20-OCT-81 17:05

MACY11 30A(1052) 21-OCT-81 08:50 L 10
ROUTINE TO SIZE MEMORY PAGE 128

SEQ 0128

6688	040540	110773	000411	070016	.WORD	110773,000411,070016,004000,150632,016640,043222,110773
6689	040546	004000	150632	016640		
6690	040554	043222	110773			
6691	040560	016635	016132	042230	.WORD	016635,016132,042230,070216,000420,062734,110714,063225
6692	040566	070216	000420	062734		
6693	040574	110714	063225			
6694	040600	002172	000401	062230	.WORD	002172,000401,062230,110661,002172,000402,062230,000772
6695	040606	110661	002172	000402		
6696	040614	062230	000772			
6697	040620	063225	007200	070216	.WORD	063225,007200,070216,042733,064214,056700,016400,057221
6698	040626	042733	064214	056700		
6699	040634	016400	057221			
6700	040640	057222	057223	054220	.WORD	057222,057223,054220,054620,112276,003000,004002,010000
6701	040646	054620	112276	003000		
6702	040654	004002	010000			
6703	040660	056413	043224	000410	.WORD	056413,043224,000410,062404,070204,076617,076600,076601
6704	040666	062404	070204	076617		
6705	040674	076600	076601			
6706	040700	076602	076603	062605	.WORD	076602,076603,062605,100467,000420,063221,070216,004000
6707	040706	100467	000420	063221		
6708	040714	070216	004000			
6709	040720	042721	004002	010000	.WORD	042721,004002,010000,056413,043224,000410,062404,070204
6710	040726	056413	043224	000410		
6711	040734	062404	070204			
6712	040740	076617	015401	016400	.WORD	076617,016401,016400,016400,016400,002401,100467,063225
6713	040746	016400	015400	002401		
6714	040754	100467	063225			
6715	040760	000500	060665	061231	.WORD	000500,060665,061231,000773,110647,000777,110647,063225
6716	040766	000773	110647	000777		
6717	040774	110647	063225			
6718	041000	060602	063221	061620	.WORD	060602,063221,061620,061620,061620,061620,063721,061620
6719	041006	061620	061620	061620		
6720	041014	063721	061620			
6721	041020	061620	063721	061620	.WORD	061620,063721,061620,063721,000600,060661,060725,063322
6722	041026	063721	000600	060661		
6723	041034	060725	063322			
6724	041040	000410	070016	042722	.WORD	000410,070016,042722,002132,060602,062230,100467,060220
6725	041046	002132	060602	062230		
6726	041054	100467	060220			
6727	041060	060614	117312	060615	.WORD	060614,117312,060615,103467,016411,002172,000401,062230
6728	041066	103467	016411	002172		
6729	041074	000401	062230			
6730	041100	100467	016465	000402	.WORD	100467,016465,000402,070016,042224,056412,043621,062225
6731	041106	070016	042224	056412		
6732	041114	043621	062225			
6733	041120	076501	043221	115062	.WORD	076501,043221,115062,074461,061230,056411,115033,042411
6734	041126	074461	061230	056411		
6735	041134	115033	042411			
6736	041140	115033	000767	110660	.WORD	115033,000767,110660,120600,116033,120620,112337,034400
6737	041146	120600	116033	120620		
6738	041154	112337	034400			
6739	041160	022420	003201	060361	.WORD	022420,003201,060361,115452,003202,060362,115452,000401
6740	041166	115452	003202	060362		
6741	041174	115452	000401			
6742	041200	070016	002671	002132	.WORD	070016,002671,002132,020400,062230,060220,002172,000400
6743	041206	020400	062230	060220		

CZKMEAO DMS11-DA DYNAMIC
CZKMEA.P11 20-OCT-81 17:05

MACY11 30A(1052) 21-OCT-81 08:50 M 10
ROUTINE TO SIZE MEMORY PAGE 129

SEQ 0129

6744	041214	002172	000400		
6745	041220	062230	100467	000404	.WORD 062230,100467,000404,062401,114423,002505,000405,070016
6746	041226	062401	114423	002505	
6747	041234	000405	070016		
6748	041240	056411	115077	056411	.WORD 056411,115077,056411,115077,000401,070016,002621,000407
6749	041246	115077	000401	070016	
6750	041254	002621	000407		
6751	041260	070016	057222	002400	.WORD 070016,057222,002400,000400,110747,060614,113231,060535
6752	041266	000400	110747	060614	
6753	041274	113231	060535		
6754	041300	113654	000402	070016	.WORD 113654,000402,070016,042224,056412,043621,062225,076501
6755	041306	042224	056412	043621	
6756	041314	062225	076501		
6757	041320	043221	115165	074461	.WORD 043221,115165,074461,061230,056411,115143,042411,115143
6758	041326	061230	056411	115143	
6759	041334	042411	115143		
6760	041340	042411	000401	070016	.WORD 042411,000401,070016,002621,120600,116134,120620,112337
6761	041346	002621	120600	116134	
6762	041354	120620	112337		
6763	041360	023002	000400	110747	.WORD 023002,000400,110747,000401,070016,002570,000407,070016
6764	041366	000401	070016	002570	
6765	041374	000407	070016		
6766	041400	120600	116150	120620	.WORD 120600,116150,120620,112337,022420,023002,000613,060762
6767	041406	112337	022420	023002	
6768	041414	000613	060762		
6769	041420	115563	000600	110747	.WORD 115563,000600,110747,000400,110747,000404,062401,114522
6770	041426	000400	110747	000404	
6771	041434	062401	114522		
6772	041440	060614	113231	060535	.WORD 060614,113231,060535,113654,002505,000405,070016,056411
6773	041446	113654	002505	000405	
6774	041454	070016	056411		
6775	041460	115210	056411	115211	.WORD 115210,056411,115211,043222,000401,070016,002621,114617
6776	041466	043222	000401	070016	
6777	041474	002621	114617		
6778	041500	054220	043222	000613	.WORD 054220,043222,000613,060762,115617,000600,114620,000400
6779	041506	060762	115617	000600	
6780	041514	114620	000400		
6781	041520	110747	002626	000410	.WORD 110747,002626,000410,070016,043222,110773,016745,002172
6782	041526	070016	043222	110773	
6783	041534	016745	002172		
6784	041540	000402	062230	100467	.WORD 000402,062230,100467,060535,113654,000402,070016,042224
6785	041546	060535	113654	000402	
6786	041554	070016	042224		
6787	041560	056412	043621	062225	.WORD 056412,043621,062225,076501,043221,115266,074461,061230
6788	041566	076501	043221	115266	
6789	041574	074461	061230		
6790	041600	056411	111215	042411	.WORD 056411,111215,042411,111215,042411,000401,070016,002626
6791	041606	111215	042411	000401	
6792	041614	070016	002626		
6793	041620	120600	116260	120620	.WORD 120600,116260,120620,112337,023002,110773,000404,062401
6794	041626	112337	023002	110773	
6795	041634	000404	062401		
6796	041640	114646	060535	113654	.WORD 114646,060535,113654,002633,000405,070016,056411,115307
6797	041646	002633	000405	070016	
6798	041654	056411	115307		
6799	041660	056411	115310	043222	.WORD 056411,115310,043222,000401,070016,002626,110773,054220

CZKMEAO DMS11-DA DYNAMIC
CZKMEA.P11 20-OCT-81 17:05

MACV11 30A(1052) 21-OCT-81 08:50 N 10
ROUTINE TO SIZE MEMORY PAGE 130

SEQ 0130

6800	041666	000401	070016	002626	
6801	041674	110773	054220		
6802	041700	043222	110773	002717	.WORD 043222,110773,002717,002172,000401,062230,100467,002732
6803	041706	002172	000401	062230	
6804	041714	100467	002732		
6805	041720	004000	000412	070016	.WORD 004000,000412,070016,002132,042230,040222,000400,002172
6806	041726	002132	042230	040222	
6807	041734	000400	002172		
6808	041740	062230	100467	016740	.WORD 062230,100467,016740,004000,000413,070016,043222,110773
6809	041746	004000	000413	070016	
6810	041754	043222	110773		
6811	041760	016400	002172	000402	.WORD 016400,002172,000402,062230,110714,060535,117751,000401
6812	041766	062230	110714	060535	
6813	041774	117751	000401		
6814	042000	110660	000766	110660	.WORD 110660,000766,110660,000000,000000,000000,000000,000000
6815	042006	000000	000000	000000	
6816	042014	000000	000000		
6817	042020	001007	114761	001027	.WORD 001007,114761,001027,114763,001047,114765,001067,114767
6818	042026	114763	001047	114765	
6819	042034	001067	114767		
6820	042040	001107	114771	001127	.WORD 001107,114771,001127,114773,001147,114775,001167,114777
6821	042046	114773	001147	114775	
6822	042054	001167	114777		
6823	042060				
6824					
6825					
6826					

CCPEND:

BOPW32:

6827	042060				
6828					
6829	042060	016777	123200	060520	.WORD 016777,123200,060520,103405,100400,000401,063230,000401
6830	042066	103405	100400	000401	
6831	042074	063230	000401		
6832	042100	063227	000402	063226	.WORD 063227,000402,063226,000777,063231,010001,004002,016403
6833	042106	000777	063231	010001	
6834	042114	004002	016403		
6835	042120	002403	000400	061220	.WORD 002403,000400,061220,061222,114400,000560,063236,004400
6836	042126	061222	114400	000560	
6837	042134	063236	004400		
6838	042140	073237	062231	047634	.WORD 073237,062231,047634,102451,020660,102054,102606,004420
6839	042146	102451	020660	102054	
6840	042154	102606	004420		
6841	042160	073016	047634	020660	.WORD 073016,047634,020660,107664,107003,004420,073417,103424
6842	042166	107664	107003	004420	
6843	042174	073417	103424		
6844	042200	100431	004420	063016	.WORD 100431,004420,063016,100446,020660,102606,002012,057635
6845	042206	100446	020660	102606	
6846	042214	002012	057635		
6847	042220	103715	140620	000403	.WORD 103715,140620,000403,070017,056406,101070,070077,100477
6848	042226	070017	056406	101070	
6849	042234	070077	100477		
6850	042240	056407	101073	100466	.WORD 056407,101073,100466,043220,000404,076400,100466,016505
6851	042246	043220	000404	076400	
6852	042254	100466	016505		
6853	042260	036600	000420	070217	.WORD 036600,000420,070217,062735,100437,016462,056222,022203
6854	042266	062735	100437	016462	
6855	042274	056222	022203		
6856	042300	056226	056227	057220	.WORD 056226,056227,057220,123221,054661,061311,055230,056411
6857	042306	123221	054661	061311	
6858	042314	055230	056411		
6859	042320	101123	056411	101631	.WORD 101123,056411,101631,000420,070217,062735,120600,102126
6860	042326	000420	070217	062735	
6861	042334	120600	102126		
6862	042340	120620	102374	100437	.WORD 120620,102374,100437,003000,016462,036600,004600,070217
6863	042346	003000	016462	036600	
6864	042354	004600	070217		
6865	042360	042727	064214	043221	.WORD 042727,064214,043221,062701,004002,010000,056407,043224
6866	042366	062701	004002	010000	
6867	042374	056407	043224		
6868	042400	000410	062404	070204	.WORD 000410,062404,070204,016604,076617,076614,076600,016400
6869	042406	016604	076617	076614	
6870	042414	076600	016400		
6871	042420	000401	062620	100437	.WORD 000401,062620,100437,016601,022600,000403,070017,056406
6872	042426	016601	022600	000403	
6873	042434	070017	056406		
6874	042440	101172	100501	056407	.WORD 101172,100501,056407,101175,100571,043220,000404,076400
6875	042446	101175	100571	043220	
6876	042454	000404	076400		
6877	042460	100571	023200	100536	.WORD 100571,023200,100536,016601,036600,100501,002052,057635
6878	042466	016601	036600	100501	
6879	042474	002052	057635		
6880	042500	103621	023605	060525	.WORD 103621,023605,060525,103302,060605,061620,102623,102244
6881	042506	103302	060605	061620	
6882	042514	102623	102244		

CZKMEAO DMS11-DA DYNAMIC
CZKMEA.P11 20-OCT-81 17:05

MACY11 30A(1052) 21-OCT-81 08:50 C 11
ROUTINE TO SIZE MEMORY PAGE 132

SEQ 0132

6883	042520	100437	020200	100715	.WORD	100437,020200,100715,000771,063224,000420,070217,076715
6884	042526	000771	063224	000420		
6885	042534	070217	076715			
6886	042540	100750	120600	102231	.WORD	100750,120600,102231,120620,102374,057625,070077,103242
6887	042546	120620	102374	057625		
6888	042554	070077	103242			
6889	042560	002533	100501	002563	.WORD	002533,100501,002563,100501,002012,074615,023204,103251
6890	042566	100501	002012	074615		
6891	042574	023204	103251			
6892	042600	062604	003200	070217	.WORD	062604,003200,070217,042700,004000,056727,004002,010000
6893	042606	042700	004000	056727		
6894	042614	004002	010000			
6895	042620	000410	056407	043220	.WORD	000410,056407,043220,062400,070200,016604,076617,076614
6896	042626	062400	070200	016604		
6897	042634	076617	076614			
6898	042640	076604	076605	023700	.WORD	076604,076605,023700,102676,002402,100437,002774,000773
6899	042646	102676	002402	100437		
6900	042654	002774	000773			
6901	042660	062274	100437	023300	.WORD	062274,100437,023300,000770,062274,000776,063224,000420
6902	042666	000770	062274	000776		
6903	042674	063224	000420			
6904	042700	070217	076715	000771	.WORD	070217,076715,000771,062274,100750,023300,000770,062274
6905	042706	062274	100750	023300		
6906	042714	000770	062274			
6907	042720	000771	062274	060615	.WORD	000771,062274,060615,061620,103037,070217,000440,062715
6908	042726	061620	103037	070217		
6909	042734	000440	062715			
6910	042740	000404	063225	000410	.WORD	000404,063225,000410,004002,010000,056407,043224,062404
6911	042746	004002	010000	056407		
6912	042754	043224	062404			
6913	042760	070204	016405	076617	.WORD	070204,016405,076617,016400,016400,016400,076605,100437
6914	042766	016400	016400	016400		
6915	042774	076605	100437			
6916	043000	004600	070217	042727	.WORD	004600,070217,042727,064214,043221,076701,000410,004002
6917	043006	064214	043221	076701		
6918	043014	000410	004002			
6919	043020	010000	056407	043220	.WORD	010000,056407,043220,062400,070200,016404,076617,076614
6920	043026	062400	070200	016404		
6921	043034	076617	076614			
6922	043040	016400	016400	062604	.WORD	016400,016400,062604,100437,063204,000500,060664,061231
6923	043046	100437	063204	000500		
6924	043054	060664	061231			
6925	043060	000773	063224	100750	.WORD	000773,063224,100750,057635,103445,147232,016132,056230
6926	043066	057635	103445	147232		
6927	043074	016132	056230			
6928	043100	000420	070216	062735	.WORD	000420,070216,062735,100445,062233,016447,016172,014401
6929	043106	100445	062233	016447		
6930	043114	016172	014401			
6931	043120	062230	042224	056406	.WORD	062230,042224,056406,105037,056225,055230,000402,070016
6932	043126	105037	056225	055230		
6933	043134	000402	070016			
6934	043140	120600	106030	120620	.WORD	120600,106030,120620,106652,036400,036420,100445,042225
6935	043146	106652	036400	036420		
6936	043154	100445	042225			
6937	043160	056407	105043	104425	.WORD	056407,105043,104425,041230,003004,056404,104426,016464
6938	043166	041230	003004	056404		

6939	043174	104426	016464		
6940	043200	002132	056230	000400	.WORD 002132,056230,000400,060220,002172,062230,100445,016574
6941	043206	060220	002172	062230	
6942	043214	100445	016574		
6943	043220	104450	062233	016457	.WORD 104450,062233,016457,104416,016132,054220,056230,042224
6944	043226	104416	016132	054220	
6945	043234	056230	042224		
6946	043240	056406	105123	056225	.WORD 056406,105123,056225,055230,056411,105100,042411,105512
6947	043246	055230	056411	105100	
6948	043254	042411	105512		
6949	043260	070216	000420	076735	.WORD 070216,000420,076735,134600,106114,120620,106652,036400
6950	043266	134600	106114	120620	
6951	043274	106652	036400		
6952	043300	036420	100445	070076	.WORD 036420,100445,070076,016533,120600,106114,120620,106652
6953	043306	016533	120600	106114	
6954	043314	120620	106652		
6955	043320	036400	036420	100445	.WORD 036400,036420,100445,042225,056407,105127,104473,041230
6956	043326	042225	056407	105127	
6957	043334	104473	041230		
6958	043340	003004	056404	104474	.WORD 003004,056404,104474,016574,043221,000411,070016,043622
6959	043346	016574	043221	000411	
6960	043354	070016	043622		
6961	043360	107170	002372	063122	.WORD 107170,002372,063122,063122,063522,062602,060522,062230
6962	043366	063122	063522	062602	
6963	043374	060522	062230		
6964	043400	060220	060601	002132	.WORD 060220,060601,002132,062230,000402,060220,002172,062230
6965	043406	062230	000402	060220	
6966	043414	002172	062230		
6967	043420	070216	054620	061620	.WORD 070216,054620,061620,107166,002615,100445,002603,100445
6968	043426	107166	002615	100445	
6969	043434	002603	100445		
6970	043440	002132	060601	062230	.WORD 002132,060601,062230,100445,054220,054220,057221,000411
6971	043446	100445	054220	054220	
6972	043454	057221	000411		
6973	043460	070016	043222	104541	.WORD 070016,043222,104541,016610,002172,000401,062230,100445
6974	043466	016610	002172	000401	
6975	043474	062230	100445		
6976	043500	002615	002172	000401	.WORD 002615,002172,000401,062230,100445,007200,070216,042727
6977	043506	062230	100445	007200	
6978	043514	070216	042727		
6979	043520	064214	056700	000411	.WORD 064214,056700,000411,070016,043225,004002,010000,056407
6980	043526	070016	043225	004002	
6981	043534	010000	056407		
6982	043540	043224	000410	062404	.WORD 043224,000410,062404,070204,016400,076617,076614,054220
6983	043546	070204	016400	076617	
6984	043554	076614	054220		
6985	043560	076605	023700	061620	.WORD 076605,023700,061620,106646,002401,100445,002774,000775
6986	043566	106646	002401	100445	
6987	043574	002774	000775		
6988	043600	062274	100445	063224	.WORD 062274,100445,063224,000500,060664,061231,000773,063224
6989	043606	000500	060664	061231	
6990	043614	000773	063224		
6991	043620	002172	000404	062230	.WORD 002172,000404,062230,104673,002172,000777,063224,000401
6992	043626	104673	002172	000777	
6993	043634	063224	000401		
6994	043640	062230	043635	103445	.WORD 062230,043635,103445,007200,070216,042727,064214,056700

CZKMEAO DMS11-DA DYNAMIC
CZKMEA.P11 20-OCT-81 17:05

MACY11 30A(1052) 21-OCT-81 08:50 E 11
ROUTINE TO SIZE MEMORY PAGE 134

SEQ 0134

6995	043646	007200	070216	042727	
6996	043654	064214	056700		
6997	043660	004002	010000	056407	.WORD 004002,010000,056407,043220,000410,062400,070200,016400
6998	043666	043220	000410	062400	
6999	043674	070200	016400		
7000	043700	076617	076614	054220	.WORD 076617,076614,054220,016400,062604,100445,043620,106721
7001	043706	016400	062604	100445	
7002	043714	043620	106721		
7003	043720	114673	002740	004001	.WORD 114673,002740,004001,002701,004600,070017,002700,004001
7004	043726	002701	004600	070017	
7005	043734	002700	004001		
7006	043740	002701	000673	062232	.WORD 002701,000673,062232,000407,062230,060220,000400,002372
7007	043746	000407	062230	060220	
7008	043754	000400	002372		
7009	043760	062230	000431	062234	.WORD 062230,000431,062234,114673,123561,107321,061620,112000
7010	043766	114673	123561	107321	
7011	043774	061620	112000		
7012	044000	106754	060521	113472	.WORD 106754,060521,113472,114673,004002,010000,056407,043220
7013	044006	114673	004002	010000	
7014	044014	056407	043220		
7015	044020	000410	062400	070200	.WORD 000410,062400,070200,016405,076617,016400,016400,023260
7016	044026	016405	076617	016400	
7017	044034	016400	023260		
7018	044040	000410	060660	061620	.WORD 000410,060660,061620,061620,061620,076620,002410,114673
7019	044046	061620	061620	076620	
7020	044054	002410	114673		
7021	044060	120400	061620	112436	.WORD 120400,061620,112436,070216,047234,043635,117673,002172
7022	044066	070216	047234	043635	
7023	044074	117673	002172		
7024	044100	000404	062230	000772	.WORD 000404,062230,000772,063224,007200,042727,064214,056700
7025	044106	063224	007200	042727	
7026	044114	064214	056700		
7027	044120	004002	010000	056407	.WORD 004002,010000,056407,043220,000410,062400,070200,016400
7028	044126	043220	000410	062400	
7029	044134	070200	016400		
7030	044140	076617	076614	016400	.WORD 076617,076614,016400,016400,016772,114673,047234,043635
7031	044146	016400	016772	114673	
7032	044154	047234	043635		
7033	044160	117673	000776	023301	.WORD 117673,000776,023301,063661,062234,060461,062234,004600
7034	044166	063661	062234	060461	
7035	044174	062234	004600		
7036	044200	042727	064214	043221	.WORD 042727,064214,043221,076701,014410,004002,010000,056407
7037	044206	076701	014410	004002	
7038	044214	010000	056407		
7039	044220	043220	062400	070200	.WORD 043220,062400,070200,016404,076617,076614,016400,016400
7040	044226	016404	076617	076614	
7041	044234	016400	016400		
7042	044240	016772	114673	123141	.WORD 016772,114673,123141,000437,063222,000740,063261,020700
7043	044246	000437	063222	000740	
7044	044254	063261	020700		
7045	044260	060662	060701	062234	.WORD 060662,060701,062234,114673,120400,061620,112605,070216
7046	044266	114673	120400	061620	
7047	044274	112605	070216		
7048	044300	047634	116673	043635	.WORD 047634,116673,043635,113521,000401,064334,043635,113524
7049	044306	113521	000401	064334	
7050	044314	043635	113524		

7051	044320	114673	060535	113527	.WORD	114673,060535,113527,110514,060535,113527,114673,054220
7052	044326	110514	060535	113527		
7053	044334	114673	054220			
7054	044340	016414	054220	054220	.WORD	016414,054220,054220,136500,136520,123160,000740,063260
7055	044346	136500	136520	123160		
7056	044354	000740	063260			
7057	044360	063120	063140	063140	.WORD	063120,063140,063140,063120,063120,076460,123141,063161
7058	044366	063120	063120	076460		
7059	044374	123141	063161			
7060	044400	123160	000401	063260	.WORD	123160,000401,063260,111156,063160,063260,063161,111166
7061	044406	111156	063160	063260		
7062	044414	063161	111166			
7063	044420	063160	111166	070076	.WORD	063160,111166,070076,002461,000407,070016,076601,000401
7064	044426	002461	000407	070016		
7065	044434	076601	000401			
7066	044440	076660	123160	000436	.WORD	076660,123160,000436,076660,000440,063260,000420,063300
7067	044446	076660	000440	063260		
7068	044454	000420	063300			
7069	044460	000403	060675	070216	.WORD	000403,060675,070216,076700,114673,047634,116673,043635
7070	044466	076700	114673	047634		
7071	044474	116673	043635			
7072	044500	113616	000401	064334	.WORD	113616,000401,064334,043635,113621,114673,060535,113624
7073	044506	043635	113621	114673		
7074	044514	060535	113624			
7075	044520	110611	060535	113624	.WORD	110611,060535,113624,114673,054220,016477,054220,136500
7076	044526	114673	054220	016477		
7077	044534	054220	136500			
7078	044540	136520	123160	000740	.WORD	136520,123160,000740,063260,063120,063140,063140,063120
7079	044546	063260	063120	063140		
7080	044554	063140	063120			
7081	044560	076520	016500	016421	.WORD	076520,016500,016421,123140,000402,076740,111671,111257
7082	044566	123140	000402	076740		
7083	044574	111671	111257			
7084	044600	123160	000401	063260	.WORD	123160,000401,063260,063160,076660,123160,110662,123160
7085	044606	063160	076660	123160		
7086	044614	110662	123160			
7087	044620	000401	076660	000436	.WORD	000401,076660,000436,076660,000403,060675,070217,076675
7088	044626	076660	000403	060675		
7089	044634	070217	076675			
7090	044640	114673	123160	000401	.WORD	114673,123160,000401,063260,063160,111700,076660,110662
7091	044646	063260	063160	111700		
7092	044654	076660	110662			
7093	044660	070077	123560	113305	.WORD	070077,123560,113305,002533,110706,002603,000412,070017
7094	044666	002533	110706	002603		
7095	044674	000412	070017			
7096	044700	110662	060220	060220	.WORD	110662,060220,060220,060220,060220,060220,060220,060220
7097	044706	060220	060220	060220		
7098	044714	060220	060220			
7099	044720	060220	060220	060220	.WORD	060220,060220,060220,060220,060220,060220,060220,060220
7100	044726	060220	060220	060220		
7101	044734	060220	060220			
7102	044740	060220	060220	060220	.WORD	060220,060220,060220,060220,060220,060220,060220,060220
7103	044746	060220	060220	060220		
7104	044754	060220	050220			
7105	044760	060220	060220	060220	.WORD	060220,060220,060220,060220,060220,060220,060220,060220
7106	044766	060220	060220	060220		

7107	044774	060220	060220		
7108	045000	060220	060220	060220	
7109	045006	060220	060220	060220	.WORD 060220,060220,060220,060220,060220,060220,060220,060220
7110	045014	060220	060220		
7111	045020	060220	060220	060220	.WORD 060220,060220,060220,060220,060220,060220,060220,060220
7112	045026	060220	060220	060220	
7113	045034	060220	060220		
7114	045040	060220	060220	060220	.WORD 060220,060220,060220,060220,060220,060220,060220,060220
7115	045046	060220	060220	060220	
7116	045054	060220	060220		
7117	045060	174610	004002	010000	.WORD 174610,004002,010000,043620,115606,076560,014410,043225
7118	045066	043620	115606	076560	
7119	045074	014410	043225		
7120	045100	062405	070205	057625	.WORD 062405,070205,057625,116167,057637,061620,061620,061620
7121	045106	116167	057637	061620	
7122	045114	061620	061620		
7123	045120	061620	061223	057234	.WORD 061620,061223,057234,057220,000560,057222,060662,061620
7124	045126	057220	000560	057222	
7125	045134	060662	061620		
7126	045140	061620	061620	061620	.WORD 061620,061620,061620,063223,055226,064214,060605,061620
7127	045146	063223	055226	064214	
7128	045154	060605	061620		
7129	045160	116500	000600	070017	.WORD 116500,000600,070017,043221,000500,076701,014420,074661
7130	045166	043221	000500	076701	
7131	045174	014420	074661		
7132	045200	061620	077303	055224	.WORD 061620,077303,055224,055225,057220,063120,063120,063120
7133	045206	055225	057220	063120	
7134	045214	063120	063120		
7135	045220	063120	000700	060660	.WORD 063120,000700,060660,060703,061227,003360,120440,060660
7136	045226	060703	061227	003360	
7137	045234	120440	060660		
7138	045240	063305	000600	060705	.WORD 063305,000600,060705,061222,060525,117602,000700,114705
7139	045246	061222	060525	117602	
7140	045254	000700	114705		
7141	045260	070217	043221	000500	.WORD 070217,043221,000500,062701,000420,060661,061620,063303
7142	045266	062701	000420	060661	
7143	045274	061620	063303		
7144	045300	060605	117552	000402	.WORD 060605,117552,000402,070017,056222,062203,056226,056227
7145	045306	070017	056222	062203	
7146	045314	056226	056227		
7147	045320	057220	123221	054661	.WORD 057220,123221,054661,061311,055230,000403,070017,054406
7148	045326	061311	055230	000403	
7149	045334	070017	054406		
7150	045340	061224	057220	061105	.WORD 061224,057220,061105,115135,114540,043220,000404,062400
7151	045346	115135	114540	043220	
7152	045354	000404	062400		
7153	045360	120600	116140	120620	.WORD 120600,116140,120620,116145,114454,000400,061231,000773
7154	045366	116145	114454	000400	
7155	045374	061231	000773		
7156	045400	061226	114454	060601	.WORD 061226,114454,060601,117112,000403,070017,056406,043220
7157	045406	117112	000403	070017	
7158	045414	056406	043220		
7159	045420	076500	115163	114512	.WORD 076500,115163,114512,043221,000404,076401,114512,054620
7160	045426	043221	000404	076401	
7161	045434	114512	054620		
7162	045440	061620	061620	061620	.WORD 061620,061620,061620,061620,061223,055224,055225,055226

7163	045446	061620	061223	055224	
7164	045454	055225	055226		
7165	045460	055227	114465	123620	.WORD 055227,114465,123620,000700,061311,114476,123400,061620
7166	045466	000700	061311	114476	
7167	045474	123400	061620		
7168	045500	117212	100425	123400	.WORD 117212,100425,123400,000600,061300,060520,117621,000625
7169	045506	000600	061300	060520	
7170	045514	117621	000625		
7171	045520	114705	000600	061231	.WORD 114705,000600,061231,000664,114705,123400,061620,117251
7172	045526	000664	114705	123400	
7173	045534	061620	117251		
7174	045540	123077	000407	063277	.WORD 123077,000407,063277,063137,063137,063137,073537,062231
7175	045546	063137	063137	063137	
7176	045554	073537	062231		
7177	045560	004600	060417	063236	.WORD 004600,060417,063236,120400,116247,116673,110504,106716
7178	045566	120400	116247	116673	
7179	045574	110504	106716		
7180	045600	104744	060600	117655	.WORD 104744,060600,117655,114606,000001,060520,117660,100425
7181	045606	114606	000001	060520	
7182	045614	117660	100425		
7183	045620	000600	061231	000664	.WORD 000600,061231,000664,114705,123400,061620,117270,114630
7184	045626	114705	123400	061620	
7185	045634	117270	114630		
7186	045640	060600	103425	114606	.WORD 060600,103425,114606,123000,000500,061260,000401,114705
7187	045646	123000	000500	061260	
7188	045654	000401	114705		
7189	045660	123440	103425	000500	.WORD 123440,103425,000500,061262,000401,063230,100425,000000
7190	045666	061262	000401	063230	
7191	045674	100425	000000		
7192	045700	000000	000000	000000	.WORD 000000,000000,000000,000000,000000,000000,000000,000000
7193	045706	000000	000000	000000	
7194	045714	000000	000000		
7195	045720	000000	000000	000000	.WORD 000000,000000,000000,000000,000000,000000,000000,000000
7196	045726	000000	000000	000000	
7197	045734	000000	000000		
7198	045740	000000	000000	000000	.WORD 000000,000000,000000,000000,000000,000000,000000,000000
7199	045746	000000	000000	000000	
7200	045754	000000	000000		
7201	045760	000000	000000	000000	.WORD 000000,000000,000000,000000,000000,000000,000000,060000
7202	045766	000000	000000	000000	
7203	045774	000000	000000		
7204	046000	000000	000000	000000	.WORD 000000,000000,000000,000000,000000,000000,000000,000000
7205	046006	000000	000000	000000	
7206	046014	000000	000000		
7207	046020	001007	114761	001027	.WORD 001007,114761,001027,114763,001047,114765,001067,114767
7208	046026	114763	001047	114765	
7209	046034	001067	114767		
7210	046040	001107	114771	001127	.WORD 001107,114771,001127,114773,001147,114775,001167,114777
7211	046046	114773	001147	114775	
7212	046054	001167	114777		
7213					
7214					
7215	046060				BOPEND:
7216					
7217	046060				BOPN32:
7218	046060	016777	123200	060520	.WORD 016777,123200,060520,103405,100400,000401,063230,000401

7219	046066	103405	100400	000401		
7220	046074	063230	000401			
7221	046100	063227	000402	063226	.WORD	063227,000402,063226,000777,063231,010001,004002,016403
7222	046106	000777	063231	010001		
7223	046114	004002	016403			
7224	046120	002403	000400	061220	.WORD	002403,000400,061220,061222,114400,000560,063236,004400
7225	046126	061222	114400	000560		
7226	046134	063236	004400			
7227	046140	073237	062231	047634	.WORD	073237,062231,047634,102451,020660,102054,102606,004420
7228	046146	102451	020660	102054		
7229	046154	102606	004420			
7230	046160	073016	047634	020660	.WORD	073016,047634,020660,107625,107001,004420,073417,103424
7231	046166	107625	107001	004420		
7232	046174	073417	103424			
7233	046200	100431	004420	063016	.WORD	100431,004420,063016,100446,020660,102606,002012,057635
7234	046206	100446	020660	102606		
7235	046214	002012	057635			
7236	046220	103713	140620	000403	.WORD	103713,140620,000403,070017,056406,101070,070077,100477
7237	046226	070017	056406	101070		
7238	046234	070077	100477			
7239	046240	056407	101073	100466	.WORD	056407,101073,100466,043220,000404,076400,100466,016505
7240	046246	043220	000404	076400		
7241	046254	100466	016505			
7242	046260	036600	000420	070217	.WORD	036600,000420,070217,062735,100437,016462,056222,022203
7243	046266	062735	100437	016462		
7244	046274	056222	022203			
7245	046300	056226	056227	057220	.WORD	056226,056227,057220,123221,054661,061311,055230,056411
7246	046306	123221	054661	061311		
7247	046314	055230	056411			
7248	046320	101123	056411	101631	.WORD	101123,056411,101631,000420,070217,062735,120600,102126
7249	046326	000420	070217	062735		
7250	046334	120600	102126			
7251	046340	120620	102372	100437	.WORD	120620,102372,100437,003000,016462,036600,004600,070217
7252	046346	003000	016462	036600		
7253	046354	004600	070217			
7254	046360	042727	064214	043221	.WORD	042727,064214,043221,062701,004002,010000,056407,043224
7255	046366	062701	004002	010000		
7256	046374	056407	043224			
7257	046400	000410	062404	070204	.WORD	000410,062404,070204,016604,076617,076614,076600,016400
7258	046406	016604	076617	076614		
7259	046414	076600	016400			
7260	046420	000401	062620	100437	.WORD	000401,062620,100437,016601,022600,000403,070017,056406
7261	046426	016601	022600	000403		
7262	046434	070017	056406			
7263	046440	101172	100501	056407	.WORD	101172,100501,056407,101175,100571,043220,000404,076400
7264	046446	101175	100571	043220		
7265	046454	000404	076400			
7266	046460	100571	023200	100536	.WORD	100571,023200,100536,016601,036600,100501,002052,057635
7267	046466	016601	036600	100501		
7268	046474	002052	05763			

7275	046540	100746	120600	102231	.WORD	100746,120600,102231,120620,102372,057625,070077,103242
7276	046546	120620	102372	057625		
7277	046554	070077	103242			
7278	046560	002533	100501	002563	.WORD	002533,100501,002563,100501,002012,074615,023204,103251
7279	046566	100501	002012	074615		
7280	046574	023204	103251			
7281	046600	062604	003200	070217	.WORD	062604,003200,070217,042700,004000,056727,004002,010000
7282	046606	042700	004000	056727		
7283	046614	004002	010000			
7284	046620	000410	056407	043220	.WORD	000410,056407,043220,062400,070200,016604,076617,076614
7285	046626	062400	070200	016604		
7286	046634	076617	076614			
7287	046640	076604	076605	060605	.WORD	076604,076605,060605,103676,002402,100437,002774,100437
7288	046646	103676	002402	100437		
7289	046654	002774	100437			
7290	046660	023300	000770	062274	.WORD	023300,000770,062274,000776,063224,000420,070217,076715
7291	046666	000776	063224	000420		
7292	046674	070217	076715			
7293	046700	000771	062274	100746	.WORD	000771,062274,100746,023300,000770,062274,000771,062274
7294	046706	023300	000770	062274		
7295	046714	000771	062274			
7296	046720	060615	061620	103037	.WORD	060615,061620,103037,070217,000440,062715,000404,063225
7297	046726	070217	000440	062715		
7298	046734	000404	063225			
7299	046740	000410	004002	010000	.WORD	000410,004002,010000,056407,043224,062404,070204,016405
7300	046746	056407	043224	062404		
7301	046754	070204	016405			
7302	046760	076617	016400	016400	.WORD	076617,016400,016400,016400,076605,100437,004600,070217
7303	046766	016400	076605	100437		
7304	046774	004600	070217			
7305	047000	042727	064214	043221	.WORD	042727,064214,043221,076701,000410,004002,010000,056407
7306	047006	076701	000410	004002		
7307	047014	010000	056407			
7308	047020	043220	062400	070200	.WORD	043220,062400,070200,016404,076617,076614,016400,016400
7309	047026	016404	076617	076614		
7310	047034	016400	016400			
7311	047040	062604	100437	063224	.WORD	062604,100437,063224,000500,060664,061231,000773,063224
7312	047046	000500	060664	061231		
7313	047054	000773	063224			
7314	047060	100746	057635	103445	.WORD	100746,057635,103445,147220,016132,056230,000420,070216
7315	047066	147220	016132	056230		
7316	047074	000420	070216			
7317	047100	062735	100445	016444	.WORD	062735,100445,016444,016172,014401,062230,042224,056406
7318	047106	016172	014401	062230		
7319	047114	042224	056406			
7320	047120	105034	056225	055230	.WORD	105034,056225,055230,000402,070016,120600,106025,120620
7321	047126	000402	070016	120600		
7322	047134	106025	120620			
7323	047140	106213	036400	036420	.WORD	106213,036400,036420,100445,042225,056407,105040,104422
7324	047146	100445				

CZKMEAO DMS11-DA DYNAMIC
CZKMEA.P11 20-OCT-81 17:05

MACY11 30A(1052) 21-OCT-81 08:50 K 11
ROUTINE TO SIZE MEMORY PAGE 140

SEQ 0140

7331	047214	016454	104413		
7332	047220	016132	054220	056230	
7333	047226	042224	056406	105112	.WORD 016132,054220,056230,042224,056406,105112,056225,055230
7334	047234	056225	055230		
7335	047240	056411	105074	042411	
7336	047246	105501	070216	000420	.WORD 056411,105074,042411,105501,070216,000420,076735,054220
7337	047254	076735	054220		
7338	047260	104503	070076	016522	
7339	047266	120600	106103	120620	.WORD 104503,070076,016522,120600,106103,120620,106213,036400
7340	047274	106213	036400		
7341	047300	036420	100445	042225	
7342	047306	056407	105116	104467	.WORD 036420,100445,042225,056407,105116,104467,041230,003004
7343	047314	041230	003004		
7344	047320	056404	104470	016556	
7345	047326	043221	000411	070016	.WORD 056404,104470,016556,043221,000411,070016,043622,107152
7346	047334	043622	107152		
7347	047340	002372	063122	063122	.WORD 002372,063122,063122,063522,062602,060522,062230,060601
7348	047346	063522	062602	060522	
7349	047354	062230	060601		
7350	047360	060220	002132	062230	.WORD 060220,0602132,062230,000402,060220,002172,062230,070076
7351	047366	000402	060220	002172	
7352	047374	062230	070076		
7353	047400	002565	100445	002132	.WORD 002565,100445,002132,060601,062230,100445,054220,054220
7354	047406	060601	062230	100445	
7355	047414	054220	054220		
7356	047420	057221	000411	070016	.WORD 057221,000411,070016,043222,104530,007200,070216,042727
7357	047426	043222	104530	007200	
7358	047434	070216	042727		
7359	047440	064214	056700	000411	.WORD 064214,056700,000411,070016,043225,004002,010000,056407
7360	047446	070016	043225	004002	
7361	047454	010000	056407		
7362	047460	043224	000410	062404	.WORD 043224,000410,062404,070204,016400,076617,076614,054220
7363	047466	070204	016400	076617	
7364	047474	076614	054220		
7365	047500	076605	002401	100445	.WORD 076605,002401,100445,063224,000500,060664,061231,000773
7366	047506	063224	000500	060664	
7367	047514	061231	000773		
7368	047520	063224	002172	000404	.WORD 063224,002172,000404,062230,104634,002172,000777,063224
7369	047526	062230	104634	002172	
7370	047534	000777	063224		
7371	047540	000401	062230	043635	.WORD 000401,062230,043635,103445,007200,070216,042727,064214
7372	047546	103445	007200	070216	
7373	047554	042727	064214		
7374	047560	056700	004002	010000	.WORD 056700,004002,010000,056407,043220,000410,062400,070200
7375	047566	056407	043220	000410	
7376	047574	062400	070200		
7377	047600	016400	076617	076614	.WORD 016400,076617,076614,054220,016400,062604,100445,043620
7378	047606	054220	016400	062604	
7379	047614	100445	043620		
7380	047620	106662	114672	002740	.WORD 106662,114672,002740,004001,002701,004600,070017,002700
7381	047626	004001	002701	004600	
7382	047634	070017	002700		
7383	047640	004001	002701	000400	.WORD 004001,002701,000400,002372,062230,000421,062234,114672
7384	047646	002372	062230	000421	
7385	047654	062234	114672		
7386	047660	123561	107262	061620	.WORD 123561,107262,061620,106334,106710,060521,113424,114672

SEQ 0141

7387	047666	106334	106710	060521		
7388	047674	113424	114672			
7389	047700	004002	010000	056407	.WORD	004002,010000,056407,043220,000410,062400,070200,016405
7390	047706	043220	000410	062400		
7391	047714	070200	016405			
7392	047720	076617	016400	016400	.WORD	076617,016400,016400,023260,000410,060660,061620,061620
7393	047726	023260	000410	060660		
7394	047734	061620	061620			
7395	047740	061620	076620	002410	.WORD	061620,076620,002410,114672,120400,061620,106770,070216
7396	047746	114672	120400	061620		
7397	047754	106770	070216			
7398	047760	047234	043635	117672	.WORD	047234,043635,117672,002172,000404,062230,007200,042727
7399	047766	002172	000404	062230		
7400	047774	007200	042727			
7401	050000	064214	056700	004002	.WORD	064214,056700,004002,010000,056407,043220,000410,062400
7402	050006	010000	056407	043220		
7403	050014	000410	062400			
7404	050020	070200	016400	076617	.WORD	070200,016400,076617,076614,016400,016400,016772,114672
7405	050026	076614	016400	016400		
7406	050034	016772	114672			
7407	050040	047234	043635	117672	.WORD	047234,043635,117672,000776,023301,063661,062234,060461
7408	050046	000776	023301	063661		
7409	050054	062234	060461			
7410	050060	062234	004600	042727	.WORD	062234,004600,042727,064214,043221,076701,014410,004002
7411	050066	064214	043221	076701		
7412	050074	014410	004002			
7413	050100	010000	056407	043220	.WORD	010000,056407,043220,062400,070200,016404,076617,076614
7414	050106	062400	070200	016404		
7415	050114	076617	076614			
7416	050120	016400	016400	016772	.WORD	016400,016400,016772,114672,123141,000437,063222,000740
7417	050126	114672	123141	000437		
7418	050134	063222	000740			
7419	050140	063261	020700	060662	.WORD	063261,020700,060662,060701,062234,114672,120400,061620
7420	050146	060701	062234	114672		
7421	050154	120400	061620			
7422	050160	112534	070216	047634	.WORD	112534,070216,047634,116672,043635,113453,000401,064334
7423	050166	116672	043635	113453		
7424	050174	000401	064334			
7425	050200	043635	113456	114672	.WORD	043635,113456,114672,060535,113461,110446,060535,113461
7426	050206	060535	113461	110446		
7427	050214	060535	113461			
7428	050220	114672	054220	016412	.WORD	114672,054220,016412,054220,054220,136500,136520,123160
7429	050226	054220	054220	136500		
7430	050234	136520	123160			
7431	050240	000700	063260	063120	.WORD	000700,063260,063120,063140,063140,063120,063120,076460
7432	050246	063140	063140	063120		
7433	050254	063120	076460			
7434	050260	123141	063161	123160	.WORD	123141,063161,123160,000401,063260,111110,063160,063260
7435	050266	000401	063260	111110		
7436	050274	063160	06326			

C	T	X	S
C	X	\$	\$
T	X	\$	\$
T	X	\$	\$
T	X	\$	\$
U	\$	\$	\$
V	\$	\$	\$
V	\$	\$	\$
W	\$	\$	\$

CZKMEAO DMS11-DA DYNAMIC
CZKMEA.P11 20-OCT-81 17:05

MACY11 30A(1052) 21-OCT-81 08:50 M 11
ROUTINE TO SIZE MEMORY PAGE 142

SEQ 0142

7443	050340	060675	070216	076700	.WORD	060675,070216,076700,114672,047634,116672,043635,113545
7444	050346	114672	047634	116672		
7445	050354	043635	113545			
7446	050360	000401	064334	043635	.WORD	000401,064334,043635,113550,114672,060535,113553,110540
7447	050366	113550	114672	060535		
7448	050374	113553	110540			
7449	050400	060535	113553	114672	.WORD	060535,113553,114672,054220,016477,054220,136500,136520
7450	050406	054220	016477	054220		
7451	050414	136500	136520			
7452	050420	123160	000740	063260	.WORD	123160,000740,063260,063120,063140,063140,063120,076520
7453	050426	063120	063140	063140		
7454	050434	063120	076520			
7455	050440	016500	016421	123140	.WORD	016500,016421,123140,000402,076740,111620,111206,123160
7456	050446	000402	076740	111620		
7457	050454	111206	123160			
7458	050460	000401	063260	063160	.WORD	000401,063260,063160,076660,123160,110611,123160,000401
7459	050466	076660	123160	110611		
7460	050474	123160	000401			
7461	050500	076660	000436	076660	.WORD	076660,000436,076660,000403,060675,070217,076675,114672
7462	050506	000403	060675	070217		
7463	050514	076675	114672			
7464	050520	123160	000401	063260	.WORD	123160,000401,063260,063160,111627,076660,110611,070077
7465	050526	063160	111627	076660		
7466	050534	110611	070077			
7467	050540	123560	113234	002533	.WORD	123560,113234,002533,110635,002603,000412,070017,110611
7468	050546	110635	002603	000412		
7469	050554	070017	110611			
7470	050560	060220	060220	060220	.WORD	060220,060220,060220,060220,060220,060220,060220,060220
7471	050566	060220	060220	060220		
7472	050574	060220	060220			
7473	050600	060220	060220	060220	.WORD	060220,060220,060220,060220,060220,060220,060220,060220
7474	050606	060220	060220	060220		
7475	050614	060220	060220			
7476	050620	060220	060220	060220	.WORD	060220,060220,060220,060220,060220,060220,060220,060220
7477	050626	060220	060220	060220		
7478	050634	060220	060220			
7479	050640	060220	060220	060220	.WORD	060220,060220,060220,060220,060220,060220,060220,060220
7480	050646	060220	060220	060220		
7481	050654	060220	060220			
7482	050660	060220	060220	060220	.WORD	060220,060220,060220,060220,060220,060220,060220,060220
7483	050666	060220	060220	060220		
7484	050674	060220	060220			
7485	050700	060220	060220	060220	.WORD	060220,060220,060220,060220,060220,060220,060220,060220
7486	050706	060220	060220	060220		
7487	050714	060220	060220			
7488	050720	060220	060220	060220	.WORD	060220,060220,060220,060220,060220,060220,060220,060220
7489	050726	060220	060220	060220		
7490	050734	060220	060220			
7491	050740	060220	060220	060220	.WORD	060220,060220,060220,060220,060220,060220,060220,060220
7492	050746	060220	060220	060220		
7493	050754	060220	060220			
7494	050760	060220	060220	060220	.WORD	060220,060220,060220,060220,060220,060220,060220,060220
7495	050766	060220	060220	060220		
7496	050774	060220	060220			
7497	051000	060220	060220	060220	.WORD	060220,060220,060220,060220,060220,060220,060220,060220
7498	051006	060220	060220	060220		

CZKMEAO DMS11-DA DYNAMIC
CZKMEA.P11 20-OCT-81 17:05

MACY11 30A(1052) 21-OCT-81 08:50 N 11
ROUTINE TO SIZE MEMORY PAGE 143

SEQ 0143

7499	051014	060220	060220		
7500	051020	060220	060220	060220	.WORD 060220,060220,060220,060220,060220,060220,060220,060220
7501	051026	060220	060220	060220	
7502	051034	060220	060220		
7503	051040	060220	060220	060220	.WORD 060220,060220,060220,060220,060220,060220,060220,060220
7504	051046	060220	060220	060220	
7505	051054	060220	060220		
7506	051060	174610	004002	010000	.WORD 174610,004002,010000,043620,115606,076560,014410,043225
7507	051066	043620	115606	076560	
7508	051074	014410	043225		
7509	051100	062405	070205	057625	.WORD 062405,070205,057625,116167,057637,061620,061620,061620
7510	051106	116167	057637	061620	
7511	051114	061620	061620		
7512	051120	061620	061223	057234	.WORD 061620,061223,057234,057220,000560,057222,060662,061620
7513	051126	057220	000560	057222	
7514	051134	060662	061620		
7515	051140	061620	061620	061620	.WORD 061620,061620,061620,063223,055226,064214,060605,061620
7516	051146	063223	055226	064214	
7517	051154	060605	061620		
7518	051160	116500	000600	070017	.WORD 116500,000600,070017,043221,000500,076701,014420,074661
7519	051166	043221	000500	076701	
7520	051174	014420	074661		
7521	051200	061620	077303	055224	.WORD 061620,077303,055224,055225,057220,063120,063120,063120
7522	051206	055225	057220	063120	
7523	051214	063120	063120		
7524	051220	063120	000700	060660	.WORD 063120,000700,060660,060703,061227,003360,120440,060660
7525	051226	060703	061227	003360	
7526	051234	120440	060660		
7527	051240	063305	000600	060705	.WORD 063305,000600,060705,061222,060525,117602,000677,114704
7528	051246	061222	060525	117602	
7529	051254	000677	114704		
7530	051260	070217	043221	000500	.WORD 070217,043221,000500,062701,000420,060661,061620,063303
7531	051266	062701	000420	060661	
7532	051274	061620	063303		
7533	051300	060605	117552	000402	.WORD 060605,117552,000402,070017,056222,062203,056226,056227
7534	051306	070017	056222	062203	
7535	051314	056226	056227		
7536	051320	057220	123221	054661	.WORD 057220,123221,054661,061311,055230,000403,070017,054406
7537	051326	061311	055230	000403	
7538	051334	070017	054406		
7539	051340	061224	057220	061105	.WORD 061224,057220,061105,115135,114540,043220,000404,062400
7540	051346	115135	114540	043220	
7541	051354	000404	062400		
7542	051360	120600	116140	120620	.WORD 120600,116140,120620,116145,114454,000400,061231,000773
7543	051366	116145	114454	000400	
7544	051374	061231	000773		
7545	051400	061226	114454	060601	.WORD 061226,114454,060601,117112,000403,070017,056406,043220
7546	051406	117112	000403	070017	
7547	051414	056406	043220		
7548	051420	076500	115163	114512	.WORD 076500,115163,114512,043221,000404,076401,114512,054620
7549	051426	043221	000404	076401	
7550	051434	114512	054620		
7551	051440	061620	061620	061620	.WORD 061620,061620,061620,061620,061223,055224,055225,055226
7552	051446	061620	061223	055224	
7553	051454	055225	055226		
7554	051460	055227	114465	123620	.WORD 055227,114465,123620,000700,061311,114476,123400,061620

7555	051466	000700	061311	114476	
7556	051474	123400	061620		
7557	051500	117212	100425	123400	.WORD 117212,100425,123400,000600,061300,060520,117621,000625
7558	051506	000600	061300	060520	
7559	051514	117621	000625		
7560	051520	114704	000600	061231	.WORD 114704,000600,061231,000663,114704,123400,061620,117251
7561	051526	000663	114704	123400	
7562	051534	061620	117251		
7563	051540	123077	000407	063277	.WORD 123077,000407,063277,063137,063137,063137,073537,062231
7564	051546	063137	063137	063137	
7565	051554	073537	062231		
7566	051560	004600	060417	063236	.WORD 004600,060417,063236,120400,116247,116672,110436,106657
7567	051566	120400	116247	116672	
7568	051574	110436	106657		
7569	051600	104700	060600	117654	.WORD 104700,060600,117654,114606,060520,117657,100425,000600
7570	051606	114606	060520	117657	
7571	051614	100425	000600		
7572	051620	061231	000663	114704	.WORD 061231,000663,114704,123400,061620,117267,114630,060600
7573	051626	123400	061620	117267	
7574	051634	114630	060600		
7575	051640	103425	114606	123000	.WORD 103425,114606,123000,000500,061260,000401,114704,123440
7576	051646	000500	061260	000401	
7577	051654	114704	123440		
7578	051660	103425	000500	061262	.WORD 103425,000500,061262,000401,063230,100425,000000,000000
7579	051666	000401	063230	100425	
7580	051674	000000	000000		
7581	051700	000000	000000	000000	.WORD 000000,000000,000000,000000,000000,000000,000000,000000
7582	051706	000000	000000	000000	
7583	051714	000000	000000		
7584	051720	000000	000000	000000	.WORD 000000,000000,000000,000000,000000,000000,000000,000000
7585	051726	000000	000000	000000	
7586	051734	000000	000000		
7587	051740	000000	000000	000000	.WORD 000000,000000,000000,000000,000000,000000,000000,000000
7588	051746	000000	000000	000000	
7589	051754	000000	000000		
7590	051760	000000	000000	000000	.WORD 000000,000000,000000,000000,000000,000000,000000,000000
7591	051766	000000	000000	000000	
7592	051774	000000	000000		
7593	052000	000000	000000	000000	.WORD 000000,000000,000000,000000,000000,000000,000000,000000
7594	052006	000000	000000	000000	
7595	052014	000000	000000		
7596	052020	001007	114761	001027	.WORD 001007,114761,001027,114763,001047,114765,001067,114767
7597	052026	114763	001047	114765	
7598	052034	001067	114767		
7599	052040	001107	114771	001127	.WORD 001107,114771,001127,114773,001147,114775,001167,114777
7600	052046	114773	001147	114775	
7601	052054	001167	114777		
7602	052060				
7603	052060	000100			ENDA:
7604	052260	201			PATCH: .BLKW 100
7605	052261	000377			TBUF: .BYTE 201 ;SET FIRST CHARACTER TO SOH
7606	052660	000200			.BLKB 255. ;PPSEUDO-RANDOM DATA
7607					RBUF: .BLKW 128.
7608		000001			.END

ADDR	030010	5546*	5562*	5567#										
ADINH	030016	5552*	5553	5557	5564*	5570#								
ADINL	030014	5550*	5555	5563*	5569#									
BADDR	002532	1559#	3126	3128	3137	3212	3217	3347	3462	3590	3593	3624	3655	3695
		5507	5513	5562	6032									
BEGIN	001542	1254	1260	1387#										
BIT0 =	000001	1184#												
BIT00 =	000001	1174#	1184											
BIT01 =	000002	1173#	1183											
BIT02 =	000004	1172#	1182											
BIT03 =	000010	1171#	1181											
BIT04 =	000020	1170#	1180											
BIT05 =	000040	1169#	1179											
BIT06 =	000100	1168#	1178											
BIT07 =	000200	1167#	1177											
BIT08 =	000400	1166#	1176	4097										
BIT09 =	001000	1165#	1175	4105	4188									
BIT1 =	000002	1183#												
BIT10 =	002000	1164#	4149											
BIT11 =	004000	1163#	4112											
BIT12 =	010000	1162#												
BIT13 =	020000	1161#	4156											
BIT14 =	040000	1160#	4083											
BIT15 =	100000	1159#	4000											
BIT2 =	000004	1182#												
BIT3 =	000010	1181#												
BIT4 =	000020	1180#												
BIT5 =	000040	1179#												
BIT6 =	000100	1178#												
BIT7 =	000200	1177#												
BIT8 =	000400	1176#												
BIT9 =	001000	1175#												
BOPEND	046060	7215#												
BOPN32	046060	5584	7217#											
BOPW32	042060	5587	6827#											
BPTVEC=	000014	1191#												
BUFADR	002522	1554#	1835*	1846*	1969*	1974*	2045*	2067*	2073*	2138*	2191*	2261*	2329*	2408*
		2438*	2505*	2607*	2643*	2730*	2737*	2840*	2848*	2907*	2911*	2985*	2989*	3054*
		3058*	3137*	3260*	3267*	3371*	3483	3485*	3491*	3624*	3909*	5832	6044*	
BUF CNT	002526	1556#	1741*	1742	1783*	1784	1837*	1842*	1847*	1970*	1975*	2047*	2048*	2071*
		2072*	2140*	2141*	2193*	2194*	2263*	2264*	2330*	2332*	2409*	2411*	2502*	2503*
		2604*	2605*	2727*	2728*	2738*	2739*	2837*	2838*	2845*	2846*	2895*	2912*	2977*
		3046*	3138*	3243*	3244*	3259*	3266*	3366						

[illegible]

[illegible]

GETLIN	031650	1712	1755	1797	1936	2008	2107	2159	2229	2293	2336	2372	2415	2466
		2509	2518	2566	2611	2620	2688	2803	2863	2940	3010	3079	3178	3314
		3740	5616	5765	5794	5906#								
GETLNO	031644	5905#	5909											
GETPHA	027666	5543#												
GNS	= ***** U	1256	1441	1454	1467	4896	4897	4898	4899	4900	4902	4904	4905	4906
		4907	4908	4949	4957	5007	5014	5023	5036	5041	5047	5055	5065	5071
		5083	5088	5104	5144	5187	5606	5645	5652	5664	5669			
GTSWR	= 104406	1436	4902#											
HADDR	002336	1506#	5509*	5998										
HOLDB	031202	5758*	5770	5772#										
HT	= 000011	1097#	4659	4717										
INPDUN	031772	1746	1788	2060	2208	2278	2351	2430	2534	2636	5751	5814	5835	5887
		5942#												
INTALL	031120	2322	2401	2496	2598	2720	5757#							
INTKMC	030664	1621	1661	1688	1733	1775	1824	1958	2038	2134	2185	2255	2320	2399
		2494	2596	2718	2828	2899	2978	3047	3129	3248	3358	3473	3599	3781
		3804	3820	3897	5700#									
INTLIN	031064	5748#	5763	5781										
INTLNE	031204	1827	1960	2040	2136	2187	2257	2830	2901	2980	3049	3131	3250	3360
		3475	3601	3910	5778#									
INTPRM	030616	1561	5681#											
INTREC	032456	3365	6052#											
INTTRM	032510	3480	6066#											
IOT	= 000004	1532#	5684											
IOTVEC	= 000020	1192#	1396*	1397*										
KIPAR0	= 172340	1225#	5376	5426	6023									
KIPAR1	= 172342	1226#												
KIPAR2	= 172344	1227#												
KIPAR3	= 172346	1228#												
KIPAR4	= 172350	1229#												
KIPAR5	= 172352	1230#												
KIPAR6	= 172354	1231#												
KIPAR7	= 172356	1232#	6031*											
KIPDR0	= 172300	1214#	6005											
KIPDR1	= 172302	1215#												
KIPDR2	= 172304	1216#												
KIPDR3	= 172306	1217#												
KIPDR4	= 172310	1218#												
KIPDR5	= 172312	1219#												
KIPDR6	= 172314	1220#												
KIPDR7	= 172316	1221#												
KMCE01	026246	5221#												
KMCE02	026260	5227#												
KMCE03	026272	5233#												
KMCE04	026304	5239#												
KMCE05	026316	5245#												
KMCE06	026330	5251#												
KMCE07	026342	5257#												
KMCE08	026354	5263#												
KMCE09	026366	5269#												
KMCE10	026400	5275#												
KMCE11	026412	5281#												
KMCE12	026424	5287#												
KMCE13	026436	5293#												
KMCE14	026450	5299#												

MACY11 30A(1052) 21-OCT-81 08:50 PAGE 150
CROSS REFERENCE TABLE -- USER SYMBOLS

KMCE15	026462	5305#
KMCE16	026474	5311#
KMCNT	024354	4935# 4978* 4982*
KMCP01	026252	5223#
KMCP02	026264	5229#
KMCP03	026276	5235#
KMCP04	026310	5241#
KMCP05	026322	5247#
KMCP06	026334	5253#
KMCP07	026346	5259#
KMCP08	026360	5265#
KMCP09	026372	5271#
KMCP10	026404	5277#
KMCP11	026416	5283#
KMCP12	026430	5289#
KMCP13	026442	5295#
KMCP14	026454	5301#
KMCP15	026466	5307#
KMCP16	026500	5313#
KMCR01	026244	5220#
KMCR02	026256	5226#
KMCR03	026270	5232#
KMCR04	026302	5238#
KMCR05	026314	5244#
KMCR06	026326	5250#
KMCR07	026340	5256#
KMCR08	026352	5262#
KMCR09	026364	5268#
KMCR10	026376	5274#
KMCR11	026410	5280#
KMCR12	026422	5286#
KMCR13	026434	5292#
KMCR14	026446	5298#
KMCR15	026460	5304#
KMCR16	026472	5310#
KMCS01	026254	5224#
KMCS02	026266	5230#
KMCS03	026300	5236#
KMCS04	026312	5242#
KMCS05	026324	5248#
KMCS06	026336	5254#
KMCS07	026350	5260#
KMCS08	026362	5266#
KMCS09	026374	5272#
KMCS10	026406	5278#
KMCS11	026420	5284#
KMCS12	026432	5290#
KMCS13	026444	5296#
KMCS14	026456	5302#
KMCS15	026470	5308#
KMCS16	026502	5314#
KMCTAB	026244	1518 5320 4938 4942 4979 5003 5218# 5322 5361 5630
KMCTBE	026504	4940 4984 5204 5316#
KMCV01	026250	5222#
KMCV02	026262	5228#
KMCV03	026274	5234#

[illegible]

PS	= 177776	3947 1101# 3959*	3952 1102	3959 3793*	3803*	3810*	3813*	3819*	3834*	3835	3837*	3908*	3947*	3952*
PSW	= 177776	1102#	1444*	1528*	1563*									
PT	024452	4944	4959	4960#										
PWRVEC	= 000024	1193#												
RBNLY	031522	2921	2998	3067	5869#									
RBUF	052660	1785	1835	1882	1884	1969	2045	2067	2261	2408	2607	2730	2769	2907
		2925	2985	3002	3054	3071	3485	3523	3909	6070	6072	6118	7606#	
RBUFSZ	002530	1557#	3232*	3237*	3257	3264								
RDCHR	= 104410	4515	4905#	4961	5175									
RDDEC	= 104413	4908#	4990											
RDLIN	= 104411	4275	4589	4906#	5113									
RDOCT	= 104412	1456	4907#	5090	5149	5163								
RECBUF	002506	1547#	6415											
RECEA	002504	1546#												
REQGO	031026	5725	5727	5729	5731	5733#								
REQNC	031000	2054	2200	2422	2629	5728#	5808	5881						
REQNI	031020	1692	5732#	5761	5770									
REQNR	030770	1779	5726#	5825										
REQNT	030760	1737	5724#	5829										
REQNX	031010	2270	2343	2527	5730#									
REQOUT	031742	1893	1982	1988	2078	2155	2215	2284	2357	2444	2461	2539	2649	2671
		2751	2796	2854	2917	2994	3064	3151	3274	3306	3384	3503	3633	3926
		5872	5932#											
RESTR	002362	1518#	4057											
RESVEC	= 000010	1188#												
RETADR	002512	1549#	3407	3516	5985*									
RETBUF	002514	1550#	3396*	3399*	3407*	3415	3516*	3523*	3526*	3531*	3540			
RETEA	002510	1548#	3406	3515	5994*									
RIN	031344	1839	1843	1972	2050	2068	2266	2413	2609	2618	2685	2732	2841	2908
		2986	3055	3139	3261	3368	3487	3619	3916	3938	5824#			
ROUT	002470	1540#	3231*	3239*	3260	6418								
RPT	031416	5826	5830	5836#										
RRET	002474	1542#	3223*	3238*	3292									
SAVE	002334	1502#												
SELO	002260	1475#	1583*	1584*	1585*	1624	1626*	1629	1631*	1634	1636	1642	1699*	1703
		3794*	3796	3811*	3827*	3828	5327	5578*	5591*	5594*	5608	5702*	5703*	5704*
		5724*	5726*	5728*	5730*	5732*	5736	5862	5896	5942*	5944	6402		
SEL1	002262	1476#												
SEL2	002264	1477#	1582*	1590	1664	1667	1854	1861	1890*	1897	1899*	1902	1984*	1991
		2080*	2088	2360	2370*	2378*	2447	2457*	2463*	2472*	2543	2556*	2653	2666*
		2677*	2760	2767	2784	2788*	2857	2922	2999	3068	3159	3162	3170*	3278
		3288	3298*	3393	3402	3419*	3512	3518	3544*	3641	3727*	3928*	3945*	3948
		3963*	5701*	5708	5733	5863	5869	5871*	5933					
SEL3	002266	1478#	1695*	1744*	1786*	1871	1912	2005	2057*	2105	2202*	2272*	2346*	2364
		2425*	2451	2530*	2548	2632*	2658	2755	5748*	5810*	5831*	5883*		
SEL4	002270	1479#	1697*	1743*	1785*	1885	1926	2059*	2204*	2274*	2348*	2427*	2532*	2634*
		2782	2926	3003	3072	3168	3282	3293	3414	3539	3650	3684	5589*	5750*
		5813*	5832*	5864	5885*									
SEL5	002272	1480#												
SEL6	002274	1481#	1696*	1742*	1784*	1866	1875	1907	1916	2058*	2100	2205*	2225	2275*
		2349*	2368	2428*	2455	2533*	2553	2635*	2663	2763	2861	2935	3008	3077
		3409	3534	3645	3659	3675	3689	5590*	5749*	5812*	5833*	5834*	5865	5886*
SEL7	002276	1482#	1877	1918	2000	2203*	2273*	2347*	2426*	2531*	2633*	5811*	5884*	
SETLIN	030244	1575	1617	1657	1684	1729	1771	1819	1954	2034	2130	2181	2251	2316

[illegible]

SW8 =	000400	1148#												
SW9 =	001000	1147#												
SYNTIM	031714	1628	1633	1964	2065	2152	2724	2834	2905	2916	2993	3063	3254	3812
TBITVE=	000014	5920#												
TBUF	052260	1189#												
TERBUF	033002	1743	1965	1974	2073	2138	2191	2329	2505	2737	2778	2848	3371	3396
TKVEC =	000060	6057	7604#											
TPVEC =	000064	2438	2643	2894	2938	6131#								
TRAPVE=	000034	1196#	4331*	4332*										
TRTVEC=	000014	1197#												
TSTMAP	032724	1195#	1400*	1401*										
TSTNUM	002460	1190#												
		6120#												
		1522*	1536#	1574*	1616*	1656*	1683*	1728*	1770*	1818*	1953*	2033*	2129*	2180*
		2250*	2315*	2394*	2488*	2590*	2713*	2823*	2886*	2968*	3037*	3105*	3206*	3339*
		3454*	3583*	3776*	3892*	3997*	6402	6404	6406	6408	6411	6415	6418	6421
		6424	6428	6431										
TSTREC	032716	6118#												
TST1	002552	1573#												
TST10	004534	1937	1952#											
TST11	005072	2009	2032#											
TST12	005536	2109	2128#											
TST13	005734	2160	2179#											
TST14	006226	2230	2249#											
TST15	006472	2294	2314#											
TST16	007042	2373	2393#											
TST17	007460	2467	2487#											
TST2	002724	1615#												
TST20	010124	2568	2589#											
TST21	010652	2690	2712#											
TST22	011404	2805	2822#											
TST23	011666	2864	2885#											
TST24	012260	2942	2967#											
TST25	012562	3012	3036#											
TST26	013064	3081	3104#											
TST27	013534	3180	3205#											
TST3	003122	1635	1655#											
TST30	014416	3316	3338#											
TST31	015166	3438	3453#											
TST32	016004	3563	3582#											
TST33	017156	3742	3775#											
TST34	017630	3844	3891#											
TST35	020304	3965	3995#											
TST36	020424	4012	4021#											
TST4	003214	1666	1682#											
TST5	003360	1713	1727#</											

TYPOC =	104402	4165	4170	4216	4240	4422	4897#	5609									
TYPON =	104404	4899#															
TYPOS =	104403	4898#	5018														
T7BUF	033022	1846	1923	1925	6140#												
UNITFG	030556	5626	5636	5647	5673#												
UNTDON	020400	4002	4014#														
VEGIN	002300	1483#	3785	3809*	3825*	3852	3902	3974									
VECOUT	002302	1484#	3854	3936*	3976												
WAS	002464	1538#	1636*	1637	1642*	1643	1667*	2085*	2947*	3247*	3282*	3283	3293*	3294			
		3349*	3409*	3410*	3411	3414*	3416	3464*	3534*	3535*	3536	3539*	3541	3597*			
		3611*	3641*	3642	3645*	3647	3650*	3656	3659*	3660	3662	3667	3671*	3675*			
		3676	3684*	3685	3689*	3690*	3691	3697*	3704*	3705	3716*	3717*	3718	3780*			
		3839*	3840*	3841	3852*	3854*	3895*	3974*	3976*	5846	6122*	6123*	6124	6404			
		6415	6418	6421	6424	6428											
XBITS	002340	1507#	5505*	5995													
XIN	031354	1849	1977	2074	2143	2196	2334	2439	2507	2516	2563	2644	2741	2849			
		2913	2990	3059	3142	3268	3375	3494	3625	3920	3942	5828#					
XOUT	002466	1539#	3230*	3267	6418	6431											
XRET	002472	1541#	3222*	3281													
\$AUTOB	001134	1289#	1438*	4412	4569												
\$BDADR	001122	1284#															
\$BDDAT	001126	1286#															
\$BELL	001172	1307#	4151	4200	4365	4562											
\$CHARC	023600	4666*	4676*	4683	4709*	4714#											
\$CKSWR	022160	4398#	4904														
\$CMTAG	001100	1272#	1389	1390	1398	1402	1403	1404									
\$CM3 =	000000	1302#															
\$CM4 =	000003	1302#	1303#	1304#	1305#												
\$CNTLC	023072	4353	4434	4562#													
\$CNTLG	023104	4419	4564#														
\$CNTLU	023077	4444	4537	4563#													
\$CORE	027340	5404	5460#														
\$CRLF	001177	1309#	4171	4182	4200	4208	4227	4232	4252	4309	4455	4542	4562	4630			
		4665	4717	5110	5161												
\$CROUT	027370	5460	5467#														
\$DBLK	024020	4741	4775	4783#													
\$DOAGN	020524	4040	4049	4055#													
\$DTBL	024010	4744	4779#														
\$E =	000035	1085#	1600	1601#	1640	1641#	1669	1670#	1708	1709#	1751	1752#	1793	1794#			
		1931	1932#	2015	2016#	2115	2116#	2165	2166#	2235	2236#	2299	2300#	2377			
		2378#	2471	2472#	2573	2574#	2695	2696#	2800	2801#	2869	2870#	2949	2950#			
		3018	3019#	3087	3088#	3186	3187#	3321	3322#	3434	3435#	3559	3560#	3752			
		3753#	3848	3849#	3970	3971#											
\$ENDAD	020514	1428	4051#	4195													
\$ENDCT	020462	1402	4042#														
\$ENDMG	020533	4044	4059#														
\$ENULL	020530	4047	4058#														
\$EOP	020426	4032#															
\$EOPCT	020454	1402*	4039#	4043													
\$ERFLG	001103	1275#	4067	4101	4103	4109*	4130	4146*	4200								
\$ERMAX	001115	1281#	1405*	4103	4125*	4130											
\$ERROR	021034	1398	4082	4144#													
\$ERRPC	001116	1282#	4153*	4154*	4155	4161*	4162*	4164	4168*	4169	4200	4214					
\$ERRTB	001202	1326#	4222														
\$ERRTY	021330	4181	4207#														
\$ERTTL	001112	1279#	4152*	4200													

\$ESCAP	001170	1306#	1404*	4124*	4191	4193	4200										
\$FILLC	001156	1300#	4669	4717													
\$FILLS	001155	1299#	4717														
\$GDADR	001120	1283#															
\$GDDAT	001124	1285#															
\$GET42	020504	4048#															
\$GTSWR	022250	4420#	4902														
\$HD =	000003	1080	1081														
\$HIOCT	021642	4297*	4308#														
\$ICNT	001104	1276#	4116*	4117	4119*	4129											
\$INTAG	001135	1290#	4417*	4436	4456	4569											
\$ITEMB	001114	1280#	4155*	4200	4211												
\$KTNEX	027332	5405	5459#														
\$KTOUT	027270	5439	5448#	5457													
\$KT11	027022	5403#	5407*	5459*	5485*	5459	5961	6002									
\$LF	001200	1310#	4200	4309	4552	4562	4630	4717									
\$LPADR	001106	1277#	1406*	4107*	4122*	4127	4129										
\$LPERR	001110	1278#	1407*	1576*	1618*	1658*	1685*	1730*	1772*	1820*	1955*	2035*	2131*	2182*			
		2252*	2317*	2396*	2490*	2592*	2715*	2825*	2888*	2970*	3039*	3107*	3208*	3341*			
		3456*	3585*	3778*	3894*	4107	4123*	4129	4190								
\$LSTAD	027446	5474*	5482#	5488	5503*	5509											
\$LSTBK	027450	5475*	5483#	5491	5493*	5495											
\$MAIL =	***** U	1424	1434	4122	4184	4653											
\$MAP	027114	5411*	5415*	5417#	5440	5450											
\$MAPRG	027120	5412	5419#														
\$MMOUT	027312	5442	5454#														
\$MNEW	023122	4423	4567#														
\$MSWR	023111	4420	4565#														
\$MXCNT	021032	4120	4129#														
\$N =	000035	1084#	1571	1577#	1613	1619#	1653	1659#	1680	1686#	1725	1731#	1767	1773#			
		1815	1821#	1950	1956#	2030	2036#	2126	2132#	2177	2183#	2247	2253#	2312			
		2318#	2391	2397#	2485	2491#	2587	2593#	2710	2716#	2820	2826#	2883	2889#			
		2965	2971#	3034	3040#	3102	3108#	3203	3209#	3336	3342#	3451	3457#	3580			
		3586#	3773	3779#	3889	3895#											
\$NOMAP	027140	5416	5424#														
\$NULL	001154	1298#	4671	4717													
\$NWTST=	000000	1572#	1614#	1654#	1681#	1726#	1768#	1816#	1951#	2031#	2127#	2178#	2248#	2313#			
		2392#	2486#	2588#	2711#	2821#	2884#	2966#	3035#	3103#	3204#	3337#	3452#	3581#			
		3774#	3890#	3994#	4020#												
\$OCNT	024252	4816*	4845*	4858#													
\$OMODE	024254	4811*	4815*	4820	4823*	4834*	4860#										
\$OVER	021016	4084	4100	4108	4118	4126#											
\$PASS	001100	1273#	4036*	4037*	4045	4058	4114	4130									
\$QUES	001176	1308#	4200	4306	4309	4474	4545	4562	4628	4630	4717	5111	5160				
\$RDCHR	022522	4487#	4905														
\$RDDEC	023134	4584#	4908														
\$RDLIN	022612	4510#	4906														
\$RDOCT	021504	4270#	4907														
\$RDSZ =	000024	4503#															
\$RTNAD	020526	4057#															
\$R2A =	***** U	4909															
\$SAVRE=	***** U	4909															
\$SCOPE	020550	1396	4076#														
\$SETUP=	000127	1265#	1395	1396	1398	1400	1402	1403	1404	1406	1428	1431	4034	4077			
		4145	4187	4195	4357	4362	4363	4393	4569								
\$SIZE	026736	5386#	5486														

\$SIZE	027374	5451	5453	5468#										
\$STOP	027116	5410*	5413*	5418#	5446									
\$STUP =	177777	1265#												
\$SVLAD	020770	4092	4121#											
\$SWR =	167400	1080	1081#	1082#	1305	1306	1307	1403	1404	1406	1407	1574	1616	1656
		1683	1728	1770	1818	1953	2033	2129	2180	2250	2315	2394	2488	2590
		2713	2823	2886	2968	3037	3105	3206	3339	3454	3583	3776	3892	3996
		4022	4029	4035	4050	4056	4058	4068	4069	4070	4071	4072	4078	4095
		4097	4098	4101	4102	4103	4110	4111	4112	4123	4126	4129	4136	4137
		4138	4139	4140	4149	4156	4184	4188	4200					
		4072	4073	4099										
\$SWRMK=	000000	4873	4909#											
\$TERM =	000030	1305#	1403*	4035*	4110*	4117	4120*	4129						
\$TIMES	001166	1295#	4312	4333	4344	4374	4402	4429	4690	4697	4717			
\$TKB	001146	4313#	4328*	4363	4380*	4494	4496*							
\$TKCNT	021644	1451	1527	4328#	4354	4415								
\$TKINT	021660	4317#	4388	4499										
\$TKQEN=	021657	4314#	4329*	4330	4386*	4387*	4388	4390*						
\$TKQIN	021646	4315#	4330*	4497	4498*	4499	4501*							
\$TKQOU	021650	4316#	4329	4390	4501									
\$TKQSR	021652	1294#	4312	4334*	4370*	4372	4378*	4400	4416*	4426	4438*	4458*	4688	4695
\$TKS	001144	4717												
\$TKSRV	021730	4331	4344#											
\$TMPO	001160	1302#	1587*	1588*	1702*	1705*	1853*	1856*	2086*	2090*	2892*	3795*	3798*	3826*
		3830*	3946*	3950*	5123*	5125	5127	5129	5131*	5132*	5134*	5705*	5706*	5735*
		5738*	5895*	5898*	5920*	5922*	5932*	5935*	5943*	5946*				
\$TMP1	001162	1303#	2893*	3924*	3929*	3935*	3964*	5921*	5924*					
\$TMP2	001164	1304#	3955*	3956*										
\$TN =	000037	1080#	1083#	1572	1574#	1614	1616#	1635	1654	1656#	1666	1681	1683#	1713
		1726	1728#	1756	1768	1770#	1798	1816	1818#	1937	1951	1953#	2009	2031
		2033#	2109	2127	2129#	2160	2178	2180#	2230	2248	2250#	2294	2313	2315#
		2373	2392	2394#	2467	2486	2488#	2568	2588	2590#	2690	2711	2713#	2805
		2821	2823#	2864	2884	2886#	2942	2966	2968#	3012	3035	3037#	3081	3103
		3105#	3180	3204	3206#	3316	3337	3339#	3438	3452	3454#	3563	3581	3583#
		3742	3774	3776#	3844	3890	3892#	3965	3994	3996#	4012	4020	4022#	
\$TPB	001152	1297#	4706*	4717										
\$TPFLG	001157	1301#	4647	4717										

SSGET4= 000000
\$OFILL 024253
\$4OCAT 000000
= 053260

4050#													
4810*	4814*	4824	4859#										
1251#	4078	4158											
1236#	1252	1256#	1261#	1271#	1311	1393	1406	1407	1455#	1468#	4058	4062	
4129	4130	4200	4255#	4309	4312	4316#	4317	4318#	4561#	4562	4569#	4630	
4717	4783#	4874	4876	4909	4950#	5024#	5037#	5042#	5048#	5056#	5066#	5105#	
5145#	5670#	5673#	6119	6401#	7603#	7605#	7606#						

CZKMEAO DMS11-DA DYNAMIC
CZKMEA.P11 20-OCT-81 17:05

MACY11 30A(1052) 21-OCT-81 08:50 PAGE 160
CROSS REFERENCE TABLE -- MACRO NAMES

SEQ 0158

ABORT	999#	2303	2382												
ALTBUF	1009#	2700													
ALTSTA	1004#	2476	2578												
BSEADR	1019#	3193													
CHRCNT	1014#	3091													
COMMEN	1199#														
DATADR	1024#	3326	3441												
DATTST	1030#	3566													
ENDCOM	1199#														
ERR	944#	1600	1640	1669	1708	1751	1793	1931	2015	2115	2165	2235	2299	2377	2471
	2573	2695	2800	2869	2949	3018	3087	3186	3321	3433	3558	3752	3848	3970	
ERROR	1093#	1600	1640	1669	1708	1751	1793	1931	2015	2115	2165	2235	2299	2377	2471
	2573	2695	2800	2869	2949	3018	3087	3186	3321	3434	3559	3752	3848	3970	
ESCAPE	1199#														
GETPRI	1199#	5399													
GETSWR	1199#	1431#													
INRDY	948#	1603													
INTIN	1032#	3755													
INTOUT	1033#	3872													
INTTST	944#	1564													
IOBUFI	966#	1717	1759												
KMCSE	1038#	5619													
LININT	958#	1672													
LINOVR	979#	1940													
LRCTST	1014#	2953													
MUL	1199#														
NABORT	99#	2168	2239												
NEWTST	1199#	1572	1614	1654	1681	1726	1768	1816	1951	2031	2127	2178	2248	2313	2392
	2486	2588	2711	2821	2884	2966	3035	3103	3204	3337	3452	3581	3774	3890	3994
	4020														
NOOUT	953#	1647													
NXM	1014#	2810													
PARTAB	938#	4910													
PARTST	1014#	3022													
PASS	1034#	3994													
POP	1199#	4298	4619	4770											
PUMP	939#	5571													
PUSH	1199#	4272	4586	4729											
REPORT	1199#														
RSTRCV	984#	2018													
RSTXMT	989#	2118													
SCOPE	1094#	1573	1615	1655	1682	1727	1769	1817	1952	2032	2128	2179	2249	2314	2393
	2487	2589	2712	2822	2885	2967	3036	3104	3205	3338	3453	3582	3775	3891	3995
	4021	4033													
SETPRI	1199#	4490													
SETTRA	4888#	4897	4898	4899	4900	4902	4904	4905	4906	4907	4908				
SETUP	1199#	1387													
SKIP	1199#	1635	1666	1713	1756	1798	1937	2009	2108	2160	2230	2294	2373	2467	2567
	2689	2804	2864	2941	3011	3080	3179	3316	3437	3562	3741	3844	3965	4012	
SLASH	1199#														
SPACE	1199#														
STARS	1199#	1267	1311	1564	1570	1572	1603	1612	1614	1647	1652	1654	1672	1678	1681
	1717	1723	1726	1759	1765	1768	1801	1813	1816	1940	1948	1951	2019	2028	2031
	2118	2124	2127	2168	2175	2178	2245	2248	2303	2311	2313	2382	2390	2392	2476
	2483	2486	2578	2585	2588	2700	2708	2711	2811	2818	2821	2873	2881	2884	2954
	2963	2966	3023	3032	3035	3092	3100	3103	3194	3201	3204	3327	3334	3337	3442

	3449	3452	3567	3581	3755	3771	3774	3872	3885	3890	3994	4020	4025	4064	4132
	4202	4258	4311	4393	4408	4479	4503	4572	4632	4719	4786	4863	5372		
SURSU	1199#	1408#													
TABLES	1068#	6131													
TERMXB	1014#	2872													
TRMTRP	4888#	4909													
TYPBIN	1199#														
TYPDEC	1199#	4045	4242	5654											
TYPNAM	1199#	1424													
TYPNUM	1199#														
TYPOCS	1199#														
TYPOCT	1199#	4164	4169	4214	4239	4421									
TYPTXT	1199#	1452	1464	4946	4955	5005	5011	5021	5034	5039	5045	5053	5063	5068	5081
	5086	5102	5141	5185	5604	5643	5650	5662	5666						
XRTST	970#	1801													
\$CLKAL	1040#	2324	2403	2498	2600										
\$CLRTA	1049#														
\$GETCO	944#	5484													
\$GETCS	1054#	1750	1792	1930	2013	2113	2163	2234	2298	2376	2470	2572	2694	2799	2867
	2948	3016	3085	3184	3750	3846	3968								
\$GETKM	1063#	1519	4011												
\$GETLI	1057#	1711	1754	1796	1935	2008	2107	2159	2228	2292	2336	2372	2415	2466	2509
	2518	2566	2611	2620	2688	2802	2863	3178	3313	3739	5765	5794			
\$GETPH	944#	5536													
\$INPDU	1062#	1746	1788	2060	2208	2278	2351	2430	2534	2636	5751	5814	5835	5887	
\$INTAL	1048#	2322	2401	2496	2598	2720									
\$INTKM	1040#	1621	1661	1688	1733	1775	1824	1958	2038	2134	2185	2255	2320	2399	2494
	2596	2718	2828	2899	2978	3047	3129	3248	3357	3472	3599	3781	3804	3820	3897
\$INTLI	1041#	5763	5781												
\$INTLN	1050#	1827	1960	2040	2136	2187	2257	2830	2901	2980	3049	3131	3250	3360	3475
	3601	3910													
\$INTRE	1060#	3365													
\$INTTR	1061#	3480													
\$LRSET	1055#	2147													
\$MAPBU	1058#	3615													
\$PARAM	938#	5216													
\$PARCK	1066#														
\$REQNC	1045#	2054	2200	2422	2629	5807	5880								
\$REQNI	1046#	1692	5761	5778											
\$REQNR	1042#	1779	5824												
\$REQNT	1043#	1737	5828												
\$REQNX	1047#	2270	2343	2527											
\$REQOU	1056#	1892	1982	1988	2155	2214	2283	2357	2444	2461	2539	2649	2670	2751	2796
	2854	2917	2994	3064	3150	3273	3383	3502	3632	3925					
\$RIN	1052#	1839	1843	1972	2050	2068	2266	2413	2609	2618	2685	2731	2841	2908	2986
	3055	3139	3261	3368	3487	3619	3916	3938							
\$STCLK	1051#	1830	1962	2042	2063	2150	2189	2259	2722	2832	2903	3133	3252	3362	3477
	3603	3791	3806	3822	3912	5791									
\$SYNTI	1064#	1628	1633	2152	2916	2993	3063	3812							
\$STSDA	1059#														
\$TSTN	944#	1571	1613	1653	1680	1725	1767	1815	1950	2030	2126	2177	2247	2312	2391
	2485	2587	2710	2820	2883	2965	3034	3102	3203	3336	3451	3580	3773	3889	
\$XIN	1053#	1849	1977	2074	2143	2196	2334	2439	2507	2516	2563	2644	2740	2849	2913
	2990	3059	3142	3268	3375	3494	3625	3920	3942						
\$SCMRE	1265#														
\$SCMTM	1265#	1302	1303	1304											

\$\$ESCA	1199#														
\$\$NEWT	1199#	1572	1614	1654	1681	1726	1768	1816	1951	2031	2127	2178	2248	2313	2392
	2486	2588	2711	2821	2884	2966	3035	3103	3204	3337	3452	3581	3774	3890	3994
	4020														
\$\$SET	4888#	4897	4898	4899	4900	4902	4904	4905	4906	4907	4908				
\$\$SKIP	1199#	1635	1666	1713	1756	1798	1937	2009	2109	2160	2230	2294	2373	2467	2568
	2690	2805	2864	2942	3012	3081	3180	3316	3438	3563	3742	3844	3965	4012	
.EQUAT	1070#	1089													
.HEADE	1070#														
.KT11	1070#	1199													
.SETLN	938#	5614													
.SETUP	1070#	1265													
.SWRHI	1070#														
.SCMTA	1070#	1265													
.SEOP	1070#	4023													
.SERRO	1070#	4130													
.SERRT	1070#	4200													
.\$GETK	938#	5318													
.SRDDE	1070#	4570													
.SRDOC	1070#	4256													
.\$READ	1070#	4309													
.\$SCOP	1070#	4062													
.\$SIZE	1070#	5370													
.\$STRAP	1070#	4861													
.\$TYPD	1070#	4717													
.\$TYPE	1070#	4630													
.\$TYPO	1070#	4784													
.\$40CA	1070#	1234													

. ABS. 053260 000

ERRORS DETECTED: 0

CZKMEA,CZKMEA/CRF/SOL/NL:TOC=CZKMEA
 RUN-TIME: 134 108 7 SECONDS
 RUN-TIME RATIO: 1513/250=6.0
 CORE USED: 52K (103 PAGES)