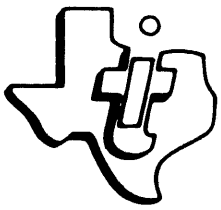
The Engineering Staff of
TEXAS INSTRUMENTS INCORPORATED
Semiconductor Group



TM 990/401-1
TIBUG MONITOR LISTING

JUNE 1977

TEXAS INSTRUMENTS
INCORPORATED

IMPORTANT NOTICE

Texas Instruments reserves the right to make changes at any time in order to improve design and to supply the best product possible.

TABLE OF CONTENTS

SECTION 1

INTRODUCTION

1.1 GENERAL

This manual contains the flowcharts and listing for the TIBUG monitor used on the TM 990/100M microcomputers. For detailed information on use of this monitor, see Section 3 of the *TM 990/100M Microcomputer User's Guide*.

Within this manual, Section 1 contains a summary of the TIBUG monitor operation, and Section 2 contains flow charts of the monitor, each chart corresponding to a particular part of the monitor listing contained in Section 3.

1.2 SUMMARY OF TIBUG OPERATION

1.2.1 STARTUP

The TIBUG Monitor is entered via an interrupt caused by the RESET switch on the microcomputer board. This interrupt is the level-zero interrupt with its vector at locations 0 and 2 in memory.

1.2.2 BAUD RATE DETECT

After startup the monitor then starts sampling the TMS 9902 for the character 'A'. The monitor then counts the width of the start bit of the first character input by looping through instructions. In fact, this character can be any character whose ASCII representation has a one in the low order bit; e.g., A = 41_{16}. This count is compared to four entries in a table. It should be noted that the number of times through the loop will depend on the clock rate the microprocessor is using. If the clock rate changes, the entries in the table must be recalculated. After the baud rate has been detected, the TMS 9902 is set up and a banner message is printed, indicating what version of TIBUG is being used.

1.2.3 SYSTEM INITIALIZATION

Following the banner message output, the monitor sets up its own workspace and initializes four flags starting at memory location $FFF4_{16}$. At this point the microterminal, if present, will send the character 'Z' to the TMS 9902. The Monitor will wait a short period for the 'Z' and if it doesn't appear, the Monitor assumes that a device other than a TM 990/301 Microterminal is connected to the TMS 9902. If a 'Z' is present, control will go to the microterminal command scanner which will accept and process entries from the microterminal. If the microterminal is not present, a question mark (?) prompt is output. The user may now enter one of several one-character commands to the monitor. A table lookup is then performed and control is transferred to the proper coding. After each command is processed, control is returned to the TIBUG Monitor.

1.2.4 INPUT/OUTPUT

All input and output functions between external peripheral devices connected to the TMS 9902, including the microterminal, are handled via Extended Operations (XOP's). All XOP vectors are in memory locations 40 to 47_{16} and 60 to $7F_{16}$. These locations reside in EPROM.

1.2.5 MICROTERMINAL OPERATIONS

Instructions sent to the Monitor from the microterminal are decoded in a jump table. If an invalid instruction is sent from the microterminal, the Monitor waits for another instruction. The microterminal part of the Monitor utilizes two unique XOP's (0,1) for sending data to and from the microterminal.

1.2.6 SINGLE STEP

The TIBUG Monitor has the unique "single step" command, which is a combination of hardware and software. Single-stepping is accomplished using the LREX instruction. This LREX instruction permits one instruction to occur and then causes an interrupt. The interrupt service routine prints the Workspace Pointer (WP), Program Counter (PC), and Status (ST) contents after the single instruction is executed.

1.2.7 MEMORY

TIBUG is EPROM-resident in memory locations 0 to $7FF_{16}$. A minimal amount of RAM (40 words, FFB0 to $FFFF_{16}$) is also needed for workspaces and flags.

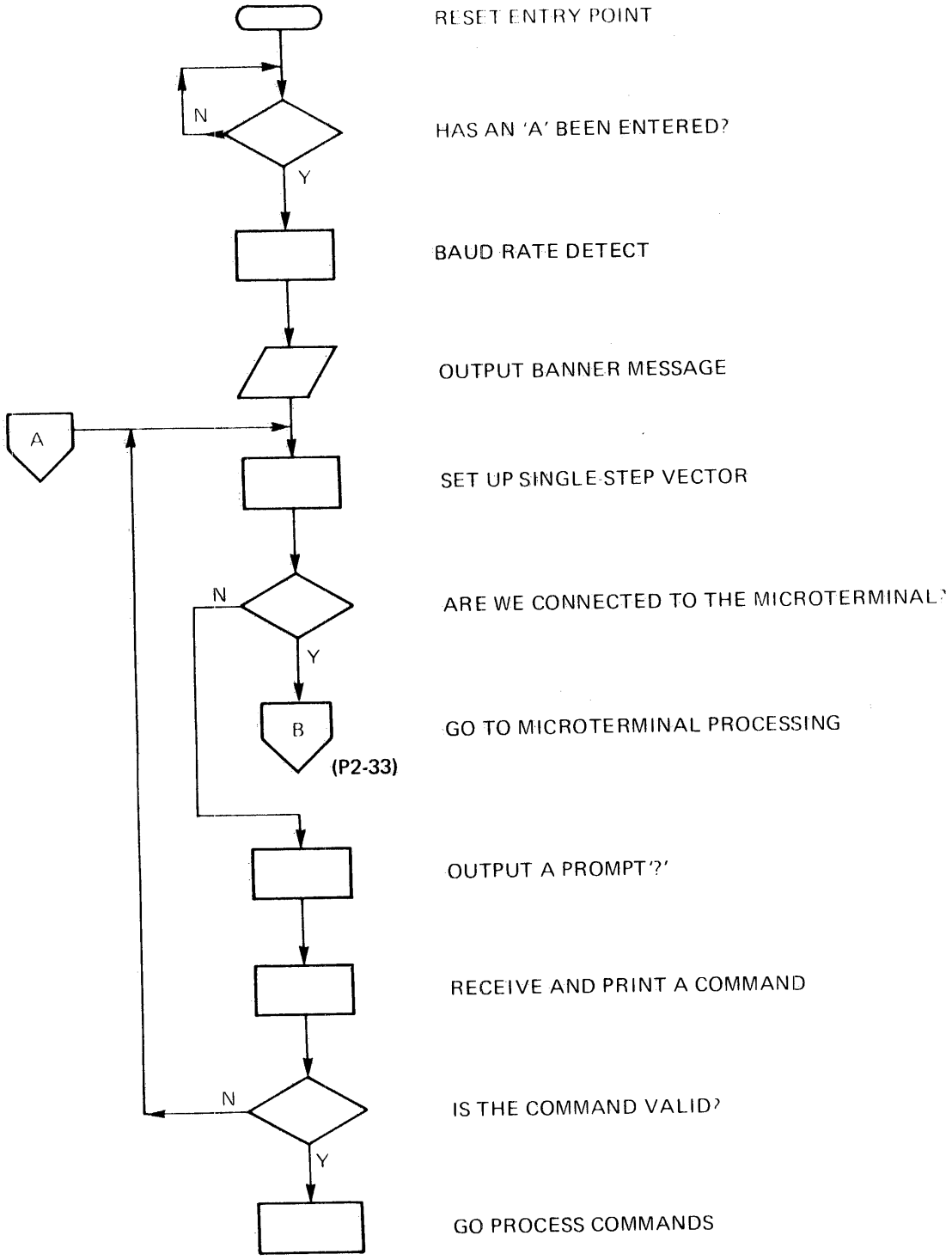
SECTION 2

TIBUG FLOW CHARTS

Flow charts are provided in the following sequence:

SYSTEM INITIALIZATION AND COMMAND SCANNER

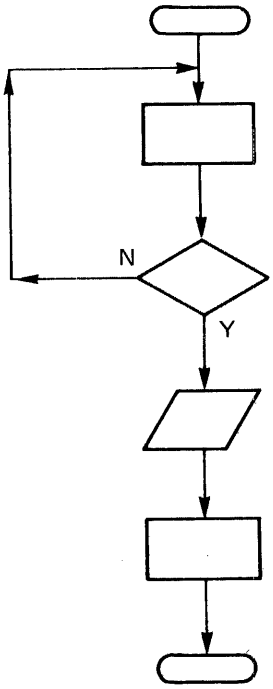
Listing Page: 0004



A0001543

XOP13, READ CHARACTER

Listing Page: 0008



ENTER READ CHARACTER XOP

SET UP 9902 CRU ADDRESS

DO WE HAVE A CHARACTER?

INPUT AND SAVE THE CHARACTER

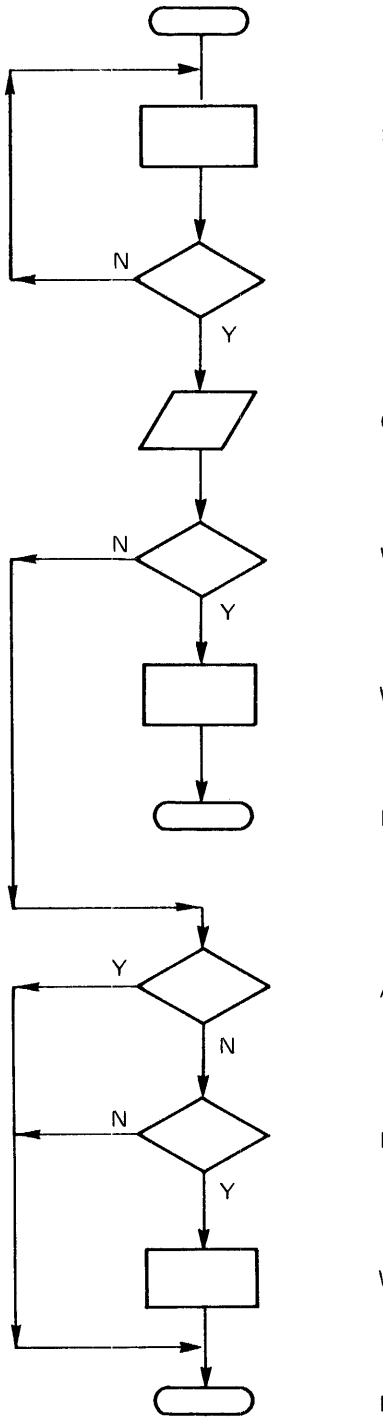
DISABLE 9902 INTERRUPTS

RETURN

A0001548

XOP 12, WRITE CHARACTER

Listing Page: 0009



ENTER WRITE CHARACTER XOP

SET UP CRU BASE OF 9902

IS TRANSMIT BUFFER REGISTER EMPTY?

OUTPUT THE CHARACTER

WAS A CARRIAGE RETURN OUTPUT?

WAIT FOR TRANSMISSION TO END

RETURN

ARE WE IN THE DUMP ROUTINE?

IS THIS ON ASR33?

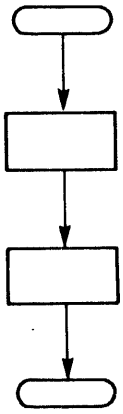
WAIT THREE NULL CHARACTERS

RETURN

A0001549

XOP 11, READ A CHARACTER AND PRINT IT OUT (ECHO)

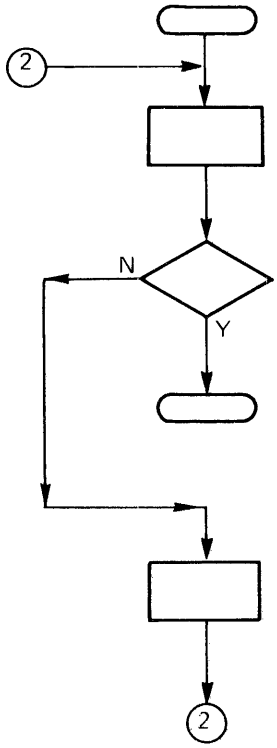
Listing Page: 0012



A0001548

XOP 14, ASCII MESSAGE OUTPUT

Listing Page: 0013



ENTER MESSAGE OUTPUT XOP

GET THE CHARACTER

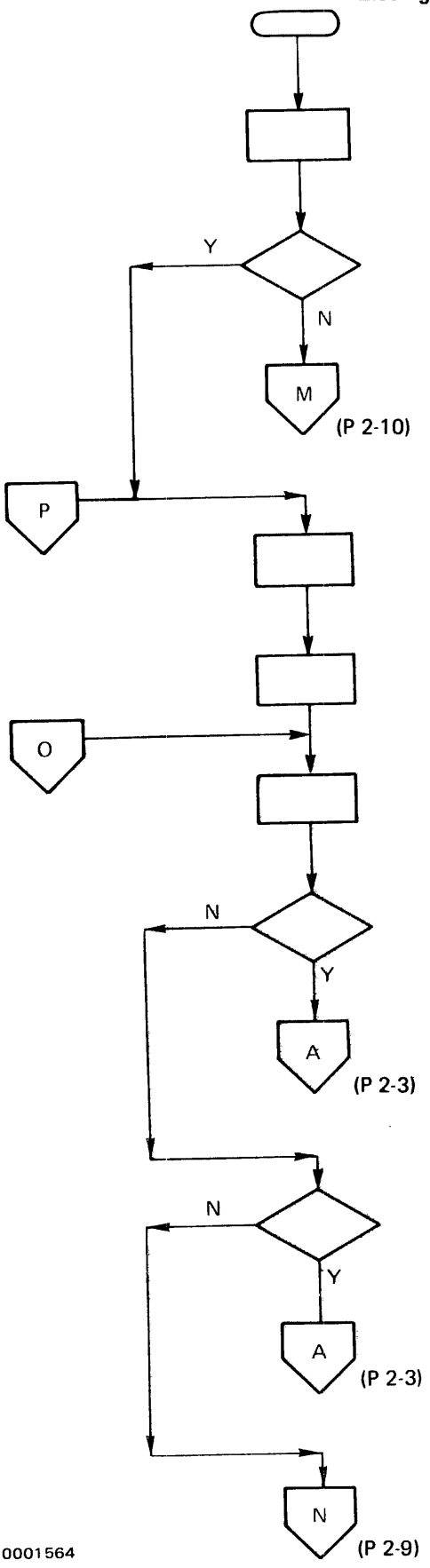
IS IT A ZERO; I.E., END OF MESSAGE?

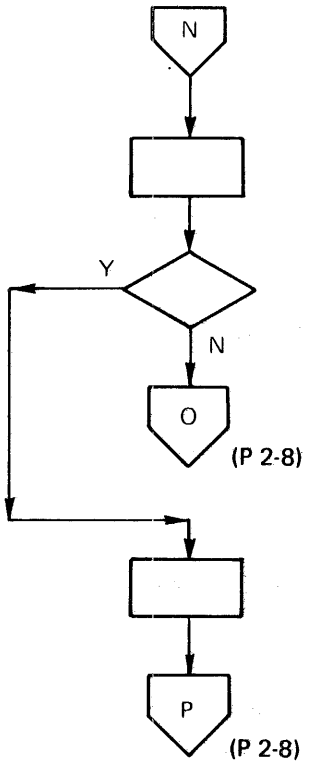
RETURN

WRITE A CHARACTER, XOP 12

A0001550

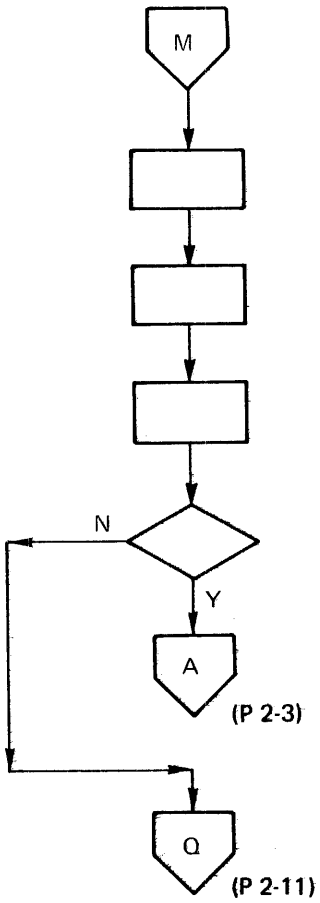
Listing Page: 0014




N
DECREMENT PRINT COUNTER
Y
IS PRINT COUNTER ZERO?
N
O
(P 2-8)
DO CARRIAGE RETURN, LINE FEED
P
(P 2-8)

A0001565

M COMMAND, MEMORY INSPECT/CHANGE (Continued)



PRINT MEMORY ADDRESS AND CONTENTS

ACCEPT NEW INPUT

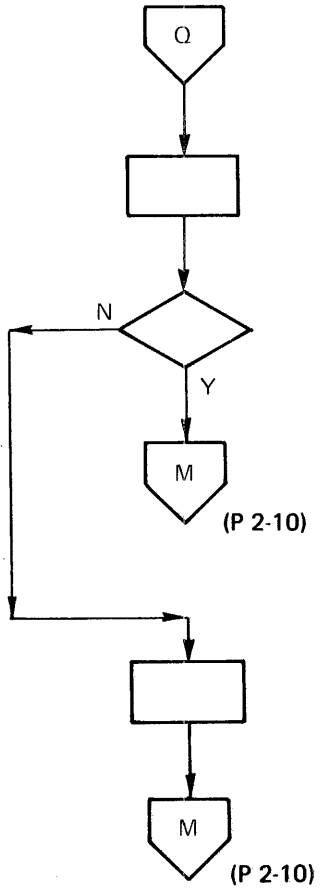
UPDATE MEMORY DATA

WAS LINE TERMINATION A CARRIAGE RETURN?

GO TO MONITOR

A0001565

M COMMAND, MEMORY INSPECT/CHANGE (Concluded)



DECREMENT MEMORY ADDRESS BY TWO TO LOOK AT PREVIOUS WORD

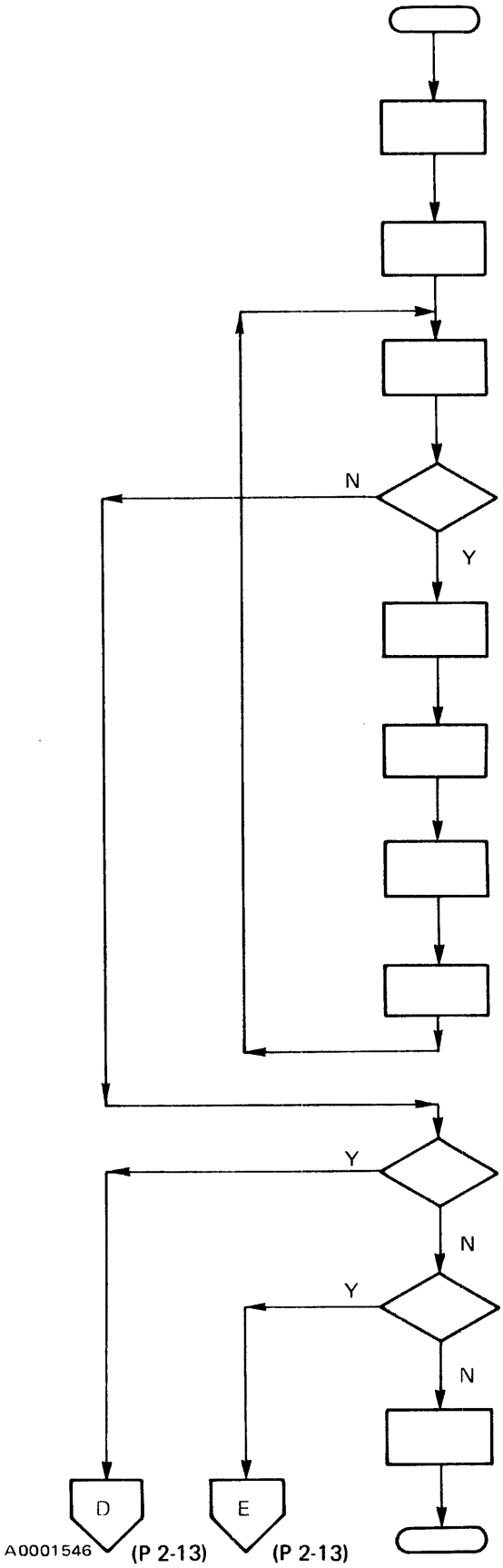
WAS LINE TERMINATION A MINUS SIGN?

INCREMENT MEMORY ADDRESS BY FOUR TO LOOK TWO WORDS AHEAD

A0001566

XOP 9, HEX INPUT ROUTINE

Listing Page: 0016



ENTER HEX INPUT ROUTINE

RESET NUMBER INPUT FLAG

CLEAR ACCUMULATOR

READ A CHARACTER, XOP 13

IS THIS A VALID HEX INPUT?

ISOLATE AND ALIGN DIGIT

SHIFT ACCUMULATOR LEFT FOUR BITS

ADD NEW HEX DIGIT TO ACCUMULATOR

INCREMENT INPUT FLAG

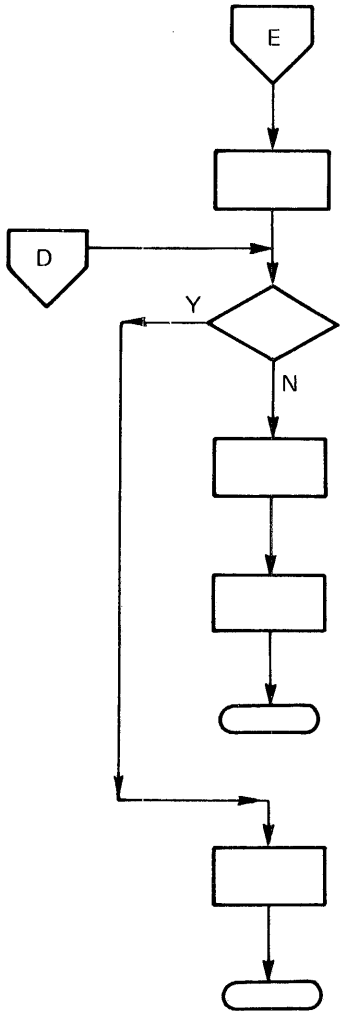
WAS THE CHARACTER A SPACE, MINUS SIGN, OR CARRIAGE RETURN?

WAS THE CHARACTER A COMMA?

SET UP ERROR POINTER

RETURN

XOP 9, HEX INPUT ROUTINE (Continued)



CHANGE THE CHARACTER TO A SPACE

IS IT A NULL CHARACTER?

RETURN VALUE AND TERMINATOR

BUMP PAST NULL AND ERROR POINTERS

RETURN

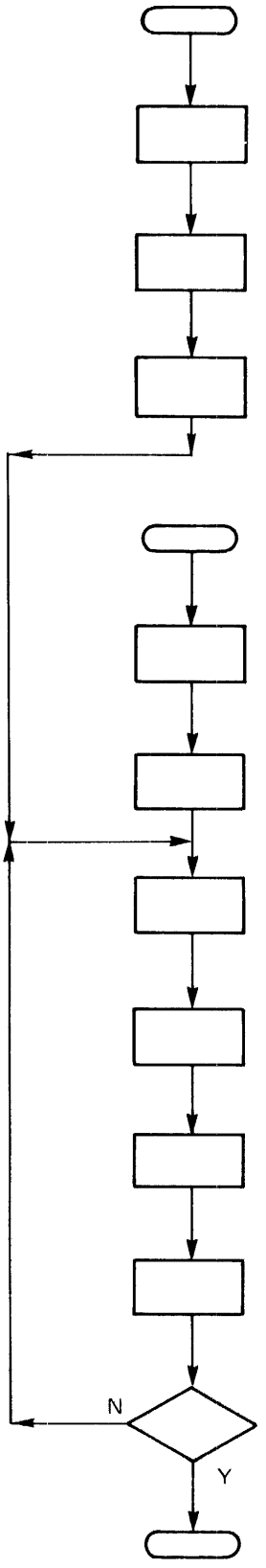
RETURN THE TERMINATOR AND GET THE POINTER

RETURN

A0001573

XOP 10, HEX OUTPUT ROUTINE

Listing Page: 0018



ENTER HEX OUTPUT ROUTINE (ONE DIGIT)

GET VALUE TO PRINT

SET UP DATA

SET DIGIT COUNT TO 1

ENTER HEX OUTPUT ROUTINE (FOUR DIGITS)

GET VALUE TO PRINT

SET DIGIT COUNT TO 4

CONVERT CHARACTER TO ASCII

WRITE A CHARACTER, XOP 12

ALIGN NEXT DIGIT

DECREMENT DIGIT COUNT

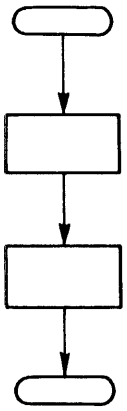
IS DIGIT COUNT ZERO?

RETURN

A0001547

S COMMAND, SINGLE STEP EXECUTION

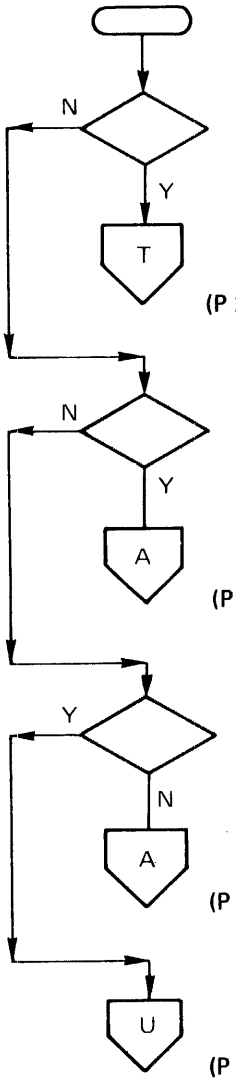
Listing Page: 0019



A0001560

UNMASKABLE LOAD INTERRUPT

Listing Page: 0019



ENTER LOAD

ARE WE IN SINGLE STEP?

IS THE HALF FLAG ON?

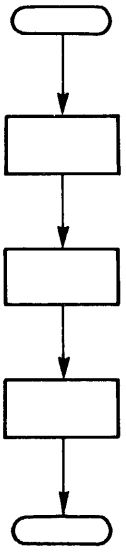
IS THE STEP FLAG SET?

GO TO THE TOP OF THE MONITOR

A0001560

B COMMAND, BREAKPOINT

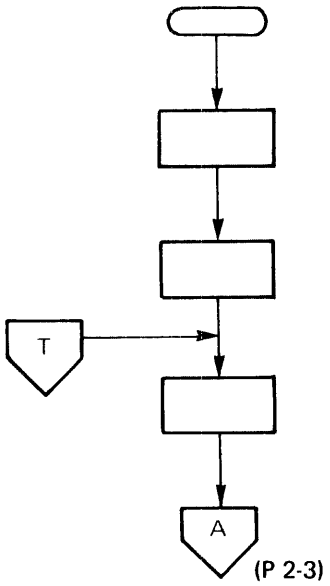
Listing Page: 0020



A0001559

XOP 15, OUTPUT WP, PC, AND ST CONTENTS

Listing Page: 0020



XOP1'S ENTRY POINT

UPDATE USER PROGRAM COUNTER

RESTORE LOCATION

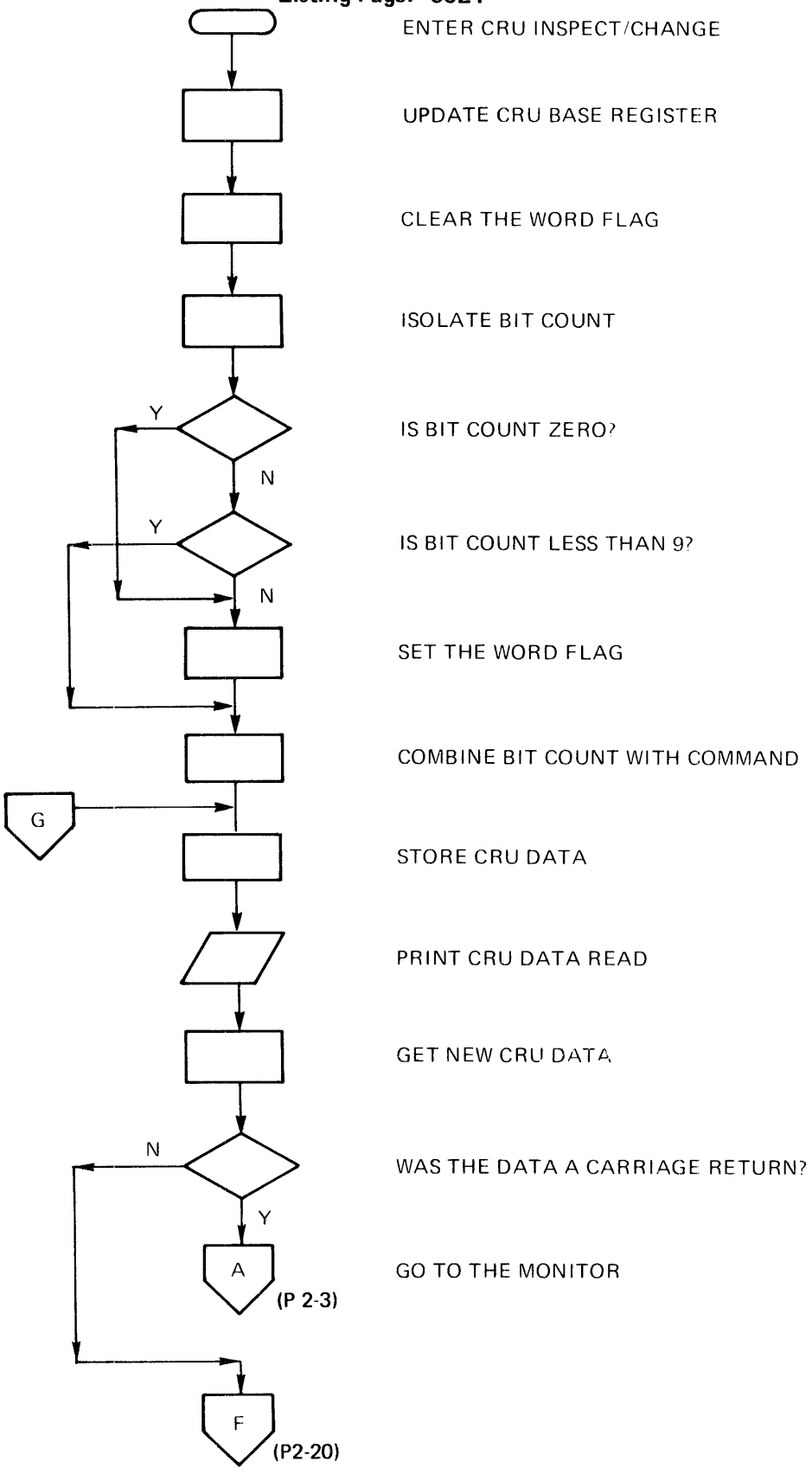
OUTPUT CONTENTS OF WORKSPACE POINTER, PROGRAM COUNTER, AND STATUS

GO TO THE TOP OF THE MONITOR

A0001559

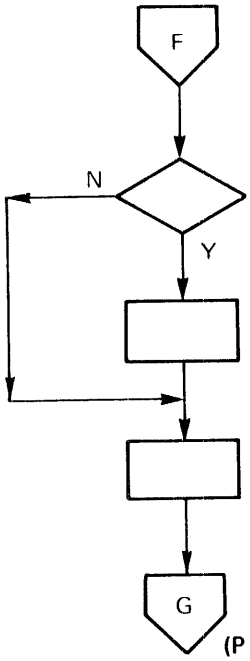
C COMMAND, CRU INSPECT/CHANGE

Listing Page: 0021



A0001571

C COMMAND, CRU INSPECT/CHANGE (Concluded)



ARE WE USING EIGHT BITS OR LESS?

ALIGN THE BYTE

COMBINE THE LOAD COMMAND WITH THE NUMBER OF BITS

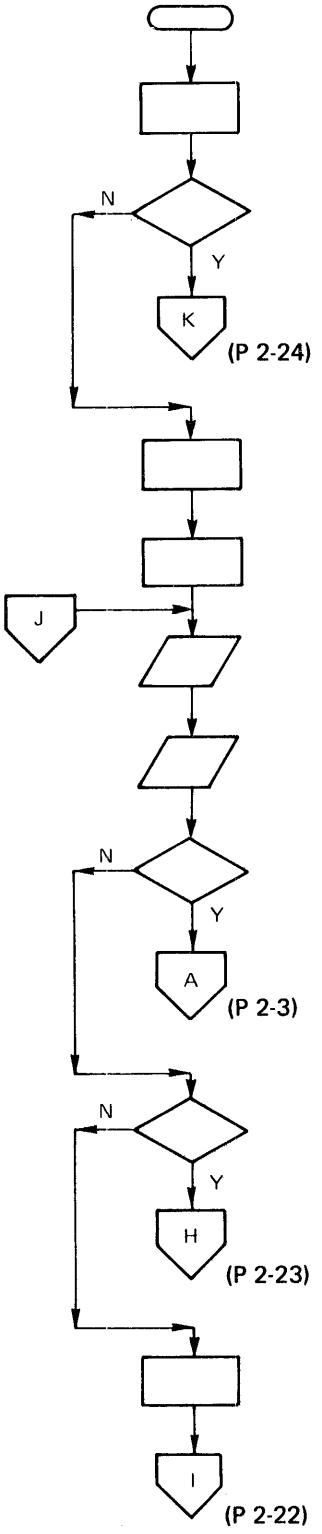
GO GET MORE CRU DATA

A0001572

(P 2-19)

W COMMAND, WORKSPACE REGISTER INSPECT/CHANGE

Listing Page: 0023



ENTER WORKSPACE REGISTER INSPECT/CHANGE

GET THE WORKSPACE POINTER

IS THE COUNT ZERO?

GO TO PRINT ALL REGISTERS

SAVE THE REGISTER NUMBER

FORM REGISTER ADDRESS

OUTPUT REGISTER CONTENTS

GET NEW CONTENTS

WAS THE DATA A CARRIAGE RETURN?

GO TO THE MONITOR

WAS THE DATA A SPACE?

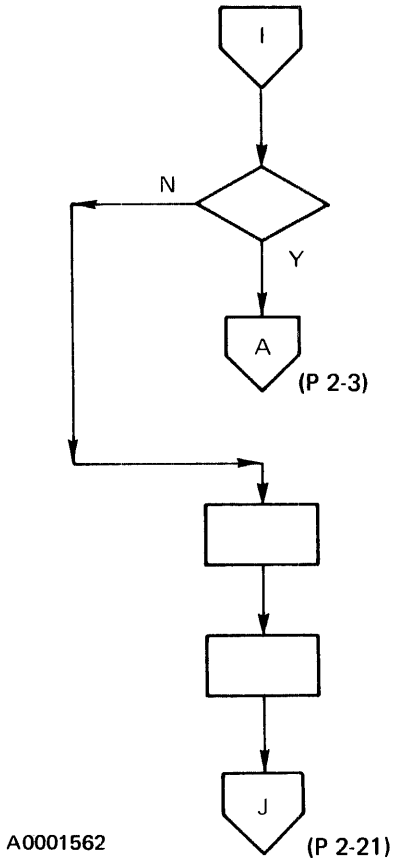
SET UP NEXT REGISTER

UPDATE REGISTER CONTENTS

SET UP PRESENT REGISTER MINUS 1

A0001561

W COMMAND, WORKSPACE REGISTER INSPECT/CHANGE (Continued)



WAS REGISTER 0 JUST UPDATED?

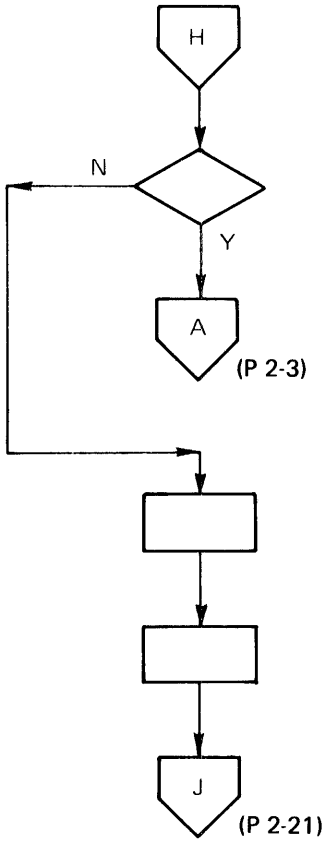
GO TO THE MONITOR

CHANGE REGISTER NUMBER TO R - 1

UPDATE REGISTER ADDRESS

GO PROCESS NEW REGISTER

A0001562



WAS REGISTER 15 JUST UPDATED?

GO TO THE MONITOR

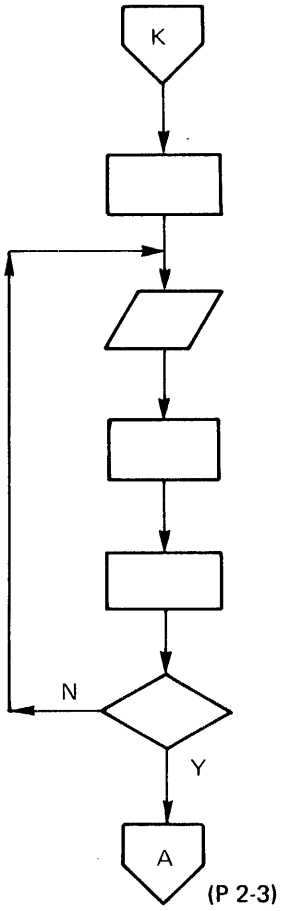
CHANGE REGISTER NUMBER TO R+1

UPDATE REGISTER ADDRESS

GO PROCESS NEW REGISTER

A0001562

W COMMAND, WORKSPACE REGISTER INSPECT/CHANGE (Concluded)



SET REGISTER NUMBER TO ZERO

PRINT CONTENTS OF THE REGISTER

UPDATE NEXT REGISTER ADDRESS

UPDATE REGISTER NUMBER

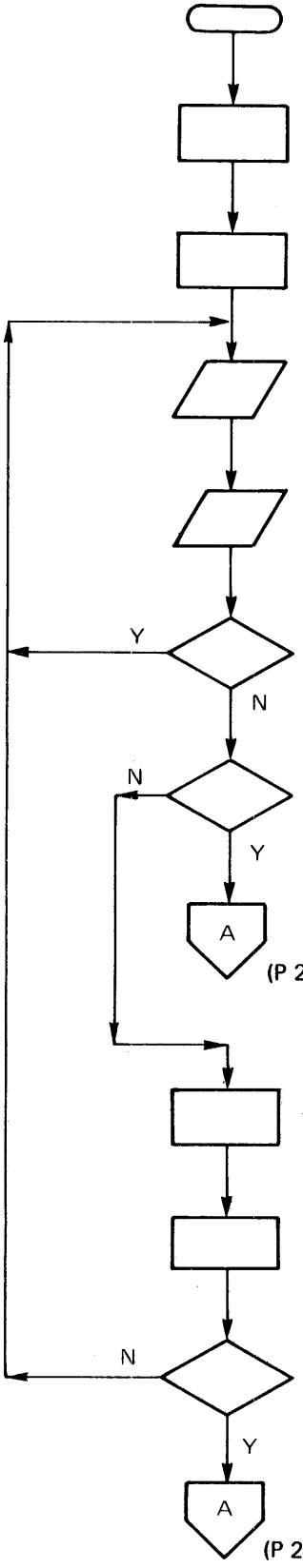
HAVE 16 REGISTERS BEEN PROCESSED?

GO TO THE MONITOR

A0001563

R COMMAND, WP, PC, AND ST REGISTERS INSPECT/CHANGE

Listing Page: 0025



ENTER DISPLAY WP, PC, ST REGISTERS

SET LOOP COUNTER TO 3

SET UP ADDRESS OF WP

OUTPUT CONTENTS OF REGISTER

INPUT NEW DATA

WAS THE DATA A MINUS SIGN?

WAS THE DATA A CARRIAGE RETURN?

GO TO THE MONITOR

UPDATE REGISTER ADDRESS

DECREMENT LOOP COUNTER

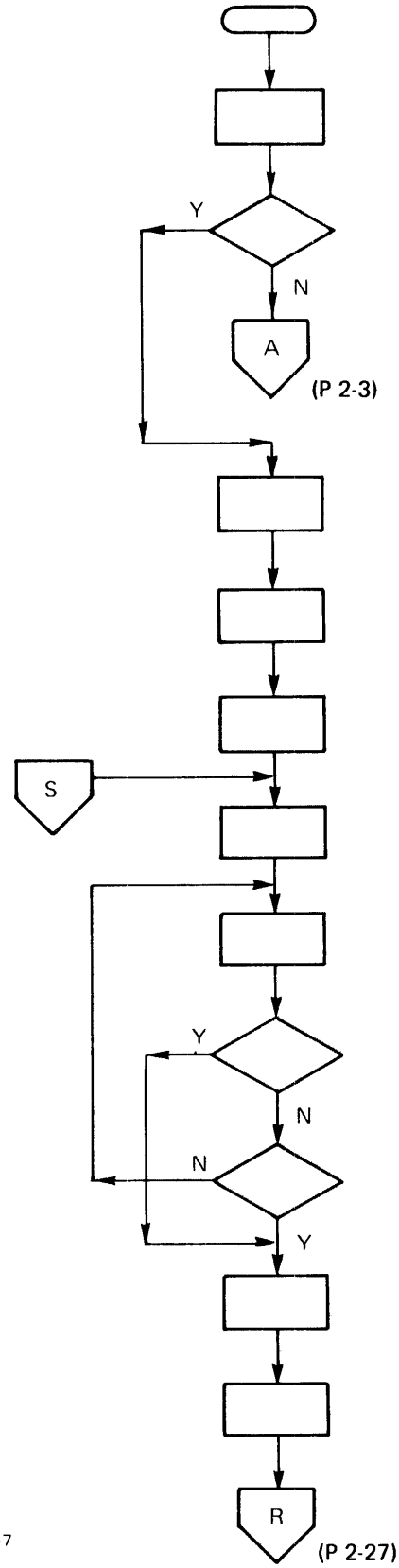
IS LOOP COUNTER ZERO?

GO TO MONITOR

A0001553

D COMMAND, TAG DUMP OF MEMORY

Listing Page: 0026



ENTER DUMP ROUTINE

GET 'IDT' FROM OPERATOR

IS OPERATOR READY?

ZERO THE CHECKSUM

OUTPUT THE IDT TO CASSETTE

UPDATE CHECKSUM

OUTPUT ENTRY ADDRESS TAG TO CASSETTE

OUTPUT MEMORY DATA TO CASSETTE

HAS THE LAST ADDRESS BEEN DUMPED?

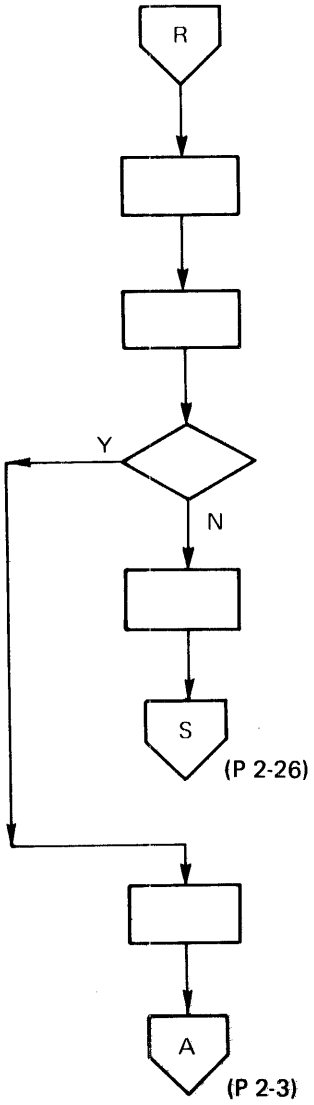
END OF RECORD?

UPDATE AND INVERT CHECKSUM

OUTPUT CHECKSUM TO CASSETTE

A0001567

D COMMAND, TAG DUMP OF MEMORY (Concluded)



ZERO THE CHECKSUM

OUTPUT F TAG AND CARRIAGE RETURN

ARE WE DONE WITH THE DUMP?

PRINT LINE FEED AND DELETE

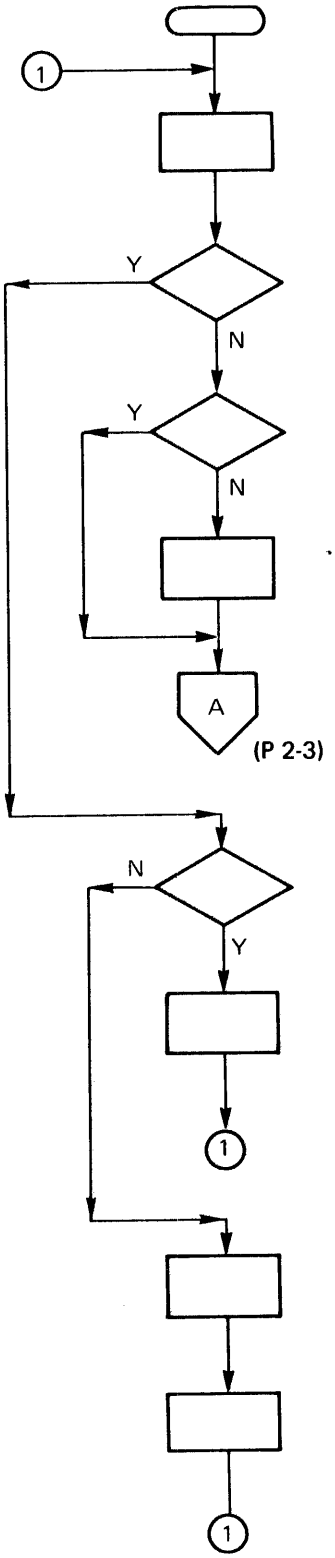
OUTPUT EOF AND NULLS TO CASSETTE

GO TO THE MONITOR

A0001568

L COMMAND, 990 TAG FORMAT LOADER

Listing Page: 0029



ENTER TAG LOADER

GET TAG OFF CASSETTE

IS THIS A VALID TAG?

IS THIS A COLON?

PRINT OUT ERROR MESSAGE

GO TO THE MONITOR

IS THIS AN 'F' TAG

READ UNTIL END OF THE RECORD

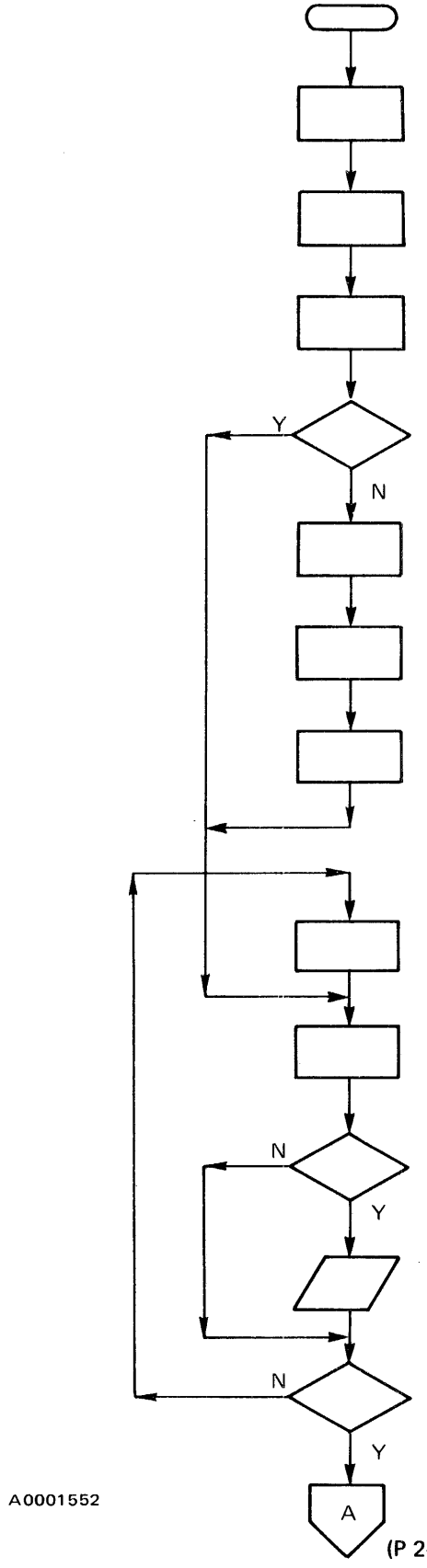
PROCESS THE TAG

GET DATA AND PUT IN MEMORY

A0001569

F COMMAND, FIND VALUE IN MEMORY

Listing Page: 0034



ENTER FIND COMMAND

WORD ALIGN STARTING ADDRESS

SET UP COMPARE COMMAND

SET UP 'INCT' COMMAND

ARE WE LOOKING FOR A WHOLE WORD?

CHANGE COMPARE COMMAND TO COMPARE BYTE

CHANGE 'INCT' COMMAND TO INC COMMAND

BYTE ALIGN DATA

EXECUTE INCREMENT COMMAND

EXECUTE COMPARE COMMAND

DID THE DATA COMPARE?

OUTPUT MATCHING ADDRESS

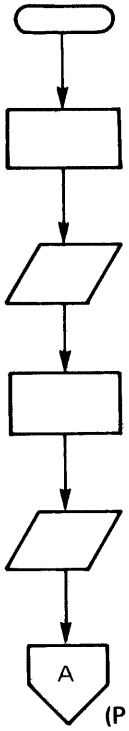
HAVE WE REACHED OUR LAST ADDRESS?

GO TO THE MONITOR

A0001552

H COMMAND, HEX ARITHMETIC

Listing Page: 0035



ENTER HEX ARITHMETIC COMMAND

ADD THE TWO NUMBERS

PRINT THE TOTAL

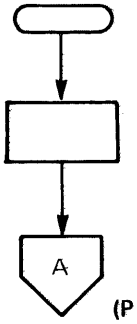
GET THE DIFFERENCE

PRINT THE DIFFERENCE

GO TO THE MONITOR

T COMMAND, SET BAUD RATE

Listing Page: 0036



ENTER T COMMAND

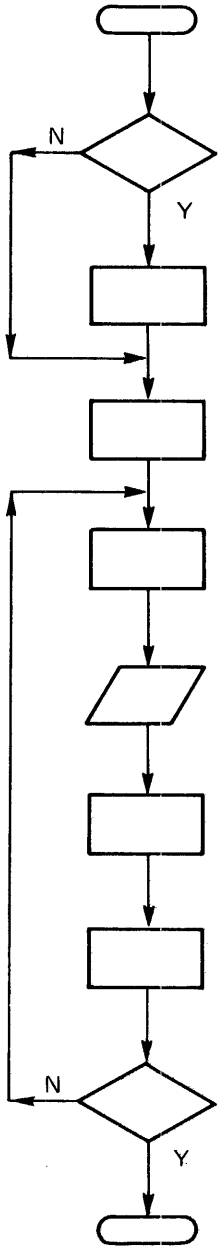
INVERT THE ASR FLAG

GO TO THE MONITOR

A0001551

XOP 0, MICROTERMINAL OUTPUT ROUTINE

Listing Page: 0037



ENTER XOP 0

DO WE USE A DELAY?

GO THROUGH TIMER LOOP

SET UP COUNTER TO 4

STRIP OUT MSB NIBBLE AND CONVERT TO ASCII

PUT OUT CHARACTER TO MICROTERMINAL ,XOP 12

SET UP NEXT NIBBLE

DECREMENT COUNTER

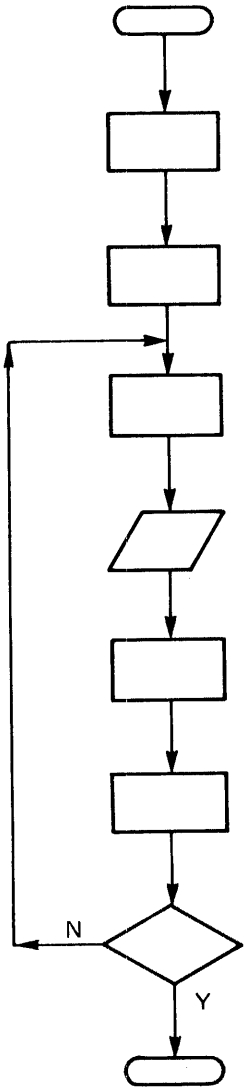
HAVE WE PUT OUT 4 HEX CHARACTERS?

RETURN

A0001544

XOP 1, MICROTERMINAL INPUT ROUTINE

Listing Page: 0038



ENTER XOP 1

SET TRANSFER COUNTER TO 4

START WITH DATA EQUAL TO ZERO

SHIFT DATA LEFT FOUR BITS (1 NIBBLE)

GET NEXT FOUR BITS FROM MICROTERMINAL, XOP 9

COMBINE NEW DATA WITH OLD DATA

DECREMENT TRANSFER COUNT

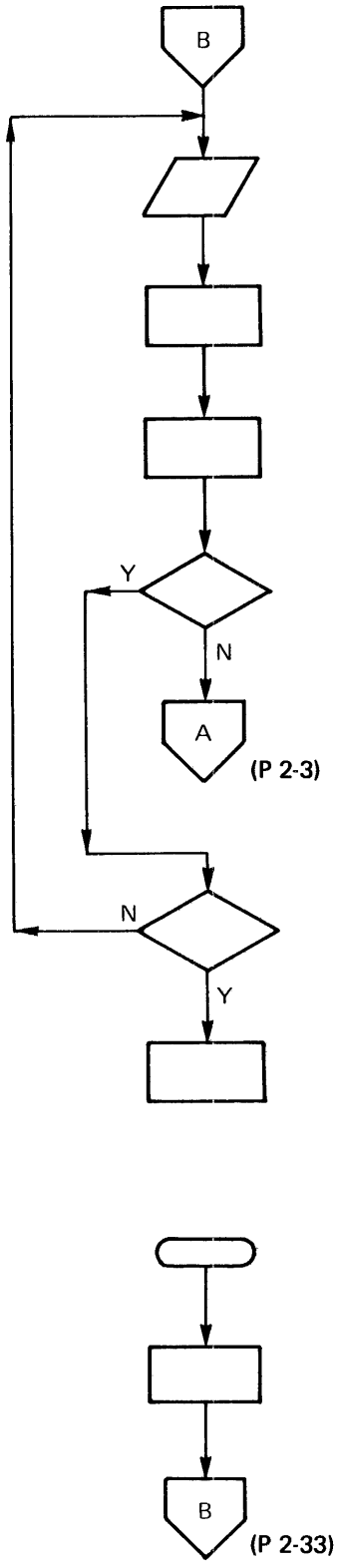
HAVE FOUR NIBBLES BEEN INPUT?

RETURN

A0001545

Z COMMAND, MICROTERMINAL COMMAND SCANNER

Listing Page: 0039



ENTER Z COMMAND FROM MONITOR

GET A CHARACTER FROM THE MICROTERMINAL XOP 9

CLEAR THE DELAY FLAG AND STEP FLAG

SET THE HALT FLAG

IS THIS REALLY THE MICROTERMINAL?

GO TO THE MONITOR

DO WE HAVE A VALID COMMAND?

GET ADDRESS OF COMMAND FROM A TABLE AND GO EXECUTE AT THAT ADDRESS

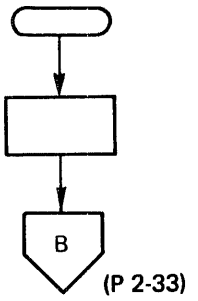
ENTER PROGRAM COUNTER (EPC)

GET PROGRAM COUNTER FROM MICROTERMINAL, XOP 1

WAIT FOR NEXT MICROTERMINAL COMMAND

A0001554

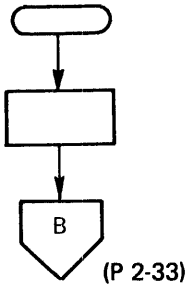
Z COMMAND, MICROTERMINAL COMMAND SCANNER (Continued)



ENTER STATUS (EST)

STORE STATUS, XOP 1

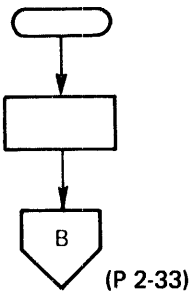
WAIT FOR NEXT MICROTERMINAL COMMAND



ENTER WORKSPACE POINTER (EWP)

STORE WORKSPACE POINTER, XOP 1

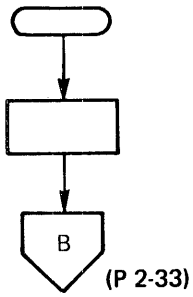
WAIT FOR NEXT MICROTERMINAL COMMAND



DISPLAY PROGRAM COUNTER (DPC)

DISPLAY PROGRAM COUNTER, XOP 0

WAIT FOR NEXT MICROTERMINAL COMMAND



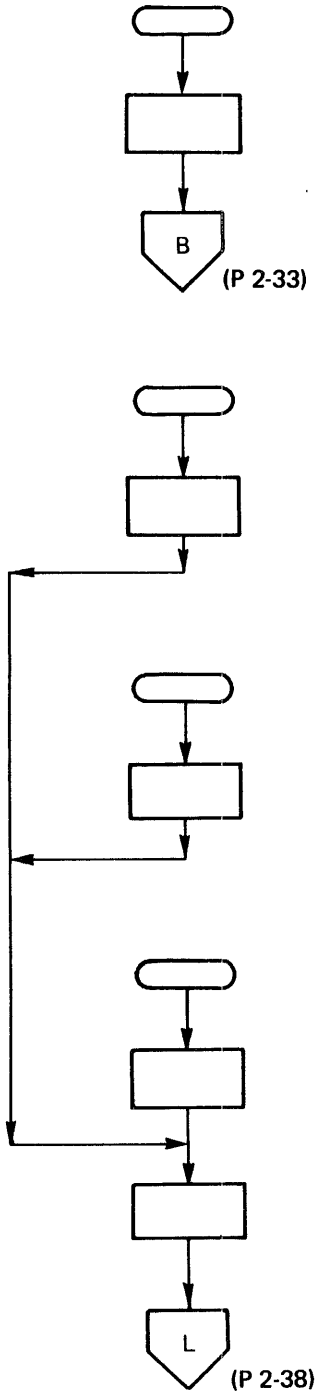
DISPLAY STATUS (DST)

DISPLAY STATUS, XOP 0

WAIT FOR NEXT MICROTERMINAL COMMAND

A0001555

Z COMMAND, MICROTERMINAL COMMAND SCANNER (Continued)



DISPLAY WORKSPACE POINTER (DWP)

OUTPUT WORKSPACE POINTER, XOP 0

WAIT FOR NEXT MICROTERMINAL COMMAND

ENTER MEMORY ADDRESS (EMA)

STORE MEMORY ADDRESS, XOP 1

ENTER MEMORY DATA (EMD)

GET DATA AND PUT IN MEMORY, XOP 1

ENTER MEMORY DATA AND INCREMENT ADDRESS (EMDI)

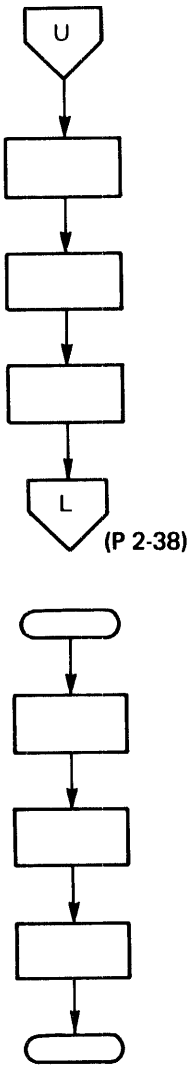
GET DATA FROM MICROTERMINAL AND PUT IN MEMORY PLUS INCREMENT ADDRESS, XOP 1

GET DATA OUT OF MEMORY

GO OUTPUT DATA TO MICROTERMINAL, XOP 0

A0001556

Z COMMAND, MICROTERMINAL COMMAND SCANNER (Continued)



STEP RT

OUTPUT PROGRAM COUNTER TO MICROTERMINAL, XOP 0

GET THE DATA AT THE PC ADDRESS

SET THE DELAY FLAG

GO OUTPUT DATA TO THE MICROTERMINAL

STEP

SET THE STEP FLAG

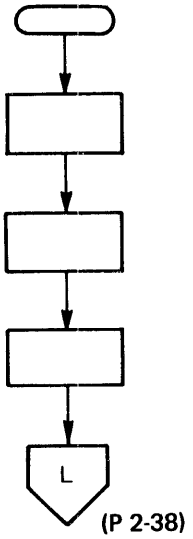
CLEAR THE HALF FLAG

EXECUTE ONE INSTRUCTION

RETURN

A0001557

Z COMMAND, MICROTERMINAL COMMAND SCANNER (Continued)



DISPLAY CRU (DCRU)

GET CONTROL WORD FROM THE MICROTERMINAL , XOP 1

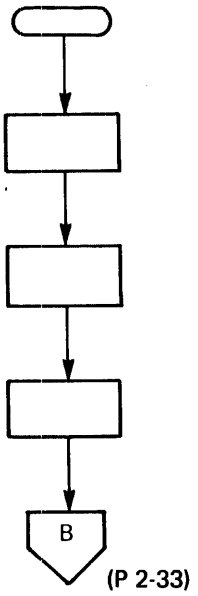
CONSTRUCT STCR COMMAND

EXECUTE STCR COMMAND

GO OUTPUT DATA TO THE MICROTERMINAL

A0001557

Z COMMAND, MICROTERMINAL COMMAND SCANNER (Concluded)



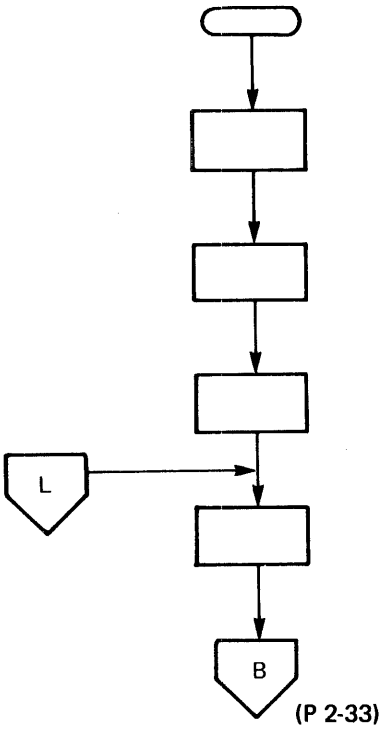
ENTER CRU (ECRU)

GET CRU DATA FROM MICROTERMINAL, XOP 1

SET UP LDCR COMMAND

EXECUTE LDCR COMMAND

WAIT FOR NEXT MICROTERMINAL COMMAND



HALT

SET THE DELAY FLAG

OUTPUT THE PROGRAM COUNTER, XOP 0

GET THE DATA OUT OF MEMORY

OUTPUT THE DATA , XOP 0

WAIT FOR NEXT MICROTERMINAL COMMAND

A0001558

SECTION 3

TIBUG LISTING

The TIBUG listing is arranged in the following order:

```
TIBUG          SDSMAC 947075 *B    16:35:23 FRIDAY, JUN 17, 1977.
*** SYSTEM DEFINITION ***                                    PAGE 0001

0003                    IDT  'TIBUG'
0004              *
0005              * TIBUG WORKSPACE DEFINITION
0006              *
0007       FFB0   MREGS  EQU  >FFB0
0008       FFBA   EREGS  EQU  >FFBA
0009       FFC6   IREGS  EQU  >FFC6
0010       FFD2   DUMYBF EQU  >FFD2
0011       FFD4   XREGS  EQU  >FFD4
0012       FFF4   ASR    EQU  >FFF4
0013       FFF6   DUMPFG EQU  >FFF6
0014       FFF8   STEPFG EQU  >FFF8
0015       FFFA   HALTFG EQU  >FFFA
0016              *
0017              * USER RAM IS FROM >FE00 TO >FFAE
0018              *
0019              * TIBUG RESERVED RAM IS FROM >FFB0 TO >FFFE
0020              *
0021              * WORKSPACE UTILIZATION
0022              *
0023              * MREGS (MONITOR SCANNER WORKSPACE)
0024              * ADDRESSES FFB0-FFCE (16 REGISTERS)
0025              * MODULES WHICH UTILIZE:
0026              ****** MONTP - SYSTEM INZ AND COMMAND SCANNER
0027              *     * MIC     = MEMORY INSPECT/CHANGE
0028              *     * EXEC    = EXECUTE/SINGLE STEP/BREAKPOINT
0029              *     * CENTRY  = CRU INSPECT/CHANGE
0030              *     * WENTY   = USER WORKSPACE INSPECT/CHANGE
0031              *     * RENTY   = STATUS INSPECT/CHANGE
0032              *     * DENTRY  = DUMP MEMORY TO CASSETTE
0033              *     * LENTRY  = LOAD MEMORY TO CASSETTE
0034              *     * FIND    = FIND WORD/BYTE
0035              *     * ARITH   = HEX ARITHMETIC
0036              ****** TENTRY - TOGGLE TERMINAL FLAG
0037              *
0038              * ALL MODULES WHICH USE MREGS
0039              * ARE TIBUG COMMAND PROCESSORS
0040              *
0041              * EREGS - ECHO ROUTINE WORKSPACE
0042              * ADDRESSES FFBA-FFC0 (4 REGISTERS)
0043              *
0044              * IREGS - I/O WORKSPACE
0045              * ADDRESSES FFC6-FFD0 (6 REGISTERS)
0046              * UTILIZED BY:
0047              ****** READ - READ 1 ASCII CHARACTER FROM TMS9902
0048              ****** WRITE - WRITE 1 ASCII CHARACTER TO TMS9902
0049              *
0050              * XREGS - XOP ROUTINES WORKSPACE
0051              * ADDRESSES FFD4-FFE0 (7 REGISTERS)
0052              * UTILIZED BY ALL XOP PROCESSORS EXCEPT 'READ'
0053              * AND 'WRITE'
0054              *
0055              ************************************************
```

```
0057                    ****************************************
0058                    * INTERRUPT VECTORS
0059 0000 FFB0          DATA MREGS,INIT
     0002 014E
0060 0004 FFFF          DATA -1,-1,-1,-1
     0006 FFFF
     0008 FFFF
     000A FFFF
0061 000C FF68          DATA >FF68,>FF88,>FF8C,>FFAC
     000E FF88
     0010 FF8C
     0012 FFAC
0062 0014 FFFF          DATA -1,-1,-1,-1,-1,-1,-1,-1,-1,-1
     0016 FFFF
     0018 FFFF
     001A FFFF
     001C FFFF
     001E FFFF
     0020 FFFF
     0022 FFFF
     0024 FFFF
     0026 FFFF
0063 0028 FFFF          DATA -1,-1,-1,-1,-1,-1,-1,-1,-1,-1
     002A FFFF
     002C FFFF
     002E FFFF
     0030 FFFF
     0032 FFFF
     0034 FFFF
     0036 FFFF
     0038 FFFF
     003A FFFF
0064 003C FFFF          DATA -1,-1
     003E FFFF
```

EPROM MA

000 →

000 →

0A

0B

NOT TO BE USED

INT VECT 0, 3, 4 USED

INT 1- EPROM BASE

WP = 02

PC 03

XOPS

```
TIBUG          SDSMAC 947075 *B    16:35:23 FRIDAY, JUN 17, 1977.
*** SYSTEM DEFINITION ***                                  PAGE 0003

0066                    * XOP VECTORS
0067 0040 FFD4 20       DATA XREGS,OTPTEN       XOP 0 = MICRO TERMINAL OUTPUT
     0042 06FE 21
0068 0044 FFD4 22       DATA XREGS,INPTEN       XOP 1 = MICRO TERMINAL INPUT
     0046 0724 23
0069 0048 FFFF 24       DATA -1,-1,-1,-1        XOPS 2-7 = NOT DEFINED
     004A FFFF 25
     004C FFFF 26
     004E FFFF 27
0070 0050 FFFF 28       DATA -1,-1,-1,-1,-1,-1,-1,-1
     0052 FFFF 29
     0054 FFFF 2A
     0056 FFFF 2B
     0058 FFFF 2C
     005A FFFF 2D
     005C FFFF 2E
     005E FFFF 2F
0071 0060 FFD4          DATA XREGS,WHXETY       XOP 8 = WRITE 1 HEX DIGIT
     0062 0332'
0072 0064 FFD4          DATA XREGS,RHENTY       XOP 9 = HEX # INPUT
     0066 02CE'
0073 0068 FFD4          DATA XREGS,WHENTY       XOP 10 = HEX # OUTPUT
     006A 033C'
0074 006C FFBA          DATA EREGS,ECHOEN       XOP 11 = ECHO
     006E 0258'
0075 0070 FFC6          DATA IREGS,WENTRY       XOP 12 = WRITE CHARACTER
     0072 01B6'
0076 0074 FFC6          DATA IREGS,RENTRY       XOP 13 = READ CHARACTER
     0076 01A6'
0077 0078 FFD4          DATA XREGS,MENTRY       XOP 14 = MESSAGE OUT
     007A 025E'
0078 007C FFB0          DATA MREGS,XOPENT       XOP 15
     007E 0392'
```

Can use 2 thru 7 XOP Loc.

MA 0048 → 005E

EPROM MF


```
0081                    ********************************************
0082                    * SYSTEM INT AND COMMAND SCANNER
0083                    ********************************************
0084                            DXOP OTPT,0
0085                            DXOP INPT,1
0086                            DXOP WHX1,8
0087                            DXOP RHEX,9
0088                            DXOP ECHO,11
0089                            DXOP MESG,14
0090                            DXOP READ,13
0091          0000      START   EQU  0
0092          0000      EREG    EQU  0
0093          0001      STOP    EQU  1
0094          0002      KEY     EQU  2
0095          0003      COUNT   EQU  3
0096          0004      VALUE   EQU  4
0097          0005      CHAR    EQU  5
0098          0006      ICOUNT  EQU  6
0099          0007      POINT   EQU  7
0100          0008      WDBDY   EQU  8
0101          000B      LINK    EQU  11
0102          000C      CRUBAS  EQU  12
0103                    *
0104                    * MONITOR ENTRY POINT >0080
0105                    *
0106          0080'     MONTOP  EQU  $              COMMAND SCAN ENTRY
0107 0080 02E0                    LWPI MREGS        INT WP
     0082 FFB0
0108 0084 04C1                    CLR  R1           REG 1 = 0
0109 0086 0202                    LI   R2,>FFFC     REG 2 = FFFC
     0088 FFFC
0110 008A CCB1                    MOV  *R1+,*R2+    PUT ADDR OF MREGS IN FFFC
0111 008C 0201                    LI   R1,LOAD      REG 1 =ADDR OF LOAD FOR SINGLE STEP
     008E 0368'
0112 0090 CC81                    MOV  R1,*R2+      PUT ADDR OF INIT IN FFFE
0113 0092 0209    TICKO   LI   9,MONTOP     INT RETURN POINTER
     0094 0080'
0114 0096 04C1                    CLR  R1           SET UP TICK COUNT
0115 0098 0641    TICK1   DECT R1             DEC COUNTER, 32 K LOOPS
0116 009A 16FE                    JNE  TICK1        IF NOT DONE, JUMP BACK
0117 009C C309                    MOV  R9,R12       REG 12 = 80 = CRU OF 9902
0118 009E 1F15                    TB   21           IS RECIEVE BUFFER REG FULL
0119 00A0 1606                    JNE  SCAN         IF NO, GO ON
0120 00A2 2F45                    READ CHAR         IF YES, GET CHAR FROM U TERMINAL
0121 00A4 0285                    CI   CHAR,>5A00   DO WE HAVE A 'Z', I.E. MICROTERM
     00A6 5A00
0122 00A8 1602                    JNE  SCAN         IF NO WAIT FOR REG COMMAND
0123 00AA 0460                    B    @MTIN        GO TO MICRO TERMINAL DSR
     00AC 073E'
0124 00AE 2FA0    SCAN    MESG @PROMPT      OUTPUT PROMPT
     00B0 0227'
0125                    *
0126                    * INITIALIZE START, STOP, KEY, ETC...
0127                    *
0128 00B2 04C0                    CLR  START
0129 00B4 04C3                    CLR  COUNT
0130 00B6 0208                    LI   WDBDY,1      WORD BOUNDRY REG
     00B8 0001
0131                    *
0132                    * WAIT FOR A COMMAND ENTRY
```

0040 →

```
0133                    *
0134 00BA 2EC5          ECHO  CHAR          RECEIVE AND ECHO A CHARACTER
0135 00BC 2FA0          MESG  @SPACE1       OUTPUT SPACE
     00BE 01FC'
0136 00C0 06A0          BL    @SRCH
     00C2 00FA'
0137                    *
0138                    * COMMAND SEARCH TABLE
0139                    *
0140 00C4   4D          TEXT 'M'            MEMORY INSPECT/CHANGE
0141 00C5   03          BYTE 3              2 POSSIBLE INPUTS
0142 00C6 0268'         DATA M              ENTRY POINT
0143 00C8   57          TEXT 'W'            USER STATUS INSPECT/CHANGE
0144 00C9   01          BYTE 1              ONE HEX INPUTS
0145 00CA 03FE'         DATA W
0146 00CC   45          TEXT 'E'            EXECUTE
0147 00CD   00          BYTE 0
0148 00CE 0366'         DATA E
0149 00D0   42          TEXT 'B'            EXECUTE WITH BREAKPOINT
0150 00D1   01          BYTE 1              ONE HEX INPUT
0151 00D2 0386'         DATA  B
0152 00D4   53          TEXT 'S'            EXECUTE SINGLE STEP
0153 00D5   00          BYTE 0
0154 00D6 0360'         DATA S
0155 00D8   4C          TEXT 'L'            LOAD MEMORY FROM CASSETTE
0156 00D9   01          BYTE 1
0157 00DA 059A'         DATA L
0158 00DC   44          TEXT 'D'            DUMPIMEMORY TO CASSETTE
0159 00DD   07          BYTE 7              3 HEX INPUTS
0160 00DE 04B0'         DATA D
0161 00E0   43          TEXT 'C'            CRU INSPECT/CHANGE
0162 00E1   03          BYTE 3
0163 00E2 03B0'         DATA C
0164 00E4   52          TEXT 'R'            USER WORKSPACE INSPECT/CHANGE
0165 00E5   00          BYTE 0
0166 00E6 0474'         DATA R
0167 00E8   46          TEXT 'F'            FIND BYTE/WORD
0168 00E9   07          BYTE 7
0169 00EA 06B6'         DATA F
0170 00EC   48          TEXT 'H'            HEX ARITHMETIC
0171 00ED   03          BYTE 3
0172 00EE 06E4'         DATA H
0173 00F0   54          TEXT 'T'            733ASR TERMINAL COMMAND
0174 00F1   00          BYTE 0
0175 00F2 06F8'         DATA T
0176 00F4 0000          DATA 0              END OF TABLE
0177                  *
0178                  * COMMAND SEARCH ROUTINE
0179                  *
0180 00F6 022B  SRCHLP AI   LINK,3          UPDATE POINTER
     00F8 0003
0181 00FA C29B  SRCH   MOV  *LINK,10        SEARCH FAIL?
0182 00FC 1322         JEQ  ERR4            YES, ERROR
0183 00FE 917B         CB   *LINK+,CHAR     INPUT MATCH TABLE ENTRY?
0184 0100 16FA         JNE  SRCHLP          NO, TO NEXT TABLE ENTRY
0185 0102 D1BB         MOVB *LINK+,ICOUNT NUMBER OF HEX INPUT FIELDS
0186 0104 C2DB         MOV  *LINK,LINK      GET ENTRY ADDRESS
0187                  *
0188                  * ICOUNT SPECIFIES NUMBER OF HEX INPUT FIELDS
0189                  *
```

```
0190 0106 0986          SRL   ICOUNT,8      ALIGN COUNT
0191 0108 0207          LI    POINT,MREGS   INT POINTER
     010A FFB0
0192 010C 0916  INLOOP  SRL   ICOUNT,1      DONE?
0193 010E 170D          JNC   CEXIT         IF YES, GO TO COMMAND SCANNER
0194 0110 2E44  HEXIN   RHEX  VALUE         ACCEPT HEX ENTRY
0195 0112 0122'         DATA  NULL,ERR2
     0114 0136'
0196 0116 CDC4          MOV   VALUE,*POINT+ SAVE HEX INPUT
0197 0118 0583  CNT     INC   COUNT         COUNTI# ENTRIES
0198 011A 0285          CI    CHAR,>0D00    END OF INPUT?
     011C 0D00
0199 011E 1305          JEQ   CEXIT         YES, TO COMMAND PROCESSOR
0200 0120 10F5          JMP   INLOOP        WAIT FOR NEXT INPUT
0201 0122 05C7  NULL    INCT  POINT         UPDATE POINTER
0202 0124 0285          CI    CHAR,>0D00    NO INPUT?
     0126 0D00
0203 0128 16F7          JNE   CNT           NO, DEFAULT PARAMETER
0204 012A 045B  CEXIT   B     *LINK         YES, TO COMMAND PROCESSOR
0205                    *
0206                    * ERROR HANDLER
0207                    *
0208 012C 04C0  ERR0    CLR   EREG          LOAD ERROR-INVALID TAG
0209 012E 100B          JMP   ERROR
0210 0130 0200  ERR1    LI    EREG,1        LOAD ERROR-CHECKSUM
     0132 0001
0211 0134 1008          JMP   ERROR
0212 0136 0200  ERR2    LI    EREG,2        TERM. CHARACTER ERROR
     0138 0002
0213 013A 1005          JMP   ERROR
0214 013C 0200  ERR3    LI    EREG,3        DUMP ADDRESS ERROR
     013E 0003
0215 0140 1002          JMP   ERROR
0216 0142 0200  ERR4    LI    EREG,4        INVALID COMMAND ERROR
     0144 0004
0217 0146 2FA0  ERROR   MESG  @ERROUT       PRINT ERROR BANNER
     0148 01F0'
0218 014A 2E00          WHX1  EREG          PRINT ERROR NUMBER
0219 014C 1099  JMMONT  JMP   MONTOP
0220                    *
0221                    * UART INITIALIZATION ROUTINE--USER INPUTS
0222                    * ONE 'A'. BAUD RATE DETECTED FROM LENGTH
0223                    * OF THE START BIT.
0224                    *
0225 014E 020C  INIT    LI    R12,ASR       REG 12 = STARTING ADDR
     0150 FFF4
0226 0152 04FC          CLR   *R12+         CLEAR ASR
0227 0154 073C          SETO  *R12+         RESET DUMP FLAG
0228 0156 04FC          CLR   *R12+         CLEAR STEPFG
0229 0158 04DC          CLR   *R12          CLEAR HALTFG
0230 015A 020C          LI    CRUBAS,>80    SET CRU BASE REG.
     015C 0080
0231                    *
0232                    * INITIALIZE TMS9902 FOR: *BAUD RATE
0233                    *                         *7 BITS/CHARACTER
0234                    *                         *EVEN PARITY
0235                    *                         *2 STOP BITS
0236                    *                         *POLLED OPERATION
0237                    *
0238 015E 1D1F          SBO   31            RESET TMS9902 UART
```

```
0239 0160 3220          LDCR @CR,8          INITIALIZE TMS9902 CONTROL REG.
     0162 01A4'
0240 0164 1E0D          SBZ  13             DO NOT INT INTERVAL REG.
0241 0166 04C3          CLR  COUNT          RESET LOOP COUNT
0242 0168 1F0F  TSTSP   TB   15             SPACE?
0243 016A 13FE          JEQ  TSTSP          NO, JUMP BACK
0244 016C 0583  SPLOOP  INC  COUNT          TIME THE START BIT
0245 016E 1F0F          TB   15             FALL OUT ON A MARK
0246 0170 16FD          JNE  SPLOOP
0247              *
0248              * TABLE SEARCH FOR BAUD RATE
0249              *
0250 0172 0207          LI   POINT,TABLE    SET POINTER TO TABLE
     0174 0194'
0251 0176 8DC3  BDLOOP  C    COUNT,*POINT+  MATCH?
0252 0178 1202          JLE  MATCH          YES, SET BAUD RATE
0253 017A 05C7          INCT POINT          NO, UPDATE POINTER
0254 017C 10FC          JMP  BDLOOP
0255      017E' MATCH   EQU  $
0256 017E 3317          LDCR *POINT,12      INT. REC./XMT. DATA RATE
0257 0180 C1D7          MOV  *POINT,POINT
0258 0182 0287          CI   POINT,>1A0     1200 BAUD ?
     0184 01A0
0259 0186 1602          JNE  BANNER         LEAVE ASR FLAG ALONE
0260 0188 0720          SETO @ASR           SET 733ASR FLAG
     018A FFF4
0261 018C 2F45  BANNER  READ CHAR
0262 018E 2FA0          MESG @LOGON         PRINT LOG ON MESSAGE
     0190 022B'
0263 0192 10DC          JMP  JMMONT         TO TOP OF MONITOR
0264 0194 0040  TABLE   DATA >40,>D0        2400 BAUD
     0196 00D0
0265 0198 0070          DATA >70,>1A0       1200 BAUD
     019A 01A0
0266 019C 0200          DATA >200,>4D0      300 BAUD
     019E 04D0
0267 01A0 0400          DATA >400,>638      110 BAUD
     01A2 0638
0268 01A4   62  CR      BYTE >62
```

```
0271                ************************************************
0272                * READ CHARACTER -- XOP  R,13
0273                *                          --       NORMAL RETURN
0274                *
0275                * READ WAITS FOR A CHARACTER TO BE ASSEMBLED IN
0276                * THE UART. THE CHARACTER IS PLACED IN THE LEFT
0277                * BYTE OF USER REGISTER R. THE RIGHT BYTE IS
0278                * ZEROED. ALL ERRORS ARE IGNORED.
0279                ************************************************
0280                *
0281 01A6 020C     RENTRY LI    CRUBAS,>80    SET CRU BASE REG.
     01A8 0080
0282 01AA 1F15            TB    21            RECEIVE BUFFER REG. FULL?
0283 01AC 16FC            JNE   RENTRY        NO, LOOP
0284 01AE 04DB            CLR   *LINK
0285 01B0 361B            STCR  *LINK,8
0286 01B2 1E12            SBZ   18
0287 01B4 0380            RTWP
```

```
0290                    ************************************************
0291                    * WRITE CHARACTER -- XOP  R,12
0292                    *                              ---      NORMAL RETURN
0293                    *
0294                    * TRANSMIT THE CHARACTER IN THE LEFT BYTE OF
0295                    * USER REGISTER R. IF THE CHARACTER IS A
0296                    * CARRIAGE RETURN, THE ROUTINE WAITS 200 MSEC FOR
0297                    * THE CARRIAGE TO RETURN. IF THE TERMINAL IS
0298                    * A 733ASR AS DENOTED IN THE T COMMAND, EACH
0299                    * CHARACTER IS PADDED WITH 25 MSEC TO REDUCE
0300                    * THE TRANSFER RATE TO 300 BAUD.
0301                    ************************************************
0302 01B6 020A  WENTRY LI   R10,3750
     01B8 0EA6
0303 01BA 020C         LI   CRUBAS,>80    SET CRU BASE REG.
     01BC 0080
0304 01BE 1D10         SBO  16            SET RTSON
0305 01C0 1F16         TB   22            TRANSMIT BUFFER REG. EMPTY?
0306 01C2 16F9         JNE  WENTRY        NO, WAIT UNTIL IT IS
0307 01C4 321B         LDCR *LINK,8       CHARACTER TO UART
0308 01C6 D2DB         MOVB *LINK,LINK
0309 01C8 1E10         SBZ  16            RESET RTSON
0310 01CA 098B         SRL  LINK,8
0311 01CC 028B         CI   LINK,>000D    CARRIAGE RETURN
     01CE 000D
0312 01D0 1608         JNE  ASR733        NO, SKIP
0313 01D2 0A3A         SLA  R10,3
0314 01D4 1F16  WLOOP1 TB   22            WAIT FOR XMISSION TO END
0315 01D6 16FE         JNE  WLOOP1
0316 01D8 1F17         TB   23
0317 01DA 16FC         JNE  WLOOP1
0318 01DC 060A  WLOOP2 DEC  R10           WAIT LOOP
0319 01DE 16FE         JNE  WLOOP2
0320 01E0 0380         RTWP
0321 01E2 C2E0  ASR733 MOV  @DUMPFG,LINK IN DUMP ROUTINE ?
     01E4 FFF6
0322 01E6 1303         JEQ  WEXIT         YES,IGNORE ASR FLAG
0323 01E8 C2E0         MOV  @ASR,LINK     ASR733 ?
     01EA FFF4
0324 01EC 16F3         JNE  WLOOP1        YES, WAIT 3 NULLS
0325 01EE 0380  WEXIT  RTWP
```

```
0328                    ********************************************************
0329                    * SYSTEM MESSAGES FOR 'MESG'
0330                    ********************************************************
0331                    *
0332                    * MONITOR MESSAGES
0333                    *
0334 01F0   0D  ERROUT BYTE >D,>A
     01F1   0A
0335 01F2   45          TEXT 'ERROR '
     01F3   52
     01F4   52
     01F5   4F
     01F6   52
     01F7   20
0336 01F8   00          BYTE 0
0337 01F9   20  SPACE4 TEXT '   '
     01FA   20
0338 01FB   20  SPACE2 TEXT ' '
0339 01FC   20  SPACE1 TEXT ' '
0340 01FD   00          BYTE 0
0341 01FE   0D  BPMSG  BYTE >D,>A
     01FF   0A
0342 0200   42          TEXT 'BP'
     0201   50
0343 0202   00          BYTE 0
0344 0203   49  IDTEQ  TEXT 'IDT='
     0204   44
     0205   54
     0206   3D
0345 0207   00          BYTE 0
0346 0208   0D  READY  BYTE >D,>A
     0209   0A
0347 020A   52          TEXT 'READY Y/N '
     020B   45
     020C   41
     020D   44
     020E   59
     020F   20
     0210   59
     0211   2F
     0212   4E
     0213   20
0348 0214   00          BYTE 0
0349 0215   46  FCR     TEXT 'F'
0350 0216   0D          BYTE >D,0
     0217   00
0351 0218   0A  EOF     BYTE >A,>7F,>3A,>D,>A,>7F,>13,>D,>14
     0219   7F
     021A   3A
     021B   0D
     021C   0A
     021D   7F
     021E   13
     021F   0D
     0220   14
0352 0221   7F  RUBOUT BYTE >7F,0
     0222   00
0353 0223   12  DC2     BYTE >12
0354 0224   0A  NR      BYTE >A,>7F,0
     0225   7F
```

```
      0226   00
0355  0227   0D  PROMPT BYTE >D,>A
      0228   0A
0356  0229   3F        TEXT '?'
0357  022A   00        BYTE 0
0358  022B   0D  LOGON  BYTE >D,>A
      022C   0A
0359  022D   54        TEXT 'TIBUG REV.A'
      022E   49
      022F   42
      0230   55
      0231   47
      0232   20
      0233   52
      0234   45
      0235   56
      0236   2E
      0237   41
0360  0238   0D  CRLF   BYTE >D,>A
      0239   0A
0361  023A 000A  TEN    DATA 10
0362  023C   57  WS     TEXT 'W'
0363  023D   00        BYTE 0
0364  023E   50        TEXT 'P'
0365  023F   00        BYTE 0
0366  0240   53        TEXT 'S'
0367  0241   00        BYTE 0
0368  0242   0D  CRLFR  BYTE >D,>A
      0243   0A
0369  0244   52  RP     TEXT 'R'
0370  0245   00        BYTE 0
0371  0246   0D  HP     BYTE >0D,>0A
      0247   0A
0372  0248   48        TEXT 'H1+H2='
      0249   31
      024A   2B
      024B   48
      024C   32
      024D   3D
0373  024E   00        BYTE 0
0374  024F   20  HM     TEXT '  H1-H2'
      0250   20
      0251   48
      0252   31
      0253   2D
      0254   48
      0255   32
0375  0256   3D  EQUSGN TEXT '='
0376  0257   00        BYTE 0
```

```
0379              ************************************************************
0380              *
0381              * READ A CHARACTER AND ECHO IT TO THE TERMINAL
0382              *                 (XOP R,11)
0383              *
0384              * CALLING SEQUENCE:  ECHO R
0385              *                            ___     NORMAL RETURN
0386              *
0387              ************************************************************
0388                    DXOP WRIT,12
0389 0258 2F5B  ECHOEN READ *LINK         READ CHARACTER
0390 025A 2F1B        WRIT *LINK         ECHO THE CHARACTER
0391 025C 0380        RTWP
```

```
0394                ****************************************************************
0395                *
0396                * MESSAGE OUTPUT  (XOP @MESSAGE,14)
0397                *
0398                * CALLING SEQUENCE: MESG @MESSAGE
0399                *                                ---        NORMAL RETURN
0400                *
0401                * OUTPUT THE ASCII STRING POINTED TO BY THE ADDRESS IN
0402                * R11. OUTPUT IS TERMINATED WHEN A ZERO IS ENCOUNTERED.
0403                *
0404                ****************************************************************
0405         000C   BUFFER EQU  12
0406                *
0407                *
0408 025E D33B  MENTRY MOVB *R11+,BUFFER      GET THE CHARACTER
0409 0260 1302         JEQ  EXIT              IF 0, EXIT
0410 0262 2F0C         WRIT BUFFER            OUTPUT CHARACTER
0411 0264 10FC         JMP  MENTRY
0412 0266 0380  EXIT   RTWP                   RETURN
```

```
0415                ***********************************************
0416                *
0417                * INSPECT/CHANGE MEMORY - 'M' COMMAND
0418                *
0419                * OPTIONS:
0420                *           1) START ADDRESS, CARRIAGE RETURN --
0421                *              DISPLAY ADDRESS, CONTENTS, AND
0422                *              OPEN THE MEMORY LOCATION FOR A CHANGE.
0423                *
0424                *           2) CARRIAGE RETURN -- SAME AS 1) BUT THE
0425                *              DEFAULT START ADDRESS IS 0000.
0426                *
0427                *           3) START ADDRESS, BLANK (OR COMMA), STOP
0428                *              STOP ADDRESS, CARRIAGE RETURN -- OUTPUT
0429                *              MEMORY CONTENTS FROM START ADDRESS TO
0430                *              STOP ADDRESS. DEFAULT VALUES FOR BOTH
0431                *              ADDRESSES ARE 0000.
0432                *
0433                ***********************************************
0434       0000     STARTA EQU  0
0435       0001     STOPA  EQU  1
0436       0005     TCHAR  EQU  5
0437                       DXOP' WHEX,10
0438                *
0439                *
0440 0268 4008      M      SZC  WDBDY,STARTA       WORD ALIGN START ADDRESS
0441 026A 4048             SZC  WDBDY,STOPA  WORD ALIGN STOP ADDRESS
0442 026C 0603             DEC  COUNT        1 INPUT?
0443 026E 131E             JEQ  MIC          YES, TO MEMORY INSPECT/CHANGE
0444                *
0445                * MEMORY DUMP ROUTINE
0446                *
0447 0270 0203      MLOOP1 LI   COUNT,8
     0272 0008
0448 0274 2FA0             MESG @CRLF        NEXT LINE
     0276 0238'
0449 0278 2E80             WHEX STARTA       PRINT ADDRESS OF FIRST LOCATION
0450 027A 2FA0             MESG @EQUSGN      DELIMITER
     027C 0256'
0451 027E 2E90      MLOOP2 WHEX *STARTA      PRINT MEMORY CONTENTS
0452 0280 1F15             TB   21           IS A CHARACTER IN ?
0453 0282 1324             JEQ  MEXIT        IF A KEY IS DEPRESSED, GET OUT
0454 0284 8040             C    STARTA,STOPA DONE?
0455 0286 1322             JEQ  MEXIT        YES, EXIT
0456 0288 05C0             INCT STARTA       NO, UPDATE ADDRESS
0457 028A 0603             DEC  COUNT
0458 028C 0283             CI   COUNT,4
     028E 0004
0459 0290 1602             JNE  MSKIP2
0460 0292 2FA0             MESG @SPACE2
     0294 01FB'
0461 0296 C0C3      MSKIP2 MOV  COUNT,COUNT  DONE WITH LINE?
0462 0298 13EB             JEQ  MLOOP1       YES, TO NEXT LINE
0463 029A 2FA0             MESG @SPACE2      DELIMITER
     029C 01FB'
0464 029E 10EF             JMP  MLOOP2
0465 02A0 0640      MIC1   DECT STARTA       LAST ADDR ?
0466 02A2 0285             CI   TCHAR,'-'*256     OPEN PRE MEMORY LOC ?
     02A4 2D00
0467 02A6 1302             JEQ  MIC
```

```
0468 02A8 0220          AI    R0,4          TWO INCT'S
     02AA 0004
0469 02AC
0470                *
0471                * MEMORY INSPECT/CHANGE ROUTINE
0472                *
0473 02AC 2FA0  MIC     MESG @CRLF          NEXT LINE
     02AE 0238'
0474 02B0 2E80          WHEX STARTA         PRINT ADDRESS
0475 02B2 2FA0          MESG @EQUSGN        PRINT ' = '
     02B4 0256'
0476 02B6 C110          MOV  *STARTA,VALUE
0477 02B8 2E84          WHEX VALUE          OUTPUT CONTENTS
0478 02BA 2FA0          MESG @SPACE2        DELIMITER
     02BC 01FB'
0479 02BE 2E44          RHEX VALUE          ACCEPT NEW INPUT
0480 02C0 02C4'         DATA MNULL,ERR2
     02C2 0136'
0481 02C4 C404  MNULL   MOV  VALUE,*STARTA UPDATE CONTENTS
0482 02C6 0285          CI   TCHAR,>D00     RETURN TO COMMAND SCANNER?
     02C8 0D00
0483 02CA 16EA          JNE  MIC1           IF NO, GO ON
0484 02CC 0459  MEXIT   B    *9             EXIT
```

```
0487                ***************************************************************
0488                * HEX INPUT ROUTINE  (XOP  R,9)
0489                *CALL:   RHEX   R
0490                *        DATA   NULL,ERROR
0491                *           ---       NORMAL RETURN
0492                *
0493                * RETURNS A 16-BIT NUMBER INPUT FROM TERMINAL. DIGITS
0494                * ARE ACCEPTED UNTIL A TERMINATION CHARACTER IS FOUND.
0495                *
0496                * TERMINATION CHARACTERS: SPACE, COMMA, CARRIAGE RETURN, MIN
0497                *
0498                * THE TERMINATION CHARACTER IS RETURNED IN THE LEFT
0499                * BYTE OF THE REGISTER FOLLOWING 'R'.
0500                *
0501                * RETURN IS TO THE NULL RETURN ADDRESS IF INPUT IS
0502                * A TERMINATION CHARACTER ONLY.
0503                * IF A FAULTY TERMINATION CHARACTER IS FOUND,
0504                * RETURN IS TO THE ERROR ENTRY.
0505                ***************************************************************
0506       000D     WP      EQU   13
0507       000E     PC      EQU   14
0508                *
0509                *
0510 02CE 04C9     RHENTY CLR   R9             RESET NUMBER INPUT FLAG
0511 02D0 04CC            CLR   BUFFER         CLEAR ACCUMULATOR
0512 02D2 2ECA     LOOP   ECHO  R10            GET A CHARACTER INPUT
0513                *
0514                * CHECK FOR VALID HEX INPUT
0515                *
0516       02D6'    ZERO   EQU   $+2
0517 02D4 028A            CI    R10,'0'*256   MIN NUMERIC
     02D6 3000
0518 02D8 1A11            JL    NOTHEX
0519 02DA 028A            CI    R10,'9'*256   MAX NUMERIC
     02DC 3900
0520 02DE 1208            JLE   GOTONE
0521 02E0 028A            CI    R10,'A'*256   MIN ALPHA
     02E2 4100
0522 02E4 1A0B            JL    NOTHEX
0523 02E6 028A            CI    R10,'F'*256   MAX ALPHA
     02E8 4600
0524 02EA 1B08            JH    NOTHEX
0525 02EC 022A            AI    R10,>900      ALPHA ADJUST
     02EE 0900
0526 02F0 0A4A     GOTONE SLA   R10,4          ISOLATE DIGIT
0527 02F2 09CA            SRC   R10,12         WORD ALIGN DIGIT
0528                *
0529                * DIGIT TO ACCUMULATOR
0530                *
0531 02F4 0A4C            SLA   BUFFER,4
0532 02F6 A30A            A     R10,BUFFER
0533 02F8 0589            INC   R9             SET INPUT FLAG
0534 02FA 10EB            JMP   LOOP
0535                *
0536                * CHECK FOR TERMINATION CHARACTER
0537                *
0538 02FC 028A     NOTHEX CI    R10,' '*256 ' '?
     02FE 2000
0539 0300 130B            JEQ   SPCK
0540       0304'    MINUS  EQU   $+2
```

```
0541 0302 028A          CI    R10,'-'*256 '-'?
     0304 2D00
0542 0306 1308          JEQ   SPCK
0543 0308 028A          CI     R10,>D00     CARRIAGE RETURN?
     030A 0D00
0544 030C 1305          JEQ   SPCK.
0545 030E 028A          CI    R10,',',*256  COMMA?
     0310 2C00
0546 0312 160D          JNE   ERR           NO, TERMINATION CHAR ERROR
0547 0314 020A          LI    R10,' '*256   CHANGE TO SPACE
     0316 2000
0548 0318 C249  SPCK    MOV   R9,R9         NULL INPUT?
0549 031A 1305          JEQ   NEXIT         YES, SKIP
0550 031C CECC          MOV   BUFFER,*R11+ RETURN VALUE
0551 031E C6CA          MOV   R10,*R11      RETURN TERMINATOR
0552 0320 05CE          INCT  PC            BUMP PAST NULL POINTER
0553 0322 05CE          INCT  PC            BUMP PAST ERROR POINTER
0554 0324 0380          RTWP
0555 0326 05CB  NEXIT   INCT  R11
0556 0328 C6CA          MOV   R10,*R11      RETURN TERMINATOR
0557 032A C39E  EXIT1   MOV   *PC,PC        GET POINTER
0558 032C 0380          RTWP
0559 032E 05CE  ERR     INCT  PC            POINT TO ERROR POINTER
0560 0330 10FC          JMP   EXIT1
```

```
0563                    ****************************************************************
0564                    *
0565                    * HEX OUTPUT ROUTINES
0566                    *
0567                    * ROUTINE 1: XOP  R,10
0568                    *
0569                    * CALL:  WHEX  R
0570                    *          ----      NORMAL RETURN
0571                    *
0572                    * OUTPUT THE BINARY CONTENTS OF 'R' AS
0573                    * 4 HEXADECIMAL DIGITS.
0574                    *
0575                    *
0576                    * ROUTINE 2: XOP  R,8
0577                    *
0578                    * CALL:  WHX1  R
0579                    *          ----        NORMAL RETURN
0580                    *
0581                    * OUTPUT RIGHT MOST HEX DIGIT IN R.
0582                    *
0583                    ****************************************************************
0584                    *
0585                    *
0586                    *
0587                    * WHX1 ENTRY POINT
0588                    *
0589 0332 C31B  WHXENTY MOV  *R11,BUFFER  GET VALUE TO PRINT
0590 0334 0ACC          SLA  BUFFER,12
0591 0336 0209          LI   R9,1         SET COUNT FOR 1 DIGIT
     0338 0001
0592 033A 1003          JMP  LOOP1
0593                    *
0594                    * WHEX ENTRY POINT
0595                    *
0596 033C C31B  WHENTY MOV  *R11,BUFFER  GET THE VALUE
0597 033E 0209          LI   R9,4         SET COUNT FOR 4 DIGITS OUT
     0340 0004
0598 0342 C28C  LOOP1  MOV  BUFFER,R10
0599 0344 09CA          SRL  R10,12       ISOLATE HEX DIGIT
0600 0346 0A8A          SLA  R10,8        BYTE ALIGN
0601 0348 028A          CI   R10,9*256    NUMERIC?
     034A 0900
0602 034C 1202          JLE  NUM          YES, SKIP
0603 034E 022A          AI   R10,7*256    ALPHA ADJUST
     0350 0700
0604 0352 022A  NUM     AI   R10,'0'*256  NUMERIC TO ASCII
     0354 3000
0605 0356 2F0A          WRIT R10          WRITE CHARACTER
0606 0358 0BCC          SRC  BUFFER,12    ALIGN NEXT DIGIT
0607 035A 0609          DEC  R9           DONE?
0608 035C 16F2          JNE  LOOP1        NO, LOOP
0609 035E 0380          RTWP
```

```
0612                ***************************************************************
0613                *
0614                * SINGLE STEP/EXECUTE ENTRY
0615                *
0616                ***************************************************************
0617                *
0618        0006    BPDATA EQU  6
0619        0007    TFLAG  EQU  7
0620                *
0621                * SINGLE STEP ENTRY
0622                *
0623        0360'   S      EQU  $
0624 0360 0207             LI   TFLAG,>9900  SET TRACE FLAG
     0362 9900
0625 0364 03E0             LREX INITIATE LOAD INTERRUPT
0626                *
0627                * EXECUTE ENTRY POINT
0628                *
0629 0366 0380     E       RTWP
0630                *
0631                * LOAD ENTRY
0632                *
0633 0368 0287     LOAD    CI   TFLAG,>9900  SINGLE STEP?
     036A 9900
0634 036C 1316             JEQ  WPSOUT       YES, TO WPS OUTPUT
0635 036E 0201             LI   R1,HALTFG    REG 1 = ADDR OF HALT FLAG
     0370 FFFA
0636 0372 C091             MOV  *R1,R2       IS HALT FLAG ON ?
0637 0374 1302             JEQ  LOAD1        IF NO, GO ON
0638 0376 0460             B    @HALT        GO TO HALT
     0378 07E0'
0639 037A 0641     LOAD1   DECT R1           REG 1 = ADDR OF STEP FLAG
0640 037C C091             MOV  *R1,R2       IS STEP FLAG SET ?
0641 037E 1316             JEQ  BRAMON       IF NO, GO TO TOP OF MONITOR
0642 0380 04D1             CLR  *R1          CLEAR STEP FLAG
0643 0382 0460             B    @STEPRT
     0384 07AA'
```

```
0646                ***************************************************************
0647                *
0648                * BREAKPOINT HANDLER
0649                *
0650                * SAVE THE CONTENTS OF THE ADDRESS INDICATED. INSTALL AN
0651                * XOP 15 (>2FC0) IN THE LOCATION AND PASS CONTROL TO
0652                * USER PROCEEDURE.
0653                *
0654                * WHEN XOP 15 IS EXECUTED, PRINT 'BP' FOLLOWED BY CONTENTS
0655                * OF USER WP, PC, AND ST AT THE BREAKPOINT. CONTROL IS THEN
0656                * PASSED TO THE MONITOR COMMAND SCANNER.
0657                *
0658                ***************************************************************
0659                *
0660                *
0661                * GET BREAKPOINT ADDRESS
0662                *
0663 0386 4008     B       SZC   WDBDY,START  WORD ALIGN ADDRESS
0664                *
0665                * SAVE DATA AND INSTALL XOP 15
0666                *
0667 0388 C190             MOV   *START,BPDATA
0668 038A C420             MOV   @BPXOP,*START
     038C 0390'
0669 038E 0380             RTWP
0670 0390 2FC0     BPXOP   DATA >2FC0
0671                *
0672                * XOP 15 ENTRY POINT
0673                *
0674 0392 064E     XOPENT  DECT PC             UPDATE USER PC
0675 0394 C406             MOV  BPDATA,*START RESTORE LOCATION
0676                *
0677                * OUTPUT 'BP' FOLLOWED BR USER WP, PC, AND ST AT BREAKPOINT
0678                *
0679 0396 2FA0             MESG @BPMSG         NEW LINE AND 'BP'
     0398 01FE'
0680 039A 04C7     WPSOUT  CLR  TFLAG          RESET FLAG
0681 039C 020A             LI   R10,-6         SET LOOP COUNT
     039E FFFA
0682 03A0 2FA0     XLOOP1  MESG @SPACE4        PRINT 4 SPACES
     03A2 01F9'
0683 03A4 2EAA             WHEX @MREGS+32(R10) PRINT STATUS
     03A6 FFD0
0684 03A8 05CA             INCT R10
0685 03AA 16FA             JNE  XLOOP1
0686 03AC 0460     BRAMON  B    @MONTOP        TO COMMAND SCANNER
     03AE 0080'
```

```
0689                ******************************************************
0690                *
0691                * CRU INSPECT/CHANGE -- 'C' COMMAND
0692                *
0693                * INPUT THE CRU BASE ADDRESS FOLLOWED BY THE BIT
0694                * COUNT. ALL INPUT AND OUTPUT TO THE CRU IS RIGHT
0695                * JUSTIFIED IN THE 16 BIT INPUT/OUTPUT DATA FIELDS.
0696                *
0697                * INPUT OF A CARR. RET. AS A TERMINATION CHARACTER
0698                * RETURNS CONTROL TO THE COMMAND SCANNER. A ' ' AS
0699                * TERMINATION CHARACTER CAUSES THE CRU INPUT BITS
0700                * TO BE OUTPUT AGAIN AS WELL AS THE CRU OUTPUT BITS
0701                * TO BE CHANGED.
0702                *
0703                ******************************************************
0704                *
0705       0001     BITCNT EQU  1
0706       0006     IOBUF  EQU  6
0707       0007     WORDFG EQU  7
0708       0008     XEC    EQU  8
0709                *
0710                *
0711 03B0 C300     C       MOV  START,CRUBAS UPDATE CRU BASE REGISTER
0712 03B2 04C7             CLR  WORDFG       RESET WORD FLAG
0713 03B4 0241             ANDI BITCNT,>F    ISOLATE BIT COUNT
     03B6 000F
0714 03B8 1303             JEQ  SETFG        YES, SET FLAG
0715 03BA 0281             CI   BITCNT,>9    BYTE JUSTIFIED I/O?
     03BC 0009
0716 03BE 1A01             JL   CSKIP1       YES, SKIP
0717 03C0 0587     SETFG   INC  WORDFG       WORD JUSTIFIED I/O FLAG
0718                *
0719                * FORM 'STCR' COMMAND AND READ CRU
0720                *
0721 03C2 0A61     CSKIP1  SLA  BITCNT,6     JUSTIFY BIT COUNT
0722 03C4 0208     CLOOP   LI   XEC,>3406
     03C6 3406
0723 03C8 E201             SOC  BITCNT,XEC
0724 03CA 0488             X    XEC          EXECUTE 'STCR'
0725                *
0726                * OUTPUT STATE OF CRU
0727                *
0728 03CC 2FA0             MESG @CRLF        NEXT LINE
     03CE 0238'
0729 03D0 2E8C             WHEX CRUBAS       PRINT BASE ADDRESS
0730 03D2 2FA0             MESG @EQUSGN      PRINT ' = '
     03D4 0256'
0731 03D6 C1C7             MOV  WORDFG,WORDFG WORD I/O?
0732 03D8 1601             JNE  CSKIP2       YES, SKIP
0733 03DA 0986             SRL  IOBUF,8      ALIGN BYTE I/O TO WORD
0734 03DC 2E86     CSKIP2  WHEX IOBUF        OUTPUT CRU STATE
0735 03DE 2FA0             MESG @SPACE2      PRINT '  '
     03E0 01FB'
0736                *
0737                * ACCEPT INPUT FOR ALTERATION OF CRU
0738                *
0739 03E2 2E44             RHEX VALUE        CRU OUTPUT?
0740 03E4 03F6'            DATA CNULL3,ERR2
     03E6 0136'
0741 03E8 C184             MOV  VALUE,IOBUF
```

```
0742 03EA C1C7          MOV   WORDFG,WORDFG  WORD I/O?
0743 03EC 1601          JNE   CSKIP3         YES, SKIP
0744 03EE 0A86          SLA   IOBUF,8        BYTE ALIGN FOR OUTPUT
0745 03F0 0248  CSKIP3  ANDI  XEC,>F3FF      'STCR' TO 'LDCR'
     03F2 F3FF
0746 03F4 0488          X     XEC            EXECUTE 'LDCR'
0747 03F6 0285  CNULL3  CI    CHAR,>D00       EXIT?
     03F8 0D00
0748 03FA 16E4          JNE   CLOOP           NO LOOP
0749 03FC 0459          B     *9              YES, TO MONITOR
```

```
0752                ***********************************************
0753                *
0754                * INSPECT/CHANGE USER WORKSPACE REGISTER -- 'W' COMMAND
0755                *
0756                * OPTIONS: 1) 'W' FOLLOWED BY CARRIAGE RETURN --
0757                *              DISPLAY THE CONTENTS OF ALL CURRENT USER
0758                *              WORKSPACE REGISTERS AND RETURN TO THE
0759                *              COMMAND SCANNER.
0760                *
0761                *          2) 'W', REGISTER NUMBER IN HEX, CARRIAGE
0762                *              RETURN -- DISPLAY THE CONTENTS OF THE
0763                *              DESIGNATED REGISTER. USER MAY ALTER
0764                *              THE CONTENTS FOLLOWED BY A TERMINATION
0765                *              CHARACTER OR MERELY ENTER A TERMINATION
0766                *              CHARACTER. THE TERMINATION CHARACTER
0767                *              SIGNIFIES WHAT IS TO BE DONE NEXT:
0768                *
0769                *     SPACE -- DISPLAY THE CONTENTS OF THE NEXT REGISTER.
0770                *     MINUS -- DISPLAY THE CONTENTS OF THE PREVIOUS REGISTER
0771                *     CARRIAGE RETURN -- TO THE COMMAND SCANNER.
0772                *
0773                ***********************************************
0774       0006     REGNUM EQU  6
0775       0007     RPOINT EQU  7
0776                *
0777                *
0778       03FE'    W      EQU  $
0779  03FE C1CD             MOV  WP,RPOINT     GET WORKSPACE POINTER
0780  0400 C0C3             MOV  COUNT,COUNT   NULL INPUT?
0781  0402 1323             JEQ  WNULL1        YES, TO FORMATTED DUMP
0782  0404 0240             ANDI START,>F      0 TO >F
      0406 000F
0783                *
0784                * INSPECT/CHANGE A WORKING REGISTER
0785                *
0786  0408 C180             MOV  START,REGNUM SAVE REGISTER NUMBER
0787  040A 0A10             SLA  START,1
0788  040C A1C0             A    START,RPOINT FORM REGISTER ADDRESS
0789  040E 2FA0  ICLOOP     MESG @CRLFR        NEXT LINE
      0410 0242'
0790  0412 2E06             WHX1 REGNUM        OUTPUT REGISTER NUMBER
0791  0414 2FA0             MESG @EQUSGN       PRINT ' = '
      0416 0256'
0792  0418 2E97             WHEX *RPOINT       PRINT REGISTER CONTENTS
0793  041A 2FA0             MESG @SPACE2       DELIMITER
      041C 01FB'
0794  041E 2E44             RHEX VALUE         NEW CONTENTS?
0795  0420 0426'            DATA WNULL2,ERR2
      0422 0136'
0796  0424 C5C4             MOV  VALUE,*RPOINT UPDATE REGISTER
0797  0426 0285  WNULL2     CI   TCHAR,>D00    RETURN TO COMMAND SCANNER?
      0428 0D00
0798  042A 1601             JNE   SKIP         NO, CHECK FOR ' '
0799  042C 0459  WEXIT1     B    *9            TO SCANNER
0800  042E 0285  SKIP       CI   TCHAR,' '*256 NEXT REGISTER?
      0430 2000
0801  0432 1305             JEQ  NREG           YES
0802                *
0803                * CHECK FOR REGISTER 0
0804                *
```

```
0805 0434 C186          MOV  REGNUM,REGNUM AT REGISTER 0?
0806 0436 13FA          JEQ  WEXIT1        YES, TO SCANNER
0807 0438 0606          DEC  REGNUM        UPDATE REGISTER NUMBER
0808 043A 0647          DECT RPOINT        UPDATE ADDRESS
0809 043C 10E8          JMP  ICLOOP
0810                *
0811                * CHECK FOR REGISTER >F
0812                *
0813 043E 0286  NREG    CI   REGNUM,>F
     0440 000F
0814 0442 13F4          JEQ  WEXIT1        REGISTER >F, TO SCANNER
0815 0444 0586          INC  REGNUM        UPDATE REGISTER NUMBER
0816 0446 05C7          INCT RPOINT        UPDATE ADDRESS
0817 0448 10E2          JMP  ICLOOP
0818                *
0819                * FORMATTED REGISTER DISPLAY
0820                *
0821 044A 04C6  WNULL1  CLR  REGNUM
0822 044C C1CD          MOV  WP,RPOINT
0823 044E 2FA0  NLINE   MESG @CRLFR        NEXT LINE
     0450 0242'
0824 0452 2E06  WLOOP   WHX1 REGNUM        REGISTER NUMBER
0825 0454 2FA0          MESG @EQUSGN       PRINT ' = '
     0456 0256'
0826 0458 2E97          WHEX *RPOINT       PRINT CONTENTS
0827 045A 0586          INC  REGNUM        TO NEXT REGISTER
0828 045C 05C7          INCT RPOINT        NEXT ADDRESS
0829 045E 0286          CI   REGNUM,>8     END OF LINE?
     0460 0008
0830 0462 13F5          JEQ  NLINE
0831 0464 0286          CI   REGNUM,>10    DONE?
     0466 0010
0832 0468 13E1          JEQ  WEXIT1        YES, TO SCANNER
0833 046A 2FA0          MESG @SPACE2       DELIMITER
     046C 01FB'
0834 046E 2FA0          MESG @RP           PRINT 'R'
     0470 0244'
0835 0472 10EF          JMP  WLOOP
```

```
0838                ***********************************************
0839                *
0840                * DISPLAY WP, PC, ST REGISTERS
0841                *
0842                * TERMINATION CHARACTERS:
0843                *     SPACE -- TO NEXT REGISTER
0844                *     CARRIAGE RETURN -- TO MONITOR SCANNER
0845                *     MINUS -- INSPECT SAME REGISTER AGAIN
0846                *
0847                * ORDER OF DISPLAY: WP, PC, ST.
0848                *
0849                ***********************************************
0850       0006     MPOINT EQU  6
0851       0007     LCOUNT EQU  7
0852                *
0853                *
0854 0474 0206     R       LI   MPOINT,WS     INIT. MESSAGE POINTER
     0476 023C'
0855 0478 0207             LI   LCOUNT,3      SET LOOP COUNT
     047A 0003
0856 047C 0208             LI   R8,MREGS+WP+WP
     047E FFCA
0857 0480 2FA0     RLOOP1  MESG @CRLF         NEXT LINE
     0482 0238'
0858 0484 2F96             MESG *MPOINT       OUTPUT REGISTER SLOGAN
0859 0486 2FA0             MESG @EQUSGN       PRINTI' = '
     0488 0256'
0860 048A C118             MOV  *R8,VALUE
0861 048C 2E98             WHEX *R8           PRINT CONTENTS
0862 048E 2FA0             MESG @SPACE2       DELIMITER
     0490 01FB'
0863 0492 2E44             RHEX VALUE         NEW DATA?
0864 0494 0498'            DATA RNULL,ERR2
     0496 0136'                    '
0865 0498 C604     RNULL   MOV  VALUE,*R8
0866 049A 0285             CI   TCHAR,'-'*256 SAME REGISTER?
     049C 2D00
0867 049E 13F0             JEQ  RLOOP1        YES, LOOP
0868 04A0 0285             CI   TCHAR,>D00    TO SCANNER?
     04A2 0D00
0869 04A4 1304             JEQ  REXIT         YES, EXIT
0870 04A6 05C6             INCT MPOINT        UPDATE MESSAGE POINTER
0871 04A8 05C8             INCT R8            UPDATE ADDRESS
0872 04AA 0607             DEC  LCOUNT        EXIT ?
0873 04AC 16E9             JNE  RLOOP1        IF >0, NO
0874 04AE 0459     REXIT   B    *9            TO SCANNER
```

```
0877                ****************************************************************
0878                *
0879                * DUMP ROUTINE -- 'D' COMMAND
0880                *
0881                * DUMP RAM IMAGE TO CASSETTE TAPE
0882                * IN 990 TAG OBJECT FORMAT
0883                *
0884                ****************************************************************
0885       0002     ENTRY  EQU  2
0886       0004     TCOUNT EQU  4
0887       0005     CKSUM  EQU  5
0888       0006     IDT    EQU  6
0889       000C     TVALUE EQU  12
0890       04B0'    D      EQU  $
0891                *
0892                * WORD ALIGN ADDRESSES
0893                *
0894 04B0 4008           SZC  WDBDY,STARTA
0895 04B2 4048           SZC  WDBDY,STOPA
0896 04B4 4088           SZC  WDBDY,ENTRY
0897                *
0898                * START ADDRESS 7 STOP ADDRESS--ERROR
0899                *
0900 04B6 8040           C    STARTA,STOPA
0901 04B8 1202           JLE  ADDROK
0902 04BA 0460           B    @ERR3          ERROR EXIT TO MONITOR
     04BC 013C'
0903                *
0904                * READ  IDT. BLANK FILL REMAINDER OF BUFFER
0905                *
0906 04BE 04C4  ADDROK   CLR  R4
0907 04C0 04C3           CLR  COUNT
0908 04C2 2FA0           MESG @IDTEQ
     04C4 0203'
0909 04C6 2EC4  RDIDT    ECHO R4             READ CHARACTER
0910 04C8 D8C4  BLANKO   MOVB R4,@MREGS+12(COUNT)
     04CA FFBC
0911 04CC 0583           INC  COUNT
0912 04CE 0283           CI   COUNT,8        BUFFER FULL?
     04D0 0008
0913 04D2 1304           JEQ  UREADY         YES, EXIT
0914 04D4 0284           CI   R4,>2000       TERMINATOR ?
     04D6 2000
0915 04D8 16F6           JNE  RDIDT
0916 04DA 10F6           JMP  BLANKO
0917                *
0918                * WAIT FOR USER READY
0919                *
0920 04DC 2FA0  UREADY   MESG @READY         OUTPUT READY MESSAGE
     04DE 0208'
0921 04E0 2F44           READ R4             WAIT FOR INPUT
0922 04E2 0284           CI   R4,'Y'*256     YES? ,IF NOT TO MONITOR
     04E4 5900
0923 04E6 1641           JNE  DEXIT
0924 04E8 04E0           CLR  @DUMPFG        SET DUMP FLAG
     04EA FFF6
0925                *
0926                * ZERO TAG  (IDT TAG)
0927                *
0928 04EC 2FA0           MESG @DC2            OUTPUT DC2,START CASSSETTE
```

```
      04EE 0223'
0929  04F0 04CA          CLR  R10
0930  04F2 04C5          CLR  CKSUM
0931  04F4 06A0          BL   @OUTTAG
      04F6 056C'
0932  04F8 3000          DATA >3000
0933  04FA 2FA0          MESG @MREGS+12      OUTPUT IDT
      04FC FFBC
0934                *
0935                * UPDATE CHECKSUM
0936                *
0937  04FE 0203          LI   COUNT,8
      0500 0008
0938  0502 D123  LLOOP1  MOVB @MREGS+11(COUNT),R4  GET CHARACTER
      0504 FFBB
0939  0506 0984          SRL  R4,8
0940  0508 A144          A    R4,CKSUM             ADD IT TO CHECKSUM
0941  050A 0603          DEC  COUNT                DONE?
0942  050C 16FA          JNE  LLOOP1         NO,ADD NEXT CHARACTER
0943                *
0944                *  ENTRY ADDRESS TAG
0945                *
0946  050E C282          MOV  ENTRY,R10      ENTRY ADDRESS TO BUFFER
0947  0510 06A0          BL   @OUTTAG
      0512 056C'
0948  0514 3100          DATA >3100
0949                *
0950                * ENTRY ADDRESS TAG (9)
0951                *
0952  0516 C280  NINE    MOV  STARTA,R10
0953  0518 06A0          BL   @OUTTAG        MEMORY ADDRESS
      051A 056C'
0954  051C 3900          DATA >3900
0955                *
0956                * MEMORY CONTENTS TAG (B)
0957                *
0958  051E C290  BTAG    MOV  *STARTA,R10
0959  0520 06A0          BL   @OUTTAG        MEMORY DATA
      0522 056C'
0960  0524 4200          DATA >4200
0961                *
0962                * DUMP COMPLETED?
0963                *
0964  0526 8040          C    STARTA,STOPA AT LAST ADDRESS?
0965  0528 1304          JEQ  EOR            YES,TERMINATE
0966  052A 05C0          INCT STARTA         NO, UPDATE ADDRESS
0967  052C 0283          CI   COUNT,60       END OF RECORD?
      052E 003C
0968  0530 1AF6          JL   BTAG           NO, OUTPUT NEXT WORD
0969                *
0970                * OUTPUT END OF RECORD
0971                *
0972  0532 0225  EOR     AI   CKSUM,>37      UPDATE CHECKSUM
      0534 0037
0973  0536 C285          MOV  CKSUM,R10
0974  0538 050A          NEG  R10            INVERT CHECKSUM
0975  053A 06A0          BL   @OUTTAG        OUTPUT 7 TAG
      053C 056C'
0976  053E 3700          DATA >3700
0977  0540 04C5          CLR  CKSUM          ZERO CHECKSUM
```

```
0978 0542 2FA0          MESG @FCR           OUTPUT F TAG AND
     0544 0215'
0979                *                  CARRIAGE RETURN
0980 0546 8040          C    STARTA,STOPA DONE WITH DUMP
0981 0548 1304          JEQ  DDUMP        YES,EXIT
0982 054A 04C3          CLR  COUNT        NO,OUTPUT NEXT
0983 054C 2FA0          MESG @NR          PRINT LEADING LF AND DEL
     054E 0224'
0984 0550 10E2          JMP  NINE         RECORD
0985                *
0986                * END DUMP ROUTINE
0987                *
0988 0552 2FA0  DDUMP   MESG @EOF         OUTPUT FINAL RECORD
     0554 0218'
0989 0556 0203          LI   COUNT,60
     0558 003C
0990 055A 2FA0  RLOOP2  MESG @RUBOUT
     055C 0221'
0991 055E 0603          DEC  COUNT
0992 0560 16FC          JNE  RLOOP2
0993 0562 0720          SETO @DUMPFG      RSET DUMP FLAG
     0564 FFF6
0994 0566 2FA0          MESG @CRLF        GET OVER TO LEFT HAND SIDE
     0568 0238'
0995 056A 1044  DEXIT   JMP  LOUT         TO MONITOR
0996                *
0997                * OUTPUT TAG ROUTINE
0998                *
0999                * TAG CHARACTER IN THE LEFT BYTE OF THE
1000                * WORD AFTER THE CALL. THE FIELD VALUE IS
1001                * IN 'VALUE'
1002                *
1003 056C C13B  OUTTAG MOV  *11+,R4       GET TAG
1004 056E 2F04          WRIT R4           OUTPUT IT
1005 0570 0984          SRL  R4,8
1006 0572 A144          A    R4,CKSUM     UPDATE CHECKSUM
1007 0574 2E8A          WHEX R10          OUTPUT FIELD
1008 0576 0223          AI   COUNT,5      UPDATE CHARACTER COUNT
     0578 0005
1009                *
1010                * UPDATE THE CHECKSUM WITH THE 'HEX/ASCII'
1011                * EQUIVALENT OF THE CONTENTS OF 'VALUE'
1012                *
1013 057A 0204          LI   TCOUNT,4     SET LOOP COUNT
     057C 0004
1014 057E 0B4A  ULOOP1  SRC  R10,4        ISOLATE A DIGIT
1015 0580 C30A          MOV  R10,TVALUE
1016 0582 09CC          SRL  TVALUE,12
1017 0584 A14C          A    TVALUE,CKSUM ADD ASCII VALUE
1018 0586 0225          AI   CKSUM,>30    TO CHECKSUM
     0588 0030
1019 058A 028C          CI   TVALUE,10
     058C 000A
1020 058E 1A02          JL   USKIP
1021 0590 0225          AI   CKSUM,7
     0592 0007
1022 0594 0604  USKIP   DEC  TCOUNT        ANOTHER DIGIT?
1023 0596 16F3          JNE  ULOOP1       YES,JUMP BACK
1024 0598 045B          B    *11          NO,RETURN
```

```
1027                ****************************************************************
1028                *
1029                * 990 TAG FORMAT LOADER -- 'L' COMMAND
1030                *
1031                * ACCEPTS LOAD BIAS FROM USER. IF INPUT IS NULL,
1032                * A BIAS OF >B0 IS THE DEFAULT.
1033                *
1034                * UPON A GOOD LOAD, THE 'IDT'IS OUTPUT.
1035                * IF AN ERROR IS DETECTED, AN ERROR MESSAGE IS
1036                * OUTPUT. IN EITHER CASE, CONTROL RETURNS TO THE
1037                * MONITOR.
1038                *
1039                ****************************************************************
1040       0000     BIAS    EQU  0
1041       0001     IDTR    EQU  1
1042       0008     JMPTAG  EQU  8
1043       0009     LOADDR  EQU  9
1044       059A'    L       EQU  $
1045                *
1046                * PLAYBACK ON
1047                *
1048 059A 0206           LI   R6,>1100
     059C 1100
1049 059E 2F06           WRIT R6              OUTPUT DC1
1050                *
1051                * GET A TAG
1052                *
1053 05A0 04C7   CLRSUM  CLR  R7
1054 05A2 04C8   TAG    CLR  JMPTAG
1055 05A4 06A0           BL   @GET1           GET TAG
     05A6 066E'
1056 05A8 100B           JMP  CHK1            ERROR EXIT
1057 05AA D22A   TAG2   MOVB @OP(R10),JMPTAG GET TAG OFFSET
     05AC 0654'
1058 05AE 132D           JEQ  ENDACT          F TAG ENCOUNTERED
1059 05B0 06A0           BL   @GET4           GET DATA VALUE
     05B2 0668'
1060 05B4 100E           JMP  CHK2
1061 05B6 0205           LI   R5,8            SET FOR SKIP 8
     05B8 0008
1062 05BA 0878           SRA  JMPTAG,7        RIGHT JUSTIFY OFFSET
1063 05BC 0468   JMP    B    @JMP(JMPTAG) GO TO TAG ROUTINE
     05BE 05BC'
1064                *
1065                * CHECK FOR G,H,I,J TAGS
1066                *
1067 05C0 0286   CHK1   CI   R6,'G'
     05C2 0047
1068 05C4 1106           JLT  CHK2
1069 05C6 0286           CI   R6,'J'
     05C8 004A
1070 05CA 1516           JGT  ERROR0
1071 05CC 0226           AI   R6,->37      ADJUST VALUE
     05CE FFC9
1072 05D0 10EC           JMP  TAG2
1073 05D2 0286   CHK2   CI   R6,>3A          COLON ?, NO MORE DATA
     05D4 003A
1074 05D6 1610           JNE  ERROR0
1075 05D8 04CA   GOODLD CLR  R10             RESET MASK FOR GOOD LOAD
1076                *
```

```
1077                    * CHARACTER TIME-OUT - WAIT FOR END OF INPUT
1078                    *
1079 05DA 0705  HANG1   SETO R5
1080 05DC 020C          LI    CRUBAS,>80    SET CRU BASE REG. FOR 9902
     05DE 0080
1081 05E0 1F0F  HANG2   TB    15            SPACE?
1082 05E2 16FB          JNE   HANG1         YES,RESET COUNT AND CONTINUE
1083 05E4 0605          DEC   R5            NO, DECREMENT COUNT
1084 05E6 16FC          JNE   HANG2         CHECK AGAIN
1085 05E8 C28A          MOV   R10,R10       GOOD LOAD?
1086 05EA 1609          JNE   LDERR         NO, OUTPUT ERROR MESSAGE
1087 05EC 2FA0          MESG  @CRLF         NEXT LINE
     05EE 0238'
1088 05F0 2FA0          MESG  @MREGS+IDTR+IDTR OUTPUT IDT
     05F2 FFB2
1089 05F4 0460  LOUT    B     @MONTOP       EXIT
     05F6 0080'
1090 05F8 04C0  ERROR0  CLR   0             TAG ERROR
1091 05FA 070A  ERROR1  SETO  R10           ERROR FLAG
1092 05FC 10EE          JMP   HANG1
1093 05FE C000  LDERR   MOV   0,0
1094 0600 1302          JEQ   ER0
1095 0602 0460          B     @ERR1
     0604 0130'
1096 0606 0460  ER0     B     @ERR0
     0608 012C'
1097                    ***************************************************
1098                    * F-TAG : SKIP TO END OF RECORD
1099                    ***************************************************
1100 060A 2F46  ENDACT  READ  R6
1101 060C 0986          SRL   R6,8
1102 060E 0286          CI    R6,>0D
     0610 000D
1103 0612 16FB          JNE   ENDACT
1104 0614 10C5          JMP   CLRSUM
1105                    ***************************************************
1106                    * LOAD ADDRESS TAGS 9 AND A
1107                    ***************************************************
1108 0616 A280  RELOAD  A     BIAS,R10      ADJUST FOR RELOCATABILITY
1109 0618 C24A  ABLOAD  MOV   R10,LOADDR    SAVE LOAD ADDRESS
1110 061A 10C3          JMP   TAG
1111                    ***************************************************
1112                    * DATA TAGS B AND C
1113                    ***************************************************
1114 061C A280  REDATA  A     BIAS,R10      ADJUST RELOCATABLE DATA
1115 061E CE4A  ABDATA  MOV   R10,*LOADDR+
1116 0620 10C0          JMP   TAG
1117                    ***************************************************
1118                    * CHECKSUM TAG 7
1119                    ***************************************************
1120 0622 A1CA  CHECK   A     R10,R7
1121 0624 13BE          JEQ   TAG
1122 0626 0200          LI    0,1           CHECKSUM ERROR
     0628 0001
1123 062A 10E7          JMP   ERROR1
1124                    ***************************************************
1125                    * IDT TAG 0
1126                    ***************************************************
1127 062C 020A  LENG    LI    R10,MREGS+IDTR+IDTR ADDRESS OF IDT BUFFER
     062E FFB2
```

```
1128 0630 1003          JMP  SKIP8
1129                    ************************************************************
1130                    * CHECKSUM FIELD 2 OF TAGS 0,3,4,5,6
1131                    ************************************************************
1132 0632 0645  SKIP6   DECT R5              ADJUST COUNT TO 6
1133 0634 020A  OLDIDT  LI   R10,DUMYBF      SET POINTER
     0636 FFD2
1134 0638 2F46  SKIP8   READ R6
1135 063A DE86          MOVB R6,*R10+        CHARACTER TO BUFFER
1136 063C 0986          SRL  R6,8
1137 063E A1C6          A    R6,R7           UPDATE CHECKSUM
1138 0640 0605          DEC  R5
1139 0642 16FA          JNE  SKIP8
1140 0644 10AE          JMP  TAG
1141                    ************************************************************
1142                    * TAGS 1 AND 2
1143                    ************************************************************
1144 0646 A280  RLENT   A    BIAS,R10
1145 0648 C38A  ABENT   MOV  R10,14          ENTRY POINT TO  PC
1146 064A 10AB          JMP  TAG
1147                    ************************************************************
1148                    * SET LOAD BIAS - TAG D
1149                    ************************************************************
1150 064C 024A  SBIAS   ANDI R10,>FFFE       WORD ALIGN
     064E FFFE
1151 0650 C00A          MOV  R10,BIAS
1152 0652 10A7          JMP  TAG
1153                    ************************************************************
1154                    * LOAD JUMP TABLE
1155                    ************************************************************
1156 0654   38  OP      BYTE LENG-JMP/2    0 PROGRAM START
1157 0655   46          BYTE ABENT-JMP/2   1 ABS ENTRY ADDRESS
1158 0656   45          BYTE RLENT-JMP/2   2 REL ENTRY ADDRESS
1159 0657   3B          BYTE SKIP6-JMP/2   3 EXT REFERENCE
1160 0658   3B          BYTE SKIP6-JMP/2   4 EXT REFERENCE
1161 0659   3B          BYTE SKIP6-JMP/2   5 EXT DEFINE
1162 065A   3B          BYTE SKIP6-JMP/2   6 EXT DEFINE
1163 065B   33          BYTE CHECK-JMP/2   7 CHECKSUM
1164 065C   F2          BYTE CLRSUM-JMP/2  8 IGNORE CHECKSUM
1165 065D   2E          BYTE ABLOAD-JMP/2  9 ABS LOAD ADDRESS
1166 065E   2D          BYTE RELOAD-JMP/2  A REL LOAD ADDRESS
1167 065F   31          BYTE ABDATA-JMP/2  B ABS DATA
1168 0660   30          BYTE REDATA-JMP/2  C REL DATA
1169 0661   48          BYTE SBIAS-JMP/2   D LOAD BIAS
1170 0662   1E          BYTE ERRORO-JMP/2  E ILLEGAL
1171 0663   00          BYTE 0             F END OF RECORD
1172 0664   3B          BYTE SKIP6-JMP/2   G REL SYMBOL
1173 0665   3B          BYTE SKIP6-JMP/2   H ABS SYMBOL
1174 0666   3C          BYTE OLDIDT-JMP/2  I OLD IDT
1175 0667   F3          BYTE TAG-JMP/2     ???
```

```
1177                ************************************************************
1178                *
1179                * GET 4 HEX DIGITS AND CONVERT THEM TO BINARY.
1180                * THE VALUE IS ASSEMBLED IN 'VALUE'. NON-HEX
1181                * INPUT RESULTS IN AN ERROR RETURN WITH THE
1182                * CHARACTER IN THE RIGHT BYTE OF 'CHAR'. GET1
1183                * IS THE ALTERNATE ENTRY  POINT FOR GETTING 1
1184                * HEX CHARACTER.
1185                *
1186                * CALL=BL @GET41
1187                *       JMP ERROR     ERROR RETURN
1188                *                     NORMAL RETURN
1189                *
1190                ************************************************************
1191                *
1192                * 4 HEX DIGIT ENTRY POINT
1193                *
1194 0668 0205    GET4    LI   R5,-4          SET COUNT FOR 4 DIGITS
     066A FFFC
1195 066C 1001          JMP  GET
1196                *
1197                * GET TAG CHARACTER ENTRY POINT
1198                *
1199 066E 0705    GET1    SETO R5             SET COUNT FOR 1 CHARACTER
1200 0670 04CA    GET     CLR  R10            CHECK FOR VALID ASCII INPUT
1201 0672 2F46    GET41   READ R6
1202 0674 0286          CI   R6,' '*256 MIN ASCII
     0676 2000
1203 0678 11FC          JLT  GET41
1204 067A 0286          CI   R6,>5F00   MAX ASCII
     067C 5F00
1205 067E 15F9          JGT  GET41
1206 0680 0986          SRL  R6,8
1207                *
1208                * CHECKSUM TAG? ( 7 )
1209                *
1210 0682 0288          CI   JMPTAG,CHECK-JMP/2*256
     0684 3300
1211 0686 1301          JEQ  SKIPCS
1212 0688 A1C6          A    R6,R7          UPDATE CHECKSUM
1213                *
1214                * CHECK FOR VALID HEX INPUT AND ISOLATE HEX DIGIT
1215                *
1216 068A 0286    SKIPCS CI    R6,'0'         MIN NUMERIC
     068C 0030
1217 068E 1112          JLT  GETERR
1218 0690 0286          CI   R6,'9'         MAX NUMERIC
     0692 0039
1219 0694 1208          JLE  GOT1
1220 0696 0286          CI   R6,'A'         MIN ALPHA
     0698 0041
1221 069A 110C          JLT  GETERR
1222 069C 0286          CI   R6,'F'         MAX ALPHA
     069E 0046
1223 06A0 1509          JGT  GETERR
1224 06A2 0226          AI   R6,9           ALPHA ADJUST
     06A4 0009
1225 06A6 0246    GOT1    ANDI R6,>F          ISOLATE BINARY
     06A8 000F
1226                *
```

```
1227                * ASSEMBLE THE VALUE IN BINARY
1228                *
1229 06AA 0A4A             SLA  R10,4
1230 06AC A286             A    R6,R10        ADD NEW DIGIT
1231 06AE 0585             INC  R5            DONE?
1232 06B0 16E0             JNE  GET41         GET ANOTHER CHARACTER
1233 06B2 05CB             INCT 11            BUMP PAST ERROR RETURN
1234 06B4 045B  GETERR     B    *11           EXIT,TAG OR INVALID INPUTIBI*11
```

```
1237                ************************************
1238                *
1239                * FIND COMMAND -- 'F'
1240                *
1241                * LOOK FROM START ADDRESS TO STOP ADDRESS FOR
1242                * THE SPECIFIED DATA PATTERN. THE TERM.
1243                * CHARACTER FOLLOWING THE DATA PATTERN DETERMINES
1244                * THE MODE:
1245                *           CARRIAGE RETURN - WORD MODE
1246                *           MINUS           - BYTE MODE.
1247                * IN THE BYTE MODE, THE RIGHT BYTE OF THE DATA
1248                * PATTERN IS SEARCHED FOR.
1249                *
1250                ************************************
1251                *
1252                *
1253 06B6 4008     F       SZC   WDBDY,STARTA WORD ALIGN
1254 06B8 0203     NULL3   LI    R3,>8402     SET UP CMD=C   KEY,*START
     06BA 8402
1255 06BC 0204             LI    R4,>5C0      SET UP 'INCT' CMD
     06BE 05C0
1256 06C0 0285             CI    TCHAR,>D00   WORD MODE ?
     06C2 0D00
1257 06C4 1307             JEQ   SKIP3        IF YES, JUMP
1258                *
1259                * ADJUST FOR COMPARE BYTE COMMAND
1260                *
1261 06C6 0223             AI    R3,>1000     CHANGE TO COMPARE BYTE
     06C8 1000
1262 06CA 0224             AI    R4,->40      CHANGE TO 'INC'
     06CC FFC0
1263 06CE 0A82             SLA   KEY,8        BYTE ALIGN DATA
1264 06D0 1001             JMP   SKIP3
1265                *
1266                * CHECK FOR DATA MATCH
1267                *
1268 06D2          SKIP1
1269 06D2 0484             X     R4           EXECUTE INCREMENT COMMAND
1270 06D4          SKIP3
1271 06D4 0483             X     R3           EXECUTE COMPARE COMMAND
1272 06D6 1603             JNE   SKIP2        IF NO MATCH, JUMP
1273 06D8 2FA0             MESG  @CRLF        NEXT LINE
     06DA 0238'
1274 06DC 2E80             WHEX  STARTA       OUTPUT MATCH ADDRESS
1275 06DE          SKIP2
1276 06DE 8040             C     STARTA,STOPA DONE?
1277 06E0 16F8             JNE   SKIP1        NO, CHECK NEXT ADDRESS
1278 06E2 0459     FEXIT   B     *9           TO MONITOR
```

```
1281                ****************************************************************
1282                *
1283                * HEX ARITHMETIC COMMAND -- 'H'
1284                *
1285                * OUTPUT SUM AND DIFFERENCE OF TWO HEX NUMBERS.
1286                *
1287                ****************************************************************
1288          0000  HO      EQU  0
1289          0001  HT      EQU  1
1290                *
1291                * H1+H2
1292                *
1293 06E4 2FA0  H       MESG @HP            PRINT 'H1+H2='
     06E6 0246'
1294 06E8 C100          MOV  HO,VALUE
1295 06EA A101          A    HT,VALUE       GET SUM
1296 06EC 2E84          WHEX VALUE          OUTPUT H1+H2
1297                *
1298                * H1-H2
1299                *
1300 06EE 2FA0          MESG @HM            PRINT 'H1-H2='
     06F0 024F'
1301 06F2 6001          S    HT,HO          GET DIFFERENCE
1302 06F4 2E80          WHEX HO             OUTPUT H1-H2
1303 06F6 0459          B    *9
```

```
 1306                ***************************************************
 1307                *
 1308                * T COMMAND
 1309                *
 1310                * IF THE TERMINAL BEING USED IS A TEXAS
 1311                * INSTRUMENTS 733ASR OPERATING AT 1200 BAUD, THE
 1312                * EFFECTIVE BAUD RATE MUST BE REDUCED TO 300 BAUD
 1313                * FOR CORRECT PRINTER OPERATION. AT SYSTEM
 1314                * INITIALIZATION, IF THE TERMINAL IS OPERATING
 1315                * AT 1200 BAUD, TIBUG ASSUMES THAT THE
 1316                * TERMINAL IS A 733ASR. ENTRY OF THE T COMMAND
 1317                * TOGGLES THE 'ASR' FLAG FOR TRUE 1200 BAUD
 1318                * OPERATION.
 1319                *
 1320                ***************************************************
 1321                *
 1322                *
 1323 06F8 0560     T       INV   @ASR          INVERT DATA
      06FA FFF4
 1324 06FC 0459     TEXIT   B     *9            EXIT
```

```
1327                ****************************************************************
1328                * MICRO TERMINAL OUTPUT ROUTINE    XOP 0
1329                *
1330                * CALL: OTPT RX     WHERE X IS NOT EQUAL TO 0
1331                *                   R0 IS USED TO PASS THE DELAY FLAG
1332                *
1333                * THIS ROUTINE IS CALLED WHEN OUTPUT TO THE MICRO TERMINAL
1334                * IS REQUIRED. THE DATA IS CONTAINED AT THE ADDRRESS
1335                * SPECIFIED BY WP11. THE DATA IS OUTPUT IN 4 MICRO
1336                * TERMINAL WORDS. THE WRITE XOP IS UTILIZED FOR OUTPUT
1337                * ONCE THE DATA IS FORMATTED. RETURN IS NORMAL.
1338                *
1339                ****************************************************************
1340 06FE          OTPTEN
1341 06FE C01D             MOV   *R13,R0       DO WE USE DELAY ?
1342 0700 1604             JNE   BDLY          IF NO, JUMP OVER DELAY
1343 0702 0200             LI    R0,>F00       REG 1 = TIMER TICK
     0704 0F00
1344 0706 0600  DLY        DEC   R0            DEC  TICK COUNT
1345 0708 16FE             JNE   DLY           IF NOT TIMED OUT GO BACK
1346 070A 0200  BDLY       LI    R0,4
     070C 0004
1347 070E C25B             MOV   *11,R9
1348 0710 C049  CNOT       MOV   R9,R1
1349 0712 09C1             SRL   R1,12         SAVE MSB
1350 0714 0A81             SLA   R1,8
1351 0716 0221             AI    R1,>3000      TURN  INTO ASCII
     0718 3000
1352 071A 2F01             WRIT  1             OUTPUT 1 MICRO TERMINAL WORD
1353 071C 0A49             SLA   R9,4
1354 071E 0600             DEC   R0            DEC NIBBLE COUNTER
1355 0720 16F7             JNE   CNOT          IF NOT DONE, JUMP BACK
1356 0722 0380             RTWP                EXIT
```

```
1359                ****************************************************************
1360                * MICRO TERMINAL INPUT              XOP 1
1361                *
1362                * CALL: INPT RX
1363                *
1364                * THIS ROUTINE IS UTILIZED WHEN THE MICRO TERMINAL REQUIRES
1365                * INPUT. DATA WILL BE PLACED AT THE ADDRESS SPECIFIED BY WP1
1366                * THE DATA IS INPUT IN 4 MICRO TERMINAL WORDS. THE READ XOP
1367                * IS UTILIZED FOR INPUT AFTER THE DATA IS FORMATTED. RETURN
1368                * IS NORMAL.
1369                *
1370                ****************************************************************
1371 0724           INPTEN
1372 0724 0201           LI   R1,4          REG 4 = # OF TRANSFERS
     0726 0004
1373 0728 04DB           CLR  *11           START WITH ALL 0'S
1374 072A C15B  CNIN     MOV  *11,5         PUT DATA (NIBBLE, 4 BITS) IN
1375                *                       REG 5
1376 072C 0A45           SLA  R5,4          MOVE DATA TO LEFT
1377 072E C6C5           MOV  R5,*11        PUT DATA BACK IN MEMORY
1378 0730 2F43           READ 3             GET NEXT 4 BITS
1379 0732 0A43           SLA  R3,4          STRIP OFF 0'S & RIGHT JUSTIFY
1380 0734 09C3           SRL  R3,12
1381 0736 E6C3           SOC  R3,*11        TURN ON 1'S IN DATA WORD
1382 0738 0601           DEC  R1            DECREMENT NIBBLE CTR
1383 073A 16F7           JNE  CNIN          IF NOT DONE, JUMP BACK
1384 073C 0380           RTWP
```

```
1387                ***************************************************************
1388                *                  Z COMMAND
1389                *
1390                * THIS ROUTINE IS ENTERED IF THE TEXAS INSTRUMENTS MICRO
1391                * TERMINAL IS CONNECTED TO THE M99/100 MX. WHEN THE RESET
1392                * PUSHBUTTON IS DEPRESSED THE MICRO TERMINAL WILL SEND OUT
1393                * AN ASCII 'A' AT 110 BAUD. TIBUG DETERMINES THE BAUD RATE.
1394                * APPROXIMATERLY 1 MS LATER THE MICRO TERMINAL SENDS OUT
1395                * THE ASCII CHARACTER 'Z' WHICH CAUSES THE COMMAND
1396                * PREPROCESSOR TO BRANCH TO THIS ROUTINE. SINCE THE MICRO
1397                * TERMINAL, FOR A HALT CONDITION, FIRES A LOAD SIGNAL,
1398                * THE HALT POINT IN THIS ROUTINE MUST BE ENTERED FROM THE
1399                * TIBUG LOAD ROUTINE WHEN THE MICRO TERMINAL STEP FLAG
1400                * IS NOT SET AND THE MICRO TERMINAL PRESENCE FLAG IS SET.
1401                * SINCE THE SINGLE INSTRUCTION EXECUTION FUNCTION ALSO
1402                * CAUSES THE LOAD SIGNAL TO BE FIRED FROM THIS ROUTINE
1403                * THE START OS THIS ROUTINE MUST BE ENTERED FROM THE
1404                * TIBUG LOAD ROUTINE IF THE MICRO TERMINAL STEP FLAG IS
1405                * SET. ALL COMMANDS FROM THE MICRO TERMINAL ARE TREATED
1406                * AS ADDRESS BIASES THAT STEER EXECUTION TO THE PROPER
1407                * OPERATION. ALL SPECIFIED OPERATIONS ARE PERFORMED IN
1408                * THIS ROUTINE EXCEPT READING COMMANDS AND INPUTTING AND
1409                * OUTPUTTING OF MICRO TERMINAL DATA WHICH IS PERFORMED
1410                * BY XOPS 1 AND 0 RESPECTIVELY.
1411                *
1412                ***************************************************************
1413 073E          MTIN
1414 073E 2F42             READ 2              GET COMMAND
1415 0740 04C0             CLR  R0             CLEAR SHORT OUT FLAG
1416 0742 04E0             CLR  @STEPFG        CLEAR STEP FLAG
     0744 FFF8
1417 0746 0720             SETO @HALTFG        TURN ON HALT FLAG
     0748 FFFA
1418 074A 0282             CI   R2,>4100
     074C 4100
1419 074E 1602             JNE  NA
1420 0750 0460             B    @TICKO
     0752 0092'
1421 0754 06C2  NA         SWPB R2             RIGHT JUSTIFY CMD CODE
1422 0756 0242             ANDI R2,>1E         STRIP OUT BITS
     0758 001E
1423 075A 0282             CI   R2,>18         TOO LARGE ?
     075C 0018
1424 075E 15EF             JGT  MTIN           IF YES, WAIT FOR NEXT KEYSTROKE
1425 0760 0222             AI   R2,JMTB        ADD JUMP TABLE BIAS
     0762 0766'
1426 0764 0452             B    *R2            GO TO EXECUTE CODE
1427 0766          JMTB
1428 0766 1018             JMP  EMA
1429 0768 100B             JMP  EPC
1430 076A 100C             JMP  EST
1431 076C 100D             JMP  EWP
1432 076E 100E             JMP  DPC
1433 0770 100F             JMP  DST
1434 0772 1010             JMP  DWP
1435 0774 1024             JMP  DCRU
1436 0776 102F             JMP  ECRU
1437 0778 1014             JMP  EMDI
1438 077A 1010             JMP  EMD
1439 077C 0380             RTWP
```

```
1440 077E 1019          JMP  STEP
1441 0780          EPC
1442 0780 2C4E          INPT 14             STORE PC
1443 0782 10DD          JMP  MTIN           AWAIT NEXT COMMAND
1444 0784          EST
1445 0784 2C4F          INPT 15             STORE ST
1446 0786 10DB          JMP  MTIN           AWAIT NEXT COMMAND
1447 0788          EWP
1448 0788 2C4D          INPT 13             STORE WP
1449 078A 10D9          JMP  MTIN           AWAIT NEXT COMMAND
1450 078C          DPC
1451 078C 2C0E          OTPT 14             OUTPUT PC
1452 078E 10D7          JMP  MTIN
1453 0790          DST
1454 0790 2C0F          OTPT 15             OUTPUT STATUS
1455 0792 10D5          JMP  MTIN           AWAIT NEXT COMMAND
1456 0794          DWP
1457 0794 2C0D          OTPT 13             OUTPUT WP
1458 0796 10D3          JMP  MTIN           AWAIT NEXT COMMAND
1459 0798          EMA
1460 0798 2C48          INPT 8              STORE MA
1461 079A 1005          JMP  EMDI1
1462 079C          EMD
1463 079C 2C49          INPT 9              GET DATA
1464 079E C609          MOV  9,*8
1465 07A0 1002          JMP  EMDI1
1466 07A2          EMDI
1467 07A2 2C49          INPT 9              GET DATA
1468 07A4 CE09          MOV  9,*8+          GET CONTENTS OF LOCATION
1469 07A6 C258     EMDI1 MOV  *8,9           GET DATA
1470 07A8 101E          JMP  HALT1          GO OUTPUT DATA
1471 07AA          STEPRT
1472 07AA 2C0E          OTPT 14             OUTPUT PC
1473 07AC C25E          MOV  *14,9
1474 07AE 0700          SETO R0             TURN ON SHORT FLAG
1475 07B0 101A          JMP  HALT1          GO OUTPUT GATA
1476 07B2          STEP
1477 07B2 0720          SETO @STEPFG        SET THE STEP FLAG
     07B4 FFF8
1478 07B6 04E0          CLR  @HALTFG        CLEAR HALT FLAG
     07B8 FFFA
1479 07BA 03E0          LREX
1480 07BC 0380          RTWP
1481 07BE          DCRU
1482 07BE 2C4A          INPT 10             GET CONTROL WORD
1483 07C0 04C9          CLR  R9             CLEAR COMMAND WORD
1484 07C2 C30A          MOV  R10,R12        REG 12 = CMD WORD
1485 07C4 024C          ANDI R12,>FFF       STRIP OFF CRU ADDR BITS
     07C6 0FFF
1486 07C8 0A1C          SLA  R12,1          CRU BASE ADDR IS NOW SET UP
1487 07CA 09CA          SRL  R10,12         STRIP OUT 0'S
1488 07CC 0A6A          SLA  R10,6          SET UP BIT COUNT
1489 07CE 022A          AI   R10,>3409      CONSTRUCT STCR COMMAND
     07D0 3409
1490 07D2 048A          X    R10            EXECUTE STCR R9,X
1491 07D4 1008          JMP  HALT1          GO OUTPUT CRU DATA
1492 07D6          ECRU
1493 07D6 2C49          INPT 9              GET CRU DATA
1494 07D8 022A          AI   R10,->400      R10 = LDCR R9,X
     07DA FC00
```

```
 1495 07DC 048A          X     R10          EXECUTE LDCR R9,X
 1496 07DE 10AF          JMP   MTIN         AWAIT NEXT COMMAND
 1497 07E0       HALT
 1498 07E0 0700          SETO  R0           SET SHORT OUT FLAG
 1499 07E2 2C0E          OTPT   14          OUTPUT PC
 1500 07E4 C25E          MOV   *14,9        GET MEMORY DATA
 1501 07E6 2C09  HALT1   OTPT   9           OUTPUT DATA
3F4→1502 07E8 10AA       JMP   MTIN         AWAIT NEXT COMMAND
 1503                    END
NO ERRORS
```

TEXAS INSTRUMENTS
INCORPORATED
Semiconductor Group
Post Office Box 1443, Houston, Texas 77001

MP 324

Printed in U.S.A.