

NASA Technical Memorandum 87646

NASA-TM-87646 19860007473

IPLIB (IMAGE PROCESSING LIBRARY) USER'S MANUAL

FOR REFERENCE

NOT TO BE TAKEN FROM THIS ROOM

NETTIE D. FAULCON, JAMES H. MONTEITH, AND  
KEITH MILLER

DECEMBER 1985

LIBRARY COPY

JAN 13 1986

LANGLEY RESEARCH CENTER  
LIBRARY, NASA  
HAMPTON, VIRGINIA

**NASA**

National Aeronautics and  
Space Administration

**Langley Research Center**  
Hampton, Virginia 23665



## INTRODUCTION

IPLIB is a collection of HP FORTRAN 77 subroutines and functions that facilitate the use of a COMTAL image processing system driven by an HP 1000 computer. These subprograms are based on work by Nettie D. Faulcon and James H. Monteith of the Acoustics and Vibration Instrumentation Section at NASA Langley Research Center, and were put into their present form by Keith Miller of the College of William and Mary's computer science department while he was a summer fellow in the ASEE/NASA program.

IPLIB uses an HP 1000 driver, DVR41, which directs all communications between the HP 1000 and the COMTAL Vision One/20 system. DVR41 was written by M. Brown of Coulter Computer Corporation, revised by R.W. Bagdazian of Hughes Aircraft Corporation, and documented by James Monteith and Keith Miller.

This user's manual is intended for programmers who want to use the HP 1000 to drive the COMTAL image processing system. It is assumed that the programmer knows HP 1000 FORTRAN 77, or at least one FORTRAN dialect. It is also assumed that the programmer has some familiarity with the COMTAL Vision One/20 system.

The manual is divided into six sections:

1. SUBPROGRAM HANDLING:

This section tells how to load and modify the subprograms in IPLIB as well as how to add subprograms to the library.

2. ALPHABETIC CATALOG:

All the subprograms are listed in alphabetic order with a few words that suggest the purpose of each. Test programs are listed in alphabetic order also.

3. CROSS REFERENCE FILE:

Each subprogram and test program is listed along with the subprograms it calls.

4. SUBPROGRAM SOURCE CODE:

The source code for each subprogram is given in its entirety. The code for all subprograms follows a documentation outline which was designed to make each subprogram understandable without reference to any other code.

5. TEST PROGRAM SOURCE CODE:

The source code for some programs used in the unit testing of the IPLIB subprograms are given. These test programs are not well documented, but they should be easily understood when used in conjunction with the documentation

N86-16943 #

of the subprograms they test.

#### 6. HP DRIVER SOURCE CODE:

The assembler code of DVR41, the HP driver for the COMTAL image processing system, is listed in its entirety.

The following references are suggested for programmers working with IPLIB:

"FORTRAN 77 Reference Manual (RTE-6/VM and RTE-A HP 1000 Computer System)  
Hewlett-Packard Company, 1981.

"Handbook on COMTAL's Image Processing System", Nettie D. Faulcon, NASA TM 85671,  
July 1983.

"Vision One/20 User's Manual", COMTAL Corporation, 1982.

## SUBPROGRAM HANDLING

### LOADING SUBPROGRAMS

When loading a program that calls IPLIB subroutines and function ("subprograms"), enter:

```
SEA %IPLIB::21
```

as part of the loading sequence. For example, the following loading sequence is used to load a program TNOTE which calls a number of IPLIB subprograms:

```
LOADR  
RE,%TNOTE  
SEA,%IPLIB  
SEA  
DI  
EN
```

### ADDING SUBPROGRAMS

In order to add subprograms to the library, 4 files must be changed: FILES::21, XREF::21, MAKELIB::21, and %IPLIB::21.

FILES::21 is an alphabetized, annotated list of the IPLIB subprograms and test programs for those subprograms. Using the text editor, add the name of the sub-

program in alphabetic order along with a short description of its effect. Also, add the name of the unit testing program to the second list in FILES::21.

XREF::21 is a list of each subprogram and unit testing program along with the subprograms it calls. Add the new subprogram and testing program to the list, using the same format that is already there.

MAKLIB::21 is a list of the load modules of all the subprograms and test programs in IPLIB. This list is in no special order, so just add the new subprogram's load module to the list. The standard convention is to use % as the first character in load modules.

%IPLIB holds all the load modules in the image processing library. If %NEW is the name of the load module to be included in IPLIB, the following sequence accomplishes the addition: (HP prompts given within parentheses.) "<CR>" signifies entering a carriage return.

```
(: )MERGE  
(ENTER DESTINATION NAMR) %IPLIB<CR>  
(ENTER COMMAND NAMR) 1<CR>  
(ENTER NAMR) %IPLIB<CR>  
(ENTER NAMR) %NEW<CR>  
(ENTER NAMR) /E<CR>
```

#### CHANGING SUBPROGRAMS

Modified programs must be tested before being changed in the library. Changes to parameters are not allowed: if such a serious modification is necessary, a NEW subprogram should be written, and added with a slightly different name. (See above for adding a subprogram to IPLIB).

If a subprogram is modified, the new source code must replace the old code. If necessary, the explanation in FILES::21 must be changed. If a different set of subprograms are called, XREF::21 must be changed. In order to replace the old load module in %IPLIB with the modified version, enter the following sequence: (HP prompts given within parenthesis.)

```
(: )MERGE  
(ENTER DESTINATION NAMR) %IPLIB  
(ENTER COMMAND NAMR) MAKLIB
```

## IPLIB CATALOG

Unless otherwise stated, these files contain subroutines.

- ADDI2 : adds 2 images, and requires a scaling factor (p. 11).
- ADDIM : adds 2 images, no scaling factor (p. 15).
- BWFDS : transfers a monochrome image to COMTAL from disk (p. 19).
- CLFDS : transfers a color image to COMTAL from disk (p. 22).
- CLRGR : clears a COMTAL graphics plane (p. 26).
- CLRIM : clears a monochrome COMTAL image (p. 28).
- CMMN2 : sends a COMTAL command given as a constant string from HP to COMTAL (p. 30).
- CMMND : sends a COMTAL command given as constant string and length parameter from HP to COMTAL (p. 33).
- COUNT : compiles the pixel count for each of the possible values, 0 - 255 (p. 36).
- DELAY : puts HP in a busy wait for number of seconds designated (p. 39).
- DIGIT : function that takes 0 - 9 integer input and returns '0' - '9' (p. 41).
- DSPBW : displays a monochrome COMTAL image (p. 43).
- DSPCL : displays a color COMTAL image (p. 45).
- DSPGR : displays a graphic plane to the COMTAL monitor (p. 48).
- DSPVD : displays the COMTAL image (5) set to the video camera (p. 50).
- HILO : scans a monochrome image and returns the high and low pixel values (p. 52).
- HISTO : displays a scaled histogram of the designated image on the COMTAL monitor (p. 54).
- ICOPY : copies one monochrome COMTAL image to another (using CMMND) (p. 57).
- ICPY2 : copies one monochrome COMTAL image to another (using RDILN & WRILN) (p. 60).
- MERGE : merges two bytes into one byte (all arguments are INTEGER) (p. 63).
- NORML : finds lowest pixel value in an image, then subtracts that value from all pixels in that image; used to get light table variations (p. 65)
- NOTE2 : writes a line of characters into a graphics plane with a given color and size at a location; takes a constant string argument (p. 67).
- NOTES : writes a line of characters into a graphics plane with a given color and size at a location; takes a character array and length (p. 72).
- PAINT : interactive "painting" of square patches on COMTAL image (p. 76).
- PROFL : gives HP access to the COMTAL profiling capabilities (p. 80).
- RANGE : logical function that determines if 1st argument is within 2nd & 3rd (p. 83).
- RDGLN : reads one horizontal line of a COMTAL graphics plane (p. 84).
- RDGPT : reads one point from a COMTAL graphics plane (p. 86).
- RDIL2 : reads one horizontal line of COMTAL pixels; 1 pixel/integer returned (p. 89).
- RDILN : reads one horizontal line of COMTAL pixels; 2 pixels/integer returned (p. 92).
- RDIPT : reads one pixel from a COMTAL monochrome image (p. 94).
- RDIRC : reads a rectangle of pixels from a COMTAL monochrome image (p.97).
- RDLUT : reads the contents of a COMTAL look-up table (p. 101).
- RDPSU : reads the contents of a COMTAL pseudo-color table (p. 104).
- RDTAB : reads the COMTAL Image/Graphics Table (p. 106).

RDTAR : reads the COMTAL cursor location (p. 109).  
 SETV : sets a COMTAL image 5 - 9 to the video camera (p. 111).  
 SPLIT : splits an integer into two bytes, both bytes stored in new integers (p. 114).  
 SPRED : finds low and high pixel values in an image, and does a linear stretch on all pixel values to expand the range to 0 - 255 (p. 116).  
 SUBI2 : subtracts two images with an offset of 128; differences <0 set to 0 (p. 119).  
 SUBIM : subtracts two images with no offset; differences <0 set to 0 (p. 123).  
 THRSH : sets pixels in output image to black(0) or white(255) depending on the corresponding pixel in input image and a threshold value (p. 126).  
 TSTI1 : generates "TeSt Image 1", increasing pixel values right and down; display appears as a darkening slash across the screen (p. 129).  
 TV2C4 : digitizes 4 images from TV camera and averages them into one image (p. 131).  
 TV2CM : digitizes an image from TV camera into a COMTAL memory plane (p. 133).  
 WAIT : halts HP processing until the HP <CR> is entered (p. 136).  
 WIPGR : removes a graphics plane from the display (p. 137).  
 WRGLN : writes a horizontal line of graphics bits to a COMTAL graphics plane (p. 139).  
 WRGPT : writes one graphics bit to a COMTAL graphics plane (p. 141).  
 WRIL2 : writes a horizontal line of pixels to a COMTAL image memory; one pixel value / integer in the buffer (p. 144).  
 WRILN : writes a horizontal line of pixels to a COMTAL image memory; two pixel values / integer in the buffer (p. 147).  
 WRIPT : writes one pixel value to a COMTAL image memory (p. 150).  
 WRIRC : writes an array of integers to a rectangle of a COMTAL image memory; one pixel value / integer in the buffer (p. 153).  
 WRLUT : writes a look-up table to the COMTAL (p. 157).  
 WRPSU : writes a pseudo-color table to the COMTAL (p. 159).  
 WRTAR : writes a target (cursor) location to the COMTAL (p. 161).

The files that follow are test programs for the subprograms above.

TADD2 : program that tests ADDI2 (p.163).  
 TADDI : program that tests ADDIM (p. 164).  
 TCLR : program that tests DSPCL, "DiSPlay CoLor" (p. 165).  
 TCLRG : program that tests CLRGR, "CLear GRaphics" (p. 166).  
 TCLRI : program that tests CLRIM, "CLear IMage" (p. 167).  
 TCMM2 : program that sends all possible single bytes to COMTAL 1 at a time (p. 168).  
 TCMMN : program that sends COMTAL commands via the HP keyboard; tests CMMND (p. 169).  
 TCNT : program that tests subroutine COUNT (p. 170).  
 TCGNS : program that tests the string concatenation facility in HP FORTRAN 77 (p. 171).  
 TCGPY : program that tests ICOPY, "Image COPY" (p. 172).  
 TDIGI : program that tests the function DIGIT (p. 173).  
 TDSP : program that tests DSPBW, "DiSPlay Black & White" and DSPCL, "DiSPlay CoLor" (p. 174).  
 THIST : program that tests HIST, "HISTogram" (p. 175).  
 TNORM : program that tests NORML, "NORMAlize" (p. 176).  
 TNOTE : program that tests NOTE2 and NOTES ( p. 177).

TPNT : program that tests PAINT (p. 178).  
 TPROF : program that tests PROFL, "PROFiling" (p. 179).  
 TRANG : program that tests the function "RANGE" (p. 180).  
 TRDTA : program that tests RDTAR, "ReaD TARget" (p. 181).  
 TSETV : program that tests SETV, "SET Video camera" (p. 182).  
 TSPRD : program that tests SPRED, "SPREaD pixel values" (p. 183).  
 TSUBI : program that tests SUBIM and SUBI2 (p.184).  
 TTHRS : program that tests THRS, "THReSHolding" (p.185).  
 TTSTI : program that tests TSTI1, "TeST Image 1" (p. 186).  
 TTV2C : program that tests TV2CM and TV2C4, "TV to CoMtal" transfers (p. 187).  
 TWAIT : program that tests WAIT (p. 188).  
 TWIPE : program that tests WIPGR and DSPGR (p. 189).  
 TXFDS : program that tests BWFDS and CLFDS, "Black & White From DiSk" and  
 "CoLor From DiSk" (p. 190).  
 TXGLN : program that tests WRGLN and RDGLN, "WRite Graphics LiNe" and  
 "ReaD Graphics LiNe" (p. 191).  
 TXGPT : program that tests WRGPT and RDGPT, "WRite Graphics Point" and  
 "ReaD Graphics Point" (p. 192).  
 TXILN : program that tests WRILN and RDILN, "WRite Image LiNe" and  
 "ReaD Image LiNe" (p. 193).  
 TXIPT : program that tests WRIPT and RDIPT, "WRite Image Point" and  
 "ReaD Image Point" (p. 194).  
 TXIRC : program that tests WRIRC and RDIRC, "WRite Image ReCtangle" and  
 "ReaD Image ReCtangle" (p. 195).  
 TXLUT : program that tests WRLUT and RDLUT, "WRite Look Up Table" and  
 "ReaD Look Up Table" (p. 196).  
 TXPSU : program that tests WRPSU and RDPSU, "WRite PSeUdo-color table" and  
 "ReaD PSeUdo-color table" (p. 197).  
 TXTAR : program that tests WRTAR and RDTAR, "WRite TARget" and "ReaD TARget" (p. 199).

#### CROSS-REFERENCE FILE

To use this file to find out which procedures or programs call a certain procedure "FRED", just use the text editor to locate all the lines that contain FRED. One of those lines is the line "FRED calls:". The remaining lines identify which procedures call FRED.

HP FORTRAN77 intrinsic functions (which need not be loaded manually) are preceded with a \*.

ADDI2 calls:  
 DIGIT CMMND RANGE  
 ADDIM calls:



CMMND DIGIT RANGE  
 BWFDS calls:  
   CMMND DIGIT OPEN RANGE READF WRILN  
 CLFDS calls:  
   CMMND DIGIT OPEN RANGE READF WRILN  
 CLRGR calls:  
   CMMND DIGIT RANGE  
 CLRIM calls:  
   CMMND DIGIT RANGE  
 CMMN2 calls:  
   \*LEN  
 CMMND calls no other procedures.  
 COUNT calls:  
   RANGE RDIL2  
 DELAY calls no other procedures.  
 DIGIT calls:  
   RANGE  
 DSPBW calls:  
   RANGE  
 DSPCL calls:  
   CMMND DIGIT RANGE  
 DSPGR calls:  
   CMMN2 DIGIT RANGE  
 DSPVD calls:  
   CMMND DIGIT  
 HILO calls:  
   RDIL2  
 HISTO calls:  
   CMMN2 DELAY DIGIT RANGE  
 ICOPY calls:  
   CMMND> DIGIT RANGE  
 ICPY2 calls:  
   CMMND DIGIT RANGE  
 MERGE calls no other procedures.  
 NORML calls:  
   HILO RDIL2 WRIL2  
 NOTE2 calls:  
   ADDGR CMMN2 CMMND DELAY DIGIT \*LEN RANGE  
 NOTES calls:  
   CMMN2 CMMND DELAY DIGIT DSPGR RANGE  
 PAINT calls:  
   CMMND RDTAR WAIT WRIRC  
 PROFL calls:  
   CHAR CMMND DIGIT RANGE WAIT  
 RANGE calls no other procedures.  
 RDGLN calls:

RANGE  
 RDGPT calls:  
     BTEST RANGE  
 RDIL2 calls:  
     \*ICAR RANGE  
 RDILN calls:  
     RANGE  
 RDIPT calls:  
     \*ICAR RANGE  
 RDIRC calls:  
     \*ICAR RANGE RDILN  
 RDLUT calls:  
     RANGE  
 RDPSU calls no other procedures.  
 RDTAB calls:  
     BTEST RANGE  
 RDTAR calls no other procedures.  
 SETV calls:  
     CMMND DIGIT RANGE  
 SPLIT calls no other procedures.  
 SPRED calls:  
     FLOAT HILO IFIX RDIL2 WRIL2  
 SSORT calls no other procedures.  
 SUBI2 calls:  
     CMMND DIGIT RANGE  
 SUBIM calls:  
     CMMND DIGIT RANGE  
 THRSH calls:  
     RANGE RDIL2 WRIL2  
 TSTI1 calls:  
     WRILN  
 TV2C4 calls:  
     ADDI2 RANGE TV2CM  
 TV2CM calls:  
     CMMND DIGIT DSPBW RANGE  
 WAIT calls no other procedures.  
 WIPGR calls:  
     CMMN2 DIGIT RANGE  
 WRGLN calls:  
     RANGE  
 WRGPT calls:  
     \*IBCLR \*IBSET RANGE  
 WRIL2 calls:  
     CHAR RANGE  
 WRILN calls:  
     RANGE

WRIPT calls:  
     RANGE  
 WRIRC calls:  
     CHAR RANGE RDIL2 WRIL2  
 WRLUT calls:  
     RANGE  
 WRPSU calls no other procedures.  
 WRTAR calls:  
     RANGE

The following are test programs for many of the procedures above:

TADD2 calls:  
     ADDI2 CMMND DIGIT RANGE  
 TADDI calls:  
     ADDIM CMMND DIGIT RANGE  
 TCLR calls:  
     DSPCL  
 TCLRG calls:  
     CLRGR CMMND DIGIT RANGE  
 TCLRI calls:  
     CLRIM CMMND DIGIT RANGE  
 TCMM2 calls:  
     CMMND WAIT  
 TCMMN calls:  
     CMMND  
 TCNT calls:  
     COUNT RANGE RDIL2  
 TCONS calls no other procedures.  
 TCOPY calls:  
     CMMND DIGIT ICOPY RANGE  
 TDELA calls:  
     DELAY CMMN2 WAIT  
 TDIGI calls:  
     DIGIT RANGE  
 TDSP calls:  
     CMMND DSPBW DSPCL RANGE WAIT  
 TDSPV calls:  
     DSPVD CMMND DIGIT WAIT DSPBW RANGE  
 THIST calls:  
     CMMN2 DIGIT HISTO RANGE  
 TNORM calls:  
     HILO NORML RANGE RDIL2 WRIL2  
 TNOTE calls:  
     CMMN2 CMMND DELAY DIGIT DSPGR NOTE2 NOTES RANGE WRTAR  
 TPNT calls:

CMMND PAINT RANGE RDILN RDTAR WRILN WRIRC  
 TPROF calls:  
 CMMND DIGIT PROFL RANGE WAIT  
 TRANG calls:  
 RANGE  
 TRDTA calls:  
 RANGE RDTAB  
 TRSET calls no other procedures.  
 TSETV calls:  
 SETV RANGE DIGIT CMMND DSPBW  
 TSPRD calls:  
 RANGE RDIL2 SPRED WRIL2  
 TSSRT calls no other procedures.  
 TSUBI calls:  
 CMMND DIGIT RANGE SUBI2 SUBIM WAIT  
 TTHRS calls:  
 RANGE RDIL2 THSH WRIL2  
 TTSTI calls no other procedures.  
 TTV2C calls:  
 ADDI2 CMMND DIGIT DSPBW DSPVD RANGE TV2C4 TV2CM WAIT  
 TWAIT calls:  
 WAIT  
 TWIPE calls:  
 CMMN2 DIGIT DSPGR RANGE WAIT WIPGR  
 TXFDS calls:  
 BWFDS CLFDS CMMND DIGIT RANGE WRILN  
 TXGLN calls:  
 RANGE RDGLN WRGLN  
 TXGPT calls:  
 RANGE RDGPT RDTAR WRGPT  
 TXILN calls:  
 RANGE RDILN WRILN  
 TXIPT calls:  
 RANGE RDIPT WAIT WRIPT  
 TXIRC calls:  
 RANGE RDILN RDIRC WRILN WRIRC  
 TXLUT calls:  
 RANGE RDLUT WRLUT  
 TXPSU calls:  
 RDPSU WRPSU RANGE WAIT  
 TXTAR calls:  
 RDTAR WRTAR RANGE

&ADDIM T=00004 IS ON CR00021 USING 00024 BLKS R=0000

```
0001 CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
0002 SUBROUTINE ADDIM(C, A, B)
0003 CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
0004 C
0005 C***PARAMETER DECLARATIONS:
0006 C
0007 INTEGER C ! the COMTAL image into which the sum of
0008 C ! image A and image B is placed by (C = A + B)
0009 INTEGER A, B ! the images whose sum is taken (C = A + B)
0010 C
0011 C***INTRODUCTION:
0012 C
0013 C The subroutine "ADD Images" takes a pixel by pixel sum
0014 C of images A and B and places the resulting image into image C.
0015 C The truecolor image B is used in the processing of ADDIM, and will
0016 C be left as the combination of C, A, and B for red, green, and blue
0017 C respectively.
0018 C
0019 C ADDIM does not do any scaling of the addition. If scaling is desired,
0020 C see the procedure ADDI2.
0021 C
0022 C***LANGUAGE:
0023 C
0024 C FORTRAN 77, the HP 1000 version for RTE-6/VM.
0025 C
0026 C***LIMITATIONS:
0027 C
0028 C The truecolor B image is destroyed during this operation. C is
0029 C obviously destroyed. This subroutine is accomplished using COMTAL
0030 C commands that exploit the pipeline processors. Because of this, the
0031 C processing steps are obscure. For example, there is no motivation
0032 C outside the COMTAL instructions for making the combination of C, A,
0033 C B a color image. Readers should be aware of these obscurities before
0034 C trying to understand the code.
0035 C
0036 C If the sum of any two pixels exceeds 255, the value in C is set to 255.
0037 C
0038 C ADDIM does not scale or offset the sum result. If you wish to scale
0039 C the sums, see the subroutine ADDI2.
0040 C
0041 C If any of the image numbers are out of range, an error message is printed
0042 C and no further processing takes place.
0043 C This subroutine assumes that 0 is not a legal image for the COMTAL
0044 C configuration.
```

```

0045 C
0046 C***SUBPROGRAMS CALLED:
0047 C
0048 C   name      source  load   remarks
0049 C   -----  -
0050 C   CMMND    &CMMND  %CMMND Sends a command to the COMTAL as if the
0051 C           command were sent from the keyboard
0052 C   RANGE    &RANGE  %RANGE logical function that determines if the 1st
0053 C           parameter is within the range of the 2nd & 3rd.
0054 C   DIGIT    &DIGIT  %DIGIT character*1 function which returns '0'-'9'
0055 C           according to integer input 0-9.
0056 C
0057 C***WRITTEN BY:
0058 C
0059 C   KEITH MILLER, July, 1984, with the support of a NASA-ASEE
0060 C   summer fellowship.
0061 C
0062 C***REVISION HISTORY:
0063 C
0064 C
0065 C***LOCAL VARIABLES:
0066 C
0067 C   INTEGER      IBUF(128) ! a buffer for passing commands to COMTAL
0068 C   CHARACTER*255 CBUF      ! character overlay for IBUF
0069 C   EQUIVALENCE  (IBUF,CBUF)
0070 C
0071 C   LOGICAL      RANGE      ! function that determines if 1st parameter
0072 C                   ! is within 2nd and 3rd parameter
0073 C
0074 C   INTEGER      IMLO, IMHI ! limits on COMTAL image numbers
0075 C   INTEGER      TERM
0076 C   CHARACTER*1  DIGIT      ! character*1 function that returns '0'-'9'
0077 C                   ! according to integer input 0-9.
0078 C
0079 C***INITIALIZATIONS:
0080 C
0081 C   DATA        IMLO/1/, IMHI/4/
0082 C   DATA        TERM/1/
0083 C
0084 C***PROCESSING
0085 C
0086 C   IF (.NOT.(RANGE(A, IMLO, IMHI))) GOTO 8001 ! error return
0087 C   IF (.NOT.(RANGE(B, IMLO, IMHI))) GOTO 8101 ! error return
0088 C   IF (.NOT.(RANGE(C, IMLO, IMHI))) GOTO 8201 ! error return
0089 C
0090 C   The following character string sends a series of keyboard

```

```

0091 C      commands to the COMTAL. In the comments below, each command
0092 C      is explained. The notation #X where X is either A, B, or C
0093 C      stands for the single character that corresponds to the single
0094 C      digit number associated with the parameter X.
0095 C      In this notation, letters in caps were entered into CBUF, and
0096 C      lower case letters are the full commands filled in by the COMTAL
0097 C      NOTE: this code assumes that the digit 0 is NOT a legal value for
0098 C      the parameters A, B, and C.
0099 C      The '$' separates COMTAL commands.
0100 C
0101 C      CBUF =
0102 C      1  'UN I B $'//
0103 C          UNassign Image B ! just in case B is already assigned.
0104 C      2  'AS T B '//DIGIT(C)//' '//DIGIT(A)//' '//DIGIT(B)//' $'//
0105 C          ASsign Truecolor image B red #C blue #A green #B
0106 C      3  'D I B $'//
0107 C          Display Image B
0108 C      4  'SE COM G + B / 1 $'//      ! sets 0 offset by default, and
0109 C          ! "/ 1" sets no scaling.
0110 C          SEt COMbine <Green + Blue> / 1
0111 C      5  'A COM $'//
0112 C          Add COMbine
0113 C      6  'I '//DIGIT(C)//' D R $'//
0114 C          Image #C = Displayed Image Red ! Red is arbitrary, since difference
0115 C          ! of images is monochrome
0116 C      7  'D I '//DIGIT(C)//' $'//
0117 C          Display Image #C
0118 C      8  'SU COM '
0119 C          SUbtract COMbine.
0120 C          CALL CMMND(IBUF, 78)
0121 C          RETURN
0122 C
0123 C***ERROR RETURNS
0124 C
0125 C      8001 WRITE(TERM, 8203) A
0126 C      8003 FORMAT(' THE 2ND IMAGE PARAMETER,', I3, ', IS OUT OF RANGE.')
0127 C          GOTO 8900
0128 C
0129 C      8101 WRITE(TERM, 8103) B
0130 C      8103 FORMAT(' THE 3RD IMAGE PARAMETER,', I3, ', IS OUT OF RANGE.')
0131 C          GOTO 8900
0132 C
0133 C      8201 WRITE(TERM, 8203) C
0134 C      8203 FORMAT(' THE 1ST IMAGE PARAMETER,', I3, ', IS OUT OF RANGE.')
0135 C          GOTO 8900
0136 C

```

```
0137 8900 WRITE(TERM, 8901)
0138 8901 FORMAT(' ADDIM RETURNS WITHOUT FURTHER PROCESSING. ')
0139     RETURN
0140 C
0141     END
0142
```



&ADDI2 T=00004 IS ON CR00021 USING 00024 BLKS R=0000

```
0001 CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
0002 SUBROUTINE ADDI2(C, A, B, SCALE)
0003 CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
0004 C
0005 C***PARAMETER DECLARATIONS:
0006 C
0007 INTEGER C ! the COMTAL image into which the sum of
0008 C ! image A and image B is placed (C = A + B)
0009 INTEGER A, B ! the images whose sum is taken (C = A + B)
0010 INTEGER SCALE! each pixel sum divided by this number.
0011 C
0012 C***INTRODUCTION:
0013 C
0014 C The subroutine "ADD Images" takes a pixel by pixel sum
0015 C of images A and B, divides each sum by SCALE, and places the results
0016 C into image C. SCALE must be between 1 and 9 inclusive.
0017 C The truecolor image B is used in the processing of ADDI2, and will
0018 C be left as the combination of C, A, and B for red, green, and blue
0019 C respectively.
0020 C
0021 C***LANGUAGE:
0022 C
0023 C FORTRAN 77, the HP 1000 version for RTE-6/VM.
0024 C
0025 C***LIMITATIONS:
0026 C
0027 C The truecolor B image is destroyed during this operation. C is
0028 C obviously destroyed. This subroutine is accomplished using COMTAL
0029 C commands that exploit the pipeline processors. Because of this, the
0030 C processing steps are obscure. For example, there is no motivation
0031 C outside the COMTAL instructions for making the combination of C, A, and B
0032 C a color image. Readers should be aware of these obscurities before trying to
0033 C understand the code.
0034 C
0035 C
0036 C The three image numbers must be distinct.
0037 C
0038 C If the sum of any two pixels exceeds 255, the value in C is set to 255.
0039 C
0040 C The SCALE factor must be between 1 and 9 inclusive.
0041 C
0042 C If any of the image numbers are out of range, an error message is printed
0043 C and no further processing takes place.
0044 C This subroutine assumes that 0 is not a legal image for the COMTAL
```

```

0045 C configuration.
0046 C
0047 C***SUBPROGRAMS CALLED:
0048 C
0049 C name source load remarks
0050 C -----
0051 C CMMND &CMMND %CMMND Sends a command to the COMTAL as if the
0052 C command were sent from the keyboard
0053 C RANGE &RANGE %RANGE logical function that determines if the 1st
0054 C parameter is within the range of the 2nd & 3rd.
0055 C DIGIT &DIGIT %DIGIT character*1 function which returns '0'-'9'
0056 C according to integer input 0-9.
0057 C
0058 C***WRITTEN BY:
0059 C
0060 C KEITH MILLER, July, 1984, with the support of a NASA-ASEE
0061 C summer fellowship.
0062 C
0063 C***REVISION HISTORY:
0064 C
0065 C
0066 C***LOCAL VARIABLES:
0067 C
0068 C INTEGER IBUF(128) ! a buffer for passing commands to COMTAL
0069 C CHARACTER*255 CBUF ! character overlay for IBUF
0070 C EQUIVALENCE (IBUF,CBUF)
0071 C
0072 C LOGICAL RANGE ! function that determines if 1st parameter
0073 C ! is within 2nd and 3rd parameter
0074 C
0075 C INTEGER IMLO, IMHI ! limits on COMTAL image numbers
0076 C INTEGER TERM
0077 C CHARACTER*1 DIGIT ! character*1 function that returns '0'-'9'
0078 C ! according to integer input 0-9.
0079 C
0080 C***INITIALIZATIONS:
0081 C
0082 C DATA IMLO/1/, IMHI/4/
0083 C DATA TERM/1/
0084 C
0085 C***PROCESSING
0086 C
0087 C IF (.NOT.(RANGE(A, IMLO, IMHI))) GOTO 8001 ! error return
0088 C IF (.NOT.(RANGE(B, IMLO, IMHI))) GOTO 8101 ! error return
0089 C IF (.NOT.(RANGE(C, IMLO, IMHI))) GOTO 8201 ! error return
0090 C

```

```

0091 C      The following character string sends a series of keyboard
0092 C      commands to the COMTAL. In the comments below, each command
0093 C      is explained. The notation #X where X is either A, B, or C
0094 C      stands for the single character that corresponds to the single
0095 C      digit number associated with the parameter X.
0096 C      In this notation, letters in caps were entered into CBUF, and
0097 C      lower case letters are the full commands filled in by the COMTAL
0098 C      NOTE: this code assumes that the digit 0 is NOT a legal value for
0099 C      the parameters A, B, and C.
0100 C      The '$' separates COMTAL commands.
0101 C
0102 C      CBUF =
0103 C      1  'UN I 8 $'//
0104 C          UNassign Image 8 ! just in case 8 is already assigned.
0105 C      2  'AS T 8 '//DIGIT(C)//' '//DIGIT(A)//' '//DIGIT(B)//' $'//
0106 C          ASsign Truecolor image 8 red #C blue #A green #B
0107 C      3  'D I 8 $'//
0108 C          Display Image 8
0109 C      4  'SE COM G + B / '//DIGIT(SCALE)//' $'//
0110 C          SEt COMbine <Green + Blue> / SCALE
0111 C      5  'A COM $'//
0112 C          Add COMbine
0113 C      6  'I '//DIGIT(C)//' D R $'//
0114 C          Image #C = Displayed image Red ! Red is arbitrary, since difference
0115 C          ! of images is monochrome
0116 C      7  'D I '//DIGIT(C)//' $'//
0117 C          Display Image #C
0118 C      8  'SU COM '
0119 C          SUbtract COMbine.
0120 C      CALL CMMND(IBUF, 77)
0121 C      RETURN
0122 C
0123 C      C***ERROR RETURNS
0124 C
0125 C      8001 WRITE(TERM, 8003) A
0126 C      8003 FORMAT(' THE 2ND IMAGE PARAMETER, ', I3, ', IS OUT OF RANGE.')
0127 C      GOTO 8900
0128 C
0129 C      8101 WRITE(TERM, 8103) B
0130 C      8103 FORMAT(' THE 3RD IMAGE PARAMETER, ', I3, ', IS OUT OF RANGE.')
0131 C      GOTO 8900
0132 C
0133 C      8201 WRITE(TERM, 8203) C
0134 C      8203 FORMAT(' THE 1ST IMAGE PARAMETER, ', I3, ', IS OUT OF RANGE.')
0135 C      GOTO 8900
0136 C

```

```
0137 8900 WRITE(TERM, 8901)
0138 8901 FORMAT(' ADDI2 RETURNS WITHOUT FURTHER PROCESSING.')
0139      RETURN
0140 C
0141      END
0142
```

&BWFD S T=00004 IS ON CR0021 USING 00018 BLKS R=0000

```
0001 CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
0002 SUBROUTINE BWFD S (IMAGE, FLNAME)
0003 CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
0004 C
0005 C***PARAMETER DECLARATIONS:
0006 C
0007 INTEGER IMAGE ! the number of the COMTAL image plane to be
0008 C ! filled from the HP disk file.
0009 INTEGER FLNAME(3)! the HP filename from which an image will be read.
0010 C
0011 C***INTRODUCTION:
0012 C
0013 C The subroutine "Black and White From Disk" transfers one b&w image
0014 C from the HP to the COMTAL. The image number and the name of the HP
0015 C disk are given as parameters.
0016 C
0017 C***LANGUAGE:
0018 C
0019 C FORTRAN 77, the HP 1000 version for RTE-6/VM.
0020 C
0021 C***LIMITATIONS:
0022 C
0023 C BWFD S only works for b&w images. Since a color image requires
0024 C three separate b&w images, another subroutine, CLFDS is available
0025 C for reading color images from the disk.
0026 C
0027 C***SUBPROGRAMS CALLED:
0028 C
0029 C name source load remarks
0030 C -----
0031 C RANGE &RANGE %RANGE logical function that determines if the 1st
0032 C argument is within the 2nd and 3rd inclusive.
0033 C OPEN ----- HP FORTRAN77 intrinsic subroutine; opens a file
0034 C and stores data block information in first param.
0035 C READF ----- HP FORTRAN77 intrinsic subroutine; reads a record
0036 C from a file; requires an OPENed data block.
0037 C WRILN &WRILN %WRILN WRites a COMTAL Image horizontal LiNe; 2 pixels
0038 C per integer in the buffer.
0039 C CMMND &CMMND %CMMND sends commands to the COMTAL as if they were
0040 C typed at the COMTAL keyboard.
0041 C DIGIT &DIGIT %DIGIT a character*1 function that returns a single
0042 C ASCII digit when given an integer 0-9
0043 C
0044 C***WRITTEN BY:
```

```

0045 C
0046 C The code on which this subprogram is based was written by
0047 C NETTIE D. FAULCON, July, 1983. This subprogram was written by
0048 C KEITH MILLER, July, 1984, with the support of a NASA-ASEE
0049 C summer fellowship.
0050 C
0051 C***REVISION HISTORY:
0052 C
0053 C
0054 C***LOCAL VARIABLES:
0055 C
0056 C     INTEGER  DBLOCK(144) ! a Data control BLOCK used in file IO.
0057 C     INTEGER  IERR        ! holds HP IO return code.
0058 C     INTEGER  IBUF(256)   ! buffers one horizontal row of COMTAL pixels.
0059 C     INTEGER  ROW         ! loop indexing which COMTAL row.
0060 C     LOGICAL  RANGE      ! logical function that determines if 1st parameter
0061 C                    ! is between 2nd and 3rd, inclusive.
0062 C     INTEGER  IMLO, IMHI ! limits on COMTAL image numbers.
0063 C     INTEGER  TERM       ! logical unit for terminal output
0064 C
0065 C     CHARACTER*1 DIGIT    ! function that returns '0','1',...,or '9'
0066 C                    ! according to a 0,1,...,or 9 integer input.
0067 C     CHARACTER*255 CBUF  ! overlays IBUF
0068 C     EQUIVALENCE (CBUF,IBUF)
0069 C
0070 C***INITIALIZATIONS:
0071 C
0072 C     DATA    IMLO/1/, IMHI/4/
0073 C     DATA    TERM/1/
0074 C
0075 C***PROCESSING
0076 C
0077 C     IF (.NOT.(RANGE(IMAGE ,IMLO,IMHI))) GOTO 8001 ! error return
0078 C
0079 C     CALL OPEN(DBLOCK, IERR, FLNAME)
0080 C     IF (IERR .LT. 0) GOTO 8201 ! error return, open failed
0081 C
0082 C     CBUF = 'D I '//DIGIT(IMAGE) ! Display the Image to be filled.
0083 C     CALL CMMND(IBUF,5)
0084 C
0085 C     DO 1000 ROW = 0, 511
0086 C         CALL READF(DBLOCK, IERR, IBUF)
0087 C         IF (IERR .LT. 0) GOTO 8301 ! error return, bad read
0088 C         CALL WRILN(IMAGE, ROW, IBUF)
0089 C     1000 CONTINUE
0090 C

```

```

0091         RETURN
0092 C
0093 C***ERROR RETURNS
0094 C
0095     8001 WRITE(TERM, 8003) IMAGE, IMLO, IMHI
0096     8003 FORMAT(' IMAGE NUMBER, ', I3, ', IS OUT OF RANGE: ', I214, '.')
0097         GOTO 8900
0098 C
0099     8201 WRITE(TERM, 8203) IERR
0100     8203 FORMAT(' ERROR OCCURED DURING IMAGE FILE OPENING: ', I4, '.')
0101         GOTO 8900
0102 C
0103     8301 WRITE(TERM, 8303) IERR
0104     8303 FORMAT(' ERROR OCCURED DURING IMAGE FILE READ: ', I4, '.')
0105         GOTO 8900
0106 C
0107     8900 WRITE(TERM, 8901)
0108     8901 FORMAT(' BWFDs FAILS. NO TRANSFER TAKES PLACE.')
0109         RETURN
0110         END

```

&CLFDS T=00004 IS ON CR00021 USING 00018 BLKS R=0000

```
0001 CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
0002 SUBROUTINE CLFDS(RED, GREEN, BLUE, COLOR, FLNAME)
0003 CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
0004 C
0005 C**PARAMETER DECLARATIONS:
0006 C
0007 INTEGER RED, GREEN, BLUE ! colors of the COMTAL image planes
0008 C | to be filled from the HP disk file.
0009 INTEGER COLOR ! the truecolor image to be formed and displayed.
0010 INTEGER FLNAME(3) ! the HP filename from which an image will be read.
0011 C
0012 C**INTRODUCTION:
0013 C
0014 C The subroutine "CoLoR image From DiSk" transfers three b&w images
0015 C from the HP to the COMTAL, and then assigns these to a truecolor
0016 C image on the COMTAL. The filename names a single file holding all
0017 C three monochrome images. The COLOR image number is automatically
0018 C unassigned and assigned by CLFDS, and after all three component
0019 C images are transferred, the color image is displayed.
0020 C
0021 C**LANGUAGE:
0022 C
0023 C FORTRAN 77, the HP 1000 version for RTE-6/VM.
0024 C
0025 C**LIMITATIONS:
0026 C
0027 C The three parameters RED, GREEN and BLUE must be distinct and
0028 C within the IMLO, IMHI range. COLOR must be within the TRLO, TRHI
0029 C range (which is mutually exclusive with IMLO-IMHI). The limits on
0030 C truecolor numbers are arbitrary. However, this subroutine enforces the
0031 C arbitrary limits. The HP image file must contain all three monochrome images
0032 C in the order RED, GREEN, and BLUE.
0033 C
0034 C**SUBPROGRAMS CALLED:
0035 C
0036 C name source load remarks
0037 C -----
0038 C RANGE &RANGE %RANGE logical function that determines if the 1st
0039 C argument is within the 2nd and 3rd inclusive.
0040 C OPEN ----- HP FORTRAN77 intrinsic subroutine; opens a file
0041 C and stores data block information in first param.
0042 C READF ----- HP FORTRAN77 intrinsic subroutine; reads a record
0043 C from a file; requires an OPENed data block.
0044 C WRILN &WRILN %WRILN WRites a COMTAL Image horizontal LiNe; 2 pixels
```



```

0045 C                                     per integer in the buffer.
0046 C   CMMND   &CMMND   %CMMND transfers a command string to the COMTAL, which
0047 C                                     accepts it as a keyboard command.
0048 C   DIGIT   &DIGIT   %DIGIT character*1 function that returns a single digit
0049 C                                     on legal integer inputs 0-9.
0050 C
0051 C***WRITTEN BY:
0052 C
0053 C   The code on which this subprogram is based was written by
0054 C   NETTIE D. FAULCON, July, 1983. This subprogram was written by
0055 C   KEITH MILLER, July, 1984, with the support of a NASA-ASEE
0056 C   summer fellowship.
0057 C
0058 C***REVISION HISTORY:
0059 C
0060 C
0061 C***LOCAL VARIABLES:
0062 C
0063 C       INTEGER   DBLOCK(144) ! a Data control BLOCK used in file 10.
0064 C       INTEGER   IERR        ! holds HP IO return code.
0065 C       INTEGER   IBUF(256)   ! buffers one horizontal row of COMTAL pixels.
0066 C       INTEGER   ROW         ! loop indexing which COMTAL row.
0067 C       LOGICAL   RANGE       ! logical function that determines if 1st parameter
0068 C                                     ! is between 2nd and 3rd, inclusive.
0069 C       INTEGER   IMLO, IMHI  ! limits on COMTAL b&w image numbers.
0070 C       INTEGER   TRLO, TRHI  ! limits on COMTAL truecolor image numbers.
0071 C       INTEGER   TERM        ! logical unit for terminal output
0072 C
0073 C       CHARACTER*255 CBUF     ! character overlay for sending COMTAL commands
0074 C       EQUIVALENCE (CBUF,IBUF)
0075 C       CHARACTER*1  DIGIT    ! function that returns '0','1',..., or '9'
0076 C                                     ! according to 0,1,....,or 9 integer input.
0077 C
0078 C***INITIALIZATIONS:
0079 C
0080 C       DATA      IMLO/1/, IMHI/4/
0081 C       DATA      TRLO/5/, TRHI/9/
0082 C       DATA      TERM/1/
0083 C
0084 C***PROCESSING
0085 C
0086 C       IF (.NOT.(RANGE(RED   ,IMLO,IMHI))) GOTO 8001 ! error return
0087 C       IF (.NOT.(RANGE(GREEN ,IMLO,IMHI))) GOTO 8101 ! error return
0088 C       IF (.NOT.(RANGE(BLUE  ,IMLO,IMHI))) GOTO 8201 ! error return
0089 C
0090 C       IF ((RED .EQ. GREEN) .OR. (GREEN .EQ. BLUE)

```

```

0091      1      .OR. (RED .EQ. BLUE))          GOTO 8301 ! error return
0092 C
0093      IF (.NOT.(RANGE(COLOR, TRLO,TRHI))) GOTO 8401 ! error return
0094 C
0095      CALL OPEN(DBLOCK, IERR, FLNAME)
0096      IF (IERR .LT. 0) GOTO 8501 ! error return, open failed
0097 C
0098      CBUF = 'D I '//DIGIT(RED) ! Display the RED Image as it is filled.
0099      CALL CMMND(IBUF,5)
0100      DO 1000 ROW = 0,511
0101          CALL READF(DBLOCK, IERR, IBUF)
0102          IF (IERR .LT. 0) GOTO 8601 ! error return, file read failed
0103          CALL WRILN(RED, ROW, IBUF)
0104      1000 CONTINUE
0105 C
0106      CBUF = 'D I '//DIGIT(GREEN) ! Display the GREEN Image as it is filled.
0107      CALL CMMND(IBUF,5)
0108      DO 2000 ROW = 0,511
0109          CALL READF(DBLOCK, IERR, IBUF)
0110          IF (IERR .LT. 0) GOTO 8701 ! error return, file read failed
0111          CALL WRILN(GREEN, ROW, IBUF)
0112      2000 CONTINUE
0113 C
0114      CBUF = 'D I '//DIGIT(BLUE) ! Display the BLUE Image as it is filled.
0115      CALL CMMND(IBUF,5)
0116      DO 3000 ROW = 0,511
0117          CALL READF(DBLOCK, IERR, IBUF)
0118          IF (IERR .LT. 0) GOTO 8801 ! error return, file read failed
0119          CALL WRILN(BLUE, ROW, IBUF)
0120      3000 CONTINUE
0121 C
0122 C      Let #C, #R, #G, #B be the DIGIT associated with COLOR, RED,
0123 C      GREEN, and BLUE respectively; then the following CMMND calls
0124 C      read as follows: UNassign Image #C; ASsign Truecolor #C
0125 C      red #R green #G blue #B; Display Image #C
0126 C
0127      CBUF = 'UN I '//DIGIT(COLOR)
0128      CALL CMMND(IBUF,6)
0129      CBUF = 'AS T '//DIGIT(COLOR)//' '//DIGIT(RED)//' '//
0130      1      DIGIT(GREEN)//' '//DIGIT(BLUE)
0131      CALL CMMND(IBUF,12)
0132      CBUF = 'D I '//DIGIT(COLOR)
0133      CALL CMMND(IBUF,5)
0134      RETURN
0135 C
0136 C***ERROR RETURNS

```

```

0137 C
0138 8001 WRITE(TERM, 8003) RED, IMLO, IMHI
0139 8003 FORMAT(' RED IMAGE NUMBER,',I3,', IS OUT OF RANGE:',214,',')
0140 GOTO 8900
0141 C
0142 8101 WRITE(TERM, 8103) GREEN, IMLO, IMHI
0143 8103 FORMAT(' GREEN IMAGE NUMBER,',I3,', IS OUT OF RANGE:',214,',')
0144 GOTO 8900
0145 C
0146 8201 WRITE(TERM, 8203) BLUE, IMLO, IMHI
0147 8203 FORMAT(' BLUE IMAGE NUMBER,',I3,', IS OUT OF RANGE:',214,',')
0148 GOTO 8900
0149 C
0150 8301 WRITE(TERM, 8303) RED, GREEN, BLUE
0151 8303 FORMAT(' 3 MONOCHROME IMAGES MUST BE DISTINCT. YOURS:',314)
0152 GOTO 8900
0153 C
0154 8401 WRITE(TERM, 8403) COLOR, CLLO, CLHI
0155 8403 FORMAT(' YOUR TRUCOLOR IMAGE,',I4,', IS OUT OF RANGE:',214)
0156 GOTO 3900
0157 C
0158 8501 WRITE(TERM, 8503) IERR
0159 8503 FORMAT(' ERROR WHILE OPENING IMAGE FILE:', 15)
0160 GOTO 8900
0161 C
0162 8601 WRITE(TERM, 8603) IERR
0163 8603 FORMAT(' ERROR WHILE READING IN THE RED MONOCHROME IMAGE:', 14)
0164 GOTO 8900
0165 C
0166 8701 WRITE(TERM, 8703) IERR
0167 8703 FORMAT(' ERROR WHILE READING IN THE GREEN MONOCHROME IMAGE:', 14)
0168 GOTO 8900
0169 C
0170 8801 WRITE(TERM, 8803) IERR
0171 8803 FORMAT(' ERROR WHILE READING IN THE BLUE MONOCHROME IMAGE:', 14)
0172 GOTO 8900
0173 C
0174 8900 WRITE(TERM, 8901)
0175 8901 FORMAT(' CLFDS FAILS. NO TRANSFER TAKES PLACE.')
0176 RETURN
0177 END

```

&CLRGR T=00004 IS ON CR0021 USING 00012 BLKS R=0000

```
0001 CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
0002 SUBROUTINE CLRGR( GRAPH )
0003 CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
0004 C
0005 C***PARAMETER DECLARATIONS:
0006 C
0007 INTEGER GRAPH ! the number of the COMTAL GRAPH to be cleared.
0008 C
0009 C***INTRODUCTION:
0010 C
0011 C CLear GRAPH clears the graphics designated by GRAPH. GRAPH should
0012 C be within the range 1-4 for the present IRD COMTAL system.
0013 C If GRAPH is out of range, an error message is printed and
0014 C no COMTAL transfer takes place.
0015 C
0016 C***LANGUAGE:
0017 C
0018 C FORTRAN 77, the HP-1000 version for RTE-6/VM.
0019 C
0020 C***LIMITATIONS:
0021 C
0022 C GRAPH must be within the limits GRLO and GRHI explained below.
0023 C
0024 C***SUBPROGRAMS CALLED:
0025 C
0026 C name source load remarks
0027 C -----
0028 C RANGE &RANGE %RANGE logical function that determines if its 1st
0029 C parameter is between (inclusive) its last parameters
0030 C CMMND &CMMND %CMMND sends a COMTAL command as if it were typed at
0031 C the COMTAL keyboard.
0032 C DIGIT &DIGIT %DIGIT returns a character (1st parameter) which is the
0033 C character equivalent of the integer 2nd parameter.
0034 C
0035 C***WRITTEN BY:
0036 C
0037 C The code on which this subprogram is based was written by
0038 C HETTIE D. FAULCON, July, 1983. This modification is by
0039 C KEITH MILLER, June, 1984.
0040 C
0041 C***REVISION HISTORY:
0042 C
0043 C***LOCAL VARIABLES:
0044 C
```

```

0045     INTEGER      GRLO,GRHI ! limits on COMTAL GRAPH numbers
0046     CHARACTER*1   WHICH
0047     LOGICAL        RANGE     ! logical function that determines if its
0048                                     ! 1st parameter lies within last 2 parameters.
0049     INTEGER      TERM      ! logical unit number of the terminal output
0050     INTEGER      IBUF(128) ! COMTAL command buffer
0051     CHARACTER*255 CBUF     ! overlays the IBUF command buffer.
0052     EQUIVALENCE   (IBUF,CBUF)
0053     C
0054     C***INITIALIZATIONS:
0055     C
0056         DATA GRLO/1/, GRHI/4/
0057         DATA TERM/1/
0058     C
0059     C***PROCESSING
0060     C
0061     IF (.NOT.(RANGE(GRAPH,GRLO,GRHI))) GOTO 8001 ! error return
0062     C ELSE...clear the GRAPH
0063         CALL DIGIT(WHICH, GRAPH)
0064         CBUF = 'CLEAR GRAPH ' // WHICH ! since CBUF overlays IBUF,
0065                                     ! this statement loads IBUF with
0066                                     ! the COMTAL command
0067         CALL CMMND(IBUF, 13)
0068         RETURN
0069     C
0070     C***ERROR RETURN
0071     C
0072     8001 WRITE(TERM, 8003) GRAPH, GRLO, GRHI
0073     8003 FORMAT(' GRAPH NUMBER, ', I4, ' IS OUT OF RANGE:', 2I3, ',.')
0074     WRITE(TERM, 8005)
0075     8005 FORMAT(' CLRGR FAILS. NO COMMAND SENT TO COMTAL. ')
0076     RETURN
0077     END

```

&CLRIM T=00004 IS ON CR00021 USING 00004 BLKS R=0000

```
0001 CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
0002 SUBROUTINE CLRIM( IMAGE )
0003 CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
0004 C
0005 C***PARAMETER DECLARATIONS:
0006 C
0007 INTEGER IMAGE ! the number of the COMTAL image to be cleared.
0008 C
0009 C***INTRODUCTION:
0010 C
0011 C CLear Image clears the image designated by IMAGE. IMAGE should
0012 C be within the range 1-4 for the present IRD COMTAL system.
0013 C If IMAGE is out of range, an error message is printed and
0014 C no COMTAL transfer takes place.
0015 C
0016 C***LANGUAGE:
0017 C
0018 C FORTRAN 77, the HP-1000 version for RTE-6/VM.
0019 C
0020 C***LIMITATIONS:
0021 C
0022 C IMAGE must be within the limits IMLO and IMHI explained below.
0023 C
0024 C***SUBPROGRAMS CALLED:
0025 C
0026 C name source load remarks
0027 C -----
0028 C RANGE &RANGE %RANGE logical function that determines if its 1st
0029 C parameter is between (inclusive) its last parameterC
0030 C CMMND &CMMND %CMMND sends a COMTAL command as if it were typed at
0031 C the COMTAL keyboard.
0032 C DIGIT &DIGIT %DIGIT a character*1 function which returns the character
0033 C associated with integer inputs 0,....,9.
0034 C
0035 C***WRITTEN BY:
0036 C
0037 C The code on which this subprogram is based was written by
0038 C KETTIE D. FAULCON, July, 1983. This modification is by
0039 C KEITH MILLER, June, 1984.
0040 C
0041 C***REVISION HISTORY:
0042 C
0043 C***LOCAL VARIABLES:
0044 C
```

```

0045     INTEGER      IMLO,IMHI ! limits on COMTAL image numbers
0046     LOGICAL      RANGE      ! logical function that determines if its
0047                                     ! 1st parameter lies within last 2 parameters.
0048     INTEGER      TERM        ! logical unit number of the terminal output
0049     INTEGER      IBUF(128) ! COMTAL command buffer
0050     CHARACTER*1   DIGIT      ! function that returns the ASCII character
0051 C                                     ! associated with integer input, 0.....9.
0052     CHARACTER*255 CBUF      ! overlays the IBUF command buffer
0053     EQUIVALENCE  (IBUF,CBUF)
0054 C
0055 C***INITIALIZATIONS:
0056 C
0057     DATA IMLO/1/, IMHI/4/
0058     DATA TERM/1/
0059 C
0060 C***PROCESSING
0061 C
0062     IF (.NOT.(RANGE(IMAGE,IMLO,IMHI))) GOTO 8001 ! error return
0063 C     ELSE...clear the image
0064         CBUF = 'CLEAR IMAGE ' // DIGIT(IMAGE) ! since CBUF overlays IBUF,
0065                                     ! this statement loads IBUF with
0066                                     ! the COMTAL command
0067         CALL CMMND(IBUF, 13)
0068         RETURN
0069 C
0070 C***ERROR RETURN
0071 C
0072     8001 WRITE(TERM, 8003) IMAGE, IMLO, IMHI
0073     8003 FORMAT(' IMAGE NUMBER, ', I4, ' IS OUT OF RANGE:', 2I3, '.')
0074     WRITE(TERM, 8005)
0075     8005 FORMAT(' CLRIM FAILS. NO COMMAND SENT TO COMTAL.')
```

&CMMN2 T=00004 IS ON CR00021 USING 00018 BLKS R=0000

```
0001 CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
0002 SUBROUTINE CMMN2(INBUF)
0003 CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
0004 C
0005 C***PARAMETERS:
0006 C
0007 CHARACTER*(*) INBUF ! the characters of a COMTAL command string
0008 C
0009 C***INTRODUCTION:
0010 C
0011 C This subroutine "CoMMaNd 2" allows a character string command to be
0012 C sent to the COMTAL much as if the command were typed at the keyboard.
0013 C CMMN2 is designed to be sent constant strings.
0014 C
0015 C The major differences are that the INBUF command string may include
0016 C multiple commands, each separated by the character "$".
0017 C A character array buffer is used in equivalence with an integer array
0018 C in this subroutine to illustrate the utility of the characters and
0019 C still allow obvious compatability.
0020 C
0021 C The subroutine CMMND is very similar, only there an integer buffer
0022 C of fixed size is used with an extra parameter to identify how many
0023 C characters are valid parts of the intended command.
0024 C
0025 C***LANGUAGE:
0026 C
0027 C FORTRAN 77, the HP-1000 version for RTE-6/VM.
0028 C
0029 C***LIMITATIONS:
0030 C
0031 C In order to make it easier to send single COMTAL commands to CMMND,
0032 C the final blank, $, and the required zero byte are added automatically
0033 C to the INBUF string.
0034 C If the last character isn't a blank, CMMND adds one.
0035 C However, the caller should NOT add the final $ or zero byte to the
0036 C string. Note that each $ in the string should be preceded by a blank.
0037 C
0038 C WARNING: When a command is sent to the COMTAL that generates a
0039 C COMTAL error, the COMTAL system is frozen until a manual
0040 C <ESC> (or perhaps several) is entered from the COMTAL keyboard.
0041 C
0042 C***SUBPROGRAMS CALLED:
0043 C
0044 C name source load remarks
```



```

0045 C -----
0046 C  LEN ----- integer function returns length of character string
0047 C
0048 C***WRITTEN BY:
0049 C
0050 C  The original code upon which this subroutine is based was written
0051 C  by NETTIE D. FAULCON , JULY, 1983.
0052 C
0053 C***REVISION HISTORY:
0054 C
0055 C  Modified by Keith Miller, 6/18/84.
0056 C
0057 C***LOCAL VARIABLES:
0058 C
0059 C      CHARACTER*1 CBUF(256) | character buffer
0060 C      INTEGER      IBUF(128) | the character buffer overlaid as integers
0061 C      INTEGER      IZERO      | constant value 0 for making a 0 byte (ZERO)
0062 C      INTEGER      WORDS      | counts number of words
0063 C      INTEGER      NUMCHR     | counts number of bytes
0064 C      INTEGER      LEN        | intrinsic HP FORTRAN77 function that returns
0065 C                               | the length of a character string.
0066 C      CHARACTER*1 BLANK, DOLLAR | special ASCII characters
0067 C      CHARACTER*1 ZERO          | zero is 00000000 binary.
0068 C      EQUIVALENCE (CBUF,IBUF), (ZERO, IZERO)
0069 C
0070 C***INITIALIZATIONS:
0071 C
0072 C      DATA          BLANK/' ', DOLLAR/'$', IZERO/0/
0073 C
0074 C      NUMCHR = LEN(INBUF)
0075 C
0076 C***PROCESSING:
0077 C
0078 C      DO 10 INDEX = 1, NUMCHR
0079 C          CBUF(INDEX) = INBUF(INDEX:INDEX)
0080 C      10 CONTINUE
0081 C
0082 C      IF (CBUF(NUMCHR) .EQ. BLANK) GOTO 30
0083 C... ELSE...
0084 C          NUMCHR = NUMCHR + 1
0085 C          CBUF(NUMCHR) = BLANK
0086 C
0087 C... ADD ENDING CHARACTERS TO COMMAND
0088 C
0089 C      30 CBUF(NUMCHR+1) = DOLLAR
0090 C          CBUF(NUMCHR+2) = ZERO

```

```

0091      CBUF(NUMCHR+3) = ZERO    ! safety precaution
0092      NUMCHR = NUMCHR + 2
0093      C
0094      C... CHANGE BYTE COUNT TO WORD COUNT
0095      C
0096      WORDS = (NUMCHR+1) / 2 ! if N is even, intentional truncation
0097      C
0098      C Programming note:
0099      C The EXEC command parameters are discussed in the HP RTE-6/VM
0100      C Programmer's Reference Manual, 2-19 ff. The COMTAL parameters
0101      C are discussed in section 5.2.4 of the COMTAL User's Manual.
0102      C
0103      C The first parameter to EXEC identifies it as a write command.
0104      C The second parameter identifies the resident HP driver (36B)
0105      C and gives a code for the operation required by this call (500B).
0106      C The third parameter is the command string, and the fourth gives
0107      C the length in words of the buffer that is to be used. The fifth
0108      C parameter is a code for the COMTAL interface that directs the
0109      C command transfer.
0110      C
0111      CALL EXEC(2, 36B + 500B, IBUF, WORDS, 24001B)
0112      RETURN
0113      C
0114      END

```

&CMMND T=00004 IS ON CR00021 USING 00012 BLKS R=0000

```
0001 CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
0002 SUBROUTINE CMMND(INBUF, INCNT)
0003 CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
0004 C
0005 C***PARAMETERS:
0006 C
0007 INTEGER INBUF(128) ! the characters of a COMTAL command string
0008 INTEGER INCNT ! the number of characters in command string
0009 C
0010 C***INTRODUCTION:
0011 C
0012 C This subroutine "CoMMaND" allows a character string command to be
0013 C sent to the COMTAL much as if the command were typed at the keyboard.
0014 C The major differences are that the INBUF command string may include
0015 C multiple commands, each separated by the character "$". Notice
0016 C that NUMCHR is in terms of characters (bytes), not words. INBUF
0017 C is an integer array to be compatible with previously written software,
0018 C but an F77 character array would probably be more appropriate.
0019 C A character array buffer is used in equivalence with an integer array
0020 C in this subroutine to illustrate the utility of the characters and
0021 C still allow obvious compatibility.
0022 C
0023 C***LANGUAGE:
0024 C
0025 C FORTRAN 77, the HP-1000 version for RTE-6/VM.
0026 C
0027 C***LIMITATIONS:
0028 C
0029 C In order to make it easier to send single COMTAL commands to CMMND,
0030 C the final blank, $, and the required zero byte are added automatically
0031 C to the INBUF string.
0032 C If the last character isn't a blank, CMMND adds one.
0033 C However, the caller should NOT add the final $ or zero byte to the
0034 C string. Note that each $ in the string should be preceded by a blank.
0035 C
0036 C WARNING: When a command is sent to the COMTAL that generates a
0037 C COMTAL error, the COMTAL system is frozen until a manual
0038 C <ESC> (or perhaps several) is entered from the COMTAL keyboard.
0039 C
0040 C***SUBPROGRAMS CALLED: NONE.
0041 C
0042 C
0043 C***WRITTEN BY:
0044 C
```

```

0045 C   The original code upon which this subroutine is based was written
0046 C   by NETTIE D. FAULCON , JULY, 1963.
0047 C
0048 C***REVISION HISTORY:
0049 C
0050 C   Modified by Keith Miller, 6/18/84.
0051 C
0052 C***LOCAL VARIABLES:
0053 C
0054     CHARACTER*1 CBUF(256) ! character buffer
0055     INTEGER     IBUF(128) ! the character buffer overlaid as integers
0056     INTEGER     IZERO     ! constant value 0 for making a 0 byte (ZERO)
0057     INTEGER     WORDS     ! counts number of words
0058     INTEGER     NUMCHR    ! counts number of bytes
0059     CHARACTER*1 BLANK, DOLLAR ! special ASCII characters
0060     CHARACTER*1 ZERO      ! zero is 00000000 binary.
0061     EQUIVALENCE (CBUF,IBUF), (ZERO, IZERO)
0062 C
0063 C***INITIALIZATIONS:
0064 C
0065     DATA        BLANK/' '/, DOLLAR/'$'/, IZERO/0/
0066 C
0067     NUMCHR = INCNT ! protects the input parameter, since NUMCHR is
0068 C                 ! reassigned in the subroutine.
0069 C
0070 C***PROCESSING:
0071 C
0072     WORDS = (NUMCHR+1) / 2
0073     DO 10 INDEX = 1, WORDS
0074         IBUF(INDEX) = INBUF(INDEX)
0075     10 CONTINUE
0076 C
0077     IF (CBUF(NUMCHR) .EQ. BLANK) GOTO 30
0078 C... ELSE...
0079         NUMCHR = NUMCHR + 1
0080         CBUF(NUMCHR) = BLANK
0081 C
0082 C... ADD ENDING CHARACTERS TO COMMAND
0083 C
0084     30 CBUF(NUMCHR+1) = DOLLAR
0085         CBUF(NUMCHR+2) = ZERO
0086         CBUF(NUMCHR+3) = ZERO ! safety precaution
0087         NUMCHR = NUMCHR + 2
0088 C
0089 C... CHANGE BYTE COUNT TO WORD COUNT
0090 C

```

```

0091      WORDS = (NUMCHR+1) / 2 ! if N is even, intentional truncation
0092 C
0093 C Programming note:
0094 C The EXEC command parameters are discussed in the HP RTE-6/VM
0095 C Programmer's Reference Manual, 2-19 ff. The COMTAL parameters
0096 C are discussed in section 5.2.4 of the COMTAL User's Manual.
0097 C
0098 C The first parameter to EXEC identifies it as a write command.
0099 C The second parameter identifies the resident HP driver (36B)
0100 C and gives a code for the operation required by this call (500B).
0101 C The third parameter is the command string, and the fourth gives
0102 C the length in words of the buffer that is to be used. The fifth
0103 C parameter is a code for the COMTAL interface that directs the
0104 C command transfer.
0105 C
0106 C CALL EXEC(2, 36B + 500B, IBUF, WORDS, 24001B)
0107 C RETURN
0108 C
0109 C END

```

&COUNT T=00004 IS ON CR00021 USING 00012 BLKS R=0000

```
0001 CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
0002 SUBROUTINE COUNT(COUNTS, IMAGE)
0003 CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
0004 C
0005 C***PARAMETER DECLARATIONS:
0006 C
0007     INTEGER*4 COUNTS(256) ! holds the counts for pixel values 0-255
0008     INTEGER    IMAGE      ! COMTAL image number of which COUNTgram is
0009 C                        ! to be taken
0010 C
0011 C***INTRODUCTION:
0012 C
0013 C The subroutine COUNT examines each pixel value in the COMTAL
0014 C image associated with the number IMAGE, and compiles a count of
0015 C how many pixels hold the values 0-255. These 256 counts are
0016 C returned in the INTEGER*4 array COUNTS.
0017 C
0018 C***LANGUAGE:
0019 C
0020 C FORTRAN 77, the HP-1000 version for RTE-6/VM.
0021 C
0022 C***LIMITATIONS:
0023 C
0024 C Note that COUNTS is an INTEGER*4 array. The 32,767 limit for INTEGER*2
0025 C is not sufficient, since there are over 250,000 pixels in a 512 X 512
0026 C COMTAL image. COUNT does no scaling or graphing.
0027 C
0028 C***SUBPROGRAMS CALLED:
0029 C
0030 C name      source  load  remarks
0031 C -----  -
0032 C RDIL2    &RDIL2  %RDIL2 reads a horizontal line of pixels from a
0033 C          COMTAL image
0034 C RANGE    &RANGE  %RANGE logical function that determines if its 1st
0035 C          parameter is within the 2nd and 3rd parameters.
0036 C
0037 C***WRITTEN BY:
0038 C
0039 C The code on which this subprogram is based was written by
0040 C NETTIE D. FAULCON, July, 1983. This subprogram was written by
0041 C KEITH MILLER, July, 1984, with the support of a NASA-ASEE
0042 C summer fellowship.
0043 C
0044 C***REVISION HISTORY:
```

```

0045 C
0046 C
0047 C***LOCAL VARIABLES:
0048 C
0049 LOGICAL RANGE ! logical function that determines if its 1st
0050 C ! parameter is within the 2nd and 3rd parameters.
0051 INTEGER INDEX ! loop index variable
0052 INTEGER LBUF(512) ! Line BUfFer holds a row of pixels.
0053 INTEGER WHICH ! changes from pixel value to count array index
0054 INTEGER IMLO, IMHI ! limits on COMTAL image numbers
0055 INTEGER ROW, COL ! loop indices
0056 INTEGER TERM ! logical unit for terminal output
0057 C
0058 C***INITIALIZATIONS:
0059 C
0060 DATA IMLO/1/, IMHI/4/
0061 DATA TERM/1/
0062 C
0063 C***PROCESSING
0064 C
0065 IF (.NOT.(RANGE(IMAGE, IMLO, IMHI))) GOTO 8001 ! error return
0066 C
0067 C initialize COUNTS to 0
0068 C
0069 DO 1000 INDEX = 1,256
0070 COUNTS(INDEX) = 0
0071 1000 CONTINUE
0072 C
0073 C collect counts
0074 C
0075 DO 3000 ROW = 0, 511
0076 CALL RDIL2(LBUF, IMAGE, ROW)
0077 DO 2000 COL = 1, 512
0078 WHICH = LBUF(COL) + 1 ! "+1" required because pixel values are
0079 C ! 0-255, COUNTS array is indexed 1-256.
0080 COUNTS(WHICH) = COUNTS(WHICH) + 1
0081 2000 CONTINUE
0082 3000 CONTINUE
0083 C
0084 RETURN
0085 C
0086 C***ERROR RETURN
0087 C
0088 8001 WRITE(TERM, 8003)IMAGE, IMLO, IMHI
0089 8003 FORMAT(' IMAGE NUMBER.', 14, ', OUT OF RANGE:', 214)
0090 WRITE(TERM, 8901)

```

```
0091 . 8901 FORMAT(' SUBROUTINE COUNT FAILS. COUNTS ARRAY NOT CHANGED.')
```

```
0092     RETURN
```

```
0093     END
```



&DELAY T=00004 IS ON CR00021 USING 00005 BLKS R=0000

```
0001 CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
0002 SUBROUTINE DELAY(SECOND)
0003 CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
0004 C
0005 C***PARAMETER DECLARATIONS:
0006 C
0007 INTEGER SECOND ! the number of seconds to delay, >= 0.
0008 C
0009 C***INTRODUCTION:
0010 C
0011 C This routine, DELAY, makes the HP busy wait for at least the
0012 C seconds given in the input parameters.
0013 C
0014 C***LANGUAGE:
0015 C
0016 C FORTRAN 77, the HP-1000 version for RTE-6/VM.
0017 C
0018 C***LIMITATIONS:
0019 C
0020 C There is no claim that the timing here is exact. However, the HP
0021 C busy waits for AT LEAST the time required by the input parameter.
0022 C The HP EXEC for time request gives tens of milliseconds, but this
0023 C procedure uses the simpler seconds measure.
0024 C
0025 C The procedure will not work properly when the Julian calendar spins
0026 C over to 0,0,0,0,0,0.
0027 C
0028 C***SUBPROGRAMS CALLED:
0029 C
0030 C None.
0031 C
0032 C***WRITTEN BY:
0033 C
0034 C The code on which this subprogram is based was written by
0035 C NETTIE D. FAULCON, July, 1983. This subprogram was written by
0036 C KEITH MILLER, July, 1984, with the support of a NASA-ASEE
0037 C summer fellowship.
0038 C
0039 C***REVISION HISTORY:
0040 C
0041 C
0042 C***LOCAL VARIABLES:
0043 C
0044 INTEGER INTIME(6) ! the milliseconds, seconds, minutes, hours,
```

```

0045
0046     INTEGER*4     INCNT      ! and Julian day when procedure entered.
0047
0048     INTEGER       NOW(6)     ! the INTIME in units of tens of milliseconds
0049
0050     INTEGER*4     NOWCNT     ! from 0,0,0,0,0.
0051
0052     INTEGER*4     NOWCNT     ! the milliseconds, seconds, minutes, hours,
0053
0054     INTEGER       TIMREQ     ! and Julian day of the latest EXEC call that
0055     INTEGER       TERM      ! NOW in units of tens of milliseconds from
0056
0057     C               ! 0,0,0,0,0.
0058     C               ! determines the time.
0059     C               ! the EXEC number for a time request.
0060     C               ! logical unit for terminal output.
0061     C
0062     C***INITIALIZATIONS:
0063     C
0064     DATA TIMREQ/11/
0065     DATA TERM/1/
0066     C
0067     C***PROCESSING:
0068     C
0069     IF (SECOND .LE. 0) GOTO 8001 ! error return
0070     C
0071     CALL EXEC(TIMREQ, INTIME)
0072     INCNT = INTIME(1) + 100*INTIME(2) + 6000*INTIME(3)
0073     1      + 360000*INTIME(4) + 360000*365*INTIME(5)
0074     C
0075     1000 CALL EXEC(TIMREQ, NOW)
0076     NOWCNT = NOW(1) + 100*NOW(2) + 6000*NOW(3)
0077     1      + 360000*NOW(4) + 360000*365*NOW(5)
0078     IF ((NOWCNT-INCNT) .LT. (SECOND*100)) GOTO 1000
0079     C
0080     RETURN
0081     C
0082     C***ERROR RETURN:
0083     C
0084     8001 WRITE(TERM,8003) SECOND
0085     8003 FORMAT(' THE SECONDS COUNT,',15,' IS <= 0. ')
0086     8900 WRITE(TERM,8901)
0087     8901 FORMAT(' DELAY FAILS. NO TIMED DELAY OCCURS. ')
0088     RETURN
0089     END

```

&DIGIT T=00004 IS ON CR00021 USING 00012 BLKS R=0000

```
0001 CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
0002 CHARACTER FUNCTION DIGIT( INTIN )
0003 CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
0004 C
0005 C***PARAMETER DECLARATIONS:
0006 C
0007 INTEGER INTIN ! INTEger INput parameter, converted to a digit.
0008 C
0009 C***INTRODUCTION:
0010 C
0011 C DIGITS converts an integer between 0 and 9 into a single
0012 C character digit. If the INTIN parameter is out of range,
0013 C an error message is printed at the terminal and DIGIT is
0014 C set to a blank.
0015 C
0016 C***LANGUAGE:
0017 C
0018 C FORTRAN 77, the HP-1000 version for RTE-6/VM.
0019 C
0020 C***LIMITATIONS:
0021 C
0022 C The INTIN parameter must be between 0 and 9.
0023 C
0024 C***SUBPROGRAMS CALLED:
0025 C
0026 C name source load remarks
0027 C -----
0028 C RANGE ✓ &RANGE %RANGE logical function determines if the first parameter
0029 C lies within (inclusive) the next two parameters.
0030 C
0031 C***WRITTEN BY:
0032 C
0033 C KEITH MILLER, ASEE NASA-Langley fellow, Summer, 1984.
0034 C
0035 C***REVISION HISTORY:
0036 C
0037 C
0038 C***LOCAL VARIABLES:
0039 C
0040 CHARACTER*1 DGTARA(10)! DiGiT ARrAy holds the digits '0'-'9'
0041 LOGICAL RANGE ! function that determines if its first parameter
0042 C ! is between (inclusive) its last two parameters.
0043 INTEGER TERM ! logical unit number for terminal output.
0044 C
```

```

0045 C**INITIALIZATIONS:
0046 C
0047 DATA DGTARA/'0','1','2','3','4','5','6','7','8','9'/
0048 DATA TERM/1/
0049 C
0050 C**PROCESSING
0051 C
0052 IF (.NOT.(RANGE(INTIN, 0, 9))) GOTO 8001 ! error return
0053 C ELSE... convert to digit and return.
0054 DIGIT = DGTARA(INTIN + 1)
0055 RETURN
0056 C
0057 C**ERROR RETURN
0058 C
0059 8001 WRITE(TERM, 8003) INTIN
0060 8003 FORMAT('THE INPUT TO DIGIT.', I4,
0061 1 ' ', IS NOT A SINGLE DIGIT. DIGIT RETURNS A BLANK.')
0062 DIGIT = ' '
0063 RETURN
0064 END
0065
0066

```

&DSPBW T=00004 IS ON CR0021 USING 00002 BLKS R=0000

```
0001 CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
0002 SUBROUTINE DSPBW(IMAGE)
0003 CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
0004 C
0005 C***PARAMETER DECLARATIONS:
0006 C
0007 INTEGER IMAGE ! a number 1-4 designating a COMTAL image
0008 C
0009 C***INTRODUCTION:
0010 C
0011 C "DiSPlay Black and White" is a subroutine that allows the caller
0012 C to send a display command to the COMTAL from an HP program.
0013 C The call can turn on one black and white image, number 1, 2, 3,
0014 C or 4. Any previous pseudocolor or function memory commands are
0015 C nullified by a DSPBW call.
0016 C
0017 C The resident driver DVR41 is called via an EXEC to accomplish the
0018 C display.
0019 C
0020 C***LIMITATIONS:
0021 C
0022 C This subroutine does NOT display truecolor images. That is
0023 C accomplished using the subroutine DSPCL.
0024 C
0025 C***SUBPROGRAMS CALLED:
0026 C
0027 C name source load remarks
0028 C -----
0029 C RANGE &RANGE %RANGE logical function that determines if the
0030 C first parameter is within the bounds defined
0031 C by the second and third parameter (inclusive).
0032 C
0033 C***WRITTEN BY:
0034 C
0035 C The code on which this subprogram is based was written by
0036 C NETTIE D. FAULCON, July, 1983. This modification is by
0037 C KEITH MILLER, June, 1984.
0038 C
0039 C***REVISION HISTORY:
0040 C
0041 C
0042 C***LOCAL VARIABLES:
0043 C
0044 LOGICAL RANGE ! a function for determining if an integer is
```

```

0045 C      ! within a certain range
0046     INTEGER TERM ! the logical unit number for the terminal
0047     INTEGER IDUMMY! fills the place of an unused EXEC parameter
0048 C
0049 C***INITIALIZATIONS:
0050 C
0051     DATA TERM/1/
0052 C
0053 C***PROCESSING
0054 C
0055     IF (RANGE(IMAGE,1,4)) GOTO 2000 ! legal image number
0056 C     ELSE... illegal image number
0057         WRITE(TERM, 1001) IMAGE
0058 1001     FORMAT( ' The image number ', I3, ' is out of range.')
0059         WRITE(TERM, 1002)
0060 1002     FORMAT( ' DSPBW fails. No action taken on command.' )
0061         RETURN
0062 C
0063 2000 CONTINUE ! send a display command to COMTAL
0064 C
0065 C     In the following call, the first parameter indicates a write
0066 C     operation. The second parameter is a combination of two codes:
0067 C     000B + 36B. 36B indicates the proper resident driver, and 000B
0068 C     informs the driver (DRV41) that we require a display operation.
0069 C     The third and fourth parameters are ignored. The final parameter
0070 C     indicates the image to be displayed. The subtraction in that
0071 C     final parameter is necessary because the COMTAL images are numbered
0072 C     0 to 3; the multiplication is necessary to push the image number
0073 C     into the proper bits in the command word sent to the COMTAL.
0074 C
0075     CALL EXEC(2, 000B + 36B, IDUMMY, 0, (IMAGE-1) * 2)
0076     RETURN
0077     END
0078
0079

```

&DSPCL T=0004 IS ON CR0021 USING 0006 BLKS R=0000

```
0001 CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
0002 SUBROUTINE DSPCL(RED, GREEN, BLUE, TCLR)
0003 CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
0004 C
0005 C***INPUT PARAMETERS:
0006 C
0007 INTEGER RED, GREEN, BLUE ! COMTAL image numbers for the 3 color
0008 C ! components of the truecolor image to
0009 C ! be displayed.
0010 INTEGER TCLR ! COMTAL image number for the truecolor
0011 C ! image formed from RED, GREEN, and BLUE.
0012 C
0013 C***INTRODUCTION:
0014 C
0015 C "DiSPlay CoLor" commands the COMTAL to display one
0016 C RGB true color image. Any previous function or
0017 C psuedocolor memory commands are nullified. See the
0018 C subroutine DSPBW to display black and white images.
0019 C
0020 C The color display is accomplished via the CMMND subroutine,
0021 C which allows COMTAL commands to be sent to the COMTAL as if
0022 C they were typed on the COMTAL keyboard.
0023 C
0024 C***LANGUAGE:
0025 C
0026 C FORTRAN 77, the HP-100 version for RTE-6/VM.
0027 C
0028 C***LIMITATIONS:
0029 C
0030 C The three parameters RED, GREEN and BLUE must be distinct and
0031 C within the IMLO, IMHI range. TCLR must be within the TRLO, TRHI
0032 C range (which is mutually exclusive with IMLO-IMHI). The limits on
0033 C truecolor numbers are arbitrary. However, this subroutine enforces
0034 C the arbitrary limits. The HP image file must contain all three
0035 C monochrome images in the order RED, GREEN, and BLUE.
0036 C
0037 C***SUBPROGRAMS CALLED:
0038 C
0039 C name source load remarks
0040 C -----
0041 C RANGE &RANGE %RANGE logical function that determines if the 1st
0042 C argument is within the 2nd and 3rd inclusive.
0043 C CMMND &CMMND %CMMND transfers a command string to the COMTAL, which
0044 C accepts it almost as a keyboard command.
```

```

0045 C DIGIT &DIGIT %DIGIT character*1 function that returns a single digit
0046 C on legal integer inputs 0-9.
0047 C
0048 C***WRITTEN BY:
0049 C
0050 C The code on which this subprogram is based was written by
0051 C NETTIE D. FAULCON, July, 1983. This subprogram was written by
0052 C KEITH MILLER, July, 1984, with the support of a NASA-ASEE
0053 C summer fellowship.
0054 C
0055 C***REVISION HISTORY:
0056 C
0057 C
0058 C***LOCAL VARIABLES:
0059 C
0060 C INTEGER IBUF(256) ! buffers COMTAL command
0061 C LOGICAL RANGE ! logical function that determines if 1st parameter
0062 C ! is between 2nd and 3rd, inclusive.
0063 C INTEGER IMLO, IMHI ! limits on COMTAL b&w image numbers.
0064 C INTEGER TRLO, TRHI ! limits on COMTAL truecolor image numbers.
0065 C INTEGER TERM ! logical unit for terminal output
0066 C
0067 C CHARACTER*255 CBUF ! character overlay for sending COMTAL commands
0068 C EQUIVALENCE (CBUF,IBUF)
0069 C CHARACTER*1 DIGIT ! function that returns '0','1',..., or '9'.
0070 C ! according to 0,1,...., or 9 integer input.
0071 C
0072 C***INITIALIZATIONS:
0073 C
0074 C DATA IMLO/1/, IMHI/4/
0075 C DATA TRLO/5/, TRHI/9/
0076 C DATA TERM/1/
0077 C
0078 C***PROCESSING
0079 C
0080 C IF (.NOT.(RANGE(RED ,IMLO,IMHI))) GOTO 8001 ! error return
0081 C IF (.NOT.(RANGE(GREEN ,IMLO,IMHI))) GOTO 8101 ! error return
0082 C IF (.NOT.(RANGE(BLUE ,IMLO,IMHI))) GOTO 8201 ! error return
0083 C
0084 C IF ((RED .EQ. GREEN) .OR. (GREEN .EQ. BLUE)
0085 C 1 .OR. (RED .EQ. BLUE)) GOTO 8301 ! error return
0086 C
0087 C IF (.NOT.(RANGE(TCLR, TRLO,TRHI))) GOTO 8401 ! error return
0088 C
0089 C DISPLAY THE COLOR IMAGE
0090 C The following commands are abbreviations of the following

```



```

0091 C      COMTAL commands, where #R, #G, #B, and #C stands for the single
0092 C      character digits corresponding to RED, GREEN, BLUE, and TCLR:
0093 C      UNassign Image #C
0094 C      ASsign Truecolor #C red #R green #G blue #B
0095 C      Display Image #C
0096 C
0097         CBUF = 'UN I '//DIGIT(TCLR)
0098         CALL CMMND(IBUF,6)
0099         CBUF = 'AS T '//DIGIT(TCLR)//' '//DIGIT(RED)//' '//
0100 1         DIGIT(GREEN)//' '//DIGIT(BLUE)
0101         CALL CMMND(IBUF,12)
0102         CBUF = 'D I '//DIGIT(TCLR)
0103         CALL CMMND(IBUF,5)
0104         RETURN
0105 C
0106 C***ERROR RETURNS
0107 C
0108 8001 WRITE(TERM, 8003) RED,  IMLO, IMHI
0109 8003 FORMAT(' RED IMAGE NUMBER, ',I3,' , IS OUT OF RANGE:',.214,')
0110      GOTO 8900
0111 C
0112 8101 WRITE(TERM, 8103) GREEN, IMLO, IMHI
0113 8103 FORMAT(' GREEN IMAGE NUMBER, ',I3,' , IS OUT OF RANGE:',.214,')
0114      GOTO 8900
0115 C
0116 8201 WRITE(TERM, 8203) BLUE,  IMLO, IMHI
0117 8203 FORMAT(' BLUE IMAGE NUMBER, ',I3,' , IS OUT OF RANGE:',.214,')
0118      GOTO 8900
0119 C
0120 8301 WRITE(TERM, 8303) RED, GREEN, BLUE
0121 8303 FORMAT(' 3 MONOCHROME IMAGES MUST BE DISTINCT. YOURS:',.314)
0122      GOTO 8900
0123 C
0124 8401 WRITE(TERM, 8403) TCLR, CLLO, CLHI
0125 8403 FORMAT(' YOUR TRUECOLOR IMAGE, ',I4,' , IS OUT OF RANGE:',.214)
0126      GOTO 8900
0127 C
0128 8900 WRITE(TERM, 8901)
0129 8901 FORMAT(' DSPCL FAILS. NO DISPLAY TAKES PLACE.')
0130      RETURN
0131      END

```

&DSPGR T=00004 IS ON CR00021 USING 00012 BLKS R=0000

```
0001 CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
0002 SUBROUTINE DSPGR(GRNUM)
0003 CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
0004 C
0005 C***PARAMETER DECLARATIONS:
0006 C
0007 INTEGER GRNUM ! a number 1-4 designating a COMTAL graphics plane.
0008 C
0009 C***INTRODUCTION:
0010 C
0011 C "DiSPlay GRaphics" is a subroutine that allows the caller
0012 C to send a display command to the COMTAL from an HP program.
0013 C The call can turn on one graphics plane, number 1, 2, 3,
0014 C or 4.
0015 C
0016 C***LIMITATIONS:
0017 C
0018 C This subroutine does not turn off previous graphics planes.
0019 C That is accomplished using the subroutine WIPGR.
0020 C It is OK to call DSPGR repeatedly without an intervening WIPGR.
0021 C The extra calls have no effect, but they don't hang up the COMTAL.
0022 C
0023 C***SUBPROGRAMS CALLED:
0024 C
0025 C name source load remarks
0026 C -----
0027 C RANGE &RANGE %RANGE logical function that determines if the
0028 C first parameter is within the bounds defined
0029 C by the second and third parameter (inclusive).
0030 C CMMN2 &CMMN2 %CMMN2 sends a constant string to the COMTAL as if
0031 C the string were typed on the COMTAL keyboard.
0032 C DIGIT &DIGIT %DIGIT character*1 function that returns '0'..'9'
0033 C according to integer input 0..9.
0034 C
0035 C***WRITTEN BY:
0036 C
0037 C The code on which this subprogram is based was written by
0038 C NETTIE D. FAULCON, July, 1983. This modification is by
0039 C KEITH MILLER, June, 1984.
0040 C
0041 C***REVISION HISTORY:
0042 C
0043 C
0044 C***LOCAL VARIABLES:
```

```

0045 C
0046 LOGICAL RANGE ! a function for determining if an integer is
0047 C ! within a certain range.
0048 INTEGER TERM ! the logical unit number for the terminal.
0049 INTEGER IDUMMY ! fills the place of an unused EXEC parameter.
0050 INTEGER GRLO,GRHI ! limits on a graphics plane number.
0051 CHARACTER*1 DIGIT ! function that returns '0'..'9' for input
0052 C ! integers 0..9.
0053 C
0054 C***INITIALIZATIONS:
0055 C
0056 DATA TERM/1/
0057 DATA GRLO/1/,GRHI/4/
0058 C
0059 C***PROCESSING
0060 C
0061 IF (.NOT.(RANGE(GRNUM,GRLO,GRHI))) GOTO 8001 ! error return
0062 C
0063 C "ADD GRaphics #GRNUM", where #GRNUM stands for the digit
0064 C corresponding to GRNUM value.
0065 C
0066 CALL CMMN2('ADD GR '//DIGIT(GRNUM))
0067 RETURN
0068 C
0069 C***ERROR RETURN:
0070 C
0071 8001 WRITE(TERM, 8003) GRNUM, GRLO, GRHI
0072 8003 FORMAT( ' THE GRAPHICS NUMBER,',I3,', OUT OF RANGE:',I2)
0073 C
0074 8900 WRITE(TERM, 8901)
0075 8901 FORMAT( ' DSPGR fails. No action taken on command.' )
0076 RETURN
0077 END

```

&DSPVD T=00004 IS ON CR00021 USING 00024 BLKS R=0000

```
0001 CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
0002 SUBROUTINE DSPVD
0003 CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
0004 C
0005 C***INPUT PARAMETERS:
0006 C
0007 C None.
0008 C
0009 C***INTRODUCTION:
0010 C
0011 C "DiSPlay ViDeo" commands the COMTAL to display the video image
0012 C which, by arbitrary convention, is always assigned to COMTAL
0013 C image #5. The video must be set to image 5 previous to this
0014 C call.
0015 C
0016 C***LANGUAGE:
0017 C
0018 C FORTRAN 77, the HP-100 version for RTE-6/VM.
0019 C
0020 C***LIMITATIONS:
0021 C
0022 C The COMTAL image #5 must have been set to video before DSPVD is called.
0023 C
0024 C***SUBPROGRAMS CALLED:
0025 C
0026 C name source load remarks
0027 C -----
0028 C CMMND &CMMND %CMMND transfers a command string to the COMTAL, which
0029 C accepts it as a keyboard command.
0030 C DIGIT &DIGIT %DIGIT character%i function that returns '0'-'9'
0031 C according to integer input 0-9.
0032 C
0033 C***WRITTEN BY:
0034 C
0035 C The code on which this subprogram is based was written by
0036 C NETTIE D. FAULCON, July, 1983. This subprogram was written by
0037 C KEITH MILLER, July, 1984, with the support of a NASA-ASEE
0038 C summer fellowship.
0039 C
0040 C***REVISION HISTORY:
0041 C
0042 C
0043 C***LOCAL VARIABLES:
0044 C
```

```

0045      INTEGER      IBUF(256)! buffers COMTAL command
0046      CHARACTER*255 CBUF      ! character overlay for sending COMTAL commands
0047      EQUIVALENCE   (CBUF,IBUF)
0048      INTEGER      TVIMAG      ! COMTAL image # for video camera
0049      CHARACTER*1   DIGIT      ! function that returns '0'-'9' according
0050 C                                          ! to integer input 0-9.
0051 C
0052 C***INITIALIZATIONS:
0053 C
0054      DATA          TVIMAG/5/      ! arbitrary choice.
0055 C
0056 C***PROCESSING
0057 C
0058 C      DISPLAY THE VIDEO IMAGE
0059 C
0060      CBUF = 'DISPLAY IMAGE '//DIGIT(TVIMAG)//' '
0061      CALL CMMND(CBUF,16)
0062 C
0063      RETURN
0064      END

```

&HILO T=00004 IS ON CR00021 USING 00004 BLKS R=0000

```
0001 CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
0002 SUBROUTINE HILO(HI, LO, IMAGE)
0003 CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
0004 C
0005 C***PARAMETER DECLARATIONS:
0006 C
0007 C     INTEGER HI, LO ! output parameters, the high and low pixel values
0008 C                   ! found in the designated image.
0009 C     INTEGER IMAGE ! the COMTAL image number of the image that is to
0010 C                   ! be searched for its high and low pixel values.
0011 C
0012 C***INTRODUCTION:
0013 C
0014 C     The subroutine "High and Low values #2" reads through an image and
0015 C     determines the highest and lowest pixel values, returning the
0016 C     values found. HILO scans the entire image.
0017 C
0018 C***LANGUAGE:
0019 C
0020 C     FORTRAN 77, the HP-1000 version for RTE-6/VM.
0021 C
0022 C***LIMITATIONS:
0023 C
0024 C     As noted above, this subroutine passes through an entire image, pixel
0025 C     by pixel. In some applications you may want to combine other processing
0026 C     during that pass, but this subroutine won't let you do that.
0027 C
0028 C***SUBPROGRAMS CALLED:
0029 C
0030 C     name      source  load  remarks.
0031 C     -----  -
0032 C     RANGE     &RANGE  %RANGE logical function that determines if 1st argument
0033 C                   is within 2nd & 3rd inclusive.
0034 C     RDIL2     &RDIL2  %RDIL2 reads a horizontal line of pixel values into
0035 C                   a 512 integer array.
0036 C
0037 C***WRITTEN BY:
0038 C
0039 C     The code on which this subprogram is based was written by
0040 C     NETTIE D. FAULCON, July, 1983. This subprogram was written by
0041 C     KEITH MILLER, July, 1984, with the support of a NASA-ASEE
0042 C     summer fellowship.
0043 C
0044 C***REVISION HISTORY:
```

```

0045 C
0046 C
0047 C***LOCAL VARIABLES:
0048 C
0049     INTEGER  IBUF(512) ! buffer for a horizontal row of COMTAL pixel values
0050     INTEGER  PXLO, PXHI! pixel value limits (for 8 bits, 0-255)
0051     INTEGER  ROW, COL  ! indexes into the COMTAL image
0052     INTEGER  LNLO, LNHI! limits on COMTAL image line numbers
0053     INTEGER  ARALO, ARAHI! limits on buffer array dimension
0054     INTEGER  IMLO, IMHI ! limits on COMTAL image numbers.
0055     LOGICAL  RANGE      ! function that determines if 1st argument is within
0056 C                          ! 2nd & 3rd arguments inclusive.
0057 C
0058 C***INITIALIZATIONS
0059 C
0060     DATA IMLO/1/, IMHI/4/
0061     DATA PXLO/0/, PXHI/255/
0062     DATA LNLO/0/, LNHI/511/
0063     DATA ARALO/1/, ARAHI/512/
0064
0065 C
0066 C***PROCESSING
0067 C
0068     IF (.NOT.(RANGE(IMAGE, IMLO, IMHI))) GOTO 8001 ! error return
0069 C
0070     HI = PXLO ! artificially low
0071     LO = PXHI ! artificially high
0072 C
0073     DO 2000 ROW = LNLO, LNHI
0074         CALL RDIL2(IBUF, IMAGE, ROW)
0075         DO 1000 COL = ARALO, ARAHI
0076             IF (IBUF(COL) .GT. HI) HI = IBUF(COL)
0077             IF (IBUF(COL) .LT. LO) LO = IBUF(COL)
0078     1000     CONTINUE
0079     2000 CONTINUE
0080 C
0081     RETURN
0082 C
0083     8001 WRITE(TERM, 8003) IMAGE , IMLO, IMHI
0084     8003 FORMAT(' IMAGE NUMBER, ', I5, ', IS OUT OF RANGE: ', I215, '.')
0085     GOTO 8900
0086 C
0087     8900 WRITE(TERM, 8901)
0088     8901 FORMAT(' HILO FAILS. HI AND LO PARAMETERS UNCHANGED.')
0089     RETURN
0090     END

```

&HISTO T=00004 IS ON CR00021 USING 00005 BLKS R=0000

```
0001 CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
0002 SUBROUTINE HISTO(IMAGE)
0003 CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
0004 C
0005 C***PARAMETER DECLARATIONS:
0006 C
0007 INTEGER IMAGE ! the COMTAL image number to take the histogram of.
0008 C
0009 C***INTRODUCTION:
0010 C
0011 C The subroutine HISTOgram uses the COMTAL "function memory" (a look-up
0012 C table) to construct a histogram of the given image, which is held
0013 C IN A SCALED VERSION in the function memory associated with IMAGE.
0014 C This histogram is displayed by HISTO.
0015 C
0016 C***LANGUAGE:
0017 C
0018 C FORTRAN 77, the HP-1000 version for RTE-6/VM.
0019 C
0020 C***LIMITATIONS:
0021 C
0022 C The function memory associated with image number IMAGE is destroyed.
0023 C IMAGE must be within the boundaries for COMTAL image memories, or
0024 C an error message is given and HTABLE is unchanged.
0025 C
0026 C Notice that the histogram is scaled so that the largest value reaches
0027 C to the top of the screen when displayed. Thus, no absolute counts can
0028 C be easily deduced from the function memory values. See the subroutine
0029 C COUNT if absolute pixel value counts are desired.
0030 C
0031 C HISTO puts the HP into a busy wait while the COMTAL determines the
0032 C histogram. Otherwise, the COMTAL ignores subsequent CMMN2 commands.
0033 C
0034 C***SUBPROGRAMS CALLED:
0035 C
0036 C name source load remarks
0037 C -----
0038 C CMMN2 &CMMN2 %CMMN2 sends a constant string to the COMTAL, which
0039 C treats it as a command typed on the COMTAL.
0040 C DELAY &DELAY %DELAY puts the HP into a busy wait for at least the
0041 C given number of seconds.
0042 C DIGIT &DIGIT %DIGIT character*I function that returns '0'-'9'
0043 C according to integer input 0-9.
0044 C RANGE &RANGE %RANGE logical function that determines if its 1st
```



```

0045 C          argument lies within its 2nd and 3rd, inclusive.
0046 C
0047 C***WRITTEN BY:
0048 C
0049 C   The code on which this subprogram is based was written by
0050 C   NETTIE D. FAULCON, July, 1983. This subprogram was written by
0051 C   KEITH MILLER, July, 1984, with the support of a NASA-ASEE
0052 C   summer fellowship.
0053 C
0054 C***REVISION HISTORY:
0055 C
0056 C
0057 C***LOCAL VARIABLES:
0058 C
0059 C          CHARACTER*1  IMCHAR  ! the single digit character corresponding to
0060 C                               ! input parameter IMAGE.
0061 C          CHARACTER*1  DIGIT   ! function that returns '0'-'9' for integer
0062 C                               ! input 0-9.
0063 C          LOGICAL      RANGE   ! function that determines if 1st argument is
0064 C                               ! within 2nd and 3rd, inclusive.
0065 C          INTEGER      TERM    ! logical unit for terminal output.
0066 C          INTEGER      IMLO,IMHI ! limits for IMAGE number.
0067 C
0068 C***INITIALIZATIONS:
0069 C
0070 C          DATA  IMLO/1/,IMHI/4/
0071 C          DATA  TERM/1/
0072 C
0073 C***PROCESSING:
0074 C
0075 C          IF (.NOT.(RANGE(IMAGE,IMLO,IMHI))) GOTO 8001 ! error return
0076 C
0077 C          IMCHAR = DIGIT(IMAGE) ! delay initialization until after IMAGE
0078 C                               ! has been found to be within its limits.
0079 C
0080 C          The following COMTAL command expands to:
0081 C          Function memory #I = Histogram of image
0082 C          where #I is the single digit associated with IMAGE.
0083 C          Image #I is automatically used for the histogram.
0084 C
0085 C          CALL CMMN2('F '//IMCHAR//' H')
0086 C
0087 C          While the COMTAL compiles the histogram, it ignores all HP
0088 C          commands; thus, we pause until the histogram is found.
0089 C
0090 C          CALL DELAY(15) ! 14 seconds experimentally determined as the

```

```

0091 C           ! time it takes to compile a histogram.
0092 CALL CMMN2('D F '//IMCHAR)! "Display Function memory #I"
0093 RETURN
0094 C
0095 C***ERROR RETURNS:
0096 C
0097 C
0098 8001 WRITE(TERM,8003)IMAGE ,IMLO,IMHI
0099 8003 FORMAT(' THE IMAGE NUMBER,',I5,', IS OUT OF RANGE:',I215,',')
0100 8900 WRITE(TERM, 8901)
0101 8901 FORMAT(' HISTO FAILS. NO HISTOGRAM TABLE ASSIGNMENT.')
```

RETURN

END

&ICOPY T=00004 IS ON CR00021 USING 00012 BLKS R=0000

```
0001 CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
0002 SUBROUTINE ICOPY(OUTIMG, INIMG)
0003 CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
0004 C
0005 C***INPUT PARAMETERS:
0006 C
0007 C      INTEGER OUTIMG ! COMTAL image number for the destination
0008 C      INTEGER INIMG ! COMTAL image number for the source
0009 C
0010 C***INTRODUCTION:
0011 C
0012 C      "Image COPY" commands the COMTAL to copy one black and
0013 C      white image into another. The OUTIMG destination image
0014 C      is, of course, wiped out by this exchange. The input
0015 C      image for ICOPY and the output image must be associated with
0016 C      a COMTAL image memory plane (1-4 currently).
0017 C      The companion subroutine ICOPY2 requires that the OUTIMG
0018 C      be a COMTAL image memory plane, but the INIMG can be in the range
0019 C      1-9.
0020 C
0021 C***LANGUAGE:
0022 C
0023 C      FORTRAN 77, the HP-100 version for RTE-6/VM.
0024 C
0025 C***LIMITATIONS:
0026 C
0027 C      ICOPY only works for the grey level images of COMTAL, not the
0028 C      truecolor images.
0029 C
0030 C***SUBPROGRAMS CALLED:
0031 C
0032 C      name      source load      remarks
0033 C      -----
0034 C      CMMND      &CMMND %CMMND  this subroutine takes an INTEGER array which
0035 C      contains a COMTAL command string, and transfers
0036 C      the command to COMTAL. The second parameter
0037 C      gives the character count of the command string.
0038 C      RANGE      &RANGE %RANGE  this logical function determines if its first
0039 C      argument is within the bounds formed by its
0040 C      last 2 arguments, inclusive.
0041 C      DIGIT      &DIGIT %DIGIT  character*1 function that returns a single digit
0042 C      '0'-'9' according to integer input 0-9.
0043 C
0044 C***WRITTEN BY:
```

```

0045 C
0046 C   The code on which this subroutine is based was written by
0047 C   NETTIE D. FAULCON in July, 1983. This modification is by
0048 C   KEITH MILLER           June, 1984.
0049 C
0050 C**REVISION HISTORY:
0051 C
0052 C
0053 C**LOCAL VARIABLES:
0054 C
0055 C       LOGICAL      RANGE      ! function determines if 1st argument is
0056 C                               ! within 2nd and 3rd argument inclusive.
0057 C       CHARACTER*1  DIGIT      ! function returns '0'-'9' according to
0058 C                               ! integer input 0-9.
0059 C       CHARACTER*255 CCOMM     ! character buffer for building up a call
0060 C                               ! to the CMMND subroutine.
0061 C       INTEGER      IBUF (128)! integer overlay of CCOMM
0062 C       EQUIVALENCE (CCOMM, IBUF)
0063 C
0064 C       INTEGER      IMLO, IMHI  ! the range of legal COMTAL image numbers
0065 C       INTEGER      TERM        ! terminal logical unit
0066 C
0067 C**INITIALIZATION:
0068 C
0069 C       DATA      IMLO/1/, IMHI/4/
0070 C       DATA      TERM/1/
0071 C
0072 C**PROCESSING:
0073 C
0074 C       IF (.NOT.(RANGE(OUTIMG, IMLO, IMHI))) GOTO 8001 ! error return
0075 C       IF (.NOT.(RANGE(INIMG, IMLO, IMHI))) GOTO 8101 ! error return
0076 C
0077 C       Legal image numbers, so do the copy
0078 C
0079 C       2000 CCOMM = 'IMAGE '//DIGIT(OUTIMG)//' = IMAGE '//DIGIT(INIMG)
0080 C
0081 C       PROGRAMMING NOTE: see COMTAL USER'S GUIDE for
0082 C       further information on the command string abbreviated
0083 C       in the string above.
0084 C
0085 C       CALL CMMND( IBUF, 17 ) ! sends copy command to COMTAL
0086 C       RETURN
0087 C
0088 C**ERROR RETURNS:
0089 C
0090 C       8001 WRITE(TERM, 8003)OUTIMG, IMLO, IMHI

```

```
0091 8003 FORMAT(' YOUR OUTPUT PARAMETER,', IS,', IS OUT OF RANGE:',.214)
0092      GOTO 8900
0093 C
0094 8101 WRITE(TERM, 8103)INIMG, IMLO, IMHI
0095 8103 FORMAT(' YOUR INPUT PARAMETER,', IS,', IS OUT OF RANGE:',.214)
0096      GOTO 8900
0097 C
0098 8900 WRITE(TERM, 8901)
0099 8901 FORMAT(' ICOPY FAILS. NO COPYING TAKES PLACE.')
```

```
0100      RETURN
0101      END
```

&ICPY2 T=00004 IS ON CR00021 USING 00018 BLKS R=0000

```
0001 CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
0002 SUBROUTINE ICPY2(OUTIMG, INIMG)
0003 CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
0004 C
0005 C***INPUT PARAMETERS:
0006 C
0007 INTEGER OUTIMG ! COMTAL image number for the destination
0008 INTEGER INIMG ! COMTAL image number for the source
0009 C
0010 C***INTRODUCTION:
0011 C
0012 C "Image CoPY #2" commands the COMTAL to copy one black and
0013 C white image into another. The OUTIMG destination image
0014 C is, of course, wiped out by this exchange. The input
0015 C image for ICPY2 can be any single digit number; the output
0016 C image must be a COMTAL image memory plane (1-4 currently).
0017 C The companion subroutine ICOPY requires that both images
0018 C be COMTAL image memory planes.
0019 C
0020 C***LANGUAGE:
0021 C
0022 C FORTRAN 77, the HP-100 version for RTE-6/VM.
0023 C
0024 C***LIMITATIONS:
0025 C
0026 C ICOPY only works for the grey level images of COMTAL, not the
0027 C truecolor images.
0028 C
0029 C***SUBPROGRAMS CALLED:
0030 C
0031 C name source load remarks
0032 C -----
0033 C CMMND &CMMND %CMMND this subroutine takes an INTEGER array which
0034 C contains a COMTAL command string, and transfers
0035 C the command to COMTAL. The second parameter
0036 C gives the character count of the command string.
0037 C RANGE &RANGE %RANGE this logical function determines if its first
0038 C argument is within the bounds formed by its
0039 C last 2 arguments, inclusive.
0040 C DIGIT &DIGIT %DIGIT character*1 function that returns a single digit
0041 C '0'-'9' according to integer input 0-9.
0042 C
0043 C***WRITTEN BY:
0044 C
```

```

0045 C   The code on which this subroutine is based was written by
0046 C   NETTIE D. FAULCON in July, 1983.  This modification is by
0047 C   KEITH MILLER      , June, 1984.
0048 C
0049 C***REVISION HISTORY:
0050 C
0051 C
0052 C***LOCAL VARIABLES:
0053 C
0054 C       LOGICAL      RANGE      ! function determines if 1st argument is
0055 C                               ! within 2nd and 3rd argument inclusive.
0056 C       CHARACTER*1  DIGIT      ! function returns '0'-'9' according to
0057 C                               ! integer input 0-9.
0058 C       CHARACTER*255 CCOMM     ! character buffer for building up a call
0059 C                               ! to the CMMND subroutine.
0060 C       INTEGER      IBUF (128)! integer overlay of CCOMM
0061 C       EQUIVALENCE  (CCOMM, IBUF)
0062 C
0063 C       INTEGER      IMLO, IMHI  ! the range of legal COMTAL image numbers
0064 C       INTEGER      TERM      ! terminal logical unit
0065 C
0066 C***INITIALIZATION:
0067 C
0068 C       DATA      IMLO/1/, IMHI/4/
0069 C       DATA      TERM/1/
0070 C
0071 C***PROCESSING:
0072 C
0073 C       IF (.NOT.(RANGE(OUTIMG, IMLO, IMHI))) GOTO 8001 ! error return
0074 C       IF (.NOT.(RANGE(INIMG, 1, 9))) GOTO 8101 ! error return
0075 C
0076 C       Legal image numbers, so do the copy
0077 C
0078 C       2000 CCOMM = 'IMAGE '//DIGIT(OUTIMG)//' = IMAGE '//DIGIT(INIMG)
0079 C
0080 C       PROGRAMMING NOTE: see COMTAL USERS GUIDE for
0081 C       further information on the command string abbreviated
0082 C       in the string above.
0083 C
0084 C       CALL CMMND( IBUF, 17 ) ! sends copy command to COMTAL
0085 C       RETURN
0086 C
0087 C***ERROR RETURNS:
0088 C
0089 C       8001 WRITE(TERM, 8003)OUTIMG, IMLO, IMHI
0090 C       8003 FORMAT(' YOUR OUTPUT PARAMETER,', I5,', IS OUT OF RANGE:', I2)

```

```
0091      GOTO 8900
0092  C
0093  8101 WRITE(TERM, 8103)INIMG
0094  8103 FORMAT(' YOUR INPUT PARAMETER, ',I5,' , IS OUT OF RANGE: 1, 9')
0095      GOTO 8900
0096  C
0097  8900 WRITE(TERM, 8901)
0098  8901 FORMAT(' ICPY2 FAILS. NO COPYING TAKES PLACE. ')
0099      RETURN
0100      END
```



&MERGE T=00004 IS ON CR00021 USING 00005 BLKS R=0000

```
0001 CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
0002 SUBROUTINE MERGE(OUTWRD, BYTE1, BYTE2)
0003 CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
0004 C
0005 C***PARAMETER DECLARATIONS:
0006 C
0007 INTEGER OUTWRD ! the output, the 2 input bytes merged into 1 integer
0008 INTEGER BYTE1 ! the left, high order input byte (in lower byte).
0009 INTEGER BYTE2 ! the right, low order input byte (in lower byte).
0010 C
0011 C***INTRODUCTION:
0012 C
0013 C The subroutine MERGE takes two integer inputs which should be non-zero
0014 C in the lower byte only, and merges these lower order bytes into a single
0015 C integer output.
0016 C
0017 C***LANGUAGE:
0018 C
0019 C FORTRAN 77, the HP-1000 version for RTE-6/VM.
0020 C
0021 C***LIMITATIONS:
0022 C
0023 C Note that MERGE does not check that the upper order byte of the
0024 C input INTEGERS are zeros. This check could be added, but will slow
0025 C down MERGE execution. This speed is important, since MERGE was designed
0026 C to be a very low level routine.
0027 C
0028 C***SUBPROGRAMS CALLED:
0029 C none.
0030 C
0031 C***WRITTEN BY:
0032 C
0033 C The code on which this subprogram is based was written by
0034 C NETTIE D. FAULCON, July, 1983. This subprogram was written by
0035 C KEITH MILLER, July, 1984, with the support of a NASA-ASEE
0036 C summer fellowship.
0037 C
0038 C***REVISION HISTORY:
0039 C
0040 C
0041 C***LOCAL VARIABLES:
0042 C
0043 INTEGER IHOLD ! an INTEGER interpretation of bits
0044 CHARACTER CHOLD(2) ! a CHARACTER interpretation of bits
```

```

0045 C
0046     INTEGER IMERGE ! an INTEGER interpretation of bits
0047     CHARACTER CSPLIT(2)! a CHARACTER interpretation of bits
0048 C
0049     EQUIVALENCE (IHOLD, CHOLD), (IMERGE, CSPLIT)
0050 C
0051 C***INITIALIZATIONS: none.
0052 C
0053 C
0054 C***PROCESSING:
0055 C
0056     IHOLD = BYTE1
0057     CSPLIT(1) = CHOLD(2)
0058     IHOLD = BYTE2
0059     CSPLIT(2) = CHOLD(2)
0060     OUTWRD = IMERGE
0061 C
0062     RETURN
0063     END
0064
0065
0066
0067
0068

```

QNRML T=00004 IS ON CR00021 USING 00004 BLKS R=0000

0001 CCC

0002 SUBROUTINE NRML(IMAGE)

0003 CCC

0004 C

0005 C\*\*\*PARAMETER DECLARATIONS:

0006 C

0007 INTEGER IMAGE ! designates a COMTAL image to "normalize" (see below)

0008 C

0009 C\*\*\*INTRODUCTION:

0010 C

0011 C The subroutine NRML searches through an image to find its lowest  
0012 C pixel value. Then NRML replaces each pixel in the image  
0013 C (call that value X) with the value (X - low). This subroutine was  
0014 C developed to obtain an image of the variations inherent in the lighting  
0015 C table that should give a constant background light, but is instead giving  
0016 C a light with a variation of as many as 10 grey scale levels out of 255.  
0017 C The "normalized" background image is subtracted from the digitized image  
0018 C to simulate a uniform background.

0019 C

0020 C\*\*\*LANGUAGE:

0021 C

0022 C FORTRAN 77, the HP-1000 version for RTE-6/VM.

0023 C

0024 C\*\*\*LIMITATIONS:

0025 C

0026 C This subroutine makes two passes through the image, one to obtain the  
0027 C lowest pixel value, and one to write out the new pixel values. In some  
0028 C applications, the programmer may want to add new processes during one  
0029 C of those passes. Also, the subroutine HILO is used here, even though  
0030 C only the lowest value is required. To optimize, create a new subroutine  
0031 C which only determines the low value.

0032 C

0033 C\*\*\*SUBPROGRAMS CALLED:

0034 C

0035 C	name	source	load	remarks
0036 C	----	----	----	-----
0037 C	RDIL2	QRDIL2	%RDIL2	reads one horizontal COMTAL image line into 0038 C an integer array, one integer/pixel.
0039 C	WRIL2	QWRIL2	%WRIL2	writes one horizontal COMTAL image line from 0040 C an integer array; one integer/pixel.
0041 C	HILO	QHILO	%HILO	determines the highest and lowest pixel value 0042 C in an image.

0043 C

0044 C\*\*\*WRITTEN BY:

```

0045 C
0046 C   The code on which this subprogram is based was written by
0047 C   NETTIE D. FAULCON, July, 1983.  This subprogram was written by
0048 C   KEITH MILLER, July, 1984, with the support of a NASA-ASEE
0049 C   summer fellowship.
0050 C
0051 C xxxxREVISION HISTORY:
0052 C
0053 C
0054 C xxxxLOCAL VARIABLES:
0055 C
0056 C       INTEGER  IBUF(512)  ! buffer for pixel values; one pixel / integer
0057 C       INTEGER  ARALO,ARAH! array bounds for a pixel buffer
0058 C       INTEGER  LNLO, LNHI ! limits on CONTAL row numbers
0059 C       INTEGER  HI, LO     ! highest and lowest pixel values in IMAGE
0060 C       INTEGER  ROW, COL  ! loop indices
0061 C
0062 C xxxxINITIALIZATIONS:
0063 C
0064 C       DATA ARALO/1/, ARAHI/512/
0065 C       DATA LNLO /0/, LNHI /511/
0066 C
0067 C xxxxPROCESSING
0068 C
0069 C       CALL HILO(HI, LO, IMAGE)
0070 C
0071 C       DO 2000 ROW = LNLO, LNHI
0072 C           CALL RDIL2(IBUF, IMAGE, ROW)
0073 C           DO 1000 COL = ARALO, ARAHI
0074 C               IBUF(COL) = IBUF(COL) - LO
0075 C           1000 CONTINUE
0076 C           CALL WRIL2(IMAGE, ROW, IBUF)
0077 C       2000 CONTINUE
0078 C
0079 C
0080 C       RETURN
0081 C       END

```

&NOTE2 T=00004 IS ON CR00021 USING 00024 BLKS R=0000

```
0001 CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
0002 SUBROUTINE NOTE2(GRNUM,XCOOR,YCOOR,COLOR,FACTOR,NOTE)
0003 CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
0004 C
0005 C***PARAMETER DECLARATIONS:
0006 C
0007 INTEGER GRNUM ! the number of the graphic to be written into;
0008 C ! NOTES doesn't clear or display this graphic.
0009 INTEGER XCOOR !
0010 INTEGER YCOOR ! XCOOR and YCOOR define the upper left corner of the
0011 C ! position of the first character in the NOTE.
0012 INTEGER FACTOR ! controls size of characters plotted; 1..16.
0013 CHARACTER*1 COLOR ! signals if you wish to stay the same color ('S')
0014 C ! or change to red ('R'), green('G'), blue('B'),
0015 C ! black('K'), white('W'), or yellow('Y').
0016 CHARACTER*(*)NOTE ! the message to be printed in graphics; NOTES
0017 C ! designed for constant strings.
0018 C
0019 C***INTRODUCTION:
0020 C
0021 C This subroutine, NOTE2 writes a line of characters into a COMTAL graphics
0022 C plane. The parameter NOTE should be a constant string.
0023 C A very similar subroutine, NOTES, uses a string variable and a length
0024 C parameter instead of the constant string.
0025 C
0026 C***LANGUAGE:
0027 C
0028 C FORTRAN 77, the HP-1000 version for RTE-6/VM.
0029 C
0030 C***LIMITATIONS:
0031 C
0032 C GRNUM must be in the range 1-4 inclusive.
0033 C XCOOR and YCOOR must be in the range 0-511.
0034 C COLOR must be one of the following: 'S' (for "Same color"),
0035 C 'R' (for "Red"), 'G' (for "Green"), 'B' (for "Blue"),
0036 C 'K' (for "black"), 'W' (for "White") or 'Y' (for "Yellow").
0037 C Other colors could be easily added. See TABLE 1 in the NOTES
0038 C section for other colors.
0039 C FACTOR must be in the range 1-16 inclusive.
0040 C If a restriction is violated, NOTES fails with an error message.
0041 C
0042 C The graphics plane named here is turned on and all other graphics
0043 C planes are turned off when this subroutine is executed.
0044 C The GRNUM plane is NOT cleared.
```

```

0045 C
0046 C   The length of time it takes the COMTAL to write a note in graphics
0047 C   causes a timing problem; the COMTAL may ignore the next COMTAL
0048 C   command sent from the HP. Therefore, we DELAY the HP for a number
0049 C   of seconds proportional to the size of the NOTE characters.
0050 C
0051 C***SUBPROGRAMS CALLED:
0052 C
0053 C   name      source    load    remarks
0054 C   -----  -
0055 C   CMMN2     &CMMN2  %CMMN2  sends constant string to COMTAL as if it
0056 C                                     were typed at the COMTAL keyboard.
0057 C   CMMND     &CMMND  %CMMND  sends command string and length parameter to
0058 C                                     COMTAL as if it were typed at the COMTAL keyboard.
0059 C   DELAY     &DELAY  %DELAY  puts the HP in a busy wait; the argument to DELAY
0060 C                                     gives the number of seconds to DELAY.
0061 C   DIGIT     &DIGIT  %DIGIT  character*1 function that returns '0'-'9'
0062 C                                     according to integer input 0-9.
0063 C   DSPGR     &DSPGR  %DSPGR  adds a graphics plane to the display.
0064 C   LEN       -----  -----  HP FORTRAN77 intrinsic integer function that
0065 C                                     returns the length of a character string.
0066 C   RANGE     &RANGE  %RANGE  logical function that determines if 1st
0067 C                                     parameter is within the 2nd & 3rd inclusive.
0068 C
0069 C***WRITTEN BY:
0070 C
0071 C   The code on which this subprogram is based was written by
0072 C   NETTIE D. FAULCON, July, 1983. This subprogram was written by
0073 C   KEITH MILLER, July, 1984, with the support of a NASA-ASEE
0074 C   summer fellowship.
0075 C
0076 C***REVISION HISTORY:
0077 C
0078 C
0079 C***LOCAL VARIABLES:
0080 C
0081 C   CHARACTER*1  DIGIT  ! function that returns '0'-'9' for integer
0082 C                                     ! input 0-9.
0083 C   INTEGER      LEN    ! intrinsic HP FORTRAN77 function that returns
0084 C                                     ! the length of a character string.
0085 C   LOGICAL      RANGE  ! logical function determines if 1st argument is
0086 C                                     ! within 2nd and 3rd arguments, inclusive.
0087 C   CHARACTER*1  GRCHAR ! the single digit that corresponds to GRNUM.
0088 C   INTEGER      LONG   ! length of the character string.
0089 C   INTEGER      START  ! the # of first character in CBUF that holds
0090 C                                     ! the first character of NOTE.

```

```

0091 C
0092     INTEGER  GRLO,GRHI ! limits for COMTAL graphics plane number.
0093     INTEGER  CMLO,CMHI ! limits on COMTAL coordinates.
0094     INTEGER  FCLO,FCHI ! limits on FACTOR.
0095     INTEGER  TERM      ! logical unit for terminal output.
0096 C
0097     INTEGER          IBUF(128) ! buffer for sending CMMND commands.
0098     CHARACTER*255 CBUF      ! overlays IBUF
0099     EQUIVALENCE      (IBUF,CBUF)
0100 C
0101 C***INITIALIZATIONS:
0102 C
0103     DATA      GRLO/1/,GRHI/4/
0104     DATA      CMLO/0/,CMHI/511/
0105     DATA      FCLO/1/,FCHI/16/
0106     DATA      TERM/1/
0107 C
0108 C***PROCESSING:
0109 C
0110     IF (.NOT.(RANGE(GRNUM, GRLO,GRHI))) GOTO 8001 ! error return
0111     IF (.NOT.(RANGE(XCOORD, CMLO,CMHI))) GOTO 8101 ! error return
0112     IF (.NOT.(RANGE(YCOORD, CMLO,CMHI))) GOTO 8201 ! error return
0113     IF (.NOT.(RANGE(FACTOR,FCLO,FCHI))) GOTO 8301 ! error return
0114 C
0115     GRCHAR = DIGIT(GRNUM) ! initialization delayed until GRNUM checked.
0116 C
0117     IF ((COLOR .EQ. 'S').OR.(COLOR .EQ. 's')) GOTO 2000 ! "Same"
0118     IF ((COLOR .EQ. 'R').OR.(COLOR .EQ. 'r')) GOTO 1000 ! "Red"
0119     IF ((COLOR .EQ. 'G').OR.(COLOR .EQ. 'g')) GOTO 1100 ! "Green"
0120     IF ((COLOR .EQ. 'B').OR.(COLOR .EQ. 'b')) GOTO 1200 ! "Blue"
0121     IF ((COLOR .EQ. 'K').OR.(COLOR .EQ. 'k')) GOTO 1300 ! "black"
0122     IF ((COLOR .EQ. 'W').OR.(COLOR .EQ. 'w')) GOTO 1400 ! "White"
0123     IF ((COLOR .EQ. 'Y').OR.(COLOR .EQ. 'y')) GOTO 1500 ! "Yellow"
0124 C     ELSE...COLOR an illegal character
0125         GOTO 8401 ! error return
0126 C
0127 C     Color graphics red
0128     1000 CALL CMMN2('CO G '//GRCHAR//' RED')
0129         GOTO 2000
0130 C
0131 C     Color graphics green
0132     1100 CALL CMMN2('CO G '//GRCHAR//' GRN')
0133         GOTO 2000
0134 C
0135 C     Color graphics blue
0136     1200 CALL CMMN2('CO G '//GRCHAR//' BLU')

```

```

0137      GOTO 2000
0138 C
0139 C      Color graphics black
0140 1300 CALL CMMN2('CO G '//GRCHAR//' BLA')
0141      GOTO 2000
0142 C
0143 C      Color graphics white
0144 1400 CALL CMMN2('CO G '//GRCHAR//' WHT')
0145      GOTO 2000
0146 C
0147 C      Color graphics white
0148 1500 CALL CMMN2('CO G '//GRCHAR//' YEL')
0149      GOTO 2000
0150 C
0151 2000 CALL DSPGR(GRNUM)          | display the chosen graphics
0152      CALL WRTAR(XCOORD, YCOORD) | position the cursor for writing
0153 C
0154      LONG = LEN(NOTE)
0155      IF (FACTOR .GE. 10) GOTO 3000
0156 C      ELSE...
0157          CBUF(1:8) = 'G '//GRCHAR//' L '//DIGIT(FACTOR)///'
0158          START = 9
0159          LONG = LONG + 8
0160          GOTO 4000
0161 C      THEN...
0162 3000  CBUF(1:9) = 'G '//GRCHAR//' L 1 '//DIGIT(FACTOR-10)///'
0163          START = 10
0164          LONG = LONG + 9
0165          GOTO 4000
0166 C
0167 4000 CBUF(START:LONG) = NOTE
0168      CALL CMMND(IBUF, LONG)
0169 C
0170 C      Put the HP in a busy wait while the COMTAL writes the note.
0171      CALL DELAY((FACTOR/4)+1)
0172      RETURN
0173 C
0174 C***:ERROR RETURNS:
0175 C
0176 8001 WRITE(TERM, 8003) GRNUM, GRLO, GRHI
0177 8003 FORMAT(' THE GRAPHICS NUMBER, ', I5, ', IS OUT OF RANGE: ', I214, '.')
0178      GOTO 8900
0179 C
0180 8101 WRITE(TERM, 8103) XCOORD, CMLO, CMHI
0181 8103 FORMAT(' THE X COORDINATE, ', I5, ', IS OUT OF RANGE: ', I214, '.')
0182      GOTO 8900

```



```
0183 C
0184 8201 WRITE(TERM,8203)YCOOR,CMLO,CMHI
0185 8203 FORMAT(' THE Y COORDINATE, ',I5,', IS OUT OF RANGE:',2I4,',')
0186 GOTO 8900
0187 C
0188 8301 WRITE(TERM,8303)FACTOR,FCLO,FCHI
0189 8303 FORMAT(' THE SCALE FACTOR, ',I5,', IS OUT OF RANGE:',2I4,',')
0190 GOTO 8900
0191 C
0192 8401 WRITE(TERM,8403)COLOR
0193 8403 FORMAT(' THE COLOR PARAMETER, ',A1,', IS NOT S, R, G, OR B.')
0194 GOTO 8900
0195 C
0196 8900 WRITE(TERM,8901)
0197 8901 FORMAT(' NOTE2 FAILS. NO LETTERING PLACED INTO GRAPHICS.')
```

```
0198 C
```

```
0199 RETURN
```

```
0200 END
```

&NOTES T=00004 IS ON CR00021 USING 00004 BLKS R=0000

```
0001 CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
0002 SUBROUTINE NOTES (GRNUM, XCOOR, YCOOR, COLOR, FACTOR, NOTE, LENGTH)
0003 CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
0004 C
0005 C***PARAMETER DECLARATIONS:
0006 C
0007 INTEGER GRNUM ! the number of the graphic to be written into;
0008 C ! NOTES doesn't clear or display this graphic.
0009 INTEGER XCOOR !
0010 INTEGER YCOOR ! XCOOR and YCOOR define the upper left corner of the
0011 C ! position of the first character in the NOTE.
0012 INTEGER FACTOR ! controls size of characters plotted; 1-16.
0013 CHARACTER*1 COLOR ! signals if you wish to stay the same color ('S')
0014 C ! or change to red ('R'), green ('G'), or blue ('B').
0015 CHARACTER*255 NOTE ! the message to be printed in graphics is in the
0016 C ! first LENGTH characters of this string.
0017 INTEGER LENGTH ! the number of NOTE characters that are to be used.
0018 C
0019 C***INTRODUCTION:
0020 C
0021 C This subroutine, NOTES writes a line of characters into a COMTAL graphics
0022 C plane. The parameter NOTE should be a declared string.
0023 C A very similar subroutine, NOTE2, uses a string constant without a length
0024 C parameter.
0025 C
0026 C***LANGUAGE:
0027 C
0028 C FORTRAN 77, the HP-1000 version for RTE-6/VM.
0029 C
0030 C***LIMITATIONS:
0031 C
0032 C GRNUM must be in the range 1-4 inclusive.
0033 C XCOOR and YCOOR must be in the range 0-511.
0034 C COLOR must be one of the following: 'S' (for "Same color"),
0035 C 'R' (for "Red"), 'G' (for "Green"), or 'B' (for "Blue").
0036 C FACTOR must be in the range 1-16 inclusive.
0037 C LENGTH must be in the range 0-255 inclusive.
0038 C If a restriction is violated, NOTES fails with an error message.
0039 C
0040 C The graphics plane named by GRNUM is turned on and all other
0041 C graphics planes are turned off. GRNUM plane is NOT cleared.
0042 C
0043 C Because the COMTAL takes a while to write the note to the screen,
0044 C there can be a timing problem between the COMTAL and the HP which
```

```

0045 C causes the COMTAL to ignore the next HP command. Therefore, the
0046 C routine DELAY is used to cause a delay proportional to the size
0047 C of the characters being printed at the COMTAL.
0048 C
0049 C***SUBPROGRAMS CALLED:
0050 C
0051 C name source load remarks
0052 C -----
0053 C CMMN2 &CMMN2 %CMMN2 sends constant string to COMTAL as if it
0054 C were typed at the COMTAL keyboard.
0055 C CMMND &CMMND %CMMND sends a fixed string command to COMTAL as if it
0056 C were typed at the COMTAL keyboard.
0057 C DELAY &DELAY %DELAY puts the HP in a busy wait for the number of
0058 C seconds designated in DELAY's parameter.
0059 C DIGIT &DIGIT %DIGIT character*1 function that returns '0'-'9'
0060 C according to integer input 0-9.
0061 C DSPGR &DSPGR %DSPGR adds a graphic plane to the display.
0062 C RANGE &RANGE %RANGE logical function that determines if the 1st
0063 C argument is within the 2nd & 3rd inclusive.
0064 C
0065 C***WRITTEN BY:
0066 C
0067 C The code on which this subprogram is based was written by
0068 C NETTIE D. FAULCON, July, 1983. This subprogram was written by
0069 C KEITH MILLER, July, 1984, with the support of a NASA-ASEE
0070 C summer fellowship.
0071 C
0072 C***REVISION HISTORY:
0073 C
0074 C
0075 C***LOCAL VARIABLES:
0076 C
0077 C CHARACTER*1 DIGIT ! function that returns '0'-'9' for integer
0078 C ! input 0-9.
0079 C LOGICAL RANGE ! function that determines if 1st argument is
0080 C ! within 2nd and 3rd arguments, inclusive.
0081 C CHARACTER*1 GRCHAR ! the single digit that corresponds to GRNUM.
0082 C INTEGER LONG ! the length of the CMMND command.
0083 C
0084 C INTEGER GRLO,GRHI ! limits for COMTAL graphics plane number.
0085 C INTEGER CMLO,CMHI ! limits on COMTAL coordinates.
0086 C INTEGER FCLO,FCHI ! limits on FACTOR.
0087 C INTEGER STLO,STHI ! limits on string LENGTH.
0088 C INTEGER TERM ! logical unit for terminal output.
0089 C
0090 C CHARACTER*255 CBUF ! buffer for CMMND commands

```

```

0091      INTEGER      IBUF(129)! overlays CBUF
0092      EQUIVALENCE  (CBUF,IBUF)
0093      C
0094      C***INITIALIZATIONS:
0095      C
0096      DATA      GRLO/1/,GRHI/4/
0097      DATA      CMLO/0/,CMHI/511/
0098      DATA      FCLO/1/,FCHI/16/
0099      DATA      STLO/0/,STHI/255/
0100      DATA      TERM/1/
0101      C
0102      C***PROCESSING:
0103      C
0104      IF (.NOT.(RANGE(GRNUM, GRLO,GRHI))) GOTO 8001 ! error return
0105      IF (.NOT.(RANGE(XCOORD, CMLO,CMHI))) GOTO 8101 ! error return
0106      IF (.NOT.(RANGE(YCOORD, CMLO,CMHI))) GOTO 8201 ! error return
0107      IF (.NOT.(RANGE(FACTOR,FCLO,FCHI))) GOTO 8301 ! error return
0108      IF (.NOT.(RANGE(LENGTH,STLO,STHI))) GOTO 8401 ! error return
0109      C
0110      GRCHAR = DIGIT(GRNUM) ! initialization delayed until GRNUM checked.
0111      C
0112      IF ((COLOR .EQ. 'S').OR.(COLOR .EQ. 's')) GOTO 2000 ! "Same"
0113      IF ((COLOR .EQ. 'R').OR.(COLOR .EQ. 'r')) GOTO 1000 ! "Red"
0114      IF ((COLOR .EQ. 'G').OR.(COLOR .EQ. 'g')) GOTO 1100 ! "Green"
0115      IF ((COLOR .EQ. 'B').OR.(COLOR .EQ. 'b')) GOTO 1200 ! "Blue"
0116      C      ELSE...COLOR an illegal character
0117      GOTO 8501 ! error return
0118      C
0119      C      Color graphics red
0120      1000 CALL CMYN2('CO G '//GRCHAR//' RED')
0121      GOTO 2000
0122      C
0123      C      Color graphics green
0124      1100 CALL CMYN2('CO G '//GRCHAR//' GRN')
0125      GOTO 2000
0126      C
0127      C      Color graphics blue
0128      1200 CALL CMYN2('CO G '//GRCHAR//' BLU')
0129      GOTO 2000
0130      C
0131      2000 CALL DSPER(GRNUM) ! display the chosen graphics
0132      CALL WRTAR(XCOORD, YCOORD) ! position the cursor for writing
0133      C
0134      IF (FACTOR .GE. 10) GOTO 3000
0135      C      ELSE...
0136      CBUF = 'G '//GRCHAR//' L '//DIGIT(FACTOR)//' '//NOTE

```

```

0137         LONG = LENGTH + 8
0138         GOTO 4000
0139 C       THEN...
0140 3000 CBUF = 'G '//GRCHAR//' L 1'//DIGIT(FACTOR-10)//' '//NOTE
0141         LONG = LENGTH + 9
0142         GOTO 4000
0143 C
0144 4000 CALL CMMND(IBUF, LONG)
0145 C
0146 C       Delay for a second or two to relieve timing problem between
0147 C       the HP and the COMTAL
0148         CALL DELAY((FACTOR/4)+1)
0149         RETURN
0150 C
0151 C***ERROR RETURNS:
0152 C
0153 8001 WRITE(TERM, 8003) GRNUM, GRLO, GRHI
0154 8003 FORMAT(' THE GRAPHICS NUMBER, ', I5, ', IS OUT OF RANGE: ', I214, ', ')
0155         GOTO 8900
0156 C
0157 8101 WRITE(TERM, 8103) XCOORD, CMLO, CMHI
0158 8103 FORMAT(' THE X COORDINATE, ', I5, ', IS OUT OF RANGE: ', I214, ', ')
0159         GOTO 8900
0160 C
0161 8201 WRITE(TERM, 8203) YCOORD, CMLO, CMHI
0162 8203 FORMAT(' THE Y COORDINATE, ', I5, ', IS OUT OF RANGE: ', I214, ', ')
0163         GOTO 8900
0164 C
0165 8301 WRITE(TERM, 8303) FACTOR, FCLO, FCHI
0166 8303 FORMAT(' THE SCALE FACTOR, ', I5, ', IS OUT OF RANGE: ', I214, ', ')
0167         GOTO 8900
0168 C
0169 8401 WRITE(TERM, 8403) LENGTH, STLO, STHI
0170 8403 FORMAT(' THE STRING LENGTH, ', I5, ', IS OUT OF RANGE: ', I214, ', ')
0171         GOTO 8900
0172 C
0173 8501 WRITE(TERM, 8503) COLOR
0174 8503 FORMAT(' THE COLOR PARAMETER, ', A1, ', IS NOT S, R, G, OR B.')
0175         GOTO 8900
0176 C
0177 8900 WRITE(TERM, 8901)
0178 8901 FORMAT(' NOTES FAILS. NO LETTERING PLACED INTO GRAPHICS.')
0179 C
0180         RETURN
0181         END

```

&PAINT T=00004 IS ON CR00021 USING 00004 BLKS R=0000

```
0001 CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
0002     SUBROUTINE PAINT(IMAGE, BRUSH, SHADE)
0003 CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
0004 C
0005 C***PARAMETER DECLARATIONS:
0006 C
0007     INTEGER IMAGE ! number of the COMTAL image to be painted
0008     INTEGER BRUSH ! size of the square brush area
0009     INTEGER SHADE ! the pixel value to be brushed on
0010 C
0011 C***INTRODUCTION:
0012 C
0013 C The subroutine PAINT allows the interactive user to paint
0014 C onto a COMTAL image, using the trackball to guide the brush.
0015 C The user uses the HP keyboard to signal when to paint.
0016 C Each time the HP <CR> is pressed, PAINT paints a BRUSH X BRUSH
0017 C square of pixels with the cursor position in the upper left corner
0018 C of the square. The maximum size for a brush has been set (arbitrarily)
0019 C to 64 pixels square.
0020 C
0021 C***LANGUAGE:
0022 C
0023 C FORTRAN 77, the HP-1000 version for RTE-6/VM.
0024 C
0025 C***LIMITATIONS:
0026 C
0027 C The COMTAL keyboard is inaccessible to the HP. Therefore, we
0028 C must use the HP keyboard even though we use the COMTAL trackball.
0029 C Because of the HP keyboard limitations, the program requires
0030 C a pointilistic painting: one square in the image is darkened
0031 C each time the HP <CR> is pressed.
0032 C
0033 C***SUBPROGRAMS CALLED:
0034 C
0035 C name      source  load  remarks
0036 C -----  -----  -----  -----
0037 C DSPBW     &DSPBW  %DSPBW display a monochrome COMTAL image.
0038 C WRIRC     &WRIRC  %WRIRC writes the contents of an array to a COMTAL image;
0039 C           the values are read into a rectangle in the image.
0040 C RDTAR     &RDTAR  %RDTAR reads the current COMTAL cursor position.
0041 C WAIT      &WAIT   %WAIT  halts HP processing until HP <CR> entered.
0042 C CMMND     &CMMND  %CMMND sends a character string to COMTAL, which
0043 C           treats the string as a COMTAL keyboard command.
0044 C
```

```

0045 C***WRITTEN BY:
0046 C
0047 C   The code on which this subprogram is based was written by
0048 C   NETTIE D. FAULCON, July, 1983. This subprogram was written by
0049 C   KEITH MILLER, July, 1984, with the support of a NASA-ASEE
0050 C   summer fellowship.
0051 C
0052 C***REVISION HISTORY:
0053 C
0054 C
0055 C***LOCAL VARIABLES:
0056 C
0057 C       INTEGER    BOX(64*64)           ! the square that acts as a paint brush-
0058 C                                           ! this rectangle placed into IMAGE whenever
0059 C                                           ! <CR> entered to locate cursor on COMTAL.
0060 C       INTEGER    XPOS                 ! loop index for initializing BOX.
0061 C       INTEGER    UPLFX, UPLFY        ! X and Y coordinates of last COMTAL cursor
0062 C                                           ! (target) position recorded via RDTAR;
0063 C                                           ! used as upper left corner of rectangle
0064 C                                           ! to be painted.
0065 C       INTEGER    TERM                 ! logical unit for terminal I/O.
0066 C       INTEGER    IMLO, IMHI          ! limits on COMTAL image number.
0067 C       INTEGER    PXLO, PXHI          ! limits on COMTAL pixel values.
0068 C       INTEGER    BRLO, BRHI          ! limits on size of brush for painting.
0069 C       CHARACTER*1 INCHAR             ! character buffer for HP keyboard input.
0070 C       LOGICAL    RANGE               ! function that determines if 1st argument
0071 C                                           ! is within the 2nd and 3rd, inclusive.
0072 C       INTEGER    IBUF(128)           ! buffer for sending CMMND strings
0073 C       CHARACTER*255 CBUF             ! character overlay for IBUF
0074 C       EQUIVALENCE (IBUF,CBUF)
0075 C
0076 C
0077 C***INITIALIZATIONS:
0078 C
0079 C       DATA IMLO/1/, IMHI/4/
0080 C       DATA BRLO/1/, BRHI/128/
0081 C       DATA PXLO/0/, PXHI/255/
0082 C       DATA TERM/1/
0083 C
0084 C***PROCESSING
0085 C
0086 C       IF (.NOT.(RANGE(IMAGE, IMLO, IMHI))) GOTO 8001 ! error return
0087 C       IF (.NOT.(RANGE(BRUSH, BRLO, BRHI))) GOTO 8101 ! error return
0088 C       IF (.NOT.(RANGE(SHADE, PXLO, PXHI))) GOTO 8201 ! error return
0089 C
0090 C       Make sure the image in question is displayed

```

```

0091 C
0092 CALL DSPBW(IMAGE)
0093 C
0094 C Initialize the paint brush (this initialization done here
0095 C instead of above to avoid processing when a parameter is bad).
0096 C
0097 C Programming note: the BOX array is filled as a one dimensional array,
0098 C but is interpreted by WRIRC below as a two dimensional array.
0099 C
0100 DO 400 XPOS = 1, (BRUSH*BRUSH)
0101     BOX(XPOS) = SHADE
0102 400 CONTINUE
0103 C
0104 C add the target and dump the image on the COMTAL
0105 C
0106 CBUF = 'ADD TARGET $DUMP IMAGE '
0107 CALL CMMND(IBUF, 23)
0108 C
0109 C Give instructions to user
0110 C
0111 WRITE(TERM, 501)
0112 501 FORMAT(' ENTER <CR> TO PAINT A SQUARE.',
0113 1         ' ENTER S<CR> TO EXIT PAINTING.')
0114 C
0115 C Loop for input/painting starts here:
0116 C
0117 1000 INCHAR = ' '
0118 READ(TERM, 1001) INCHAR
0119 1001 FORMAT(1A1)
0120 IF (INCHAR .EQ. ' ') GOTO 2000 ! paint another square and continue.
0121 IF ((INCHAR .EQ. 'S') .OR. (INCHAR .EQ. 's')) GOTO 9000 ! terminate
0122 C ELSE...illegal entry
0123 WRITE(TERM, 1003) INCHAR
0124 1003 FORMAT(' THE CHARACTER ENTERED, "', 1A1, '" IS NOT LEGAL',
0125 1         ' FOR PAINTING.',/,/, ' PLEASE TRY AGAIN.')
0126 GOTO 1000
0127 C
0128 C Actual painting takes place here:
0129 C
0130 2000 CALL RDTAR(UPLFX, UPLFY)
0131 CALL WRIRC(IMAGE, UPLFX, UPLFY, BOX, BRUSH, BRUSH)
0132 GOTO 1000
0133 C
0134 C***ERROR RETURNS:
0135 C
0136 8001 WRITE(TERM, 8003) IMAGE, IMLO, IMHI

```



```
0137 8003 FORMAT(' IMAGE NUMBER.', 15, ', OUT OF RANGE:', 214, ',.')
0138     GOTO 8900
0139 C
0140 8101 WRITE(TERM, 8103) BRUSH, BRLO, BRHI
0141 8103 FORMAT(' BRUSH ARGUMENT.', 15, ', OUT OF RANGE:', 214, ',.')
0142     GOTO 8900
0143 C
0144 8201 WRITE(TERM, 8203) SHADE, PXLO, PXSH
0145 8203 FORMAT(' SHADE ARGUMENT.', 15, ', OUT OF RANGE:', 214, ',.')
0146     GOTO 8900
0147 C
0148 8900 WRITE(TERM, 8901)
0149 8901 FORMAT(' PAINT SUBROUTINE FAILS.')
```

```
0150 C
0151 9000 CONTINUE
0152     RETURN
0153     END
0154
```

&PROFL T=00004 IS ON CR00021 USING 00004 BLKS R=0000

```
0001 CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
0002 SUBROUTINE PROFL(GRAPH, IMAGE)
0003 CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
0004 C
0005 C***PARAMETER DECLARATIONS:
0006 C
0007 INTEGER GRAPH ! the number of the COMTAL graphics plane in which
0008 C ! the profile is to be displayed.
0009 INTEGER IMAGE ! the number of the COMTAL image which is going to
0010 C ! be "profiled."
0011 C
0012 C***INTRODUCTION:
0013 C
0014 C The subroutine PROFILE gives HP access to the COMTAL's interactive
0015 C profiling facilities. This subroutine initializes the COMTAL for
0016 C taking profiles of IMAGE and displaying them in the specified GRAPH
0017 C plane. Note that this subroutine clears the specified graphics plane.
0018 C Control passes to COMTAL for profiling, and then a <CR> on the HP
0019 C terminal restores control to the HP terminal.
0020 C
0021 C***LANGUAGE:
0022 C
0023 C FORTRAN 77, the HP-1000 version for RTE-6/VM.
0024 C
0025 C***LIMITATIONS:
0026 C
0027 C GRAPH and IMAGE are checked, and if they are out of bounds, no
0028 C profiling takes place and an error message is printed. The shift
0029 C from the HP terminal to the COMTAL terminal may result in some
0030 C confusion, but this switch exploits the COMTAL circuitry much more
0031 C efficiently than would be possible using the HP terminal alone.
0032 C
0033 C***SUBPROGRAMS CALLED:
0034 C
0035 C name source load remarks
0036 C -----
0037 C RANGE &RANGE %RANGE logical function that determines if 1st argu-
0038 C ment is within 2nd and 3rd, inclusive.
0039 C CMMND &CMMND %CMMND sends a command to the COMTAL as if it were
0040 C typed at the COMTAL keyboard.
0041 C WAIT &WAIT %WAIT halts HP processing until <CR> is pressed on
0042 C the HP keyboard.
0043 C CHAR ----- intrinsic HP FORTRAN77 function that converts
0044 C integers into characters
```

```

0045 C DIGIT &DIGIT %DIGIT character*1 function that returns '0'..'9'
0046 C
0047 C according to integer input 0..9.
0048 C***WRITTEN BY:
0049 C
0050 C The code on which this subprogram is based was written by
0051 C NETTIE D. FAULCON, July, 1983. This subprogram was written by
0052 C KEITH MILLER, July, 1984, with the support of a NASA-ASEE
0053 C summer fellowship.
0054 C
0055 C***REVISION HISTORY:
0056 C
0057 C***LOCAL VARIABLES:
0058 C
0059 CHARACTER*1 GCHAR ! the character that corresponds to the single
0060 C ! digit argument, GRAPH
0061 CHARACTER*1 ICHAR ! the character that corresponds to the single
0062 C ! digit argument, IMAGE
0063 CHARACTER*1 BELL ! the ASCII code for ESCAPE.
0064 CHARACTER*1 DIGIT ! function that returns '0'..'9' according to
0065 C ! integer argument 0..9.
0066 CHARACTER CHAR ! intrinsic FORTRAN77 function for integer to
0067 C ! character conversion.
0068 C
0069 INTEGER IBUF(128) ! integer buffer for sending COMTAL commands
0070 CHARACTER*255 CBUF ! overlays IBUF
0071 EQUIVALENCE (IBUF,CBUF)
0072 C
0073 INTEGER GRLO,GRHI ! limits on numbers of graphics planes in COMTAL
0074 INTEGER IMLO,IMHI ! limits on numbers of image planes in COMTAL
0075 LOGICAL RANGE ! logical function that determines if 1st argu-
0076 C ! ment is within 2nd and 3rd argument, inclusive.
0077 INTEGER TERM ! logical unit for terminal output
0078 C
0079 C***INITIALIZATIONS:
0080 C
0081 DATA GRLO/1/,GRHI/4/
0082 DATA IMLO/1/,IMHI/4/
0083 DATA TERM/1/
0084 BELL = CHAR(7) ! HP bell
0085 C
0086 C***PROCESSING
0087 C
0088 IF (.NOT.(RANGE(GRAPH,GRLO,GRHI))) GOTO 8001
0089 IF (.NOT.(RANGE(IMAGE,IMLO,IMHI))) GOTO 8101
0090 C

```

```

0091 C.   set up COMTAL display for profiling
0092 C
0093     GCHAR = DIGIT(GRAPH)
0094     ICHAR = DIGIT(IMAGE)
0095 C
0096 C     the following COMTAL command reads as follows (#G stands for
0097 C     the single digit associated with GRAPH; and #I, with IMAGE):
0098 C     Display Image #I; Add Graphics #G; CLear Graphics #G;
0099 C     COLor Graphics #G RED.
0100 C
0101     CBUF = 'D I '//ICHAR//' $ADD G '//GCHAR//' $CL G '//GCHAR//
0102 1     ' $CO G '//GCHAR//' RED'
0103     CALL CMMND(IBUF, 34)
0104 C
0105 C     the following COMTAL command reads: Add TArget:
0106 C     Graphics #G = PROfile of image #I.
0107 C
0108     CBUF = 'A TA $G '//GCHAR//' PRO '//ICHAR
0109     CALL CMMND(IBUF, 15)
0110 C
0111 C     print out instructions for the COMTAL keyboard interaction
0112 C
0113     WRITE(TERM, 7001)
0114 7001 FORMAT(' COMTAL function switches control profiling.',
0115 1     /,' Switch 2 toggles X profiling on and off.', /,
0116 2     ' Switch 3 toggles Y profiling on and off.', /,
0117 3     ' Switch 1 requests new profiles.', /,
0118 4     ' Press <ESC> on the COMTAL keyboard and then press',
0119 5     ' <CR> on the HP keyboard to end profiling.')
0120     CALL WAIT
0121     RETURN
0122 C
0123 C***ERROR RETURNS:
0124 C
0125 8001 WRITE(TERM, 8003)GRAPH, GRLO, GRHI
0126 8003 FORMAT(' THE GRAPH PARAMETER, ',I4,', IS OUT OF RANGE:',2I3)
0127     GOTO 8900
0128 C
0129 8101 WRITE(TERM, 8103)IMAGE, IMLO, IMHI
0130 8103 FORMAT(' THE IMAGE PARAMETER, ',I4,', IS OUT OF RANGE:',2I3)
0131     GOTO 8900
0132 C
0133 8900 WRITE(TERM, 8901)
0134 8901 FORMAT(' PROFL FAILS. NO PROFILING DONE.')
0135     RETURN
0136     END

```

&RANGE T=00004 IS ON CR00021 USING 00003 BLKS R=0000

```
0001 CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
0002 LOGICAL FUNCTION RANGE(OBJECT, LOW, HIGH)
0003 CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
0004 C
0005 C***PARAMETER DECLARATIONS:
0006 C
0007 INTEGER OBJECT ! the number to be examined
0008 INTEGER LOW, HIGH ! the limits on the number
0009 C
0010 C***INTRODUCTION:
0011 C
0012 C This function returns .TRUE. if the OBJECT is within the range
0013 C between LOW and HIGH (inclusive), and .FALSE. if it is outside
0014 C that range. If LOW .GT. HIGH, an error message is printed,
0015 C and .FALSE. is returned.
0016 C
0017 C***LIMITATIONS:
0018 C
0019 C No error code is returned. Only a message is printed out.
0020 C
0021 C***WRITTEN BY:
0022 C
0023 C Keith Miller, NASA-Langley ASEE fellow, 1984
0024 C
0025 C***REVISION HISTORY:
0026 C
0027 C
0028 C***LOCAL VARIABLES:
0029 C
0030 INTEGER TERM ! logical unit for terminal output
0031 C
0032 C***INITIALIZATIONS:
0033 C
0034 DATA TERM/1/
0035 C
0036 C***PROCESSING:
0037 C
0038 IF (LOW .GT. HIGH) WRITE(TERM, 1001) LOW, HIGH
0039 1001 FORMAT(' LOW, ', I5, ', .GT. HIGH, ', I5, '. RANGE fails.')
0040 RANGE = ((OBJECT .GE. LOW) .AND. (OBJECT .LE. HIGH))
0041 RETURN
0042 END
0043
0044
```

&RDGLN T=00004 IS ON CR00021 USING 00018 BLKS R=0000

```
0001 CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
0002 SUBROUTINE RDGLN (ONOFFS, GRAPH, LINE)
0003 CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
0004 C
0005 C***PARAMETER DECLARATIONS:
0006 C
0007 INTEGER ONOFFS(32) ! 16 bits per integer, 512 bits in a line.
0008 INTEGER GRAPH ! COMTAL image graphics # to be written to.
0009 INTEGER LINE ! which horizontal line to be written to;
0010 C ! lines numbered from 1 (screen top) to 512.
0011 C
0012 C***INTRODUCTION:
0013 C
0014 C This subroutine Reads a Graphics Line from the COMTAL. The line of
0015 C bits is coded as 1 for on and 0 for off.
0016 C
0017 C***LANGUAGE:
0018 C
0019 C FORTRAN 77, the HP-1000 version for RTE-6/VM.
0020 C
0021 C***LIMITATIONS:
0022 C
0023 C If GRAPH or LINE is out of range, an error message is printed
0024 C and no transfer takes place.
0025 C
0026 C***SUBPROGRAMS CALLED:
0027 C
0028 C name source load remarks
0029 C -----
0030 C RANGE &RANGE %RANGE logical function that determines if its
0031 C first parameter is within the last two parameters.
0032 C
0033 C***WRITTEN BY:
0034 C
0035 C The code on which this subprogram is based was written by
0036 C NETTIE D. FAULCON, July, 1983. This modification is by
0037 C KEITH MILLER, June, 1984.
0038 C
0039 C***REVISION HISTORY:
0040 C
0041 C
0042 C***LOCAL VARIABLES:
0043 C
0044 LOGICAL RANGE ! function that ascertains if its first parameter
```

```

0045 C          ! is between (inclusive) its last 2 parameters
0046 C          INTEGER TERM          ! the logical unit for terminal output
0047 C          INTEGER GRLO, GRHI ! the limits on COMTAL monochrome image numbers
0048 C          INTEGER LNLO, LNHI ! the limits on COMTAL image line numbers
0049 C
0050 C***INITIALIZATIONS:
0051 C
0052 C          DATA    TERM/1/
0053 C          DATA    GRLO/1/, GRHI/4/
0054 C          DATA    LNLO/0/, LNHI/511/
0055 C
0056 C***PROCESSING
0057 C
0058 C          IF (.NOT.(RANGE(GRAPH,GRLO,GRHI))) GOTO 8001 ! error return
0059 C          IF (.NOT.(RANGE(LINE, LNLO,LNHI))) GOTO 8101 ! error return
0060 C
0061 C          Programming note:
0062 C          The EXEC call is explained in detail in the
0063 C          HP Programmer's Reference Manual for RTE-6/VM,p.2-19ff. This
0064 C          transfer function for the COMTAL is discussed in the
0065 C          COMTAL User's Manual, Section 5.2.2.1. In the EXEC call
0066 C          that follows, the HP resident driver, DVR41, is called as
0067 C          follows: the first parameter (1) signifies a read; the
0068 C          second parameter is in two parts: 36B identifies the resident
0069 C          DVR41 driver, and 100B identifies the line transfer operation
0070 C          of that driver; the third parameter (ONOFFS) holds the data to be
0071 C          transferred, and the fourth parameter gives ONOFFS' length in words
0072 C          (32); and the final parameter is a COMTAL command code for the transfer
0073 C          CALL EXEC(1,36B+100B,ONOFFS,32,(GRAPH-1)*2048 + LINE + 512)
0074 C          RETURN
0075 C
0076 C***ERROR RETURNS
0077 C
0078 C          8001 WRITE(TERM, 8003) GRAPH, GRLO, GRHI
0079 C          8003 FORMAT(' GRAPHICS NUMBER,', I3, ' OUT OF RANGE:', 2I2, '.')
0080 C          GOTO 8900
0081 C
0082 C          8101 WRITE(TERM, 8103) LINE, LNLO, LNHI
0083 C          8103 FORMAT(' LINE NUMBER,', I4, ' OUT OF RANGE:', 2I3, '.')
0084 C          GOTO 8900
0085 C
0086 C          8900 WRITE(TERM, 8901)
0087 C          8901 FORMAT(' RDGLN FAILS. NO TRANSFER.')
```

```

0088 C          RETURN
0089 C          END
```

&RDGPT T=00004 IS ON CR00021 USING 00024 BLKS R=0000

0001 CCC  
0002 SUBROUTINE RDGPT(VALUE, GRAPH, XCOORD, YCOORD)  
0003 CCC  
0004 C

0005 C\*\*\*PARAMETER DECLARATIONS:  
0006 C

0007 INTEGER VALUE ! 0-1 graphics value to be retrieved.  
0008 INTEGER GRAPH ! COMTAL graph number to be read from.  
0009 INTEGER XCOORD, YCOORD! point where value is to be read from.  
0010 C

0011 C\*\*\*INTRODUCTION:  
0012 C

0013 C This subroutine Reads a Graph Point from the COMTAL. The value  
0014 C of the graphics is an on/off decision, represented  
0015 C by an integer 1 or 0.  
0016 C If the graph number or coordinates are out of range, an error  
0017 C message is printed and no transfer takes place.  
0018 C

0019 C\*\*\*LANGUAGE:  
0020 C

0021 C FORTRAN 77, the HP-1000 version for RTE-6/VM.  
0022 C

0023 C\*\*\*LIMITATIONS:  
0024 C

0025 C If GRAPH, XCOORD, YCOORD, or VALUE are out of range, an error message  
0026 C is printed at the terminal and no transfer takes place.  
0027 C

0028 C\*\*\*SUBPROGRAMS CALLED:  
0029 C

0030 C	name	source	load	remarks
0031 C	-----	-----	-----	-----
0032 C	RANGE	&RANGE	%RANGE	logical function that determines if its 0033 C first parameter is within the last two parameters.
0034 C	BTEST	-----	-----	HP FORTRAN77 intrinsic function that takes an 0035 C integer argument and returns .TRUE. or .FALSE. 0036 C according to the bit selected by its second 0037 C argument; bit = 0, FALSE returned. 0038 C

0039 C\*\*\*WRITTEN BY:  
0040 C

0041 C The code on which this subprogram is based was written by  
0042 C NETTIE D. FAULCON, July, 1983. This modification is by  
0043 C KEITH MILLER, June, 1984.  
0044 C



```

0045 C***REVISION HISTORY:
0046 C
0047 C
0048 C***LOCAL VARIABLES:
0049 C
0050 LOGICAL RANGE ! function that ascertains if its first parameter
0051 C ! is between (inclusive) its last 2 parameters
0052 LOGICAL BTEST ! is a certain bit on or off.
0053 INTEGER TERM ! the logical unit for terminal output
0054 INTEGER GRLO, GRHI ! the limits on COMTAL monochrome graph numbers
0055 INTEGER LNLO, LNHI ! the limits on COMTAL graph line numbers
0056 INTEGER BTLO, BTHI ! the limits on COMTAL graphics values
0057 INTEGER BITS(32) ! a buffer to read & write a COMTAL graphics line
0058 INTEGER WORD ! which word of BITS holds the bit selected by XCOORD.
0059 INTEGER BIT ! which bit in BITS(WORD) holds the bit selected by
0060 C ! XCOORD: bits numbered 0-15, right to left.
0061 C
0062 C***INITIALIZATIONS:
0063 C
0064 DATA TERM/1/
0065 DATA GRLO/1/, GRHI/4/
0066 DATA LNLO/0/, LNHI/511/
0067 C
0068 C***PROCESSING
0069 C
0070 IF (.NOT.(RANGE(GRAPH,GRLO,GRHI))) GOTO 8001 ! error return
0071 IF (.NOT.(RANGE(XCOORD,LNLO,LNHI))) GOTO 8101 ! error return
0072 IF (.NOT.(RANGE(YCOORD,LNLO,LNHI))) GOTO 8201 ! error return
0073 C
0074 C Programming note:
0075 C The EXEC calls below are to the DVR41 driver. The first call
0076 C is identical to the call made in RDILN. The second EXEC call
0077 C is identical to the one in WRILN. See the documentation for
0078 C those subroutines for details on these calls.
0079 C
0080 C Read the COMTAL line (horizontal) that contains the point in question:
0081 C
0082 CALL EXEC(1,368+1008,BITS,32,(GRAPH-1)*2048 + 512 + YCOORD)
0083 C
0084 C Find the single bit that has been selected:
0085 C
0086 WORD = (XCOORD/16) + 1
0087 BIT = (16*WORD) - XCOORD - 1
0088 C
0089 VALUE = 0 ! bit is clear until proven set.
0090 IF (BTEST(BITS(WORD),BIT)) VALUE = 1

```

```

0091     RETURN
0092 C
0093 C***ERROR RETURNS
0094 C
0095     8001 WRITE(TERM, 8003) GRAPH, GRLO, GRHI
0096     8003 FORMAT(' GRAPH NUMBER,', I3, ' OUT OF RANGE:', 2I2, '.')
0097     GOTO 8900
0098 C
0099     8101 WRITE(TERM, 8103)XCOORD, LNLO, LNHI
0100     8103 FORMAT(' X COORDINATE,', I4, ' OUT OF RANGE:', 2I4, '.')
0101     GOTO 8900
0102 C
0103     8201 WRITE(TERM, 8203)YCOORD, LNLO, LNHI
0104     8203 FORMAT(' Y COORDINATE,', I4, ' OUT OF RANGE:', 2I4, '.')
0105     GOTO 8900
0106 C
0107     8900 WRITE(TERM, 8901)
0108     8901 FORMAT(' RDGPT FAILS. NO TRANSFER.')
0109     RETURN
0110     END

```

&RDIL2 T=00004 IS ON CR00021 USING 00018 BLKS R=0000

```
0001 CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
0002 SUBROUTINE RDIL2(INTS, IMAGE, LINE)
0003 CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
0004 C
0005 C***PARAMETER DECLARATIONS:
0006 C
0007 INTEGER INTS(512) ! 512 pixel values to be read, one integer/pixel
0008 INTEGER IMAGE ! COMTAL image number to be read from
0009 INTEGER LINE ! which horizontal line is to be read from;
0010 C ! lines numbered from 1 (screen top) to 512.
0011 C
0012 C***INTRODUCTION:
0013 C
0014 C This subroutine Reads an Image Line from the COMTAL. The line of
0015 C pixels is made up of 8 bit (0-255) grey scale intensities. The
0016 C PIXELS array will be filled to capacity by RDIL2.
0017 C
0018 C***LANGUAGE:
0019 C
0020 C FORTRAN 77, the HP-1000 version for RTE-6/VM.
0021 C
0022 C***LIMITATIONS:
0023 C
0024 C IMAGE must be a monochrome image. The LINE parameter must be between
0025 C 0 and 511. If IMAGE or LINE is out of range, an error message is printed
0026 C and no transfer takes place.
0027 C
0028 C***SUBPROGRAMS CALLED:
0029 C
0030 C name source load remarks
0031 C -----
0032 C RANGE %RANGE %RANGE logical function that determines if its
0033 C first parameter is within the last two parameters.
0034 C ICHAR ----- intrinsic HP FORTRAN77 function that converts
0035 C a character (1 byte) into an integer (2 bytes)
0036 C
0037 C***WRITTEN BY:
0038 C
0039 C The code on which this subprogram is based was written by
0040 C NETTIE D. FAULCON, July, 1983. This modification is by
0041 C KEITH MILLER, June, 1984.
0042 C
0043 C***REVISION HISTORY:
0044 C
```

```

0045 C
0046 C***LOCAL VARIABLES:
0047 C
0048     INTEGER  PIXELS(256)! 512 bytes, 1 pixel/byte, transferred to COMTAL
0049     CHARACTER*1 CPIX(512)! overlays PIXELS
0050     EQUIVALENCE (PIXELS,CPIX)
0051 C
0052     LOGICAL RANGE      ! function that ascertains if its first parameter
0053 C                      ! is between (inclusive) its last 2 parameters
0054     INTEGER TERM       ! the logical unit for terminal output
0055     INTEGER IMLO, IMHI ! the limits on COMTAL monochrome image numbers
0056     INTEGER LNLO, LNHI ! the limits on COMTAL image line numbers
0057     INTEGER INDEX      ! loop index for stepping through the arrays.
0058 C
0059 C***INITIALIZATIONS:
0060 C
0061     DATA  TERM/1/
0062     DATA  IMLO/1/, IMHI/4/
0063     DATA  LNLO/0/, LNHI/511/
0064 C
0065 C***PROCESSING
0066 C
0067     IF (.NOT.(RANGE(IMAGE, IMLO, IMHI))) GOTO 8001 ! error return
0068     IF (.NOT.(RANGE(LINE, LNLO, LNHI))) GOTO 8101 ! error return
0069 C
0070 C     Programming note:
0071 C     The EXEC call is explained in detail in the
0072 C     HP Programmer's Reference Manual for RTE-6/VM,p.2-19ff. This
0073 C     transfer function for the COMTAL is discussed in the
0074 C     COMTAL User's Manual, Section 5.2.2.1. In the EXEC call
0075 C     that follows, the HP resident driver, DVR41, is called as
0076 C     follows: the first parameter (1) signifies a read; the
0077 C     second parameter is in two parts: 36B identifies the resident
0078 C     DVR41 driver, and 100B identifies the line transfer operation
0079 C     of that driver; the third parameter (PIXELS) holds the data to be
0080 C     transferred, and the fourth parameter gives PIXELS' length in words
0081 C     (256); and the final parameter is a COMTAL command code for the transfer
0082     CALL EXEC(1, 36B+100B, PIXELS, 256, (IMAGE-1)*2048 + LINE)
0083 C
0084 C     transform the 512 8 bit values into 512 16 bit integers
0085 C
0086     DO 1000 INDEX = 1,512
0087         INTS(INDEX) = ICHAR(CPIX(INDEX)) ! CPIX overlays PIXELS
0088     1000 CONTINUE
0089 C
0090     RETURN

```

```
0091 C
0092 C***ERROR RETURNS
0093 C
0094 8001 WRITE(TERM, 8003) IMAGE, IMLO, IMHI
0095 8003 FORMAT(' IMAGE NUMBER.', I3, ' OUT OF RANGE:', 212, '.')
0096 GOTO 8900
0097 C
0098 8101 WRITE(TERM, 8103) LINE, LNLO, LNHI
0099 8103 FORMAT(' LINE NUMBER.', I4, ' OUT OF RANGE:', 213, '.')
0100 GOTO 8900
0101 C
0102 8900 WRITE(TERM, 8901)
0103 8901 FORMAT(' RDIL2 FAILS. NO TRANSFER.')
0104 RETURN
0105 END
```

&RDILN T=00004 IS ON CR00021 USING 00018 BLKS R=0000

```

0001 CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
0002          SUBROUTINE RDILN(PIXELS, IMAGE, LINE)
0003 CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
0004 C
0005 C***PARAMETER DECLARATIONS:
0006 C
0007          INTEGER PIXELS(256)! 512 bytes (pixels) to be read
0008          INTEGER IMAGE      ! COMTAL image number to be read from
0009          INTEGER LINE       ! which horizontal line to be read from:
0010 C                           ! lines numbered from 1 (screen top) to 512.
0011 C
0012 C***INTRODUCTION:
0013 C
0014 C This subroutine Reads an Image Line from the COMTAL. The line of
0015 C pixels is made up of 8 bit (0-255) grey scale intensities. The
0016 C PIXELS array will be filled to capacity by RDILN.
0017 C
0018 C***LANGUAGE:
0019 C
0020 C FORTRAN 77, the HP-1000 version for RTE-6/VM.
0021 C
0022 C***LIMITATIONS:
0023 C
0024 C IMAGE must be a monochrome image. The LINE parameter must be between
0025 C 0 and 511. If IMAGE or LINE is out of range, an error message is printed
0026 C and no transfer takes place.
0027 C
0028 C***SUBPROGRAMS CALLED:
0029 C
0030 C      name      source  load   remarks
0031 C      -----  -
0032 C RANGE  GRANGE  %RANGE logical function that determines if its
0033 C                          first parameter is within the last two parameters.
0034 C
0035 C***WRITTEN BY:
0036 C
0037 C The code on which this subprogram is based was written by
0038 C NETTIE D. FAULCON, July, 1983. This modification is by
0039 C KEITH MILLER,      June, 1984.
0040 C
0041 C***REVISION HISTORY:
0042 C
0043 C***LOCAL VARIABLES:
0044 C

```

```

0045 LOGICAL RANGE ! function that ascertains if its first parameter
0046 C ! is between (inclusive) its last 2 parameters
0047 INTEGER TERM ! the logical unit for terminal output
0048 INTEGER IMLO, IMHI ! the limits on COMTAL monochrome image numbers
0049 INTEGER LNLO, LNHI ! the limits on COMTAL image line numbers
0050 C
0051 C***INITIALIZATIONS:
0052 C
0053 DATA TERM/1/
0054 DATA IMLO/1/, IMHI/4/
0055 DATA LNLO/0/, LNHI/511/
0056 C
0057 C***PROCESSING
0058 C
0059 IF (.NOT.(RANGE(IMAGE, IMLO, IMHI))) GOTO 8001 ! error return
0060 IF (.NOT.(RANGE(LINE, LNLO, LNHI))) GOTO 8101 ! error return
0061 C
0062 C Programming note:
0063 C The EXEC call is explained in detail in the
0064 C HP Programmer's Reference Manual for RTE-6/VM, p.2-19ff. This
0065 C transfer function for the COMTAL is discussed in the
0066 C COMTAL User's Manual, Section 5.2.2.1. In the EXEC call
0067 C that follows, the HP resident driver, DVR41, is called as
0068 C follows: the first parameter (1) signifies a read; the
0069 C second parameter is in two parts: 36B identifies the resident
0070 C DVR41 driver, and 100B identifies the line transfer operation
0071 C of that driver; the third parameter (PIXELS) holds the data to be
0072 C transferred, and the fourth parameter gives PIXELS' length in words
0073 C (256); and the final parameter is a COMTAL command code for the transfer
0074 C CALL EXEC(1, 36B+100B, PIXELS, 256, (IMAGE-1)*2048 + LINE)
0075 C RETURN
0076 C
0077 C***ERROR RETURNS
0078 C
0079 8001 WRITE(TERM, 8003) IMAGE, IMLO, IMHI
0080 8003 FORMAT(' IMAGE NUMBER.', I3, ' OUT OF RANGE:', I2, '.')
0081 GOTO 8900
0082 C
0083 8101 WRITE(TERM, 8103) LINE, LNLO, LNHI
0084 8103 FORMAT(' LINE NUMBER.', I4, ' OUT OF RANGE:', I3, '.')
0085 GOTO 8900
0086 C
0087 8900 WRITE(TERM, 8901)
0088 8901 FORMAT(' RDILN FAILS. NO TRANSFER.')
0089 RETURN
0090 END

```

&RDIPT T=0004 IS ON CR0021 USING 0018 BLKS R=0000

```
0001 CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
0002 SUBROUTINE RDIPT(VALUE, IMAGE, XCOOR, YCOOR)
0003 CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
0004 C
0005 C***PARAMETER DECLARATIONS:
0006 C
0007 INTEGER VALUE ! receives pixel value, converted from byte
0008 C ! to integer.
0009 INTEGER IMAGE ! COMTAL image number to be read from.
0010 INTEGER XCOOR, YCOOR! point where value is to be read from.
0011 C
0012 C***INTRODUCTION:
0013 C
0014 C This subroutine Reads an Image Point from the COMTAL. The value
0015 C of the pixel is an 8 bit (0-255) grey scale intensity. If the
0016 C image number or coordinates are out of range, an
0017 C message is printed and no transfer takes place.
0018 C
0019 C***LANGUAGE:
0020 C
0021 C FORTRAN 77, the HP-1000 version for RTE-6/VM.
0022 C
0023 C***LIMITATIONS:
0024 C
0025 C The IMAGE designated
0026 C must be a monochrome image. The YCOOR parameter must be between
0027 C 0 and 511. If IMAGE or YCOOR is out of range, an error message is printed
0028 C and no transfer takes place.
0029 C
0030 C***SUBPROGRAMS CALLED:
0031 C
0032 C name source load remarks
0033 C -----
0034 C RANGE &RANGE %RANGE logical function that determines if its
0035 C first parameter is within the last two parameters.
0036 C ICHAR ----- HP FORTRAN77 intrinsic function; converts a byte
0037 C into its integer code.
0038 C
0039 C***WRITTEN BY:
0040 C
0041 C The code on which this subprogram is based was written by
0042 C NETTIE D. FAULCON, July, 1983. This modification is by
0043 C KEITH MILLER, June, 1984.
0044 C
```



```

0045 C***REVISION HISTORY:
0046 C
0047 C
0048 C***LOCAL VARIABLES:
0049 C
0050 LOGICAL RANGE ! function that ascertains if its first parameter
0051 C ! is between (inclusive) its last 2 parameters
0052 INTEGER TERM ! the logical unit for terminal output
0053 INTEGER IMLO, IMHI ! the limits on COMTAL monochrome image numbers
0054 INTEGER LNLO, LNHI ! the limits on COMTAL image line numbers
0055 INTEGER PIXELS(256)! a buffer to read a COMTAL line
0056 CHARACTER*1 BYTES(512)! overlay for PIXELS buffer
0057 EQUIVALENCE (PIXELS,BYTES)
0058 C
0059 C***INITIALIZATIONS:
0060 C
0061 DATA TERM/1/
0062 DATA IMLO/1/, IMHI/4/
0063 DATA LNLO/0/, LNHI/511/
0064 C
0065 C***PROCESSING
0066 C
0067 IF (.NOT.(RANGE(IMAGE,IMLO,IMHI))) GOTO 8001 ! error return
0068 IF (.NOT.(RANGE(YCOOR,LNLO,LNHI))) GOTO 8101 ! error return
0069 IF (.NOT.(RANGE(XCOOR,LNLO,LNHI))) GOTO 8201 ! error return
0070 C
0071 C Programming note:
0072 C The EXEC call below is to the DVR41 driver, and
0073 C is identical to the call made in RDILN.
0074 C See the RDILN documentation for the details.
0075 C
0076 C Read the COMTAL line (horizontal) that contains the point in question:
0077 C
0078 CALL EXEC(1, 36B+100B, PIXELS, 256, (IMAGE-1)*2048 + YCOOR)
0079 C
0080 C convert the byte that is to be read:
0081 C
0082 VALUE = ICHAR(BYTES(XCOOR+1))! ICHAR is an intrinsic F77 function
0083 C ! which converts a character into its
0084 C ! integer code.
0085 C ! The '+1' changes from pixels, which are
0086 C ! #ed 0-511 to the FORTRAN array, 1-512
0087 RETURN
0088 C
0089 C***ERROR RETURNS
0090 C

```

```
0091 8001 WRITE(TERM, 8003) IMAGE, IMLO, IMHI
0092 8003 FORMAT(' IMAGE NUMBER.', I3, ' OUT OF RANGE:', 2I2, '.')
0093      GOTO 8900
0094 C
0095 8101 WRITE(TERM, 8103) YCOORD, LNLO, LNHI
0096 8103 FORMAT(' Y COORDINATE.', I4, ' OUT OF RANGE:', 2I4, '.')
0097      GOTO 8900
0098 C
0099 8201 WRITE(TERM, 8203) XCOORD, LNLO, LNHI
0100 8203 FORMAT(' Y COORDINATE.', I4, ' OUT OF RANGE:', 2I4, '.')
0101      GOTO 8900
0102 C
0103 8900 WRITE(TERM, 8901)
0104 8901 FORMAT(' RDIPT FAILS. NO TRANSFER.')
0105      RETURN
0106      END
```

&RDIRC T=0004 IS ON CR00021 USING 00004 BLKS R=0000

```
0001 CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
0002          SUBROUTINE RDIRC(OUTARA, XDIM, YDIM, IMAGE, UPLFX, UPLFY)
0003 CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
0004 C
0005 C***PARAMETER DECLARATIONS:
0006 C
0007          INTEGER XDIM, YDIM          ! dimensions of the output array, OUTARA
0008          INTEGER OUTARA(XDIM, YDIM) ! the array to be filled
0009          INTEGER IMAGE                ! the number of the COMTAL image from
0010 C                                  ! which OUTARA is to be filled
0011          INTEGER UPLFX, UPLFY        ! the image coordinates of the upper left
0012 C                                  ! corner of the rectangle of pixels that is
0013 C                                  ! to be read into OUTARA.
0014 C
0015 C***INTRODUCTION:
0016 C
0017 C
0018 C The subroutine Read Image ReCtangle transfers pixel values from a
0019 C designated section of an image to an integer array. Note that although
0020 C pixel values are generally stored with 1 byte/pixel, RDIRC places each
0021 C numeric value into a 2 byte integer in OUTARA.
0022 C XDIM, YDIM, IMAGE, UPLFX, and UPLFY are all checked for possible out
0023 C of range errors before any transfer is attempted.
0024 C
0025 C***LANGUAGE:
0026 C
0027 C FORTRAN 77, the HP-1000 version for RTE-6/VM.
0028 C
0029 C***LIMITATIONS:
0030 C
0031 C RDIRC does a great deal of error checking before initiating the
0032 C transfer. If the programmer can verify that all calls to RDIRC
0033 C will be legal, these checks could be commented out to improve
0034 C machine efficiency. Also, the call to the subroutine RDILN could
0035 C be replaced inline by the RDILN code (which is short).
0036 C
0037 C***SUBPROGRAMS CALLED:
0038 C
0039 C name      source  load  remarks
0040 C -----  -
0041 C RDILN    &RDILN  %RDILN transfers a line of bytes from a COMTAL image
0042 C RANGE    &RANGE  %RANGE logical function that determines if its first
0043 C          parameter is within the last two parameters
0044 C ICHAR    -----  ----- HP FORTRAN77 intrinsic function; converts a
```

```

0045 C                               byte into its integer code.
0046 C
0047 C***WRITTEN BY:
0048 C
0049 C   The code on which this subprogram is based was written by
0050 C   NETTIE D. FAULCON, July, 1983. This code was written by
0051 C   KEITH MILLER,      July, 1984.
0052 C
0053 C***REVISION HISTORY:
0054 C
0055 C
0056 C***LOCAL VARIABLES:
0057 C
0058 C   INTEGER    IMROW,  IMCOL  ! a location in the image
0059 C   INTEGER    ARAROW, ARACOL ! a location in OUTARA
0060 C   INTEGER    ICHAR      ! intrinsic byte to integer conversion
0061 C                               ! function
0062 C   LOGICAL     RANGE      ! function that ascertains if the first
0063 C                               ! parameter is within the last two parameters.
0064 C   INTEGER     IMLO,  IMHI  ! limits on COMTAL image numbers
0065 C   INTEGER     LNLO,  LNHI  ! limits on COMTAL pixel coordinates
0066 C   INTEGER     LNCNT      ! LNHI-LNLO+1, # of pixels in an image line
0067 C   INTEGER     TERM       ! logical unit for terminal output
0068 C   INTEGER     IBUF(256)  ! buffer to hold COMTAL horizontal line
0069 C   CHARACTER*1 CBUF(512) ! overlay for IBUF
0070 C   EQUIVALENCE (IBUF, CBUF)
0071 C
0072 C***INITIALIZATIONS:
0073 C
0074 C   DATA  TERM/1/
0075 C   DATA  IMLO/1/, IMHI/4/
0076 C   DATA  LNLO/0/, LNHI/511/, LNCNT/512/
0077 C
0078 C***PROCESSING
0079 C
0080 C   IF (.NOT.(RANGE(IMAGE,      IMLO,IMHI )))GOTO 8001 ! error return
0081 C   IF (.NOT.(RANGE(XDIM,      1,  LNCNT)))GOTO 8101 ! error return
0082 C   IF (.NOT.(RANGE(YDIM,      1,  LNCNT)))GOTO 8201 ! error return
0083 C   IF (.NOT.(RANGE(UPLFX,     LNLO,LNHI )))GOTO 8301 ! error return
0084 C   IF (.NOT.(RANGE(UPLFY,     LNLO,LNHI )))GOTO 8401 ! error return
0085 C   IF (.NOT.(RANGE(UPLFX+XDIM-1,LNLO,LNHI )))GOTO 8501 ! error return
0086 C   IF (.NOT.(RANGE(UPLFY+YDIM-1,LNLO,LNHI )))GOTO 8601 ! error return
0087 C
0088 C   we get to this point if the transfer is to take place
0089 C
0090 C   IMROW = UPLFY

```

```

0091      DO 2000 ARAROW = 1, YDIM
0092      CALL RDILN(IBUF, IMAGE, IMROW)
0093      IMROW = IMROW + 1 ! increment for next pass thru 2000 loop
0094 C
0095 C      ! the next line initializes the column pointer;
0096      IMCOL = UPLFX + 1 ! the "+1" is necessary because COMTAL image
0097 C      ! coordinates range from 0 to 511 and the
0098 C      ! FORTRAN array indices range from 1 to 512.
0099      DO 1000 ARACOL = 1, XDIM
0100      OUTARA(ARAROW, ARACOL) = ICHAR(CBUF(IMCOL))
0101      IMCOL = IMCOL + 1
0102 1000    CONTINUE
0103 2000    CONTINUE
0104      RETURN      ! successful termination
0105 C
0106 C***ERROR RETURNS:
0107 C
0108 8001 WRITE(TERM, 8003) IMAGE, IMLO, IMHI
0109 8003 FORMAT(' IMAGE NUMBER,', I3, ' OUT OF RANGE:', I2, '.')
0110      GOTO 8900
0111 C
0112 8101 WRITE(TERM, 8103) XDIM, 1, LNCNT
0113 8103 FORMAT(' X DIMENSION,', I4, ' OUT OF RANGE:', I2, '.')
0114      GOTO 8900
0115 C
0116 8201 WRITE(TERM, 8203) YDIM, 1, LNCNT
0117 8203 FORMAT(' Y DIMENSION,', I4, ' OUT OF RANGE:', I2, '.')
0118      GOTO 8900
0119 C
0120 8301 WRITE(TERM, 8303) UPLFX, LNLO, LNHI
0121 8303 FORMAT(' X COORDINATE FOR CORNER,', I4, ' OUT OF RANGE:', I2, '.')
0122      GOTO 8900
0123 C
0124 8401 WRITE(TERM, 8403) UPLFY, LNLO, LNHI
0125 8403 FORMAT(' Y COORDINATE FOR CORNER,', I4, ' OUT OF RANGE:', I2, '.')
0126      GOTO 8900
0127 C
0128 8501 WRITE(TERM, 8503) UPLFX, XDIM, LNLO, LNHI
0129 8503 FORMAT(' X COORDINATE FOR THE CORNER AND THE X DIMENSION ',
0130      1      ' OF THE ARRAY', //, ' OVERFLOW IMAGE BOUNDARIES.',
0131      2      ' X COORDINATE =', I4, ' X DIMENSION =', I4, //,
0132      3      ' IMAGE COORDINATE LIMITS ARE ', I2, '.')
0133      GOTO 8900
0134 C
0135 8601 WRITE(TERM, 8603) UPLFY, YDIM, LNLO, LNHI
0136 8603 FORMAT(' Y COORDINATE FOR THE CORNER AND THE Y DIMENSION ',

```

```
0137 - 1      ' OF THE ARRAY',/, ' OVERFLOW IMAGE BOUNDARIES.',
0138 - 2      ' X COORDINATE =', I4, 'X DIMENSION =', I4, /,
0139 - 3      ' IMAGE COORDINATE LIMITS ARE ', 2I5, '.')
0140      GOTO 8900
0141 C
0142 8900 WRITE(TERM, 8903)
0143 8903 FORMAT(' RDIRC FAILS. NO TRANSFER TAKES PLACE.')
0144      RETURN
0145      END
0146
0147
```

&RDLUT T=00004 IS ON CR00021 USING 00018 BLKS R=0000

```
0001 CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
0002 SUBROUTINE RDLUT(TABLE, LUTNUM)
0003 CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
0004 C
0005 C***PARAMETER DECLARATIONS:
0006 C
0007 INTEGER TABLE(256)! the values of the specified COMTAL look-up
0008 C ! table are read into this array. The 0 entry
0009 C ! goes into TABLE(1), ..., the 255 entry goes
0010 C ! into TABLE(256).
0011 INTEGER LUTNUM ! the number of the COMTAL look-up table (called
0012 C ! "function memory" in the COMTAL literature).
0013 C
0014 C***INTRODUCTION:
0015 C
0016 C The subroutine Read Look-Up Table (LUT) reads the COMTAL mapping from
0017 C the integers 0-255 into the integer array TABLE. This LUT can be used
0018 C for grey level enhancements in the COMTAL. A similar subroutine
0019 C called RDPSU is used to read from a pseudocolor look-up table. This
0020 C routine is only used for grey scale look-up tables.
0021 C
0022 C***LANGUAGE:
0023 C
0024 C FORTRAN 77, the HP-1000 version for RTE-6/VM.
0025 C
0026 C***LIMITATIONS:
0027 C
0028 C Although 8 bits are sufficient for the look up table values, full
0029 C integers are used in TABLE. This format is dictated by the COMTAL
0030 C conventions as given in section 5.2.3.1.
0031 C
0032 C***SUBPROGRAMS CALLED:
0033 C
0034 C name source load remarks
0035 C -----
0036 C RANGE @RANGE %RANGE logical function which determines if its 1st
0037 C parameter is within its 2nd and 3rd inclusive.
0038 C
0039 C***WRITTEN BY:
0040 C
0041 C The code on which this subprogram is based was written by
0042 C NETTIE D. FAULCON, July, 1983. This subprogram was written by
0043 C KEITH MILLER, July, 1984, with the support of a NASA-ASEE
0044 C summer fellowship.
```

```

0045 C
0046 C***REVISION HISTORY:
0047 C
0048 C
0049 C***LOCAL VARIABLES:
0050 C
0051 LOGICAL RANGE      | logical function which determines if its 1st
0052 C                  | parameter is within its 2nd and 3rd inclusive.
0053 INTEGER TERM       | logical unit for terminal output
0054 INTEGER LUTLO,LUTHI | limits for COMTAL function memories
0055 C
0056 C***INITIALIZATIONS:
0057 C
0058 DATA TERM/1/
0059 DATA LUTLO/1/, LUTHI/4/
0060 C
0061 C***PROCESSING
0062 C
0063 IF (.NOT.(RANGE(LUTNUM,LUTLO,LUTHI))) GOTO 8001 ! error return
0064 C
0065 C Programming notes:
0066 C The EXEC command parameters are discussed in the HP RTE-6/VM
0067 C Programmer's Reference Manual, 2-19ff. The COMTAL parameters
0068 C are discussed in section 5.2.3 of the COMTAL User's Manual.
0069 C
0070 C The first parameter to EXEC identifies the EXEC command as
0071 C a read command. The second parameter identifies the resident
0072 C HP driver (36B) and gives the code (200B) that identifies this
0073 C operation, a transfer to a COMTAL function memory (Look-Up Table).
0074 C The third parameter gives the Look-Up Table values (TABLE),
0075 C and the fourth parameter gives the length of TABLE in words.
0076 C The fifth parameter is a COMTAL code that is described bit by
0077 C bit in the User's Manual. In short, bit 15 signifies write to
0078 C COMTAL, bit 14 designates function memory instead of pseudocolor,
0079 C bit 12 signifies standard replacement, and bits 8&9 identify the
0080 C function memory to be used. (Bits are numbered 15 high, 0 low).
0081 C
0082 CALL EXEC( 1, 36B+200B, TABLE, 256, ((LUTNUM-1)*256) )
0083 RETURN
0084 C
0085 C***ERROR RETURN
0086 C
0087 8001 WRITE(TERM, 8003) LUTNUM, LUTLO, LUTHI
0088 8003 FORMAT(' THE FUNCTION MEMORY NUMBER, ',I4,', IS OUT OF RANGE:',
0089 1 214, '.')
0090 8900 WRITE(TERM, 8901)

```



0091 8901 FORMAT(' RDLUT FAILS. NO TRANSFER FROM COMTAL.')

0092

END

&RDPSU T=00004 IS ON CR00021 USING 00018 BLKS R=0000

```
0001 CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
0002 SUBROUTINE RDPSU(TABLE)
0003 CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
0004 C
0005 C***PARAMETER DECLARATIONS:
0006 C
0007 INTEGER TABLE(768)! the values of the specified COMTAL look-up
0008 C ! table are read into this array. The RED table
0009 C ! is in TABLE(1:256); the GREEN, in TABLE(257:512);
0010 C ! and the BLUE, in TABLE(513:768).
0011 C
0012 C***INTRODUCTION:
0013 C
0014 C The subroutine Read the PSeUdocoloR table reads the 3 COMTAL mappings
0015 C from 0-255 which comprise the pseudocolor table. Note that the values
0016 C are placed into TABLE in the order RED, GREEN, and BLUE.
0017 C
0018 C***LANGUAGE:
0019 C
0020 C FORTRAN 77, the HP-1000 version for RTE-6/VM.
0021 C
0022 C***LIMITATIONS:
0023 C
0024 C Although 8 bits are sufficient for the look-up table values, full
0025 C integers are used in TABLE. This format is dictated by the COMTAL
0026 C conventions as given in section 5.2.3.1.
0027 C
0028 C***SUBPROGRAMS CALLED:
0029 C
0030 C NONE.
0031 C
0032 C***WRITTEN BY:
0033 C
0034 C The code on which this subprogram is based was written by
0035 C NETTIE D. FAULCON, July, 1983. This subprogram was written by
0036 C KEITH MILLER, July, 1984, with the support of a NASA-ASEE
0037 C summer fellowship.
0038 C
0039 C***REVISION HISTORY:
0040 C
0041 C
0042 C***LOCAL VARIABLES:
0043 C
0044 C NONE.
```

```

0045 C
0046 C***INITIALIZATIONS:
0047 C
0048 C  NONE
0049 C
0050 C***PROCESSING
0051 C
0052 C  Programming notes:
0053 C  The EXEC command parameters are discussed in the HP RTE-6/VM
0054 C  Programmer's Reference Manual, 2-19ff.  The COMTAL parameters
0055 C  are discussed in section 5.2.3 of the COMTAL User's Manual.
0056 C
0057 C  The first parameter to EXEC identifies the EXEC command as
0058 C  a read command.  The second parameter identifies the resident
0059 C  HP driver (36B) and gives the code (300B) that identifies this
0060 C  operation, a transfer from the COMTAL pseudocolor table.
0061 C  The third parameter gives the array that will hold the values,
0062 C  and the fourth parameter gives the length of TABLE in words.
0063 C  The fifth parameter is a COMTAL code that is described bit by
0064 C  bit in the User's Manual.  The DVR41 driver takes care of all the
0065 C  bits except 8&9 which identify the color to be transferred.
0066 C
0067 C  Note that we make three separate calls to EXEC.  Each call fills a
0068 C  different section of TABLE with a different color of the COMTAL's
0069 C  pseudocolor table.
0070 C
0071 C  CALL EXEC( 1, 36B+300B, TABLE(1), 256, 1*256 ) ! red
0072 C  CALL EXEC( 1, 36B+300B, TABLE(257), 256, 0*256 ) ! green
0073 C  CALL EXEC( 1, 36B+300B, TABLE(513), 256, 2*256 ) ! blue
0074 C  RETURN
0075 C  END

```

&RDTAB T=00004 IS ON CR00021 USING 00018 BLKS R=0000

```
0001 CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
0002 SUBROUTINE RDTAB(TABLE, CNUMB, IMOGR)
0003 CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
0004 C
0005 C***PARAMETER DECLARATIONS:
0006 C
0007 INTEGER TABLE(16) ! holds the image/graphics table from COMTAL.
0008 INTEGER CNUMB ! COMTAL image or graph number.
0009 INTEGER IMOGR ! "Image Or Graphics", 0-Image, 1-Graphics.
0010 C
0011 C***INTRODUCTION:
0012 C
0013 C This subroutine Reads the TABLE associated with the image or graphics
0014 C memory plane identified by CNUMB (the COMTAL image or graphics number)
0015 C and IMOGR (which is either a 0, indicating an image, or 1, indicating
0016 C a graphics plane. The 16 word, 32 byte table goes into TABLE.
0017 C
0018 C***LANGUAGE:
0019 C
0020 C FORTRAN 77, the HP-1000 version for RTE-6/VM.
0021 C
0022 C***LIMITATIONS:
0023 C
0024 C IMOGR must be either a 0 or a 1, and CNUMB must be between 1 and 9
0025 C inclusive. If either number is out of range, no transfer takes place.
0026 C
0027 C***SUBPROGRAMS CALLED:
0028 C
0029 C name source load remarks
0030 C -----
0031 C RANGE &RANGE %RANGE logical function that determines if its
0032 C first parameter is within the last two parameters.
0033 C BTEST ----- HP FORTRAN77 intrinsic function that takes an
0034 C integer argument and returns .TRUE. or .FALSE.
0035 C according to the bit selected by its second
0036 C argument; bit = 0, FALSE returned.
0037 C
0038 C***WRITTEN BY:
0039 C
0040 C The code on which this subprogram is based was written by
0041 C NETTIE D. FAULCON, July, 1983. This modification is by
0042 C KEITH MILLER, June, 1984.
0043 C
0044 C***REVISION HISTORY:
```

```

0045 C
0046 C
0047 C***LOCAL VARIABLES:
0048 C
0049 LOGICAL RANGE      ! function that ascertains if its first parameter
0050 C                  ! is between (inclusive) its last 2 parameters
0051 LOGICAL BTEST      ! is a certain bit on or off.
0052 INTEGER TERM       ! the logical unit for terminal output
0053 INTEGER DGLO, DGHI ! the limits on single DiGit image/graphics numbers
0054 INTEGER BTLO, BTHI ! the limits on COMTAL graphics values
0055 C                  ! XCOOR: bits numbered 0-15, right to left.
0056 C
0057 C***INITIALIZATIONS:
0058 C
0059 DATA TERM/1/
0060 DATA DGLO/1/, DGHI/9/
0061 DATA BTLO/0/, BTHI/1/
0062 C
0063 C***PROCESSING
0064 C
0065 IF (.NOT.(RANGE(CNUMB,DGLO,DGHI))) GOTO 8001 ! error return
0066 IF (.NOT.(RANGE(IMOGR,BTLO,BTHI))) GOTO 8101 ! error return
0067 C
0068 C Programming note:
0069 C The EXEC call below is to the DVR41 driver.
0070 C The first argument, "1", identifies the operation as a read.
0071 C The second argument has two parts: "36B" identifies the DVR41 driver,
0072 C and "500B" selects a transfer code = 3 operation of that driver.
0073 C The third argument, "TABLE", is the buffer that will hold the IGT
0074 C ("Image/Graphics Table") information after the EXEC is completed;
0075 C the fourth argument, "16", gives the length of TABLE in words.
0076 C The final parameter is a code to the COMTAL which identifies the
0077 C mode of the transfer ("7*4096"), mode 7; selects either image or
0078 C graphics ("IMOGR*128"); and gives the number of the image/graphics
0079 C memory plane ("CNUMB-1").
0080 C
0081 CALL EXEC(1,36B+500B,TABLE,16, 7*4096 + IMOGR*128 + CNUMB-1)
0082 RETURN
0083 C
0084 C***ERROR RETURNS
0085 C
0086 8001 WRITE(TERM, 8003) CNUMB, DGLO, DGHI
0087 8003 FORMAT(' COMTAL NUMBER,', I3, ' OUT OF RANGE:', 2I2, '.')
0088 GOTO 8900
0089 C
0090 8101 WRITE(TERM, 8103) IMOGR, BTLO, BTHI

```

```
0091 8103 FORMAT(' IMAGE/GRAPHICS ARGUMENT MUST BE 0 OR 1, NOT '.14.'.')
0092      GOTO 8900
0093 C
0094 8900 WRITE(TERM, 8901)
0095 8901 FORMAT(' RDTAB FAILS. NO TRANSFER.')
```

```
RETURN
```

```
END
```

&RDTAR T=00004 IS ON CR00021 USING 00012 BLKS R=0000

```
0001 CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
0002 SUBROUTINE RDTAR(XCOOR, YCOOR)
0003 CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
0004 C
0005 C***PARAMETER DECLARATIONS:
0006 C
0007 INTEGER XCOOR ! X coordinate of the COMTAL target location.
0008 INTEGER YCOOR ! Y coordinate of the COMTAL target location.
0009 C
0010 C***INTRODUCTION:
0011 C
0012 C This subroutine Reads the TARGET (cursor) location from the COMTAL.
0013 C
0014 C***LANGUAGE:
0015 C
0016 C FORTRAN 77, the HP-1000 version for RTE-6/VM.
0017 C
0018 C***LIMITATIONS: NONE
0019 C
0020 C***SUBPROGRAMS CALLED: NONE
0021 C
0022 C***WRITTEN BY:
0023 C
0024 C The code on which this subprogram is based was written by
0025 C NETTIE D. FAULCON, July, 1983. This modification is by
0026 C KEITH MILLER, June, 1984.
0027 C
0028 C***REVISION HISTORY:
0029 C
0030 C
0031 C***LOCAL VARIABLES:
0032 C
0033 INTEGER IBUF(2) ! the buffer to hold the COMTAL data transfer
0034 C
0035 C***INITIALIZATIONS: NONE
0036 C
0037 C***PROCESSING
0038 C
0039 C Programming note:
0040 C The EXEC call is explained in detail in the
0041 C HP Programmer's Reference Manual for RTE-6/VM, p.2-19ff. This
0042 C transfer function for the COMTAL is discussed in the
0043 C COMTAL User's Manual, Section 5.2.4. In the EXEC call
0044 C that follows, the HP resident driver, DVR41, is called as
```

```
0045 C follows: the first parameter (1) signifies a read; the
0046 C second parameter is in two parts: 36B identifies the resident
0047 C DVR41 driver, and 400B identifies the target transfer operation
0048 C of that driver; the third parameter (IBUF) will hold the COMTAL data
0049 C to be transfered, and the fourth parameter gives the length in words
0050 C (2); the final parameter is a COMTAL command code for the transfer,
0051 C (00000B).
0052 C
0053 CALL EXEC(1, 36B+400B, IBUF, 2, 00000B)
0054 XCOOR = IBUF(1)
0055 YCOOR = IBUF(2)
0056 RETURN
0057 END
```



&SETV T=00004 IS ON CR00021 USING 00004 BLKS R=0000

```
0001 CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
0002 SUBROUTINE SETV(IMAGE)
0003 CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
0004 C
0005 C***PARAMETER DECLARATIONS:
0006 C
0007 INTEGER IMAGE ! the COMTAL image number to be assigned to the
0008 C ! video camera; 5 is traditional.
0009 C
0010 C***INTRODUCTION:
0011 C
0012 C The subroutine "SET Video" establishes the image number that is to be
0013 C associated with the video camera, and displays the camera input.
0014 C
0015 C***LANGUAGE:
0016 C
0017 C FORTRAN 77, the HP-1000 version for RTE-6/VM.
0018 C
0019 C***LIMITATIONS:
0020 C
0021 C SETV should only be called once during a COMTAL session. If the
0022 C camera is already set to IMAGE and SETV is called, the COMTAL
0023 C freezes. Manually reset the COMTAL with the SHIFT~ to release any
0024 C previous SETV command.
0025 C
0026 C Note that IMAGE should not be an image number used for COMTAL memory.
0027 C For programming convenience, we insist on a single digit. 5 is
0028 C traditional in this lab, but 5-9 will do.
0029 C
0030 C***SUBPROGRAMS CALLED:
0031 C
0032 C name source load remarks
0033 C -----
0034 C CMMND &CMMND %CMMND sends character strings to the COMTAL where
0035 C they are much like COMTAL keyboard commands.
0036 C DIGIT &DIGIT %DIGIT character*1 function that returns '0'-'9'
0037 C according to integer input 0-9.
0038 C RANGE &RANGE %RANGE logical function which determines if its 1st
0039 C argument is within the 2nd and 3rd, inclusive.
0040 C
0041 C***WRITTEN BY:
0042 C
0043 C The code on which this subprogram is based was written by
0044 C NETTIE D. FAULCON, July, 1983. This subprogram was written by
```

```

0045 C KEITH MILLER, July, 1984, with the support of a NASA-ASEE
0046 C summer fellowship.
0047 C
0048 C***REVISION HISTORY:
0049 C
0050 C
0051 C***LOCAL VARIABLES:
0052 C
0053 LOGICAL RANGE ! function which determines if its 1st argument
0054 C ! is within the 2nd & 3rd arguments, inclusive.
0055 CHARACTER*1 DIGIT ! function that returns '0'-'9' according
0056 C ! to integer input 0-9.
0057 INTEGER IBUF(128) ! holds COMTAL keyboard command strings for
0058 C ! use with CMMND.
0059 CHARACTER*255 CBUF ! overlays IBUF.
0060 EQUIVALENCE (IBUF,CBUF)
0061 INTEGER VDLO,VDHI ! the limits on video image number; one digit,
0062 C ! but not a # reserved for image memory.
0063 INTEGER TERM ! logical unit number for terminal input
0064 C
0065 C***INITIALIZATIONS:
0066 C
0067 DATA VDLO/5/, VDHI/9/
0068 DATA TERM/1/
0069 C
0070 C***PROCESSING:
0071 C
0072 IF (.NOT.(RANGE(IMAGE,VDLO,VDHI))) GOTO 8001 ! error return
0073 C
0074 C
0075 C the following COMTAL command is expanded to:
0076 C Set Video image #I; Display Image #I
0077 C where #I is the digit equal to IMAGE.
0078 C This command will hang up the COMTAL if the camera is already
0079 C set to IMAGE. However, we can't do a RELEASE just to be sure,
0080 C because if the camera is NOT set to IMAGE, then a RELEASE also
0081 C hangs up! (catch-22.) If the mode 7 IGP table transfer is
0082 C incorporated into DVR41, perhaps the table can be inquired
0083 C about SET or not SET. However, the COMTAL Users Manual is not
0084 C clear on that matter.
0085 C
0086 CBUF = 'S V '//DIGIT(IMAGE)//' $D I '//DIGIT(IMAGE)
0087 CALL CMMND(IBUF,12)
0088 RETURN
0089 C
0090 C***ERROR RETURNS:

```

```
0091 C
0092 8001 WRITE(TERM,8003)IMAGE, VDLO, VDHI
0093 8003 FORMAT(' YOUR IMAGE ARGUMENT,', I4,', IS OUT OF RANGE:', 2I4)
0094 C
0095 WRITE(TERM, 8901)
0096 8901 FORMAT(' SETV FAILS. NO ACTION TAKEN.')
```

```
0097 C
```

```
0098 RETURN
```

```
0099 END
```

```
0100
```

&SPLIT T=00004 IS ON CR00021 USING 00005 BLKS R=0000

```
0001 CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
0002 SUBROUTINE SPLIT(BYTE1, BYTE2, INBYTE)
0003 CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
0004 C
0005 C***PARAMETER DECLARATIONS:
0006 C
0007 INTEGER BYTE1 ! on output, this integer holds the left (high order)
0008 C ! byte of INBYTE in its right (low order) byte.
0009 INTEGER BYTE2 ! on output, this integer holds the right (low order)
0010 C ! byte of INBYTE in its right (low order) byte.
0011 INTEGER INBYTE ! the input integer whose two bytes are to be SPLIT.
0012 C
0013 C***INTRODUCTION:
0014 C
0015 C The subroutine SPLIT takes each of the two bytes of INBYTE and
0016 C interprets each byte as a separate integer. The two resulting
0017 C numbers are assigned to BYTE1 (which gets the high order byte of
0018 C INBYTE) and to BYTE2 (which gets the low order byte).
0019 C
0020 C***LANGUAGE:
0021 C
0022 C FORTRAN 77, the HP-1000 version for RTE-6/VM.
0023 C
0024 C***LIMITATIONS:
0025 C
0026 C None.
0027 C
0028 C***SUBPROGRAMS CALLED:
0029 C
0030 C None.
0031 C
0032 C***WRITTEN BY:
0033 C
0034 C The code on which this subprogram is based was written by
0035 C NETTIE D. FAULCON, July, 1983. This subprogram was written by
0036 C KEITH MILLER, July, 1984, with the support of a NASA-ASEE
0037 C summer fellowship.
0038 C
0039 C***REVISION HISTORY:
0040 C
0041 C
0042 C***LOCAL VARIABLES:
0043 C
0044 INTEGER IHOLD ! an INTEGER interpretation of bits
```

```
0045 - CHARACTER CHOLD(2) ! a CHARACTER interpretation of bits
0046 C
0047 INTEGER IMERGE ! an INTEGER interpretation of bits
0048 CHARACTER CSPLIT(2)! a CHARACTER interpretation of bits
0049 C
0050 EQUIVALENCE (IHOLD, CHOLD), (IMERGE, CSPLIT)
0051 C
0052 IMERGE = INBYTE
0053 IHOLD = 0 ! zero out high order bits
0054 CHOLD(2) = CSPLIT(1)
0055 BYTE1 = IHOLD
0056 CHOLD(2) = CSPLIT(2)
0057 BYTE2 = IHOLD
0058 C
0059 RETURN
0060 END
```

&SPRED T=00004 IS ON CR00021 USING 00004 BLKS R=0000

```
0001 CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
0002 SUBROUTINE SPRED(IMAGE)
0003 CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
0004 C
0005 C***PARAMETER DECLARATIONS:
0006 C
0007 C INTEGER IMAGE ! the number of the COMTAL image to be contrast spread
0008 C
0009 C***INTRODUCTION:
0010 C
0011 C The subroutine SPREaD takes as input and output a COMTAL image.
0012 C The subroutine requires two passes through the image. On the
0013 C first pass, SPRED determines the high and low pixel values in
0014 C the image. On the second pass, SPRED replaces each pixel value
0015 C X with  $(X - \text{lowest}) * (255 / (\text{highest} - \text{lowest}))$ . If the lowest and
0016 C highest value are identical, no pixels are replaced and no message
0017 C is printed. If the lowest value is 0 and the highest 255, no
0018 C pixels are replace and no message is printed.
0019 C
0020 C***LANGUAGE:
0021 C
0022 C FORTRAN 77, the HP-1000 version for RTE-6/VM.
0023 C
0024 C***LIMITATIONS:
0025 C
0026 C The present implementation reads from the COMTAL image twice. This
0027 C could be speeded up by placing the pixel values into a virtual array
0028 C during the first pass. Also, the decision not to change a uniform
0029 C grey image at all is arbitrary (but defensible...how do you stretch
0030 C a constant value?).
0031 C
0032 C***SUBPROGRAMS CALLED:
0033 C
0034 C name source load remarks
0035 C -----
0036 C RDIL2 &RDIL2 %RDIL2 reads a horizontal line of pixels from the
0037 C COMTAL, and places the values in INTEGER array.
0038 C WRIL2 &WRIL2 %WRIL2 writes a horizontal line of pixels from an
0039 C integer array to a COMTAL image.
0040 C HILO &HILO %HILO passes through a COMTAL image and returns the
0041 C highest and lowest pixel value found.
0042 C IFIX ----- FORTRAN IV intrinsic function; converts from
0043 C real to integer.
0044 C FLOAT ----- FORTRAN IV intrinsic function; converts from
```

```

0045 C integer to real.
0046 C
0047 C***WRITTEN BY:
0048 C
0049 C The code on which this subprogram is based was written by
0050 C NETTIE D. FAULCON. July, 1983. This subprogram was written by
0051 C Keith Miller, July, 1984, with the support of a NASA-ASEE
0052 C summer fellowship.
0053 C
0054 C***REVISION HISTORY:
0055 C
0056 C
0057 C***LOCAL VARIABLES:
0058 C
0059 C INTEGER IBUF(512) ! holds pixel values read/written to/from COMTAL
0060 C INTEGER HIGH, LOW ! high and low pixel values in the IMAGE
0061 C INTEGER PXHI, PXLO ! limits of pixel values.
0062 C INTEGER LNLO, LNHI ! limits on COMTAL line numbers
0063 C INTEGER ROW, COL ! indices for COMTAL images and the arrays.
0064 C INTEGER TERM ! logical unit for terminal output
0065 C REAL FACTOR ! the scaling factor for doing the contrast spread
0066 C
0067 C***INITIALIZATIONS:
0068 C
0069 C DATA PXLO/0/, PXHI/255/
0070 C DATA LNLO/0/, LNHI/511/
0071 C DATA TERM/1/
0072 C
0073 C***PROCESSING
0074 C
0075 C CALL HILO(HIGH, LOW, IMAGE) ! get highest and lowest pixel values
0076 C ! in the image
0077 C
0078 C IF (HIGH .LE. LOW) GOTO 9999 ! no processing required
0079 C WRITE(TERM, 3501) HIGH, LOW
0080 C 3501 FORMAT(' SPRED DIAGNOSTIC. HIGH AND LOW ARE ', 2I4, '.')
0081 C IF ((HIGH .EQ. PXHI) .AND. (LOW .EQ. PXLO)) GOTO 9999 ! no processing
0082 C
0083 C FACTOR = 255.0 / FLOAT(HIGH-LOW)
0084 C
0085 C During the second pass, replace each pixel in the image with a
0086 C new pixel that has been spread linearly according to HIGH and LOW
0087 C
0088 C DO 4000 ROW = 1,512
0089 C CALL RDIL2(IBUF, IMAGE, ROW)
0090 C DO 3000 COL = 1,512

```

```
0091          IBUF(COL) = IFIX(FLOAT(IBUF(COL)-LOW) * FACTOR)
0092 3000      CONTINUE
0093          CALL WRIL2(IMAGE,ROW,IBUF)
0094 4000      CONTINUE
0095 C
0096 C
0097 9999      RETURN
0098          END
```



&SUBI2 T=00004 IS ON CR00021 USING 00024 BLKS R=0000

```
0001 CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
0002 SUBROUTINE SUBI2(C, A, B)
0003 CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
0004 C
0005 C***PARAMETER DECLARATIONS:
0006 C
0007 INTEGER C ! the COMTAL image into which the difference between
0008 C ! image A and image B is placed by SUBI2 (C = A - B)
0009 INTEGER A, B ! the images whose difference is taken
0010 C
0011 C***INTRODUCTION:
0012 C
0013 C The subroutine "SUBtract Images #2" takes a pixel by pixel difference
0014 C of images A and B and places the resulting image into image C.
0015 C The truecolor image B is used in the processing of SUBI2, and will
0016 C be left as the combination of C, A, and B for red, green, and blue
0017 C respectively.
0018 C
0019 C Unlike SUBIM, which does no scaling or offsetting, SUBI2 offsets the
0020 C result of the subtraction by adding 128 to each pixel difference.
0021 C Thus, a 128 pixel value in image C after the call means that
0022 C the true value of the difference was 0. This offset can be handy when
0023 C many of the values of the difference are less than 0. After the offset
0024 C is added, any pixel values less than 0 are set to 0.
0025 C
0026 C***LANGUAGE:
0027 C
0028 C FORTRAN 77, the HP-1000 version for RTE-6/VM.
0029 C
0030 C***LIMITATIONS:
0031 C
0032 C The truecolor B image is destroyed during this operation. C is
0033 C obviously destroyed. This subroutine is accomplished using COMTAL
0034 C commands that exploit the pipeline processors. Because of this, the
0035 C processing steps are obscure. For example, there is no motivation
0036 C outside the COMTAL instructions for making the combination of C, A, and B
0037 C a color image. Readers should be aware of these obscurities before trying
0038 C to understand the code.
0039 C
0040 C If any offset difference is less than 0, the pixel value is set to 0.
0041 C
0042 C If any image number is out of range, an error message is printed and
0043 C no further processing takes place.
0044 C This subroutine assumes that 0 is not a legal image for the COMTAL
```

```

0045 C configuration.
0046 C
0047 C***SUBPROGRAMS CALLED:
0048 C
0049 C name source load remarks
0050 C -----
0051 C CMMND &CMMND %CMMND Sends a command to the COMTAL as if the
0052 C command were sent to the Keyboard
0053 C RANGE &RANGE %RANGE logical function that determines if the 1st
0054 C parameter is within the range of the 2nd & 3rd.
0055 C DIGIT &DIGIT %DIGIT character*1 function that returns '0'-'9'
0056 C according to integer input 0-9.
0057 C
0058 C***WRITTEN BY:
0059 C
0060 C KEITH MILLER, July, 1984, with the support of a NASA-ASEE
0061 C summer fellowship.
0062 C
0063 C***REVISION HISTORY:
0064 C
0065 C
0066 C***LOCAL VARIABLES:
0067 C
0068 C INTEGER IBUF(128) ! a buffer for passing commands to COMTAL
0069 C CHARACTER*255 CBUF ! character overlay for IBUF
0070 C EQUIVALENCE (IBUF,CBUF)
0071 C
0072 C LOGICAL RANGE ! function that determines if 1st parameter
0073 C ! is within 2nd and 3rd parameter
0074 C
0075 C INTEGER IMLO, IMHI ! limits on COMTAL image numbers
0076 C INTEGER TERM
0077 C CHARACTER*1 DIGIT ! returns a single digit character '0'
0078 C ! to '9' for integer input 0-9.
0079 C
0080 C***INITIALIZATIONS:
0081 C
0082 C DATA IMLO/1/, IMHI/4/
0083 C DATA TERM/1/
0084 C
0085 C***PROCESSING
0086 C
0087 C IF (.NOT.(RANGE(A,IMLO,IMHI))) GOTO 8001 ! error return
0088 C IF (.NOT.(RANGE(B,IMLO,IMHI))) GOTO 8101 ! error return
0089 C IF (.NOT.(RANGE(C,IMLO,IMHI))) GOTO 8201 ! error return
0090 C

```

```

0091 C      The following character string sends a series of keyboard
0092 C      commands to the COMTAL. In the comments below, each command
0093 C      is explained. The notation #X where X is either A, B, or C
0094 C      stands for the single character that corresponds to the single
0095 C      digit number associated with the parameter X.
0096 C      In this notation, letters in caps were entered into CBUF, and
0097 C      lower case letters are the full commands filled in by the COMTAL.
0098 C      NOTE: this code assumes that the digit 0 is NOT a legal value for
0099 C      the parameters A, B, and C.
0100 C      The '$' separates COMTAL commands.
0101 C
0102 C      CBUF =
0103 C      1  'UN I 8 $'//
0104 C          UNassign Image 8 ! just in case 8 is already assigned.
0105 C      2  'AS T 8 '//DIGIT(C)//' '//DIGIT(A)//' '//DIGIT(B)//' $'//
0106 C          ASsign Truecolor image 8 red #C green #A blue #B
0107 C      3  'D I 8 $'//
0108 C          Display Image 8
0109 C      4  'SE COM G - B $'//
0110 C          SE\ COMbine <Green - Blue> ! +128 offset and no scaling is the
0111 C          ! default.
0112 C      5  'A COM $'//
0113 C          Add COMbine
0114 C      6  'I '//DIGIT(C)//' D R $'//
0115 C          Image #C = Displayed Image Red ! Red is arbitrary, since difference
0116 C          ! of images is monochrome
0117 C      7  'D I '//DIGIT(C)//' $'//
0118 C          Display Image #C
0119 C      8  'SU COM '
0120 C          SUbtract COMbine.
0121 C          CALL CMMND(IBUF, 74)
0122 C          RETURN
0123 C
0124 C ***ERROR RETURNS
0125 C
0126 C      8001 WRITE(TERM, 8003) A
0127 C      8003 FORMAT(' THE 2ND IMAGE PARAMETER,', I3, ', IS OUT OF RANGE.')
0128 C          GOTO 8900
0129 C
0130 C      8101 WRITE(TERM, 8103) B
0131 C      8103 FORMAT(' THE 3RD IMAGE PARAMETER,', I3, ', IS OUT OF RANGE.')
0132 C          GOTO 8900
0133 C
0134 C      8201 WRITE(TERM, 8203) C
0135 C      8203 FORMAT(' THE 1ST IMAGE PARAMETER,', I3, ', IS OUT OF RANGE.')
0136 C          GOTO 8900

```

```
0137 C
0138 8900 WRITE(TERM, 8901)
0139 8901 FORMAT(' SUBI2 RETURNS WITHOUT FURTHER PROCESSING.')
```

```
0140 RETURN
```

```
0141 END
```

```
0142
```

&SUBIM T=00004 IS ON CR00021 USING 00004 BLKS R=0000

```
0001 CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
0002 SUBROUTINE SUBIM(C, A, B)
0003 CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
0004 C
0005 C***PARAMETER DECLARATIONS:
0006 C
0007 INTEGER C ! the COMTAL image into which the difference between
0008 C ! image A and image B is placed by SUBIM (C = A - B)
0009 INTEGER A, B ! the images whose difference is taken(C = A - B)
0010 C
0011 C***INTRODUCTION:
0012 C
0013 C The subroutine "SUBtract IMages" takes a pixel by pixel difference
0014 C of images A and B and places the resulting image into image C.
0015 C The truecolor image B is used in the processing of SUBIM, and will
0016 C be left as the combination of C, A, and B for red, green, and blue
0017 C respectively.
0018 C
0019 C***LANGUAGE:
0020 C
0021 C FORTRAN 77, the HP-1000 version for RTE-6/VM.
0022 C
0023 C***LIMITATIONS:
0024 C
0025 C The truecolor B image is destroyed during this operation. C is
0026 C obviously destroyed. This subroutine is accomplished using COMTAL
0027 C commands that exploit the pipeline processors. Because of this, the
0028 C processing steps are obscure. For example, there is no motivation
0029 C outside the COMTAL instructions for making the combination of C, A, and B
0030 C a color image. Readers should be aware of these obscurities before trying
0031 C to understand the code.
0032 C
0033 C The 3 images C, A, and B must be distinct.
0034 C
0035 C If any difference is less than 0, the pixel value is set to 0.
0036 C
0037 C SUBIM does no scaling or offsetting. SUBI2 does an automatic
0038 C scale and offset.
0039 C
0040 C If any image number is out of range, an error message is printed and
0041 C no further processing takes place.
0042 C This subroutine assumes that 0 is not a legal image for the COMTAL
0043 C configuration.
0044 C
```

```

0045 C***SUBPROGRAMS CALLED:
0046 C   name      source  load  remarks
0047 C   -----  -----  -----  -----
0048 C   CMMND    &CMMND  %CMMND  Sends a command to the COMTAL as if the
0049 C           command were sent to the keyboard
0050 C   RANGE    &RANGE  %RANGE  logical function that determines if the 1st
0051 C           parameter is within the range of the 2nd & 3rd.
0052 C   DIGIT    &DIGIT  %DIGIT  character*1 function that returns '0'-'9'
0053 C           according to integer input 0-9.
0054 C
0055 C***WRITTEN BY:
0056 C
0057 C   KEITH MILLER, July, 1984, with the support of a NASA-ASEE
0058 C   summer fellowship.
0059 C
0060 C***REVISION HISTORY:
0061 C
0062 C***LOCAL VARIABLES:
0063 C
0064 C           INTEGER      IBUF(128)  ! a buffer for passing commands to COMTAL
0065 C           CHARACTER*255 CBUF      ! character overlay for IBUF
0066 C           EQUIVALENCE (IBUF,CBUF)
0067 C
0068 C           LOGICAL      RANGE      ! function that determines if 1st parameter
0069 C           is within 2nd and 3rd parameter
0070 C
0071 C           INTEGER      IMLO, IMHI ! limits on COMTAL image numbers
0072 C           INTEGER      TERM
0073 C           CHARACTER*1  DIGIT      ! returns a single digit character '0'
0074 C           ! to '9' for integer input 0-9.
0075 C
0076 C***INITIALIZATIONS:
0077 C
0078 C           DATA      IMLO/1/, IMHI/4/
0079 C           DATA      TERM/1/
0080 C
0081 C***PROCESSING
0082 C
0083 C           IF (.NOT.(RANGE(A,IMLO,IMHI))) GOTO 8001 ! error return
0084 C           IF (.NOT.(RANGE(B,IMLO,IMHI))) GOTO 8101 ! error return
0085 C           IF (.NOT.(RANGE(C,IMLO,IMHI))) GOTO 8201 ! error return
0086 C
0087 C           The following character string sends a series of keyboard
0088 C           commands to the COMTAL. In the comments below, each command
0089 C           is explained. The notation #X where X is either A, B, or C
0090 C           stands for the single character that corresponds to the single

```

```

0091 C      digit number associated with the parameter X.
0092 C      In this notation, letters in caps were entered into CBUF, and
0093 C      lower case letters are the full commands filled in by the COMTAL
0094 C      NOTE: this code assumes that the digit 0 is NOT a legal value for
0095 C      the parameters A, B, and C.
0096 C      The '$' separates COMTAL commands.
0097 C
0098 C      CBUF =
0099 C      1  'UN I 8 $'//
0100 C          UNassign Image 8 ! just in case 8 is already assigned.
0101 C      2  'AS T 8 '//DIGIT(C)//' '//DIGIT(A)//' '//DIGIT(B)//' '$'//
0102 C          ASsign Truecolor image 8 red #C blue #A green #B
0103 C      3  'D I 8 $'//
0104 C          Display Image 8
0105 C      4  'SE COM G - B + 0 $'//
0106 C          SEt COMbine <Green - Blue> offset 0! no scaling by default.
0107 C      5  'A COM $'//
0108 C          Add COMbine
0109 C      6  'I '//DIGIT(C)//' D R $'//
0110 C          Image #C = Displayed Image Red ! Red is arbitrary, since difference
0111 C                               ! of images is monochrome
0112 C      7  'D I '//DIGIT(C)//' '$'//
0113 C          Display Image #C
0114 C      8  'SU COM '
0115 C          SUbtract COMbine.
0116 C          CALL CMMND(IBUF, 77)
0117 C          RETURN
0118 C
0119 C      ***ERROR RETURNS
0120 C
0121 C      8001 WRITE(TERM, 8003) A
0122 C      8003 FORMAT(' THE 2ND IMAGE PARAMETER,', I3, ', IS OUT OF RANGE.')
```

```

0123 C          GOTO 8900
0124 C
0125 C      8101 WRITE(TERM, 8103) B
0126 C      8103 FORMAT(' THE 3RD IMAGE PARAMETER,', I3, ', IS OUT OF RANGE.')
```

```

0127 C          GOTO 8900
0128 C
0129 C      8201 WRITE(TERM, 8203) C
0130 C      8203 FORMAT(' THE 1ST IMAGE PARAMETER,', I3, ', IS OUT OF RANGE.')
```

```

0131 C          GOTO 8900
0132 C
0133 C      8900 WRITE(TERM, 8901)
0134 C      8901 FORMAT(' SUBIM RETURNS WITHOUT FURTHER PROCESSING.')
```

```

0135 C          RETURN
0136 C          END

```

&THRSH T=00004 IS ON CR00021 USING 00018 BLKS R=0000

```
0001 CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
0002 SUBROUTINE THRSH(OUTIMG, INIMG, THRESH)
0003 CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
0004 C
0005 C***PARAMETER DECLARATIONS:
0006 C
0007 INTEGER OUTIMG! the COMTAL image number for the thresholded image.
0008 INTEGER INIM ! the COMTAL image # for the image to be thresholded.
0009 INTEGER THRESH! the threshold pixel value; < threshold -> 0,
0010 C ! >= threshold -> 255.
0011 C
0012 C***INTRODUCTION:
0013 C
0014 C The subroutine THRESHold creates a new image OUTIMG by examining the
0015 C pixel values in INIMG. If an INIMG pixel value is < THRESH, then the
0016 C corresponding pixel value in OUTIMG is 0. If the INIMG pixel value
0017 C is >= THRESH, the corresponding pixel value in OUTIMG is 255.
0018 C OUTIMG and INIMG need not be distinct.
0019 C
0020 C***LANGUAGE:
0021 C
0022 C FORTRAN 77, the HP-1000 version for RTE-6/VM.
0023 C
0024 C***LIMITATIONS:
0025 C
0026 C OUTIMG and INIMG must be COMTAL memory planes (1-4). If not, an
0027 C error message is printed and no thresholding takes place.
0028 C THRESH must be a value between 0-255, or a message is printed instead
0029 C of any thresholding.
0030 C
0031 C***SUBPROGRAMS CALLED:
0032 C
0033 C name source load remarks
0034 C -----
0035 C RANGE &RANGE %RANGE logical function that determines if its 1st
0036 C argument is within the 2nd and 3rd inclusive.
0037 C RDIL2 &RDIL2 %RDIL2 reads a horizontal line of COMTAL pixels;
0038 C each pixel value put into its own integer.
0039 C WRIL2 &WRIL2 %WRIL2 writes a horizontal line of pixels to a COMTAL
0040 C image from an integer array; 1 pixel/integer.
0041 C
0042 C***WRITTEN BY:
0043 C
0044 C The code on which this subprogram is based was written by
```



```

0045 C NETTIE D. FAULCON, July, 1983. This subprogram was written by
0046 C KEITH MILLER, July, 1984, with the support of a NASA-ASEE
0047 C summer fellowship.
0048 C
0049 C***REVISION HISTORY:
0050 C
0051 C
0052 C***LOCAL VARIABLES:
0053 C
0054 C     INTEGER IBUF(512)    ! buffer for read/write of horizontal line of
0055 C                          ! COMTAL pixels.
0056 C     INTEGER LINE        ! loop index that keeps track of the line
0057 C                          ! in INIMG and OUTIMG being processed.
0058 C     INTEGER PXCNT       ! loop index that counts pixels in a line.
0059 C     LOGICAL RANGE       ! function that determines if its 1st argument
0060 C                          ! lies within its 2nd and 3rd inclusive.
0061 C     INTEGER IMLO, IMHI  ! range of legal COMTAL image numbers.
0062 C     INTEGER PXLO, PXHI  ! range of legal COMTAL pixel values.
0063 C     INTEGER LNLO, LNHI  ! range of legal COMTAL line numbers.
0064 C     INTEGER ARALO, ARAHI ! range of array holding a line of pixels;
0065 C                          ! (+1 of LNLO and LNHI).
0066 C     INTEGER TERM        ! logical unit for terminal output.
0067 C
0068 C***INITIALIZATIONS:
0069 C
0070 C     DATA  IMLO/1/, IMHI/4/
0071 C     DATA  PXLO/0/, PXHI/255/
0072 C     DATA  LNLO/0/, LNHI/511/
0073 C     DATA  ARALO/1/, ARAHI/512/
0074 C     DATA  TERM/1/
0075 C
0076 C***PROCESSING:
0077 C
0078 C     IF (.NOT.(RANGE(OUTIMG, IMLO, IMHI))) GOTO 8001 ! error return
0079 C     IF (.NOT.(RANGE(INIMG , IMLO, IMHI))) GOTO 8101 ! error return
0080 C     IF (.NOT.(RANGE(THRESH, PXLO, PXHI))) GOTO 8201 ! error return
0081 C
0082 C     DO 2000 LINE = LNLO, LNHI
0083 C         CALL RDIL2(IBUF, INIMG, LINE)
0084 C         DO 1000 PXCNT = ARALO, ARAHI
0085 C             IF (IBUF(PXCNT) .LT. THRESH) GOTO 500
0086 C             ELSE...
0087 C                 IBUF(PXCNT) = 255
0088 C                 GOTO 1000
0089 C             THEN...
0090 C     500         IBUF(PXCNT) = 0

```

```

0091 1000 CONTINUE
0092 CALL WRIL2(OUTIMG, LINE, IBUF)
0093 2000 CONTINUE
0094 RETURN
0095 C
0096 C***ERROR RETURNS:
0097 C
0098 8001 WRITE(TERM, 8003) OUTIMG, IMLO, IMHI
0099 8003 FORMAT(' THE OUTPUT IMAGE NUMBER,',I5,', IS OUT OF RANGE:',I4)
0100 GOTO 8900
0101 C
0102 8101 WRITE(TERM, 8103) INIMG, IMLO, IMHI
0103 8103 FORMAT(' THE INPUT IMAGE NUMBER,',I5,', IS OUT OF RANGE:',I4)
0104 GOTO 8900
0105 C
0106 8201 WRITE(TERM, 8203) THRESH, PXLO, PXHI
0107 8203 FORMAT(' THE THRESHOLD VALUE,',I5,', IS OUT OF RANGE:',I4)
0108 GOTO 8900
0109 C
0110 8900 WRITE(TERM, 8901)
0111 8901 FORMAT(' THRSH FAILS. OUTIMG NOT CHANGED.')
0112 RETURN
0113 END

```

&TSTI1 T=00004 IS ON CR00021 USING 00004 BLKS R=0000

```
0001 CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
0002 SUBROUTINE TSTI1(WHICH)
0003 CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
0004 C
0005 C***PARAMETER DECLARATIONS:
0006 C
0007 INTEGER WHICH ! designates the COMTAL image in which the test
0008 C ! image is to be generated
0009 C
0010 C***INTRODUCTION:
0011 C
0012 C The subroutine TeST Image 1 generates a distinctive pattern in one
0013 C of the COMTAL image memories. The pattern is generated with ascending
0014 C pixel values as you move to the right and down in the image. However,
0015 C when 255 is reached in either the x or y directions, the pixel values
0016 C restart at 0. This gives a slash across the screen appearance at the
0017 C discontinuity, and a gradual change in grey scale elsewhere.
0018 C
0019 C***LANGUAGE:
0020 C
0021 C FORTRAN 77, the HP-1000 version for RTE-6/VM.
0022 C
0023 C***LIMITATIONS:
0024 C
0025 C The test pattern developed here is always the same. An enhancement
0026 C might be to vary the repetition of the pixel values (here, each pixel
0027 C is identical to one of its horizontal neighbors and one of its vertical
0028 C neighbors).
0029 C
0030 C***SUBPROGRAMS CALLED:
0031 C
0032 C name source load remarks
0033 C -----
0034 C WRILN &WRILN %WRILN given an integer buffer of at least 512 bytes,
0035 C WRILN writes a horizontal line of pixels to a
0036 C designated COMTAL image.
0037 C
0038 C***WRITTEN BY:
0039 C
0040 C The code on which this subprogram is based was written by
0041 C NETTIE D. FAULCON, July, 1983. This subprogram was written by
0042 C KEITH MILLER, July, 1984, with the support of a NASA-ASEE
0043 C summer fellowship.
0044 C
```

```

0045 C***REVISION HISTORY:
0046 C
0047 C
0048 C***LOCAL VARIABLES:
0049 C
0050     INTEGER BIGBUF(512) ! 1024 bytes of pixel values
0051     INTEGER I           ! pointer into BIGBUF
0052     INTEGER ROW        ! designates a COMTAL image row
0053 C
0054 C***INITIALIZATIONS:
0055 C
0056     DO 1000 I = 0,255
0057         BIGBUF(I+1) = I * 257      ! I*256 numbers the high byte
0058 C                                 ! I*1  numbers the low byte
0059         BIGBUF(I+257)= BIGBUF(I+1) ! facilitates the wraparound effect
0060     1000 CONTINUE
0061 C
0062 C***PROCESSING
0063 C
0064     DO 2000 ROW = 0,511
0065 C
0066 C         by starting WRILN at different places in BIGBUF, the
0067 C         desired wrap-around effect is acheived.
0068 C
0069         CALL WRILN( WHICH, ROW, BIGBUF((ROW/2)+1) )
0070     2000 CONTINUE
0071 C
0072     RETURN
0073     END

```

&TV2C4 T=00004 IS ON CR00021 USING 00006 BLKS R=0000

```
0001 CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
0002 SUBROUTINE TV2C4
0003 CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
0004 C
0005 C***PARAMETER DECLARATIONS:
0006 C
0007 C None.
0008 C
0009 C***INTRODUCTION:
0010 C
0011 C The subroutine "TV to CoMtal image transfer #4" digitizes four
0012 C copies of the current TV image, and averages them together into a
0013 C single image stored in COMTAL image #1. COMTAL images 1, 2, and 3
0014 C are used for storage. Image 5 must be set to the video camera image
0015 C before TV2C4 is called.
0016 C
0017 C***LANGUAGE:
0018 C
0019 C FORTRAN 77, the HP-1000 version for RTE-6/VM.
0020 C
0021 C***LIMITATIONS:
0022 C
0023 C Images 1, 2, 3, and 4 are changed by this routine.
0024 C
0025 C This subroutine assumes that image 5 has been set to the video
0026 C camera previous to the call. If 5 is not set to video, the COMTAL
0027 C hangs up.
0028 C
0029 C
0030 C***SUBPROGRAMS CALLED:
0031 C
0032 C name source load remarks
0033 C -----
0034 C RANGE &RANGE %RANGE logical function that determines if its 1st
0035 C argument lies within 2nd & 3rd argument, inclusive.
0036 C TV2CM &TV2CM %TV2CM digitizes the camera associated with COMTAL image
0037 C 5 into the 1st argument, an image memory.
0038 C ADDI2 &ADDI2 %ADDI2 adds two images, pixel by pixel, and divides the
0039 C sums by last argument, to produce a new image.
0040 C
0041 C***WRITTEN BY:
0042 C
0043 C The code on which this subprogram is based was written by
0044 C NETTIE D. FAULCON, July, 1983. This subprogram was written by
```

```

0045 C KEITH MILLER, July, 1984, with the support of a NASA-ASEE
0046 C summer fellowship.
0047 C
0048 C***REVISION HISTORY:
0049 C
0050 C
0051 C***LOCAL VARIABLES:
0052 C
0053 LOGICAL RANGE ! function that determines of its 1st argument
0054 C ! is within its 2nd & 3rd, inclusive.
0055 INTEGER TVIMAG ! COMTAL image number associated with the video
0056 C ! camera.
0057 C
0058 C***INITIALIZATIONS:
0059 C
0060 DATA TVIMAG/5/ ! arbitrary convention used at our lab.
0061 DATA TERM/1/
0062 C
0063 C***PROCESSING:
0064 C
0065 CALL DSPVD
0066 CALL TV2CM(1)
0067 CALL TV2CM(2)
0068 CALL ADDI2(3,1,2,2) ! put average of images 1&2 into image 1
0069 C
0070 CALL TV2CM(1)
0071 CALL TV2CM(2)
0072 CALL ADDI2(4,1,2,2) ! put average of images 2&3 into image 2
0073 C
0074 CALL ADDI2(1,3,4,2) ! put average of images 1&2 into image 1
0075 C
0076 CALL DSPBW(1)
0077 C
0078 RETURN
0079 END

```

&TV2CM T=0004 IS ON CR00021 USING 00004 BLKS R=0000

```
0001 CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
0002 SUBROUTINE TV2CM(CMIMAG)
0003 CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
0004 C
0005 C***PARAMETER DECLARATIONS:
0006 C
0007 INTEGER CMIMAG ! "COMtal IMAGE" number where the digitized image
0008 C ! is to be stored.
0009 C
0010 C***INTRODUCTION:
0011 C
0012 C The subroutine TV2CM digitizes
0013 C a "snapshot" of the current TV image into the COMTAL image
0014 C number CMIMAG. Previous to the TV2CM call, image 5 must be set
0015 C to the video camera.
0016 C
0017 C***LANGUAGE:
0018 C
0019 C FORTRAN 77, the HP-1000 version for RTE-6/VM.
0020 C
0021 C***LIMITATIONS:
0022 C
0023 C The video camera must be associated with TVIMAG before TV2CM is
0024 C called. TVIMAGE is 5 as an arbitrary convention in this lab.
0025 C
0026 C CMIMAG must identify a COMTAL image memory (IMLO to IMHI).
0027 C
0028 C***SUBPROGRAMS CALLED:
0029 C
0030 C name source load remarks
0031 C -----
0032 C CMMND &CMMND %CMMND sends character strings to COMTAL which
0033 C interprets them as COMTAL keyboard commands.
0034 C DSPBW &DSPBW %DSPBW displays the indicated monochrome COMTAL image.
0035 C RANGE &RANGE %RANGE logical function that determines if 1st
0036 C argument is within the 2nd & 3rd, inclusive.
0037 C DIGIT &DIGIT %DIGIT character%1 function that returns '0'..'9'
0038 C according to integer input 0..9.
0039 C
0040 C***WRITTEN BY:
0041 C
0042 C The code on which this subprogram is based was written by
0043 C NETTIE D. FAULCON, July, 1983. This subprogram was written by
0044 C KEITH MILLER, July, 1984, with the support of a NASA-ASEE
```

```

0045 C   summer fellowship.
0046 C
0047 C***REVISION HISTORY:
0048 C
0049 C
0050 C***LOCAL VARIABLES:
0051 C
0052     INTEGER      IMLO, IMHI ! limits on COMTAL image memory numbers.
0053     INTEGER      TVIMAGE    ! the COMTAL image # associated with the
0054 C              ! video camera.
0055     LOGICAL      RANGE      ! function that determines if the 1st argument
0056 C              ! is within the 2nd and 3rd arguments, inclusive.
0057     CHARACTER*1  DIGIT      ! function that returns '0'..'9' according
0058 C              ! to integer input 0..9.
0059     INTEGER      TERM       ! logical unit for terminal output.
0060 C
0061     INTEGER      IBUF(128) ! buffer for CMMND COMTAL command strings.
0062     CHARACTER*255 CBUF      ! overlays IBUF.
0063     EQUIVALENCE  (IBUF,CBUF)
0064     INTEGER      TVIMAG     ! COMTAL image # associated with video image;
0065 C              ! arbitrarily set to 5 in this lab.
0066 C
0067 C***INITIALIZATIONS:
0068 C
0069     DATA      IMLO/1/, IMHI/4/
0070     DATA      TVIMAG/5/      ! arbitrary convention for our lab.
0071     DATA      TERM/1/
0072 C
0073 C***PROCESSING:
0074 C
0075     IF (.NOT.(RANGE(CMIMAG,IMLO,IMHI))) GOTO 8001 ! error return
0076 C
0077 C
0078 C   let #C and #V be the digits associated with CMIMAG and TVIMAG;
0079 C   then the following CMMND string is expanded by the COMTAL into:
0080 C       Display Image #V; Image #C = Displayed image Red
0081 C
0082     CBUF = 'D I '//DIGIT(TVIMAG)//' SI '//DIGIT(CMIMAG)//' D R '
0083     CALL CMMND(IBUF, 15)
0084     CALL DSPBW(CMIMAG)
0085     RETURN
0086 C
0087 C***ERROR RETURNS:
0088 C
0089     8001 WRITE(TERM,8003) CMIMAG, IMLO, IMHI
0090     8003 FORMAT(' THE COMTAL IMAGE NUMBER,',I4,', IS OUT OF RANGE:',I2)

```



```
0091 - GOTO 8900
0092 C
0093 C
0094 8900 WRITE(TERM,8901)
0095 8901 FORMAT(' TV2CM FAILS. NO DIGITIZING TAKES PLACE.')
```

```
0096 C
0097 RETURN
0098 END
```

&WAIT T=0004 IS ON CR00021 USING 00002 BLKS R=0000

```
0001 CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
0002 SUBROUTINE WAIT
0003 CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
0004 C
0005 C***PARAMETERS: NONE
0006 C
0007 C
0008 C***INTRODUCTION:
0009 C
0010 C WAIT pauses until <CR> is pressed on the HP keyboard.
0011 C
0012 C***LANGUAGE:
0013 C
0014 C FORTRAN 77, the HP-1000 version for RTE-6/VM.
0015 C
0016 C***SUBPROGRAMS CALLED: NONE
0017 C
0018 C
0019 C***WRITTEN BY:
0020 C
0021 C KEITH MILLER, NASA-Langley ASEE fellow, 1984
0022 C
0023 C***REVISION HISTORY: NONE
0024 C
0025 C
0026 C***LOCAL VARIABLES:
0027 C
0028 C INTEGER TERM ! logical unit of the terminal
0029 C INTEGER IDUMMY ! facilitates the read that forces a pause
0030 C
0031 C***INITIALIZATIONS:
0032 C
0033 C DATA TERM/1/
0034 C
0035 C***PROCESSING:
0036 C
0037 C WRITE(TERM, 1001)
0038 C 1001 FORMAT(' PUSH <CR> TO CONTINUE.')
0039 C READ(TERM,2001) IDUMMY
0040 C 2001 FORMAT(I2)
0041 C RETURN
0042 C END
0043
0044
```

&WIPGR T=0004 IS ON CR0021 USING 0012 BLKS R=0000

```
0001 CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
0002          SUBROUTINE WIPGR(GRNUM)
0003 CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
0004 C
0005 C***PARAMETER DECLARATIONS:
0006 C
0007          INTEGER   GRNUM ! a number, 1-4, designating a COMTAL graphics plane.
0008 C
0009 C***INTRODUCTION:
0010 C
0011 C   "WIPe GRaphics" is a subroutine that allows the caller
0012 C   to send an "un-display" command to the COMTAL from an HP program.
0013 C   The call can turn off a graphics plane, number 1, 2, 3,
0014 C   or 4.
0015 C
0016 C***LIMITATIONS:
0017 C
0018 C   It is OK to call WIPGR repeatedly without an intervening DSPGR.
0019 C   The extra calls have no effect, but they don't hang up the COMTAL.
0020 C
0021 C***SUBPROGRAMS CALLED:
0022 C
0023 C   name      source load   remarks
0024 C   -----
0025 C   RANGE     &RANGE %RANGE logical function that determines if the
0026 C               first parameter is within the bounds defined
0027 C               by the second and third parameter (inclusive).
0028 C   CMMN2     &CMMN2 %CMMN2 sends a constant string to the COMTAL as if
0029 C               the string were typed on the COMTAL keyboard.
0030 C   DIGIT     &DIGIT %DIGIT character*1 function that returns '0'-'9'
0031 C               according to integer input 0-9.
0032 C
0033 C***WRITTEN BY:
0034 C
0035 C   The code on which this subprogram is based was written by
0036 C   NETTIE D. FAULCON, July, 1983. This modification is by
0037 C   KEITH MILLER,      June, 1984.
0038 C
0039 C***REVISION HISTORY:
0040 C
0041 C
0042 C***LOCAL VARIABLES:
0043 C
0044          LOGICAL     RANGE      ! a function for determining if an integer is
```

```

0045 C                                ! within a certain range.
0046     INTEGER      TERM           ! the logical unit number for the terminal.
0047     INTEGER      IDUMMY          ! fills the place of an unused EXEC parameter.
0048     INTEGER      GRLO,GRHI       ! limits on a graphics plane number.
0049     CHARACTER*1  DIGIT           ! function that returns '0'-'9' for input
0050 C                                ! integers 0-9.
0051 C
0052 C***INITIALIZATIONS:
0053 C
0054     DATA      TERM/1/
0055     DATA      GRLO/1/,GRHI/4/
0056 C
0057 C***PROCESSING
0058 C
0059     IF (.NOT.(RANGE(GRNUM,GRLO,GRHI))) GOTO 8001 ! error return
0060 C
0061 C     "SUBtract GRaphics #GRNUM", where #GRNUM stands for the digit
0062 C           corresponding to GRNUM value.
0063 C
0064     CALL CMMN2('SUB GR '//DIGIT(GRNUM))
0065     RETURN
0066 C
0067 C***ERROR RETURN:
0068 C
0069     8001 WRITE(TERM, 8003) GRNUM, GRLO, GRHI
0070     8003 FORMAT( ' THE GRAPHICS NUMBER,',I3,', OUT OF RANGE:',I2I4)
0071 C
0072     8900 WRITE(TERM, 8901)
0073     8901 FORMAT( ' DSPGR fails. No action taken on command.' )
0074     RETURN
0075     END

```

&WRGLN T=0004 IS ON CR0021 USING 0018 BLKS R=0000

```
0001 CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
0002 SUBROUTINE WRGLN(GRAPH, LINE, ONOFFS)
0003 CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
0004 C
0005 C***PARAMETER DECLARATIONS:
0006 C
0007 INTEGER GRAPH ! COMTAL image graphics # to be written to.
0008 INTEGER LINE ! which horizontal line to be written to;
0009 C ! lines numbered from 1 (screen top) to 512.
0010 INTEGER ONOFFS(32) ! 16 bits per integer, 512 bits in a line.
0011 C
0012 C***INTRODUCTION:
0013 C
0014 C This subroutine WRites a Graphics LiNe to the COMTAL. The line of
0015 C bits is coded as 1 for on and 0 for off. Each bit in ONOFFS must be
0016 C set by the caller of WRGLN.
0017 C
0018 C***LANGUAGE:
0019 C
0020 C FORTRAN 77, the HP-1000 version for RTE-6/VM.
0021 C
0022 C***LIMITATIONS:
0023 C
0024 C If GRAPH or LINE is out of range, an error message is printed
0025 C and no transfer takes place.
0026 C
0027 C***SUBPROGRAMS CALLED:
0028 C
0029 C name source load remarks
0030 C -----
0031 C RANGE &RANGE %RANGE logical function that determines if its
0032 C first parameter is within the last two parameters.
0033 C
0034 C***WRITTEN BY:
0035 C
0036 C The code on which this subprogram is based was written by
0037 C NETTIE D. FAULCON, July, 1983. This modification is by
0038 C KEITH MILLER, June, 1984.
0039 C
0040 C***REVISION HISTORY:
0041 C
0042 C***LOCAL VARIABLES:
0043 C
0044 LOGICAL RANGE ! function that ascertains if its first parameter
```

```

0045 C          ! is between (inclusive) its last 2 parameters
0046     INTEGER TERM          ! the logical unit for terminal output
0047     INTEGER GRLO, GRHI    ! the limits on COMTAL monochrome image numbers
0048     INTEGER LNLO, LNHI    ! the limits on COMTAL image line numbers
0049 C
0050 C***INITIALIZATIONS:
0051 C
0052     DATA    TERM/1/
0053     DATA    GRLO/1/, GRHI/4/
0054     DATA    LNLO/0/, LNHI/511/
0055 C
0056 C***PROCESSING
0057 C
0058     IF (.NOT.(RANGE(GRAPH,GRLO,GRHI))) GOTO 8001 ! error return
0059     IF (.NOT.(RANGE(LINE, LNLO,LNHI))) GOTO 8101 ! error return
0060 C
0061 C     Programming note:
0062 C     The EXEC call is explained in detail in the
0063 C     HP Programmer's Reference Manual for RTE-6/VM,p.2-19ff. This
0064 C     transfer function for the COMTAL is discussed in the
0065 C     COMTAL User's Manual, Section 5.2.2.1. In the EXEC call
0066 C     that follows, the HP resident driver called DVR41 is called as
0067 C     follows: the first parameter (2) signifies a write; the
0068 C     second parameter is in two parts: 36B identifies the resident
0069 C     DVR41 driver, and 100B identifies the line transfer operation
0070 C     of that driver; the third parameter (ONOFFS) holds the data to be
0071 C     transferred, and the fourth parameter gives ONOFFS' length in words
0072 C     (32); and the final parameter is a COMTAL command code for the transfer.
0073 C
0074     CALL EXEC(2,36B+100B,ONOFFS,32,(GRAPH-1)*2048 + LINE + 512)
0075     RETURN
0076 C
0077 C***ERROR RETURNS
0078 C
0079     8001 WRITE(TERM, 8003) GRAPH, GRLO, GRHI
0080     8003 FORMAT(' GRAPHICS NUMBER,', I3, ' OUT OF RANGE:', 2I2, ',.')
0081     GOTO 8900
0082 C
0083     8101 WRITE(TERM, 8103) LINE, LNLO, LNHI
0084     8103 FORMAT(' LINE NUMBER,', I4, ' OUT OF RANGE:', 2I3, ',.')
0085     GOTO 8900
0086 C
0087     8900 WRITE(TERM, 8901)
0088     8901 FORMAT(' WRGLN FAILS. NO TRANSFER. ')
0089     RETURN
0090     END

```

&WRGPT T=00004 IS ON CR00021 USING 00018 BLKS R=0000

```
0001 CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
0002 SUBROUTINE WRGPT(GRAPH, XCOOR, YCOOR, VALUE)
0003 CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
0004 C
0005 C***PARAMETER DECLARATIONS:
0006 C
0007 INTEGER GRAPH ! COMTAL graph number to be written to.
0008 INTEGER XCOOR, YCOOR! point where new value is to be written to.
0009 INTEGER VALUE ! 0-1 graphics value to be written to graph pt.
0010 C
0011 C***INTRODUCTION:
0012 C
0013 C This subroutine WRites a Graph Point to the COMTAL. The value
0014 C of the graphics is an on/off decision, represented in the call
0015 C by an integer that must be a 1 or a 0. If the
0016 C graph number, coordinates, or on/off vdlue are out of range, an error
0017 C message is printed and no transfer takes place.
0018 C
0019 C***LANGUAGE:
0020 C
0021 C FORTRAN 77, the HP-1000 version for RTE-6/VM.
0022 C
0023 C***LIMITATIONS:
0024 C
0025 C If GRAPH, XCOOR, YCOOR, or VALUE are out of range, an error message
0026 C is printed at the terminal and no transfer takes place.
0027 C
0028 C***SUBPROGRAMS CALLED:
0029 C
0030 C name source load remarks
0031 C -----
0032 C RANGE &RANGE %RANGE logical function that determines if its
0033 C first parameter is within the last two parameters.
0034 C IBSET ----- HP FORTRAN77 intrinsic function that takes an
0035 C integer argument and returns that integer with
0036 C one bit set, according to the second argument.
0037 C IBCLR ----- HP FORTRAN77 intrinsic function that takes an
0038 C integer argument and returns that integer with
0039 C one bit cleared, according to the second argument.
0040 C
0041 C***WRITTEN BY:
0042 C
0043 C The code on which this subprogram is based was written by
0044 C NETTIE D. FAULCON, July, 1983. This modification is by
```

```

0045 C KEITH MILLER, June, 1984.
0046 C
0047 C***REVISION HISTORY:
0048 C
0049 C
0050 C***LOCAL VARIABLES:
0051 C
0052 LOGICAL RANGE ! function that ascertains if its first parameter
0053 C ! is between (inclusive) its last 2 parameters
0054 INTEGER TERM ! the logical unit for terminal output
0055 INTEGER GRLO, GRHI ! the limits on COMTAL monochrome graph numbers
0056 INTEGER LNLO, LNHI ! the limits on COMTAL graph line numbers
0057 INTEGER BTLO, BTHI ! the limits on COMTAL graphics values
0058 INTEGER BITS(32) ! a buffer to read & write a COMTAL graphics line
0059 INTEGER WORD ! which word of BITS holds the bit selected by XCOORD.
0060 INTEGER BIT ! which bit in BITS(WORD) holds the bit selected by
0061 C ! XCOORD: bits numbered 0-15, right to left.
0062 C
0063 C***INITIALIZATIONS:
0064 C
0065 DATA TERM/1/
0066 DATA GRLO/1/, GRHI/4/
0067 DATA LNLO/0/, LNHI/511/
0068 DATA BTLO/0/, BTHI/1/
0069 C
0070 C***PROCESSING
0071 C
0072 IF (.NOT.(RANGE(GRAPH,GRLO,GRHI))) GOTO 8001 ! error return
0073 IF (.NOT.(RANGE(XCOORD,LNLO,LNHI))) GOTO 8101 ! error return
0074 IF (.NOT.(RANGE(YCOORD,LNLO,LNHI))) GOTO 8201 ! error return
0075 IF (.NOT.(RANGE(VALUE,BTLO,BTHI))) GOTO 8301 ! error return
0076 C
0077 C Programming note:
0078 C The EXEC calls below are to the DVR41 driver. The first call
0079 C is identical to the call made in RDILN. The second EXEC call
0080 C is identical to the one in WRILN. See the documentation for
0081 C those subroutines for details on these calls.
0082 C
0083 C Read the COMTAL line (horizontal) that contains the point in question:
0084 C
0085 CALL EXEC(1,36B+100B,BITS,32,(GRAPH-1)*2048 + 512 + YCOORD)
0086 C
0087 C Change the single bit that has been selected:
0088 C
0089 WORD = (XCOORD/16) + 1
0090 BIT = (16*WORD) - XCOORD - 1

```



```

0091 C
0092     IF (VALUE .EQ. 0) GOTO 1000
0093 C     ELSE...VALUE .EQ. 1
0094         BITS(WORD) = IBSET(BITS(WORD),BIT)
0095         GOTO 2000
0096 C     THEN...VALUE .EQ. 0
0097     1000     BITS(WORD) = IBCLR(BITS(WORD),BIT)
0098         GOTO 2000
0099 C
0100 C     Write the graph line with one changed graphics to COMTAL
0101 C
0102     2000 CALL EXEC(2,36B+100B,BITS,32,(GRAPH-1)*2048 + 512 + YCOOR)
0103         RETURN
0104 C
0105 C***ERROR RETURNS
0106 C
0107     8001 WRITE(TERM, 8003) GRAPH, GRLO, GRHI
0108     8003 FORMAT(' GRAPH NUMBER,', I3, ' OUT OF RANGE:', 2I2, '.')
0109         GOTO 8900
0110 C
0111     8101 WRITE(TERM, 8103)XCOOR, LNLO, LNHI
0112     8103 FORMAT(' X COORDINATE,', I4, ' OUT OF RANGE:', 2I4, '.')
0113         GOTO 8900
0114 C
0115     8201 WRITE(TERM, 8203)YCOOR, LNLO, LNHI
0116     8203 FORMAT(' Y COORDINATE,', I4, ' OUT OF RANGE:', 2I4, '.')
0117         GOTO 8900
0118 C
0119     8301 WRITE(TERM, 8303) VALUE, BTLO, BTHI
0120     8303 FORMAT(' BIT VALUE,', I4, ' OUT OF RANGE:', 2I4, '.')
0121         GOTO 8900
0122 C
0123     8900 WRITE(TERM, 8901)
0124     8901 FORMAT(' WRGPT FAILS. NO TRANSFER.')
0125         RETURN
0126         END

```

&WRIL2 T=00004 IS ON CR00021 USING 00018 BLKS R=0000

```
0001 CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
0002 SUBROUTINE WRIL2(IMAGE, LINE, INTS)
0003 CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
0004 C
0005 C***PARAMETER DECLARATIONS:
0006 C
0007 INTEGER IMAGE ! COMTAL image number to be written to
0008 INTEGER LINE ! which horizontal line to be written to;
0009 C ! lines numbered from 1 (screen top) to 512.
0010 INTEGER INTS(512) ! 512 values, one integer per pixel, to
0011 C ! be transferred.
0012 C
0013 C***INTRODUCTION:
0014 C
0015 C This subroutine, WRite Image LiNe #2, writes a line of pixels to the
0016 C COMTAL. The input array INTS has a two byte integer for each pixel,
0017 C but the COMTAL only uses the lower order byte of each integer. Therefore,
0018 C WRIL2 strips off the upper byte before sending the pixels to the COMTAL.
0019 C WRIL2 is very similar to WRILN, which writes out a line of bytes.
0020 C
0021 C***LANGUAGE:
0022 C
0023 C FORTRAN 77, the HP-1000 version for RTE-6/VM.
0024 C
0025 C***LIMITATIONS:
0026 C
0027 C The INTS array must be completely "full". The IMAGE designated
0028 C must be a monochrome image. The LINE parameter must be between
0029 C 0 and 511. If IMAGE or LINE is out of range, an error message is printed
0030 C and no transfer takes place.
0031 C
0032 C***SUBPROGRAMS CALLED:
0033 C
0034 C name source load remarks
0035 C -----
0036 C RANGE %RANGE %RANGE logical function that determines if its
0037 C first parameter is within the last two parameters.
0038 C CHAR ----- intrinsic HP FORTRAN77 function that strips off
0039 C the upper byte of an integer and returns the lower
0040 C byte as a character.
0041 C
0042 C***WRITTEN BY:
0043 C
0044 C The code on which this subprogram is based was written by
```

```

0045 C   NETTIE D. FAULCON, July, 1983.  This modification is by
0046 C   KEITH MILLER,      June, 1984.
0047 C
0048 C***REVISION HISTORY:
0049 C
0050 C
0051 C***LOCAL VARIABLES:
0052 C
0053     INTEGER      PIXELS(256)! holds lower order bytes of INTS values
0054     CHARACTER*1  CPIX  (512)! overlays PIXELS
0055     EQUIVALENCE (PIXELS,CPIX)
0056 C
0057     LOGICAL RANGE      ! function that ascertains if its first parameter
0058 C                       ! is between (inclusive) its last 2 parameters
0059     INTEGER TERM       ! the logical unit for terminal output
0060     INTEGER IMLO, IMHI ! the limits on COMTAL monochrome image numbers
0061     INTEGER LNLO, LNHI ! the limits on COMTAL image line numbers
0062     INTEGER INDEX      ! indexes into arrays in a loop
0063 C
0064 C***INITIALIZATIONS:
0065 C
0066     DATA  TERM/1/
0067     DATA  IMLO/1/, IMHI/4/
0068     DATA  LNLO/0/, LNHI/511/
0069 C
0070 C***PROCESSING
0071 C
0072     IF (.NOT.(RANGE(IMAGE, IMLO, IMHI))) GOTO 8001 ! error return
0073     IF (.NOT.(RANGE(LINE, LNLO, LNHI))) GOTO 8101 ! error return
0074 C
0075 C     strip off the upper bytes of the pixel values in INTS
0076 C
0077     DO 1000 INDEX=1,512
0078         CPIX(INDEX) = CHAR(INTS(INDEX)) ! CPIX overlays PIXELS
0079     1000 CONTINUE
0080 C
0081 C     Programming note:
0082 C     The EXEC call is explained in detail in the
0083 C     HP Programmer's Reference Manual for RTE-6/VM,p.2-19ff.  This
0084 C     transfer function for the COMTAL is discussed in the
0085 C     COMTAL User's Manual, Section 5.2.2.1.  In the EXEC call
0086 C     that follows, the HP resident driver called DVR41 is called as
0087 C     follows: the first parameter (2) signifies a write; the
0088 C     second parameter is in two parts: 36B identifies the resident
0089 C     DVR41 driver, and 100B identifies the line transfer operation
0090 C     transferred, and the fourth parameter gives PIXELS' length in words

```

```

0091 C      (256); and the final parameter is a COMTAL command code for the transfer.
0092 C
0093      CALL EXEC(2, 36B+100B, PIXELS, 256, (IMAGE-1)*2048 + LINE)
0094      RETURN
0095 C
0096 C***ERROR RETURNS
0097 C
0098      8001 WRITE(TERM, 8003) IMAGE, IMLO, IMHI
0099      8003 FORMAT(' IMAGE NUMBER,', I3, ' OUT OF RANGE:', 212, '.')
0100      GOTO 8900
0101 C
0102      8101 WRITE(TERM, 8103) LINE, LNLO, LNHI
0103      8103 FORMAT(' LINE NUMBER,', I4, ' OUT OF RANGE:', 213, '.')
0104      GOTO 8900
0105 C
0106      8900 WRITE(TERM, 8901)
0107      8901 FORMAT(' WRIL2 FAILS. NO TRANSFER.')
0108      RETURN
0109      END

```

&WRILN T=00004 IS ON CR00021 USING 00004 BLKS R=0000

```
0001 CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
0002 SUBROUTINE WRILN(IMAGE, LINE, PIXELS)
0003 CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
0004 C
0005 C***PARAMETER DECLARATIONS:
0006 C
0007 INTEGER IMAGE ! COMTAL image number to be written to
0008 INTEGER LINE ! which horizontal line to be written to;
0009 C ! lines numbered from 1 (screen top) to 512.
0010 INTEGER PIXELS(256)! 512 bytes (pixels) to be transferred
0011 C
0012 C***INTRODUCTION:
0013 C
0014 C This subroutine WRites an Image LiNe to the COMTAL. The line of
0015 C pixels is made up of 8 bit (0-255) grey scale intensities. The
0016 C PIXELS array is assumed to be completely full.
0017 C
0018 C***LANGUAGE:
0019 C
0020 C FORTRAN 77, the HP-1000 version for RTE-6/VM.
0021 C
0022 C***LIMITATIONS:
0023 C
0024 C The PIXELS array must be completely "full". The IMAGE designated
0025 C must be a monochrome image. The LINE parameter must be between
0026 C 0 and 511. If IMAGE or LINE is out of range, an error message is printed
0027 C and no transfer takes place.
0028 C
0029 C***SUBPROGRAMS CALLED:
0030 C
0031 C name source load remarks
0032 C -----
0033 C RANGE @RANGE %RANGE logical function that determines if its
0034 C first parameter is within the last two parameters.
0035 C
0036 C***WRITTEN BY:
0037 C
0038 C The code on which this subprogram is based was written by
0039 C NETTIE D. FAULCON, July, 1983. This modification is by
0040 C KEITH MILLER, June, 1984.
0041 C
0042 C***REVISION HISTORY:
0043 C
0044 C
```

```

0045 C***LOCAL VARIABLES:
0046 C
0047 LOGICAL RANGE ! function that ascertains if its first parameter
0048 C ! is between (inclusive) its last 2 parameters
0049 INTEGER TERM ! the logical unit for terminal output
0050 INTEGER IMLO, IMHI ! the limits on COMTAL monochrome image numbers
0051 INTEGER LNLO, LNHI ! the limits on COMTAL image line numbers
0052 C
0053 C***INITIALIZATIONS:
0054 C
0055 DATA TERM/1/
0056 DATA IMLO/1/, IMHI/4/
0057 DATA LNLO/0/, LNHI/511/
0058 C
0059 C***PROCESSING
0060 C
0061 IF (.NOT.(RANGE(IMAGE, IMLO, IMHI))) GOTO 8001 ! error return
0062 IF (.NOT.(RANGE(LINE, LNLO, LNHI))) GOTO 8101 ! error return
0063 C
0064 C Programming note:
0065 C The EXEC call is explained in detail in the
0066 C HP Programmer's Reference Manual for RTE-6/VM, p.2-19ff. This
0067 C transfer function for the COMTAL is discussed in the
0068 C COMTAL User's Manual, Section 5.2.2.1. In the EXEC call
0069 C that follows, the HP resident driver called DVR41 is called as
0070 C follows: the first parameter (2) signifies a write; the
0071 C second parameter is in two parts: 36B identifies the resident
0072 C DVR41 driver, and 100B identifies the line transfer operation
0073 C of that driver; the third parameter (PIXELS) holds the data to be
0074 C transferred, and the fourth parameter gives PIXELS' length in words
0075 C (256); and the final parameter is a COMTAL command code for the transfer.
0076 C
0077 CALL EXEC(2, 36B+100B, PIXELS, 256, (IMAGE-1)*2048 + LINE)
0078 RETURN
0079 C
0080 C***ERROR RETURNS
0081 C
0082 8001 WRITE(TERM, 8003) IMAGE, IMLO, IMHI
0083 8003 FORMAT(' IMAGE NUMBER.', I3, ' OUT OF RANGE:', 2I2, '.')
0084 GOTO 8900
0085 C
0086 8101 WRITE(TERM, 8103) LINE, LNLO, LNHI
0087 8103 FORMAT(' LINE NUMBER.', I4, ' OUT OF RANGE:', 2I3, '.')
0088 GOTO 8900
0089 C
0090 8900 WRITE(TERM, 8901)

```

```
0091 8901 FORMAT(' WRILN FAILS. NO TRANSFER.')
```

```
0092     RETURN
```

```
0093     END
```

&LRIPT T=00004 IS UN CR00021 USING 00018 BLKS R=0000

```
0001 CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
0002 SUBROUTINE WRIPT(IMAGE, XCOOR, YCOOR, VALUE)
0003 CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
0004 C
0005 C***PARAMETER DECLARATIONS:
0006 C
0007 INTEGER IMAGE ! COMTAL image number to be written to.
0008 INTEGER XCOOR, YCOOR! point where new value is to be written to.
0009 INTEGER VALUE ! 0-255 pixel value to be written to image pt.
0010 C
0011 C***INTRODUCTION:
0012 C
0013 C This subroutine WRites an Image Point to the COMTAL. The value
0014 C of the pixel is an 8 bit (0-255) grey scale intensity. If the
0015 C image number, coordinates, or value are out of range, an error
0016 C message is printed and no transfer takes place.
0017 C
0018 C***LANGUAGE:
0019 C
0020 C FORTRAN 77, the HP-1000 version for RTE-6/VM.
0021 C
0022 C***LIMITATIONS:
0023 C
0024 C The PIXELS array must be completely "full". The IMAGE designated
0025 C must be a monochrome image. The LINE parameter must be between
0026 C 0 and 511. If IMAGE, XCOOR, or YCOOR are out of range, an error message
0027 C is printed at the terminal and no transfer takes place.
0028 C
0029 C***SUBPROGRAMS CALLED:
0030 C
0031 C name source load remarks
0032 C -----
0033 C RANGE &RANGE %RANGE logical function that determines if its
0034 C first parameter is within the last two parameters.
0035 C
0036 C***WRITTEN BY:
0037 C
0038 C The code on which this subprogram is based was written by
0039 C NETTIE D. FAULCON, July, 1983. This modification is by
0040 C KEITH MILLER, June, 1984.
0041 C
0042 C***REVISION HISTORY:
0043 C
0044 C
```



```

0045 C***LOCAL VARIABLES:
0046 C
0047 LOGICAL RANGE ! function that ascertains if its first parameter
0048 C ! is between (inclusive) its last 2 parameters
0049 INTEGER TERM ! the logical unit for terminal output
0050 INTEGER IMLO, IMHI ! the limits on COMTAL monochrome image numbers
0051 INTEGER LNLO, LNHI ! the limits on COMTAL image line numbers
0052 INTEGER PXLO, PXHI ! the limits on COMTAL pixel values
0053 INTEGER PIXELS(256) ! a buffer to read & write a COMTAL line
0054 CHARACTER*1 BYTES(512) ! overlay for PIXELS buffer
0055 EQUIVALENCE (PIXELS,BYTES)
0056 INTEGER IHOLD ! hold an integer for byte conversion
0057 CHARACTER*1 CHARS(2) ! overlay for IHOLD
0058 EQUIVALENCE (IHOLD,CHARS)
0059 C
0060 C***INITIALIZATIONS:
0061 C
0062 DATA TERM/1/
0063 DATA IMLO/1/, IMHI/4/
0064 DATA LNLO/0/, LNHI/511/
0065 DATA PXLO/0/, PXHI/255/
0066 C
0067 C***PROCESSING
0068 C
0069 IF (.NOT.(RANGE(IMAGE,IMLO,IMHI))) GOTO 8001 ! error return
0070 IF (.NOT.(RANGE(XCOORD,LNLO,LNHI))) GOTO 8101 ! error return
0071 IF (.NOT.(RANGE(YCOORD,LNLO,LNHI))) GOTO 8201 ! error return
0072 IF (.NOT.(RANGE(VALUE,PXLO,PXHI))) GOTO 8301 ! error return
0073 C
0074 C Programming note:
0075 C The EXEC calls below are to the DVR41 driver. The first call
0076 C is identical to the call made in RDILN. The second EXEC call
0077 C is identical to the one in WRILN. See the documentation for
0078 C those subroutines for details on these calls.
0079 C
0080 C Read the COMTAL line (horizontal) that contains the point in question:
0081 C
0082 CALL EXEC(1, 368+1008, PIXELS, 255, (IMAGE-1)*2048 + YCOORD)
0083 C
0084 C Change the single byte that needs changing:
0085 C
0086 IHOLD = VALUE ! transfers the pixel VALUE to CHARS(2).
0087 BYTES(XCOORD+1) = CHARS(2) ! transfers VALUE to PIXELS in proper position
0088 C ! '+1' converts from 0-255 pixels to 1-256
0089 C ! FORTRAN array.
0090 C

```

```

0091 C   Write the image line with one changed pixel to COMTAL
0092 C
0093     CALL EXEC(2, 36B+100B, PIXELS, 256, (IMAGE-1)*2048 + YCOORD)
0094     RETURN
0095 C
0096 C***ERROR RETURNS
0097 C
0098     8001 WRITE(TERM, 8003) IMAGE, IMLD, IMHI
0099     8003 FORMAT(' IMAGE NUMBER,', 13, ' OUT OF RANGE:', 212, '.')
0100     GOTO 8900
0101 C
0102     8101 WRITE(TERM, 8103) XCOORD, LNLO, LNHI
0103     8103 FORMAT(' X COORDINATE,', 14, ' OUT OF RANGE:', 214, '.')
0104     GOTO 8900
0105 C
0106     8201 WRITE(TERM, 8203) YCOORD, LNLO, LNHI
0107     8203 FORMAT(' Y COORDINATE,', 14, ' OUT OF RANGE:', 214, '.')
0108     GOTO 8900
0109 C
0110     8301 WRITE(TERM, 8303) VALUE, PXLO, PXHI
0111     8303 FORMAT(' PIXEL VALUE,', 14, ' OUT OF RANGE:', 214, '.')
0112     GOTO 8900
0113 C
0114     8900 WRITE(TERM, 8901)
0115     8901 FORMAT(' WRIPT FAILS. NO TRANSFER.')
0116     RETURN
0117     END

```

&WRIRC T=0004 IS ON CR00021 USING 00030 BLKS R=0000

```

0001 CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
0002 SUBROUTINE WRIRC(IMAGE, UPLFX, UPLFY, INPARA, XDIM, YDIM)
0003 CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
0004 C
0005 C***PARAMETER DECLARATIONS:
0006 C
0007 INTEGER IMAGE | the number of the COMTAL image to which
0008 C | INPARA is to be written
0009 INTEGER UPLFX, UPLFY | the image coordinates of the upper left
0010 C | corner of the rectangle of pixels that is
0011 C | to be written into.
0012 INTEGER XDIM, YDIM | dimensions of the input array, INPARA
0013 INTEGER INPARA(XDIM, YDIM) | the array holding the new pixel values
0014 C
0015 C***INTRODUCTION:
0016 C
0017 C
0018 C The subroutine WRite Image ReCtangle transfers pixel values from an
0019 C integer array to a portion of a COMTAL image. Note that although
0020 C pixel values are generally stored with 1 byte/pixel, WRIRC takes as
0021 C input an array of integers in which each integer holds one pixel value.
0022 C XDIM, YDIM, IMAGE, UPLFX, and UPLFY are all checked for possible out
0023 C of range errors before any transfer is attempted.
0024 C
0025 C***LANGUAGE:
0026 C
0027 C FORTRAN 77, the HP-1000 version for RTE-6/VM.
0028 C
0029 C***LIMITATIONS:
0030 C
0031 C WRIRC does a great deal of error checking before initiating the
0032 C transfer. If the programmer can verify that all calls to WRIRC
0033 C will be legal, these checks could be commented out to improve
0034 C machine efficiency. Also, the calls to the subroutines RDIL2
0035 C and WRIL2 could be replace by inline code, which is not long.
0036 C
0037 C***SUBPROGRAMS CALLED:
0038 C
0039 C name source load remarks
0040 C -----
0041 C RDIL2 &RDIL2 %RDIL2 transfers a line of bytes from a COMTAL image
0042 C into an HP array of integers.
0043 C WRIL2 &WRIL2 %WRIL2 transfers a buffer of integers to a COMTAL image
0044 C as a line of bytes (1 byte/integer).

```

```

0045 C RANGE &RANGE %RANGE logical function that determines of its first
0046 C parameter is within the last two parameters
0047 C CHAR ----- ----- HP FORTRAN77 intrinsic function: converts a
0048 C 2 byte integer into a one byte char (removes
0049 C high order byte).
0050 C
0051 C***WRITTEN BY:
0052 C
0053 C The code on which this subprogram is based was written by
0054 C NETTIE D. FAULCON, July, 1983. This code was written by
0055 C KEITH MILLER, July, 1984.
0056 C
0057 C***REVISION HISTORY:
0058 C
0059 C
0060 C***LOCAL VARIABLES:
0061 C
0062 INTEGER IMROW, IMCOL ! a location in the image
0063 INTEGER ARAROW, ARACOL ! a location in INPARA
0064 INTEGER ICHAR ! intrinsic byte to integer conversion
0065 C ! function
0066 LOGICAL RANGE ! function that ascertains if the first
0067 C ! parameter is within the last two parameters.
0068 INTEGER IMLO, IMHI ! limits on COMTAL image numbers
0069 INTEGER LNLO, LNHI ! limits on COMTAL pixel coordinates
0070 INTEGER DIMLIM ! limit on the dimensions of INPARA
0071 INTEGER TERM ! logical unit for terminal output
0072 INTEGER IBUF(512) ! buffer to hold COMTAL horizontal line
0073 C
0074 C***INITIALIZATIONS:
0075 C
0076 DATA TERM/1/
0077 DATA IMLO/1/, IMHI/4/
0078 DATA LNLO/0/, LNHI/511/
0079 DATA DIMLIM/64/
0080 C
0081 C***PROCESSING
0082 C
0083 IF (.NOT.(RANGE(IMAGE, IMLO,IMHI )))GOTO 8001 ! error return
0084 IF (.NOT.(RANGE(XDIM, 1, DIMLIM)))GOTO 8101 ! error return
0085 IF (.NOT.(RANGE(YDIM, 1, DIMLIM)))GOTO 8201 ! error return
0086 IF (.NOT.(RANGE(UPLFX, LNLO,LNHI )))GOTO 8301 ! error return
0087 IF (.NOT.(RANGE(UPLFY, LNLO,LNHI )))GOTO 8401 ! error return
0088 IF (.NOT.(RANGE(UPLFX+XDIM-1,LNLO,LNHI )))GOTO 8501 ! error return
0089 IF (.NOT.(RANGE(UPLFY+YDIM-1,LNLO,LNHI )))GOTO 8601 ! error return
0090 C

```

```

0091 C      we get to this point if the transfer is to take place
0092 C
0093      IMROW = UPLFY
0094      DO 2000 ARAROW = 1, YDIM
0095          CALL RDIL2(IBUF, IMAGE, IMROW)
0096 C              ! the next line initializes the column pointer;
0097      IMCOL = UPLFX + 1 ! the "+1" is necessary because COMTAL image
0098 C              ! coordinates range from 0 to 511 and the
0099 C              ! FORTRAN array indices range from 1 to 512.
0100      DO 1000 ARACOL = 1, XDIM
0101          IBUF(IMCOL) = INPARA(ARAROW,ARACOL)
0102          IMCOL = IMCOL + 1 ! increment for next 1000 loop pass
0103      1000 CONTINUE
0104          CALL WRIL2(IMAGE, IMROW, IBUF)
0105          IMROW = IMROW + 1 ! increment for next 2000 loop pass
0106      2000 CONTINUE
0107      RETURN      ! successful termination
0108 C
0109 C***ERROR RETURNS:
0110 C
0111      8001 WRITE(TERM, 8003) IMAGE, IMLO, IMHI
0112      8003 FORMAT(' IMAGE NUMBER,', I3, ' OUT OF RANGE:', I214, '.')
0113      GOTO 8900
0114 C
0115      8101 WRITE(TERM, 8103) XDIM, 1, DIMLIM
0116      8103 FORMAT(' X DIMENSION,', I4, ' OUT OF RANGE:', I215, '.')
0117      GOTO 8900
0118 C
0119      8201 WRITE(TERM, 8203) YDIM, 1, DIMLIM
0120      8203 FORMAT(' Y DIMENSION,', I4, ' OUT OF RANGE:', I215, '.')
0121      GOTO 8900
0122 C
0123      8301 WRITE(TERM, 8303) UPLFX, LNLO, LNHI
0124      8303 FORMAT(' X COORDINATE FOR CORNER,', I4, ' OUT OF RANGE:', I215, '.')
0125      GOTO 8900
0126 C
0127      8401 WRITE(TERM, 8403) UPLFY, LNLO, LNHI
0128      8403 FORMAT(' Y COORDINATE FOR CORNER,', I4, ' OUT OF RANGE:', I215, '.')
0129      GOTO 8900
0130 C
0131      8501 WRITE(TERM, 8503) UPLFX, XDIM, LNLO, LNHI
0132      8503 FORMAT(' X COORDINATE FOR THE CORNER AND THE X DIMENSION ',
0133          1      ' OF THE ARRAY', //, ' OVERFLOW IMAGE BOUNDARIES.',
0134          2      ' X COORDINATE =', I4, ' X DIMENSION =', I4, //,
0135          3      ' IMAGE COORDINATE LIMITS ARE ', I215, '.')
0136      GOTO 8900

```

```
0137 C-
0138 8601 WRITE(TERM, 8603) UPLFY, YDIM, LNLO, LNHI
0139 8603 FORMAT(' Y COORDINATE FOR THE CORNER AND THE Y DIMENSION ',
0140 1 ' OF THE ARRAY',/, ' OVERFLOW IMAGE BOUNDARIES.',
0141 2 ' X COORDINATE =', I4, ' X DIMENSION =', I4, /,
0142 3 ' IMAGE COORDINATE LIMITS ARE ', 2I5, '.')
0143 GOTO 8900
0144 C
0145 8900 WRITE(TERM, 8903)
0146 8903 FORMAT(' WRIRC FAILS. NO TRANSFER TAKES PLACE.')
```

```
0147 RETURN
```

```
0148 END
```

```
0149
```

```
0150
```

&WRLUT T=00004 IS ON CR00021 USING 00004 BLKS R=0000

```
0001 CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
0002 SUBROUTINE WRLUT(LUTNUM, TABLE)
0003 CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
0004 C
0005 C***PARAMETER DECLARATIONS:
0006 C
0007 INTEGER LUTNUM ! the number of the COMTAL look-up table (called
0008 C ! "function memory" in the COMTAL literature).
0009 INTEGER TABLE(256)! the values to be placed in the look-up table;
0010 C ! notice that 0 maps to TABLE(1), 1 maps to
0011 C ! TABLE(2), ..., and 255 maps to TABLE(256).
0012 C
0013 C***INTRODUCTION:
0014 C
0015 C The subroutine Write Look-Up Table (LUT) establishes a COMTAL mapping
0016 C from the integers 0-255 to the elements in TABLE. This LUT can be
0017 C used for grey level enhancements in the COMTAL. A similar subroutine
0018 C called WRPSU is used to establish a pseudocolor look-up table. This
0019 C routine is only used for grey scale look-up tables.
0020 C
0021 C***LANGUAGE:
0022 C
0023 C FORTRAN 77, the HP-1000 version for RTE-6/VM.
0024 C
0025 C***LIMITATIONS:
0026 C
0027 C Although 8 bits are sufficient for the look-up table values, full
0028 C integers are used in TABLE. This format is dictated by the COMTAL
0029 C conventions as given in section 5.2.3.1.
0030 C
0031 C***SUBPROGRAMS CALLED:
0032 C
0033 C name source load remarks
0034 C -----
0035 C RANGE %RANGE %RANGE logical function which determines if its 1st
0036 C parameter is within its 2nd and 3rd inclusive.
0037 C
0038 C***WRITTEN BY:
0039 C
0040 C The code on which this subprogram is based was written by
0041 C NETTIE D. FAULCON, July, 1983. This subprogram was written by
0042 C KEITH MILLER, July, 1984, with the support of a NASA-ASEE
0043 C summer fellowship.
0044 C
```

```

0045 C***REVISION HISTORY:
0046 C
0047 C***LOCAL VARIABLES:
0048 C
0049 LOGICAL RANGE ! logical function which determines if its 1st
0050 C ! parameter is within its 2nd and 3rd inclusive.
0051 INTEGER TERM ! logical unit for terminal output
0052 INTEGER LUTLO,LUTHI ! limits for COMTAL function memories
0053 C
0054 C***INITIALIZATIONS:
0055 C
0056 DATA TERM/1/
0057 DATA LUTLO/1/, LUTHI/4/
0058 C
0059 C***PROCESSING
0060 C
0061 IF (.NOT.(RANGE(LUTNUM,LUTLO,LUTHI))) GOTO 8001 ! error return
0062 C
0063 C Programming notes:
0064 C The EXEC command parameters are discussed in the HP RTE-6/VM
0065 C Programmer's Reference Manual, 2-19ff. The COMTAL parameters
0066 C are discussed in section 5.2.3 of the COMTAL User's Manual.
0067 C
0068 C The first parameter to EXEC identifies the EXEC command as
0069 C a write command. The second parameter identifies the resident
0070 C HP driver (36B) and gives the code (200B) that identifies this
0071 C operation, a transfer to a COMTAL function memory (Look-Up Table).
0072 C The third parameter gives the Look-Up Table values (TABLE),
0073 C and the fourth parameter gives the length of TABLE in words.
0074 C The fifth parameter is a COMTAL code that is described bit by
0075 C bit in the User's Manual. In short, bit 15 signifies write to
0076 C COMTAL, bit 14 designates function memory instead of pseudocolor,
0077 C bit 12 signifies standard replacement, and bits 8&9 identify the
0078 C function memory to be used. (Bits are numbered 15 high, 0 low).
0079 C
0080 CALL EXEC( 2, 36B+200B, TABLE, 256, ((LUTNUM-1)*256) )
0081 RETURN
0082 C
0083 C***ERROR RETURN
0084 C
0085 8001 WRITE(TERM, 8003) LUTNUM, LUTLO, LUTHI
0086 8003 FORMAT(' THE FUNCTION MEMORY NUMBER, ',I4,', IS OUT OF RANGE:',
0087 1, 2I4, ',.')
0088 8900 WRITE(TERM, 8901)
0089 8901 FORMAT(' WRLUT FAILS. NO TRANSFER TO COMTAL. ')
0090 END

```



&WRPSU T=00004 IS ON CR00021 USING 00012 BLKS R=0000

```
0001 CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
0002 SUBROUTINE WRPSU(TABLE)
0003 CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
0004 C
0005 C***PARAMETER DECLARATIONS:
0006 C
0007 INTEGER TABLE(768)! the values of the COMTAL look-up
0008 C ! table are read from this array. The RED table
0009 C ! is in TABLE(1:256); the GREEN, in TABLE(257:512);
0010 C ! and the BLUE, in TABLE(513:768).
0011 C
0012 C***INTRODUCTION:
0013 C
0014 C The subroutine WRite the PSeUdocolor table writes the 3 COMTAL mappings
0015 C from 0-255 which comprise the pseudocolor table. Note that the values
0016 C are placed into TABLE in the order RED, GREEN, and BLUE.
0017 C
0018 C***LANGUAGE:
0019 C
0020 C FORTRAN 77, the HP-1000 version for RTE-6/VM.
0021 C
0022 C***LIMITATIONS:
0023 C
0024 C Although 8 bits are sufficient for the look-up table values, full
0025 C integers are used in TABLE. This format is dictated by the COMTAL
0026 C conventions as given in section 5.2.3.1.
0027 C
0028 C***SUBPROGRAMS CALLED:
0029 C
0030 C NONE.
0031 C
0032 C***WRITTEN BY:
0033 C
0034 C The code on which this subprogram is based was written by
0035 C NETTIE D. FAULCON, July, 1983. This subprogram was written by
0036 C KEITH MILLER, July, 1984, with the support of a NASA-ASEE
0037 C summer fellowship.
0038 C
0039 C***REVISION HISTORY:
0040 C
0041 C
0042 C***LOCAL VARIABLES:
0043 C
0044 C NONE.
```

```

0045 C
0046 C***INITIALIZATIONS:
0047 C
0048 C NONE.
0049 C
0050 C***PROCESSING
0051 C
0052 C Programming notes:
0053 C The EXEC command parameters are discussed in the HP RTE-6/VM
0054 C Programmer's Reference Manual, 2-19ff. The COMTAL parameters
0055 C are discussed in section 5.2.3 of the COMTAL User's Manual.
0056 C
0057 C The first parameter to EXEC identifies the EXEC command as
0058 C a write command. The second parameter identifies the resident
0059 C HP driver (36B) and gives the code (300B) that identifies this
0060 C operation, a transfer from the COMTAL pseudocolor table.
0061 C The third parameter gives the array that will hold the values,
0062 C and the fourth parameter gives the length of TABLE in words.
0063 C The fifth parameter is a COMTAL code that is described bit by
0064 C bit in the User's Manual. The DVR41 driver takes care of all the
0065 C bits except 8&9 which identify the color to be transferred.
0066 C
0067 C Note that we make three separate calls to EXEC. Each call fills a
0068 C different section of the pseudocolor table from the TABLE array.
0069 C
0070 C CALL EXEC( 2, 36B+300B, TABLE(1), 256, 1*256 ) ! red
0071 C CALL EXEC( 2, 36B+300B, TABLE(257), 256, 0*256 ) ! green
0072 C CALL EXEC( 2, 36B+300B, TABLE(513), 256, 2*256 ) ! blue
0073 C RETURN
0074 C END

```

&WRTAR T=00004 IS ON CR00021 USING 00002 BLKS R=0000

```
0001 CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
0002 SUBROUTINE WRTAR(XCOOR, YCOOR)
0003 CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
0004 C
0005 C***PARAMETER DECLARATIONS:
0006 C
0007 INTEGER XCOOR ! X coordinate of the desired target location.
0008 INTEGER YCOOR ! Y coordinate of the desired target location.
0009 C
0010 C***INTRODUCTION:
0011 C
0012 C This subroutine WRites a TARget location to the COMTAL, thereby
0013 C "positioning" the COMTAL cursor.
0014 C
0015 C***LANGUAGE:
0016 C
0017 C FORTRAN 77, the HP-1000 version for RTE-6/VM.
0018 C
0019 C***LIMITATIONS:
0020 C
0021 C Both the XCOOR and YCOOR must be within the range 0 to 511. If not,
0022 C an error message is printed and no transfer takes place to the COMTAL.
0023 C
0024 C***SUBPROGRAMS CALLED:
0025 C
0026 C name source load remarks
0027 C -----
0028 C RANGE &RANGE %RANGE logical function that determines if its
0029 C first parameter is within the last two parameters.
0030 C
0031 C***WRITTEN BY:
0032 C
0033 C The code on which this subprogram is based was written by
0034 C NETTIE D. FAULCON, July, 1983. This modification is by
0035 C KEITH MILLER, June, 1984.
0036 C
0037 C***REVISION HISTORY:
0038 C
0039 C
0040 C***LOCAL VARIABLES:
0041 C
0042 LOGICAL RANGE ! function that ascertains if its first parameter
0043 C ! is between (inclusive) its last 2 parameters
0044 INTEGER TERM ! the logical unit for terminal output
```

```

0045 - INTEGER LNLO, LNHI ! the limits on COMTAL image line numbers
0046 INTEGER IBUF(2) ! buffer for passing coordinates to COMTAL
0047 INTEGER IDUMMY ! an ignored EXEC call parameter
0048 C
0049 C***INITIALIZATIONS:
0050 C
0051 DATA TERM/1/
0052 DATA LNLO/0/, LNHI/511/
0053 C
0054 C***PROCESSING
0055 C
0056 IF (.NOT.(RANGE(XCOORD, LNLO, LNHI))) GOTO 8001 ! error return
0057 IF (.NOT.(RANGE(YCOORD, LNLO, LNHI))) GOTO 8101 ! error return
0058 C
0059 C Programming note:
0060 C The EXEC call is explained in detail in the
0061 C HP Programmer's Reference Manual for RTE-6/VM, p.2-19ff. This
0062 C transfer function for the COMTAL is discussed in the
0063 C COMTAL User's Manual, Section 5.2.4. In the EXEC call
0064 C that follows, the HP resident driver called DVR41 is called as
0065 C follows: the first parameter (2) signifies a write; the
0066 C second parameter is in two parts: 36B identifies the resident
0067 C DVR41 driver, and 400B identifies the target transfer operation
0068 C of that driver; the third parameter (IBUF) contains the two coordinates
0069 C to be transferred, and the fourth parameter gives the length of the
0070 C buffer in words. The last parameter is ignored.
0071 C
0072 IBUF(1) = XCOORD
0073 IBUF(2) = YCOORD
0074 CALL EXEC(2, 36B+400B, IBUF, 2, IDUMMY)
0075 RETURN
0076 C
0077 C***ERROR RETURNS
0078 C
0079 8001 WRITE(TERM, 8003) XCOORD, LNLO, LNHI
0080 8003 FORMAT(' XCOORD, ', I5, ' OUT OF RANGE:', 2I4, ', ')
0081 GOTO 8900
0082 C
0083 8101 WRITE(TERM, 8103) YCOORD, LNLO, LNHI
0084 8103 FORMAT(' YCOORD, ', I5, ' OUT OF RANGE:', 2I4, ', ')
0085 GOTO 8900
0086 C
0087 8900 WRITE(TERM, 8901)
0088 8901 FORMAT(' WRTAR FAILS. NO TRANSFER. ')
0089 RETURN
0090 END

```

TEST PROGRAM SOURCE CODES

&TADD2\_T=00004 IS ON CR00021 USING 00002 BLKS R=0000

```
0001      PROGRAM TADD2
0002 C
0003 C      TEST THE ADDPROGRAM ADDI2 (ADD IMAGES)
0004 C
0005 C      KEITH MILLER 7/17/84
0006 C
0007 C      LOAD MODULES: %TADD2, %ADDI2, %CMMND, %RANGE, %DIGIT
0008 C
0009      CALL ADDI2(0,1,2,1) ! SHOULD GIVE OUT OF RANGE ERROR.
0010      CALL ADDI2(1,0,2,1) ! "
0011      CALL ADDI2(1,2,0,1) ! "
0012 C
0013      CALL ADDI2(3,1,2,1) ! SHOULD WORK.
0014 C
0015      STOP
0016      END
```

&TADDI T=00004 IS ON CR00021 USING 00002 BLKS R=0000

```
0001      PROGRAM TADDI
0002 C
0003 C      TEST THE ADDPROGRAM ADDIM (ADD IMAGES)
0004 C
0005 C      KEITH MILLER 7/17/84
0006 C
0007 C      LOAD MODULES: %TADDI, %ADDIM, %CMMND, %RANGE, %DIGIT
0008 C
0009      CALL ADDIM(0,1,2,1) ! SHOULD GIVE OUT OF RANGE ERROR
0010      CALL ADDIM(1,0,2,1) ! "
0011      CALL ADDIM(1,2,0,1) ! "
0012 C
0013      CALL ADDIM(3,1,2,1)
0014 C
0015      STOP
0016      END
```

&TCLR T=00004 IS ON CR00021 USING 00001 BLKS R=0000

```
0001      PROGRAM TCLR
0002 C
0003 C      tests the DSPCL subroutine
0004 C
0005      CALL DSPCL(8,1,2,3)
0006      STOP
0007      END
0008
0009
0010
0011
0012
0013
0014
```

&TCLRG T=00004 IS ON CR00021 USING 00002 BLKS R=0000

```
0001      PROGRAM TCLRG
0002 C
0003 C      TEST CLEAR GRAPHICS
0004 C
0005 C      KEITH MILLER, 7/2/84
0006 C
0007 C      LOAD MODULES: %TCLRG, %CLRGR, %RANGE, %CMMND, %DIGIT
0008 C
0009      INTEGER IBUF(128)
0010      CHARACTER*20 CBUF  ! OVERLAYS THE FIRST 10 ELEMENTS OF IBUF
0011      EQUIVALENCE (IBUF,CBUF)
0012 C
0013      CBUF = 'CLEAR GR 2'
0014 C
0015      CALL CLRGR(0)
0016      CALL CLRGR(1)
0017      CALL CMMND(IBUF, 10)
0018      CALL CLRGR(5)
0019 C
0020      STOP
0021      END
0022
0023
0024
0025
0026
```



&TCLRI T=00004 IS ON CR00021 USING 00002 BLKS R=0000

```
0001      PROGRAM TCLRI
0002 C
0003 C      TEST CLEAR IMAGE
0004 C
0005 C      KEITH MILLER, 7/2/84
0006 C
0007 C      LOAD MODULES: %TCLRI, %CLRIM, %RANGE, %COMMND, %DIGIT
0008 C
0009      INTEGER IBUF(128)
0010      CHARACTER*20 CBUF ! OVERLAYS THE FIRST 10 ELEMENTS OF IBUF
0011      EQUIVALENCE (IBUF,CBUF)
0012 C
0013      CBUF = 'CLEAR IMAGE 2'
0014 C
0015      CALL CLRIM(0)
0016      CALL CLRIM(1)
0017      CALL COMMND(IBUF, 13)
0018      CALL CLRIM(5)
0019 C
0020      STOP
0021      END
0022
0023
0024
0025
0026
```

&TCMM2 T=00004 IS ON CR00021 USING 00003 BLKS R=0000

```
0001      PROGRAM TCMM2
0002 C
0003 C      TEST CMMND AGAIN
0004 C
0005 C      KEITH MILLER, 7/12/84
0006 C
0007 C      LOAD MODULES %TCMM2, %CMMND, %WAIT
0008 C
0009      INTEGER INNUM,I, IBUF(128)
0010      CHARACTER*255 CBUF
0011      EQUIVALENCE (IBUF,CBUF)
0012 C
0013      DO 1000 I = 0.256
0014          WRITE(1,500) I
0015 500      FORMAT('TESTING CHARACTER #', I4)
0016          CALL WAIT
0017          CBUF = 'G 2 PRO 1'
0018          CALL CMMND(IBUF,9)
0019          CBUF = CHAR(I)
0020          CALL CMMND(IBUF,1)
0021 1000 CONTINUE
0022      STOP
0023      END
```

&TCMMN T=00004 IS ON CR00021 USING 00002 BLKS R=0000

```
0001      PROGRAM TCMMN
0002 C
0003 C      Keith Miller, 6/16/84
0004 C
0005 C      Testing SUBROUTINE CMMND
0006 C
0007 C      LOAD MODULES REQUIRED: %TCMMN, %CMMND
0008 C
0009      INTEGER      IBUF(128)
0010 C
0011      WRITE(1,1)-
0012 1      FORMAT(' TO EXIT THIS TEST, ENTER AN EMPTY STRING')
0013 C
0014 5      WRITE(1,10)
0015 10     FORMAT("ENTER ASCII STRING")
0016      READ(1,20)IBUF
0017 20     FORMAT(128A2)
0018      N = ITLOG() ! ITLOG gives the number of characters typed in
0019      IF (N .EQ. 0) GOTO 999
0020 C
0021      CALL CMMND(IBUF,N)
0022      GO TO 5
0023      999 END
```

&TCNT T=0004 IS ON CR0021 USING 0002 BLKS R=0000

```
0001      PROGRAM TCNT
0002 C
0003 C      TEST THE SUBROUTINE COUNT
0004 C
0005 C      KEITH MILLER 7/11/84
0006 C
0007 C      LOAD MODULES: %TCNT, %COUNT, %RANGE, %RDIL2
0008 C
0009      INTEGER*4 I4BUF(256)
0010      INTEGER  INDEX
0011 C
0012      CALL COUNT(I4BUF, 1)
0013      DO 1000 INDEX = 0, 31
0014          WRITE(1,999) (I4BUF((INDEX*8)+J), J=1,8)
0015      999      FORMAT(8I9)
0016      1000 CONTINUE
0017 C
0018      STOP
0019      END
```

&TCONS T=0004 IS ON CR0021 USING 0002 BLKS R=0000

```
0001      PROGRAM TCONS
0002 C
0003 C      TEST CONCATENATION OF CHARACTER STRING, FORTRAN 77
0004 C
0005      INTEGER IHOLD
0006      CHARACTER*1 CHOLD(2)
0007      EQUIVALENCE (IHOLD,CHOLD)
0008      CHARACTER*5 STRING
0009 C
0010      IHOLD = 6 + 605
0011      STRING = '**' // CHOLD(2) // '**'
0012      WRITE(1,1000) CHOLD(2),STRING
0013 1000 FORMAT(' LETTER=', A1, '. STRING=', A5, '.')
0014      STOP
0015      END
```

&TCOPY T=0004 IS ON CR00021 USING 00001 BLKS R=0000

```
0001      PROGRAM TCOPY
0002 C
0003 C      TEST ICOPY
0004 C
0005 C      LOAD MODULES REQUIRED: %TCOPY,%ICOPY,%CMMND,%RANGE,
0006 C                               %DIGIT
0007 C
0008      CALL ICOPY(1,2)
0009      CALL ICOPY(5,1)  ! SHOULD GIVE OUT OF RANGE ERROR
0010      CALL ICOPY(2,-3) ! SHOULD GIVE OUT OF RANGE ERROR
0011      STOP
0012      END
```

&TDIGI T=00004 IS ON CR00021 USING 00002 BLKS R=0000

```
0001      PROGRAM TDIGI
0002 C
0003 C      TEST DIGIT
0004 C      KEITH MILLER, 7/2/84
0005 C
0006 C      LOAD MODULES: %TDIGI, %DIGIT, %RANGE
0007 C
0008      INTEGER INT
0009      CHARACTER*1 HCHAR
0010 C
0011      DO 1000 INT = -1, 12
0012          CALL DIGIT(HCHAR, INT)
0013          WRITE(1,501) INT, HCHAR
0014      501  FORMAT(' INTEGER INPUT AND CHARACTER OUTPUT:',1I3,1A1,'.')
0015      1000 CONTINUE
0016 C
0017      STOP
0018      END
0019
0020
0021
0022
0023
0024
0025
```

&TDSP T=00004 IS ON CR00021 USING 00001 BLKS R=0000

```
0001          PROGRAM TDSP
0002 C
0003 C          TEST DSPBW AND DSPCL
0004 C
0005 C          LOAD MODULES REQUIRED: %TDSP, %DSPBW, %DSPCL, %CMMND,
0006 C                                     %WAIT, %RANGE, %DIGIT
0007 C
0008          CALL DSPBW(1)
0009          CALL WAIT
0010          CALL DSPBW(2)
0011          CALL WAIT
0012          CALL DSPBW(3)
0013          CALL WAIT
0014          CALL DSPBW(0)
0015          CALL WAIT
0016          CALL DSPCL(1,2,3,8)
0017          STOP
0018          END
0019
0020
0021
```



&THIST T=00004 IS ON CR00021 USING 00002 BLKS R=0000

```
0001      PROGRAM THIST
0002 C
0003 C      TEST THE SUBROUTINE HISTO
0004 C
0005 C      KEITH MILLER, 7/31/84
0006 C
0007 C      LOAD MODULES: %THIST, %HISTO, %DIGIT, %CMMN2, %RANGE
0008 C
0009      INTEGER TABLE(256)
0010      INTEGER I          ! implicit do loop index
0011 C
0012      CALL HISTO(1)
0013      STOP
0014      END
```

&TNORM T=00004 IS ON CR00021 USING 00002 BLKS R=0000

```
0001      PROGRAM TNORM
0002 C
0003 C      TEST THE PROCEDURE NORML.
0004 C
0005 C      KEITH MILLER JULY 6, 1983
0006 C
0007 C      LOAD MODULES: %TNORM, %NORML, %HILO, %RDIL2, %WRIL2, %RANGE
0008 C
0009 C      CALL NORML(2)
0010 C      STOP
0011 C      END
```

&TNOTE T=00004 IS ON CR00021 USING 00002 BLKS R=0000

```
0001      PROGRAM TNOTE
0002 C
0003 C      TEST THE SUBROUTINES NOTES AND NOTE2
0004 C
0005 C      KEITH MILLER, 7/26/84
0006 C
0007 C      LOAD MODULES: %TNOTE, %NOTES, %NOTE2, %DSPGR, %WRTAR,
0008 C                    %CMMN2, %CMMND, %DIGIT, %RANGE, %DELAY
0009 C
0010      CHARACTER*255 CBUF
0011      INTEGER      IBUF(120)
0012      EQUIVALENCE (CBUF,IBUF)
0013 C
0014      CALL NOTE2(1,100,100,'B',3,'1.BLUE')
0015      CALL NOTE2(1,100,200,'R',3,'2.RED')
0016      CALL NOTE2(1,100,300,'G',3,'3.GREEN')
0017      CALL NOTE2(1,100,400,'S',3,'4.SAME')
0018 C
0019      CBUF(1:11) = 'USING NOTES'
0020      CALL NOTES(1,10,10,'R',1,CBUF,11)
0021 C      CALL NOTES(5,10,10,'R',1,CBUF,11)
0022 C      CALL NOTES(1,512,10,'R',1,CBUF,11)
0023 C      CALL NOTES(1,-1,10,'R',1,CBUF,11)
0024 C      CALL NOTES(1,10,512,'R',1,CBUF,11)
0025 C      CALL NOTES(1,10,-1,'R',1,CBUF,11)
0026 C      CALL NOTES(1,10,10,'r',1,CBUF,11)
0027 C      CALL NOTES(1,10,10,' ',1,CBUF,11)
0028 C      CALL NOTES(1,10,10,'R',0,CBUF,11)
0029 C      CALL NOTES(1,10,10,'R',17,CBUF,11)
0030 C
0031      CALL NOTE2(2,120,240,'B',3,'USING NOTE2')
0032      CALL NOTE2(0,120,240,'B',3,'USING NOTE2')
0033 C      CALL NOTE2(5,120,240,'B',3,'USING NOTE2')
0034 C      CALL NOTE2(2,512,240,'B',3,'USING NOTE2')
0035 C      CALL NOTE2(2,-1,240,'B',3,'USING NOTE2')
0036 C      CALL NOTE2(2,120,512,'B',3,'USING NOTE2')
0037 C      CALL NOTE2(2,120,-1,'B',3,'USING NOTE2')
0038 C      CALL NOTE2(2,120,240,'b',3,'USING NOTE2')
0039 C      CALL NOTE2(2,120,240,'B',0,'USING NOTE2')
0040 C      CALL NOTE2(2,120,240,'B',17,'USING NOTE2')
0041      CALL DSPGR(1)
0042      CALL DSPGR(2)
0043      STOP
0044      END
```

&TPNT T=00004 IS ON CR00021 USING 00001 BLKS R=0000

```
0001      PROGRAM TPNT
0002 C
0003 C      TEST THE SUBROUTINE PAINT
0004 C
0005 C      KEITH MILLER, 7/16/84
0006 C
0007 C      LOAD MODULES: %TPNT, %PAINT, %RANGE, %WRIRC, %RDTAR,
0008 C                  %RDIL2, %WRIL2, %CMMND, %DSPBW
0009 C
0010      CALL PAINT(1,35, 200)
0011      STOP
0012      END
```

&TPROF T=00004 IS ON CR00021 USING 00002 BLKS R=0000

```
0001      PROGRAM TPROF
0002      C
0003      C      TEST PROFL SUBROUTINE
0004      C
0005      C      KEITH MILLER, 7/12/84
0006      C
0007      C      LOAD MODULES: %TPROF, %PROFL, %RANGE, %CMMND, %WAIT,
0008      C                      %DIGIT
0009      C
0010      CALL PROFL(1,1)
0011      WRITE(1,1001)
0012      1001 FORMAT(' SUCCESSFULLY RETURNED TO CALLER OF PROFILER')
0013      STOP
0014      END
```

&TRANG T=00004 IS ON CR00021 USING 00002 BLKS R=0000

```
0001      PROGRAM TRANG
0002 C
0003 C      TESTS THE RANGE FUNCTION
0004 C
0005 C      LOAD MODULES REQUIRED: %TRANG, %RANGE
0006 C
0007 C
0008      LOGICAL RANGE, ANSWER
0009 C
0010      WRITE(1,1)
0011      1 FORMAT(' SHOULD BE T F ERROR-F')
0012 C
0013      WRITE(1,2001) RANGE(2,1,3)
0014      2001 FORMAT(' ', L1)
0015      WRITE(1,2001) RANGE(1,2,3)
0016      ANSWER = RANGE(3,2,1)
0017      WRITE(1,2001) ANSWER
0018 C
0019      STOP
0020      END
0021
0022
0023
0024
0025
0026
0027
```

&TRDTA T=00004 IS ON CR00021 USING 00002 BLKS R=0000

```
0001      PROGRAM TRDTA
0002 C
0003 C      TEST THE SUBROUTINE RDTAB
0004 C
0005 C      KEITH MILLER, 7/19/84
0006 C
0007 C      LOAD MODULES: %TRDTA, %RDTAB, %RANGE
0008 C
0009 C      INTEGER TABLE(16), INDEX, WHICH
0010 C
0011 C      DO 2000 WHICH = 1,4
0012 C          CALL RDTAB(TABLE, WHICH, 0)
0013 C          WRITE(1,999)WHICH, TABLE(1), TABLE(2), TABLE(3), TABLE(4)
0014 C      999      FORMAT(I2, ' ') ', 409)
0015 C      2000 CONTINUE
0016 C
0017 C      STOP
0018 C      END
```

&TSETV T=0004 IS ON CR00021 USING 00002 BLKS R=0000

```
0001      PROGRAM TSETV
0002 C
0003 C      TEST THE SETV SUBROUTINE
0004 C
0005 C      KEITH MILLER. 7/17/84
0006 C
0007 C      LOAD MODULES: %TSETV, %SETV, %RANGE, %DIGIT, %CMMND, %DSPBW
0008 C
0009      CALL SETV(2)  !SHOULD BE AN ERROR
0010      CALL SETV(11) !SHOULD BE AN ERROR
0011      CALL SETV(5)  !SHOULD WORK
0012 C
0013      STOP
0014      END
```



&TSPRD T=00004 IS ON CR00021 USING 00003 BLKS R=0000

```
0001      PROGRAM TSPRD
0002 C
0003 C      TESTS THE PROGRAM TSPRED.
0004 C
0005 C      KEITH MILLER, JULY 5, 1984
0006 C
0007 C      LOAD MODULES: %TSPRD, %SPRED, %RDIL2, %WRIL2, %RANGE
0008 C
0009      INTEGER IBUF(512), I1, I2, I3, INDEX
0010 C
0011 C      WRITE(1, 1001)
0012 C1001 FORMAT(' GIVE THREE SHADES, USING THE FORMAT 314:')
0013 C      READ (1, 1003) I1, I2, I3
0014 C1003 FORMAT(3I4)
0015 C
0016 C      DO 2000 INDEX=1,170
0017 C          IBUF(INDEX) = I1
0018 C          IBUF(INDEX+170) = I2
0019 C          IBUF(INDEX+340) = I3
0020 C2000 CONTINUE
0021 C
0022 C          IBUF(511) = I3
0023 C          IBUF(512) = I3
0024 C
0025 C      DO 3000 INDEX=0.511
0026 C          CALL WRIL2(1, INDEX, IBUF)
0027 C3000 CONTINUE
0028 C
0029 C          CALL SPRED(1)
0030 C
0031 C          STOP
0032 C          END
```

&TSUBI T=00004 IS ON CR00021 USING 00002 BLKS R=0000

```
0001      PROGRAM TSUBI
0002 C
0003 C      TESTS SUBROUTINES SUBIM AND SUBI2
0004 C
0005 C      KEITH MILLER 7/23/84
0006 C
0007 C      LOAD MODULES: %TSUBI, %SUBIM, %SUBI2, %RANGE, %DIGIT,
0008 C                  %WAIT, %CMMND
0009 C
0010 C
0011 C      CALL SUBIM(1,3,2) | no offset
0012 C      CALL WAIT
0013 C      CALL SUBI2(1,3,2) | 128 offset
0014 C
0015 C      STOP
0016 C      END
```

&TTHRS T=00004 IS ON CR00021 USING 00003 BLKS R=0000

```
0001      PROGRAM TTHRS
0002 C
0003 C      TESTS THE SUBROUTINE THRESHOLD
0004 C
0005 C      KEITH MILLER 7/25/84
0006 C
0007 C      LOAD MODULES: %TTHRS, %THRSH, %RANGE, %RDIL2, %WRIL2
0008 C
0009      INTEGER THRESH! THRESHOLD FROM USER
0010      INTEGER IN,OUT! TWO IMAGES
0011 C
0012      WRITE(1,1001)
0013 1001 FORMAT(' GIVE THE OUTPUT IMAGE NUMBER:')
0014      READ(1,1003)OUT
0015 1003 FORMAT(I1)
0016      WRITE(1,1005)
0017 1005 FORMAT(' GIVE THE INPUT IMAGE NUMBER:')
0018      READ(1,1003)IN
0019      WRITE(1,1007)
0020 1007 FORMAT(' GIVE THE THRESHOLD PIXEL VALUE:')
0021      READ(1,1009)THRESH
0022 1009 FORMAT(I3)
0023      CALL THRSH(OUT,IN,THRESH)
0024      STOP
0025      END
```

&TTSTI T=0004 IS ON CR00021 USING 00001 BLKS R=0000

```
0001 PROGRAM TTSTI
0002 C
0003 C TESTS THE TEST IMAGES
0004 C
0005 C LOAD MODULES: %TTSTI, %TSTI1, %WRILN, %RANGE
0006 C
0007 CALL TSTI1(1)
0008 STOP
0009 END
```

&TTV2C T=00004 IS ON CR00021 USING 00002 BLKS R=0000

```
0001      PROGRAM TTV2C
0002 C
0003 C      TEST THE SUBROUTINES TV2CM AND TV2C4
0004 C
0005 C      KEITH MILLER, 7/17/84
0006 C
0007 C      LOAD MODULES: %TTV2C, %TV2C4, %TV2CM, %CMMND, %DSPBW,
0008 C                    %RANGE, %DIGIT, %WAIT, %ADD12, %DSPVD
0009 C
0010      WRITE(1,1001)
0011 1001 FORMAT(' MAKE SURE TV CAMERA IS SET TO IMAGE 5. ')
0012      CALL WAIT
0013      CALL TV2C4
0014      STOP
0015      END
```

&TWAIT T=0004 IS ON CR00021 USING 00002 BLKS R=0000

```
0001      PROGRAM TWAIT
0002 C
0003 C      TESTING THE WAIT SUBROUTINE
0004 C      KEITH MILLER, JUNE 8, 1984
0005 C
0006      INTEGER IERR, IX, IY
0007 C
0008      WRITE(1,1000)
0009 1000 FORMAT( ' START WAIT TEST')
0010      CALL WAIT(IERR)
0011      CALL KMRTA( IX,IY,IERR)
0012      WRITE(1, 2000)
0013 2000 FORMAT( ' END WAIT TEST')
0014 C
0015      STOP
0016      END
0017
```

&TWIPE T=0004 IS ON CR00021 USING 00002 BLKS R=0000

```
0001      PROGRAM TWIPE
0002 C
0003 C      TESTS THE PROCEDURES DSPGR AND WIPGR
0004 C
0005 C      KEITH MILLER 7/30/84
0006 C
0007 C      LOAD MODULES: %TWIPE, %DSPGR, %WIPGR, %WAIT,
0008 C                  %RANGE, %DIGIT, %CMMN2
0009 C
0010      CALL DSPGR(1)
0011      CALL DSPGR(2)
0012      CALL DSPGR(3)
0013 C
0014      CALL WAIT
0015 C
0016      CALL WIPGR(3)
0017      CALL WAIT
0018      CALL WIPGR(2)
0019      CALL WAIT
0020      CALL WIPGR(1)
0021      CALL WAIT
0022      CALL WIPGR(1)
0023      STOP
0024      END
```

&TXFDS J=00004 IS ON CR00021 USING 00001 BLKS R=0000

```
0001      PROGRAM TXFDS
0002 C
0003 C      TEST PROGRAM FOR BWFDS, CLFDS
0004 C
0005 C      KEITH MILLER, 7/12/84
0006 C
0007 C      LOAD MODULES: %TXFDS, %BWFDS, %CLFDS, %RANGE, %WRILN, %CMND, %DIGIT
0008 C
0009      INTEGER FLNAME(3)
0010      CHARACTER*6 CNAME
0011      EQUIVALENCE (FLNAME,CNAME)
0012 C
0013      CNAME = 'CFXRAY'
0014      CALL BWFDS(1, FLNAME)
0015 C
0016 C      CNAME = 'CFMAND'
0017 C      CALL CLFDS(1,2,3,8, FLNAME)
0018      STOP
0019      END
```



&TXGLN T=00004 IS ON CR00021 USING 00002 BLKS R=0030

```
0001          PROGRAM TXGLN
0002 C
0003 C          TEST WRGLN AND RDGLN
0004 C
0005 C          LOAD MODULES REQUIRED: %TXGLN,%RDGLN,%WRGLN,%RANGE
0006 C
0007 C          INTEGER LINE ! number of line in image being processed
0008 C          INTEGER IBUF(32) ! buffer to hold ONOFF values, one line/time
0009 C                      ! note that 16 ONOFF bits fit in one INTEGER
0010 C
0011 C          DO 2000 LINE = 1, 512
0012 C              CALL RDGLN(IBUF, 2, LINE)
0013 C
0014 C              CALL WRGLN(1, LINE, IBUF)
0015 C          2000 CONTINUE
0016 C          END
0017 C
0018 C
```

&TXGPT T=00004 IS ON CR00021 USING 00002 BLKS R=0000

```
0001      PROGRAM TXGPT
0002 C
0003 C      TEST RDGPT AND WRGPT
0004 C
0005 C      KEITH MILLER, 6/22/84
0006 C
0007 C      LOAD MODULES: %TXGPT, %RDGPT, %WRGPT, %RANGE, &RDTAR
0008 C
0009 C      INTEGER INDEX I LOOP INDEX
0010 C
0011      DO 1000 INDEX = 1, 20
0012          CALL WRGPT(1, INDEX, INDEX, 1)
0013          CALL WRGPT(1, INDEX, 21-INDEX, 1)
0014 1000 CONTINUE
0015 C
0016      CALL WAIT
0017 C
0018 C      DO 2000 INDEX = 1, 20
0019 C          CALL WRGPT(1, INDEX, INDEX, 0)
0020 C          CALL WRGPT(1, INDEX, 21-INDEX, 0)
0021 C2000 CONTINUE
0022 C
0023      DO 3000 INDEX=1,5
0024          CALL WAIT
0025          CALL RDTAR(IX, IY)
0026          CALL RDGPT(IVALUE, 1, IX, IY)
0027          WRITE(1, 2001) IX, IY, IVALUE
0028 2001  FORMAT(' AT POINT ', 214, ' GRAPHICS VALUE=', I2, '.')
0029 3000 CONTINUE
0030 C
0031      STOP
0032      END
```

TXILN T=0004 IS ON CR00021 USING 0004 BLKS R=0000

```
0001      PROGRAM TXILN
0002 C
0003 C      TEST WRILN AND RDILN
0004 C
0005 C      LOAD MODULES REQUIRED: %TXILN,%RDILN,%WRILN,%RANGE
0006 C
0007      INTEGER LINE ! number of line in image being processed
0008      INTEGER IBUF(256) ! buffer to hold pixel values, one line/time
0009 C                      ! note that 2 pixels fit in one INTEGER
0010 C
0011      DO 2000 LINE = 0, 511
0012          CALL RDILN(IBUF, 1, LINE)
0013 C
0014          CALL WRILN(2, LINE, IBUF)
0015      2000 CONTINUE
0016      END
0017
0018
```

&TXIPT T=00004 IS ON CR00021 USING 00002 BLKS R=0000

```
0001      PROGRAM TXIPT
0002 C
0003 C      TEST RDIPT AND WRIPT
0004 C
0005 C      KEITH MILLER, 6/21/84
0006 C
0007 C      LOAD MODULES: %TXIPT, %RDIPT, %WRIPT, %RANGE, %WAIT
0008 C
0009      INTEGER XCOOR, YCOOR, VALUE
0010 C
0011      CALL RDIPT(VALUE, 1, 200, 100)
0012      WRITE(1,1001) VALUE
0013 1001  FORMAT(' VALUE =', I4 )
0014      CALL WRIPT(1,200,100,005)
0015      CALL RDIPT(VALUE, 1, 200, 100)
0016      WRITE(1,1001) VALUE
0017 C
0018      STOP
0019      END
0020
0021
0022
0023
```

&TXIRC T=00004 IS ON CR00021 USING 00002 BLKS R=0000

```
0001      PROGRAM TXIRC
0002 C
0003 C      TEST RDIRC (READ IMAGE RECTANGLE)
0004 C      & WRIRC (WRITE IMAGE RECTANGLE)
0005 C
0006 C      KEITH MILLER, JULY 3, 1984
0007 C
0008 C      LOAD MODULES: %TXIRC, %RDIRC, %WRIRC, %RANGE, %RDILN, %WRILN
0009 C
0010      INTEGER BUFFER(10,10)
0011      INTEGER ROW, COL
0012 C
0013      CALL RDIRC(BUFFER, 10, 10, 1, 0, 0)
0014      CALL WRIRC(1, 100, 100, BUFFER, 10, 10)
0015      STOP
0016      END
0017
0018
```

&TXLUT T=00004 IS ON CR00021 USING 00002 BLKS R=0000

```
0001      PROGRAM TXLUT
0002 C
0003 C      TEST WRLUT AND RDLUT
0004 C
0005 C      KEITH MILLER, 7/10/84
0006 C
0007 C      LOAD MODULES: %TXLUT, %WRLUT, %RDLUT, %RANGE
0008 C
0009      INTEGER TABLE(256), TABLE2(256)
0010      INTEGER INDEX
0011 C
0012      DO 1000 INDEX = 1,256
0013          TABLE(INDEX) = 256-INDEX
0014 1000 CONTINUE
0015 C
0016 C      CALL WRLUT(1, TABLE)
0017 C
0018      CALL RDLUT(TABLE2, 1)
0019      DO 2000 INDEX = 1,256
0020          WRITE (1, 1999) TABLE2(INDEX)
0021 1999      FORMAT(I4)
0022 2000 CONTINUE
0023 C
0024      STOP
0025      END
```

&TXPSU T=00004 IS ON CR00021 USING 00004 BLKS R=0000

```
0001      PROGRAM TXPSU
0002 C
0003 C      TEST PSEUDO COLOR TABLE TRANSFERS: RDPSU AND WRPSU
0004 C
0005 C      LOAD MODULES: %TXPSU, %RDPSU, %WRPSU, %RANGE, %WAIT
0006 C
0007 C      KEITH MILLER, 7/10/84
0008 C
0009      INTEGER TABLE(768)
0010      INTEGER INDEX
0011 C
0012      CALL RDPSU(TABLE)
0013 C
0014      WRITE(1,999)
0015 999  FORMAT(' THE REDS:')
0016      DO 2000 INDEX=1,256
0017          WRITE(1,1999)TABLE(INDEX)
0018 1999  FORMAT(I4)
0019 2000  CONTINUE
0020 C
0021      CALL WAIT
0022 C
0023      WRITE(1,2999)
0024 2999  FORMAT(' THE GREENS:')
0025      DO 4000 INDEX=257,512
0026          WRITE(1,3999)TABLE(INDEX)
0027 3999  FORMAT(I4)
0028 4000  CONTINUE
0029 C
0030      CALL WAIT
0031 C
0032      WRITE(1,4999)
0033 4999  FORMAT(' THE BLUES:')
0034      DO 6000 INDEX=513,768
0035          WRITE(1,5999)TABLE(INDEX)
0036 5999  FORMAT(I4)
0037 6000  CONTINUE
0038 C
0039      DO 1000 INDEX = 1,256
0040          TABLE(INDEX) = 256-INDEX
0041          TABLE(INDEX+256) = INDEX
0042          TABLE(INDEX+512) = 122
0043 1000  CONTINUE
0044 C
```

```
0045 - CALL WRPSU(TABLE)  
0046 STOP  
0047 END
```



QTXTAR T=00004 IS ON CR00021 USING 00003 BLKS R=0000

```
0001      PROGRAM TXTAR
0002 C
0003 C      TEST RDTAR AND WRTAR
0004 C
0005 C      KEITH MILLER, 6/21/84
0006 C
0007 C      LOAD MODULES: %TXTAR, %RDTAR, %WRTAR, %RANGE
0008 C
0009 C      INTEGER XCOOR, YCOOR ! CURSOR COORDINATES
0010 C      INTEGER INDEX      ! LOOP INDEX
0011 C
0012 C      CALL WRTAR(12,34) ! INITIAL POSITION OF TARGET
0013 C
0014 C      DO 1000 INDEX = 1, 20
0015 C          CALL RDTAR(XCOOR, YCOOR)
0016 C          WRITE(1,500)XCOOR, YCOOR
0017 C      500  FORMAT(2I5)
0018 C          CALL WRTAR(INDEX, INDEX)
0019 C      1000 CONTINUE
0020 C
0021 C      CALL WRTAR(513,1) ! SHOULD BE AN ERROR ON X
0022 C      CALL WRTAR(0, 1) ! SHOULD BE AN ERROR ON X
0023 C      CALL WRTAR(512,-1)! SHOULD BE AN ERROR ON Y
0024 C      STOP
0025 C      END
0026
0027
0028
0029
```

HP DRIVER, DVR41, SOURCE CODE

&DOCU T=00004 IS ON CR00021 USING 00084 BLKS R=0000

```

0001 ASMB,L
0002     NAM DVR41.
0003     *****
0004     *****
0005     *****
0006     ***** WRITTEN BY M.BROWNE  COULTER COMPUTER CORPORATION.
0007     ***** REWRITTEN BY R.W. Bagdazian  HUGHES AIRCRAFT COMPANY.
0008     ***** WRITTEN NOV.,1979
0009     ***** REWRITTEN APR 1980
0010     *****
0011     ** DOCUMENTATION ADDED AUG. 1984, JIM MONTEITH & KEITH MILLER.
0012     **
0013     ** TO INVOKE THIS DRIVER FROM A FORTRAN ROUTINE, MAKE AN EXEC
0014     ** CALL LIKE THE ONE SHOWN BELOW. FOR MORE INFORMATION ON HOW
0015     ** THE EXEC CALL WORKS, SEE CHAPTER 2 IN THE HP MANUAL
0016     ** "RTE-6/VM PROGRAMMER'S REFERENCE MANUAL". FOR A LIST OF THE
0017     ** EQUIPMENT TABLE WORDS AND BITS, SEE SECTION L OF THE QUICK
0018     ** REFERENCE GUIDE FOR THE HP-1000.
0019     **
0020     ** SAMPLE CALL:
0021     **
0022     ** CALL EXEC(ICODE, LU+IFUNC, IBUFF, ILEN, ICMND)
0023     **
0024     ** ICODE: THIS PARAMETER IS EITHER A 1, 2, OR 3.
0025     ** 1: SIGNIFIES A READ OPERATION. (COMTAL -> HP)
0026     ** 2: SIGNIFIES A WRITE OPERATION. (HP -> COMTAL)
0027     ** 3: SIGNIFIES A CONTROL OPERATION; THIS MODE DOESN'T
0028     ** SEEM TO WORK IN OUR SYSTEM.
0029     ** THE EXEC CALL SETS BITS 0 AND 1 OF WORD 6 OF THE
0030     ** EQUIPMENT TABLE (EQT6 IN THE CODE BELOW) ACCORDING
0031     ** TO THE ICODE VALUE.
0032     **
0033     ** LU : LOGICAL UNIT NUMBER. THE LOGICAL UNIT NUMBER FOR
0034     ** THE COMTAL IS SET AT SYSTEM GENERATION. AT THE MOMENT
0035     ** IT IS 36B.
0036     **
0037     ** IFUNC: THIS PARAMETER SIGNALS THE DRIVER AS TO THE TYPE OF
0038     ** HP <=> COMTAL COMMUNICATION THAT IS BEING REQUESTED.
0039     ** ALTHOUGH THERE IS SOMETHING OF A RELATIONSHIP BETWEEN THE
0040     ** IFUNC CODES AND THE TRANSFER CODES OF SECTION 5.2 IN THE
0041     ** COMTAL USERS MANUAL, THE RELATIONSHIP IS HARD TO EXPLAIN,
0042     ** SO WE WON'T TRY. INSTEAD, WE'LL GIVE A CHART THAT GIVES
0043     ** THE IFUNC VALUES AND THE RELEVANT COMTAL USER MANUAL
0044     ** SECTION NUMBERS WHERE FURTHER DETAILS ON THE TRANSFERS

```

0045 \*\* ARE AVAILABLE. THE EXEC CALL STORES THE IFUNC CODE IN  
 0046 \*\* IN BITS 6-10 OF WORD 6 OF THE EQUIPMENT TABLE (EQT6 IN  
 0047 \*\* THIS DRIVER).  
 0048 \*\*

IFUNC FOR DVR	COMTAL MANUAL	REMARKS
0052 ** 000B	5.2.1	DISPLAY COMMANDS
0053 ** 100B	5.2.2	IMAGE/GRAPHIC TRANSFERS
0054 ** 200B	5.2.3	FUNCTION MEMORY (LOOK UP TABLE) TRANSFERS
0056 ** 300B	5.2.3	PSEUDO-COLOR TABLE TRANSFERS
0057 ** 400B	5.2.4.1	CURSOR AND TRACKBALL REQUESTS
0058 ** 500B	5.2.4.3	MACRO & COMMAND BLOCK TRANSFERS
0059 **	5.2.4.4	IMAGE/GRAPHICS PARAMETER BLOCK
0060 **	5.2.4.6	CODE BLOCK TRANSFERS
0061 **	5.2.4.8	IMAGE/GRAPHICS TABLE READ
0062 ** 600B	5.2.2.3	LINK SUBFUNCTION ESTABLISHES A CORRESPONDENCE BETWEEN THE NUMBERS USED TO IDENTIFY HP CONTROLLED IMAGES AND THE INTERNAL COMTAL IMAGES. SINCE WE USE THE COMTAL 1 USERS AT A TIME, WE DON'T USE THIS SUBFUNCTION.

0070 \*\* IBUFF: INTEGER ARRAY BUFFER WHICH IS USED TO FERRY DATA BETWEEN  
 0071 \*\* COMTAL AND THE HP. SOMETIMES THIS BUFFER PARAMETER  
 0072 \*\* IS IGNORED. THE EXEC CALL PLACES THE IBUFF ADDRESS IN  
 0073 \*\* SLOT 7 OF THE EQUIPMENT TABLE, AND THIS DRIVER REFERS  
 0074 \*\* TO IBUFF USING EQT7.  
 0075 \*\*

0076 \*\* ILEN : INTEGER LENGTH, IN WORDS, OF IBUFF.  
 0077 \*\* THIS LENGTH IS STORED BY THE EXEC CALL IN WORD 8 OF  
 0078 \*\* THE EQUIPMENT TABLE, EQT8 IN THIS DRIVER.  
 0079 \*\*

0080 \*\* ICMND: INTEGER CODE THAT IS USED TO GIVE CODED INFORMATION TO  
 0081 \*\* THE COMTAL. IN THE COMTAL MANUAL, THE BITS DESCRIBED IN  
 0082 \*\* THE RELEVANT 5.2 SUBSECTION ARE CODED INTO ICMND.  
 0083 \*\* THIS PARAMETER IS SOMETIMES IGNORED.  
 0084 \*\* ICMND IS STORED IN WORD 9 OF THE EQUIPMENT TABLE,  
 0085 \*\* EQT9 IN THIS DRIVER.  
 0086 \*\*

0087 \*\*  
 0088 \*\*

NOTES:

1. USES STANDARD INTERRUPTS FOR TRANSFER OF DEVICE  
 COMMANDS AND TERMINATION SEQUENCES.

0089 \*\*  
 0090 \*\*

```

0091  **
0092  **
0093  **
0094  **
0095  **
0096  **
0097  **
0098  **
0099  **
0100  **
0101  **
0102  **
0103  ENT I.41.C.41
0104  *
0105  *
0106  ***** INITIATION SECTION FOLLOWS *****
0107  *
0108  *
0109  I.41  NOP
0110      JSB SETIO
0111      CLB
0112      LDA EQT6,I      GET CONTROL SUBFUNCTION
0113      AND =B3700
0114      STA FUNC        SAVE FUNCTION MEMORY
0115      LDA EQT6,I      GET CONTROL WORD OF REQUEST
0116      AND =B3         ISOLATE THE FUNCTION
0117      CPA =B2         CHECK FOR WRITE ( SKIP IF NOT WRITE )
0118      JMP RORWI       IF REQUEST FUNCTION = 2 (WRITE)
0119      LDB BIT15       SET READ FLAG FOR ACTUAL BIT USED
0120      CPA =B1         CHECK FOR READ (SKIP IF NOT READ )
0121      JMP RORWI       IF REQUEST FUNCTION = 1 (READ)
0122  * ANYTHING ELSE IS CONSIDERED A CONTROL REQUEST
0123      LDA FUNC        GET CONTROL WORD
0124      SZA
0125      JMP REJCT       FOR NOW ONLY CONTROL REQUEST "0" IS VALID
0126  CLR1  CLC DAT,C
0127  CLR2  CLC CST,C
0128      LDA =B4
0129      JMP I.41,I
0130  *
0131  *
0132  REJCT LDA =B2      SET A=2, FOR ILLEGAL CONTROL REQUEST
0133      JMP I.41,I    RETURN TO IOC
0134  *
0135  *
0136  *

```

```

0137 ***** SETIO ROUTINE FOR INITIATION SECTION FOLLOWS
0138 SETIO NOP
0139     STA B           SAVE SELECT CODE FROM A (STATUS CARD SC)
0140     IOR CLC        OR IN "CLC" INSTRUCTION
0141     STA CLR2
0142     STA RORW1
0143     INA           INCREMENT TO HIGHER SELECT CODE (DATA CARD SC)
0144     STA CLR1
0145     STA RORW
0146     LDA LIA       "LIA INSTRUCTION" TO A
0147     IOR B         OR IN SELECT CODE
0148     STA RORW2
0149     LDA STC       "STC INSTRUCTION" TO A
0150     IOR B
0151     STA IO1L3     STORE INSTRUCTION AT LABEL "IO1L3"
0152     LDA OTA
0153     IOR B
0154     STA IO1L2
0155     LDA OTB
0156     INB          INCREMENT SELECT CODE IN B
0157     IOR B
0158     STA IO1L1
0159     JMP SETIO,I  RETURN
0160 *****
0161 ***** READ OR WRITE REQUEST *****
0162 *****
0163 * CHECK FOR COMTAL IN REMOTE
0164 *
0165 RORWI STB RFLAG
0166 RORW  CLC DAT,C   CLEAR CONTROL AND FLAG
0167 RORW1 CLC CST,C
0168 RORW2 LIA CST     GET STATUS
0169     AND NBSY       FIX FOR COMTAL VO/20  R.W.B.
0170     CPA NBSY       FIX FOR COMTAL VO/20  R.W.B.
0171     RSS           SKIP IF YES
0172     JMP RORW2     FIX FOR COMTAL VO 20  R.W.B.
0173 *****
0174 ***** DETERMINE WHICH SUBFUNCTION *****
0175 *****
0176 ***** NOTE THAT THE DISPLAY COMMAND AND TRACK BALL REQUESTS ARE
0177 * PERFORMED USING DEVICE INTERRUPTS.
0178 * HOWEVER IMAGE LINE, FUNCTION, AND PSEUDO-COLOR TRANSFERS ARE
0179 * PERFORMED USING DEVICE INTERRUPTS TO TRANSMIT THE COMMAND AND
0180 * ENDING SEQUENCE, BUT WITH A DMA TRANSFER IN BETWEEN.
0181 *
0182     LDA FUNC       GET CONTROL WORD

```

```

0183 *
0184      CPA DSPLY      DISPLAY COMMAND REQUEST?
0185      JMP DSPLY
0186 *
0187      CPA ILTRV      IMAGE LINE TRANSFER?
0188      JMP CHDCP
0189 *
0190      CPA FMTRV      FUNCTION MEMORY TRANSFER?
0191      JMP CHDCP
0192 *
0193      CPA PCMTV      PSEUDO-COLOR TRANSFER
0194      JMP CHDCP
0195 *
0196      CPA TBALV      TRACK BALL REQUEST
0197      JMP TBALL
0198 *
0199      CPA SUPTV      CHECK FOR TYPE 3 DMA SUPPORT CONTROL
0200      JMP CHDCP
0201 *
0202      CPA LINKV      CHECK FOR LINK SUBFUNCTION
0203      JMP LINK
0204 *
0205 *????????????????? ILLEGAL SUBFUNCTION HERE ???????????????
0206 *
0207 *
0208 *XXXXXXXXXXXXXXXXX PROCESS DISPLAY COMMAND REQUEST XXXXXXXXXXXXXXXXXXXX
0209 *
0210      DSPLY LDA =D1      SET STATE CONTROL VARIABLE
0211      STA EQT12,I      WHAT TO DO
0212      LDB EQT9,I      GET OPTION
0213      CLA              CLEAR OUT A
0214      JMP IO1          GO TO START IO ROUTINE
0215 *
0216 *XXXXXXXXXXXXXXXXX PROCESS TRACK BALL REQUEST XXXXXXXXXXXXXXXXXXXX
0217 *
0218 *
0219 *
0220      TBALL LDB RFLAG
0221      CPB BIT15      READ FLAG SET? ( SKIP IF WRITE)
0222      JMP RTBAL      JUMP IF SET FOR READ
0223 *
0224 **** TRANSMIT WRITE COMMAND AND X-POSITION.
0225 *
0226      WTBAL LDA =D2      SET STATE CONTROL VARIABLE.
0227      STA EQT12,I
0228      LDB EQT7,I

```

```

0229      LDA B,I          CONTENTS OF BUFFER TO A
0230      AND =B0777      MASK FOR BITS 0-8
0231      STA B
0232      LDA =B3         TRANSFER CODE OF 3 TO A
0233      JMP IO1         GO START I/O
0234      *
0235      ***** READ TRACK BALL *****
0236      ***** ALSO READS LITTLE RED SWITCHES ON TRACK BALL APPARATUS.
0237      *
0238      RTBAL LDA =D5     SET FLAG TO TELL CONT. SECTION WHAT TO DO
0239      STA EQT12,I
0240      LDA =B3         TRANSFER CODE=3
0241      JMP IO1         GO TO START I/O
0242      *
0243      *
0244      ***** CHECK FOR DCPC CHANNEL ASSIGNED *****
0245      *
0246      CHDCP DLD INTBA,I ACCESS DCPC ASSIGNMENT WORDS FROM THE
0247      *                THE INTERRUPT TABLE WHOSE ADDRESS IS IN THE BASE PAGE
0248      *                COMMUNICATIONS AREA.
0249      CPA EQT1        IS CHANNEL 1 ASSIGNED TO THIS DRIVER
0250      JMP CH1         IF YES
0251      CPB EQT1        IS CHANNEL 2 ASSIGNED TO THIS DRIVER
0252      JMP CH2         IF YES
0253      LDA =B5         SET RETURN CODE TO 5
0254      JMP I.41,I     RETURN TO IOC REQUESTING A DCPC CHANNEL
0255      CH1  LDB =B2     LOW SELECT CODE OF CHANNEL 1 TO B
0256      RSS             SKIP
0257      CH2  LDB =B3     LOW SELECT CODE OF CHANNEL 2 TO B
0258      STB EQT11,I    STORE IN EQUIPMENT TABLE FOR USE BY COMPLETION SECTION
0259      *
0260      ***** WHICH OF THE DMS SUBFUNCTIONS *****
0261      *
0262      CHFTN LDA FUNC    CONTROL WORD TO A
0263      CPA FMTRV
0264      JMP FMTR        FUNCTION MEMORY TRANSFER
0265      CPA PCMTV
0266      JMP PCMTR       PSEUDO COLOR TRANSFER
0267      CPA SUPTV
0268      JMP SUPPT       SUPPORT TRANSFER
0269      *
0270      ***** IMAGE LINE TRANSFER ROUTINE *****
0271      *
0272      ILTR  LDA =D8     SET STATE CONTROL VARIABLE
0273      STA EQT12,I
0274      LDA EQT9,I      GET INSTRUCTIONS FOR LINE TRANSFER

```

```

0275      IOR RFLAG      SET READ OR WRITE AS PER READ FLAG
0276  ILTR2 STA B
0277      LDA =B1        SET TRANSFER CODE TO 1
0278      STA TRAN       SAVE TRANSFER CODE
0279      JMP IO1        GO START I/O
0280  *XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX
0281  * FUNCTION MEMORY OR PSUEDO COLOR TRANSFER
0282  *XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX
0283  PCMTR CLA          CLEAR A FOR PSUEDO COLOR TRANSFER
0284      JMP PCMT2      CONTINUE TRANSFER
0285  *
0286  FMTR  LDA BIT14    SET FUNCTION MEMORY BIT
0287  *
0288  PCMT2 LDB =D8      SET STATE VARIABLE
0289      STB EQT12,I    SAVE STATE FOR CONTINUATION
0290      IOR EQT9,I     MERGE IN IOPTN1 WORD
0291      IOR BIT13     MERGE IN 8000S COMPATIBILITY BIT
0292      IOR RFLAG     MERGE IN READ OR WRITE BIT
0293  FMTR2 STA B        PLACE CONTROL WORD IN B
0294      LDA =B2        SET TRANSFER CODE
0295      STA TRAN       SAVE TRANSFER CODE FOR LATER
0296      JMP IO1
0297  *XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX
0298  * S U P P O R T   F U N C T I O N   B L O C K   T R A N S F E R   (   T R   C O D E   =   3   )
0299  *XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX
0300  SUPPT LDA =D8      SET DMA TRANSFER RETURN STATUS FOR CONTINUATION SECTION
0301      STA EQT12,I   AND SAVE IT
0302      LDA EQT9,I    GET OPTIONS
0303      IOR RFLAG     SET READ OR WRITE BIT
0304      STA B         SAVE IN B
0305      LDA =B3       SET TRANSFER CODE
0306      STA TRAN      SAVE TRANSFER CODE
0307      JMP IO1      AND GO DO IT
0308  *
0309  * S P E C I A L   P R O C E S S I N G   F O R   T R   1   M O D E   6   C A L L
0310  *
0311  LINK  LDA =D1      SET NO BLOCK TRANSFER
0312      STA EQT12,I   SAVE THIS
0313      LDB EQT9,I    GET COMMAND OPTION
0314      LDA =B1       SET TRANSFER CODE
0315      JMP IO1       GO DO IT
0316  *
0317  *
0318  *
0319  * S P E C I A L   T H I S   R O U T I N E   S T A R T S   T H E   D A T A   T R A N S F E R   O F   A   C O M M A N D
0320  *           T O   T H E   C O M T A L

```



```

0321 *
0322 *
0323 I01  NOP
0324 I01L1 OTB DAT      PUT TRANSFER COMMAND ON DATA LINES
0325      IOR GOBIT
0326 I01L2 OTA CST      PUT TRANSFER CODE ON COMMAND LINES
0327 I01L3 STC CST,C    SEND THE GO PULSE , START TRANSFER
0328      CLA          NOW RETURN TO IOC WITH
0329      JMP I.41,I    OPERATION INITIATED (A=0=OK)
0330 *
0331 *
0332 *
0333 *
0334 *
0335 *
0336 *****      COMPLETION SECTION FOLLOWS      *****
0337 *
0338 *
0339 *
0340 C.41  NOP
0341      JSB CS10
0342      LDA EQT1,I    CHECK FOR SPURIOUS INTERUPT
0343      AND =B77777   GET I/O REQUEST LIST POINTER
0344      SZA          IS A REQUEST IN PROGRESS
0345      JMP COMP2     IF YES GO PROCESS REQUEST
0346      STA EQT15,I   NO,ITS SPURIOUS SO ZERO TIME-OUT CLOCK TO PREVENT TIME-OUT
0347 SPUR2 ISZ C.41    ADJUST RETURN TO P+2 (CONT.)
0348      JMP C.41,I    MAKE CONTINUATION RETURN TO CIC
0349 *
0350 *
0351 COMP2 NOP
0352      LDA CTABA     STATE CONTROL TABLE BASE ADDRESS TO A
0353      ADA EQT12,I   ADD THE STATE CONTROL VARIABLE
0354      JMP A,I       JUMP TO ADDRESS JUMP TABLE
0355 *
0356 ***** DISPLAY COMMAND COMPLETION.
0357 *
0358 SCA1  LDA IDSC     INTERRUPTING DEVICE SELECT CODE TO A
0359      CPA SSC       IS INTERRUPTING SELECT CODE THE STATUS CARD?
0360      RSS          SKIP IF YES?
0361      JMP SPUR2     JUMP TO SPURIOUS INTERUPT IF NOT
0362      LDB =B1       SET B FOR TRANS. LOG
0363      JMP CEND1
0364 *
0365 ***** TRACK BALL 1
0366 *

```

0367	SCA2	LDA IDSC	INTERUPTING DEVICE SELECT CODE TO A
0368		CPA SSC	IS INTERUPTING SELECT CODE THE STATUS CARD?
0369		RSS	SKIP IF YES?
0370		JMP SPUR2	JUMP TO SPURIOUS INTERUPT RETURN IF NOT@
0371		LDA =D3	
0372		STA EQT12.I	SET STATE CONTROL VAR.
0373		LDB EQT7.I	ADDRESS OF BUFFER TO B
0374		INB	INCREMENT B (BUFFER ADDRESS)
0375		LDA B.I	CONTENTS OF BUFFER+1 TO A
0376		AND =B0777	
0377		IOR BIT14	SIGNIFYS Y-POSITION
0378		STA B	
0379		LDA =B3	TRANSFER CODE OF 3
0380		IOR GOBIT	
0381		JMP CIO1	
0382	*		
0383	**	TRACK BALL 2	
0384	*		
0385	SCA3	LDA IDSC	INTERUPTING DEVICE SELECT CODE TO A
0386		CPA SSC	IS INTERUPTING SELECT CODE THE STATUS CARD?
0387		RSS	SKIP IF YES?
0388		JMP SPUR2	JUMP TO SPURIOUS INTERUPT IF NOT SET
0389		LDA =D4	
0390		STA EQT12.I	SET STATE CONTROL VARIABLE
0391		JMP TBDUN	
0392	*		
0393	**	TRACK BALL 3	
0394	*		
0395	SCA5	LDA IDSC	INTERUPTING DEVICE SELECT CODE TO A
0396		CPA SSC	IS INTERUPTING SELECT CODE THE STATUS CARD?
0397		RSS	SKIP IF YES?
0398		JMP SPUR2	JUMP TO SPURIOUS INTERUPT IF NOT
0399		LDA =D6	
0400		STA EQT12.I	SET STATE CONTROL VARIABLE
0401		JMP CIO2	JUMP TO START A READ
0402	*		
0403	**	TRACK BALL 4	
0404	*		
0405	SCA6	LDA IDSC	INTERUPTING DEVICE SELECT CODE TO A
0406		CPA DSC	IS INTERUPTING SELECT CODE THE DSCA CARD?
0407		RSS	SKIP IF YES?
0408		JMP SPUR2	JUMP TO SPURIOUS INTERUPT IF NOT
0409		LDA =D7	
0410		STA EQT12.I	SET STATE CONTROL VARIABLE
0411	GET1	LIA DAT	DATA FROM DATA LINES TO A (X-POSITION &SWITCH 1)
0412	*	STA EQT13.I	SAVE FOR SWITCH 1

```

0413 *      AND =B0777
0414      LDB EQT7,I      ADDRESS OF BUFFER TO B
0415      STA B,I        STORE X-POSITION IN BUFFER
0416      JMP CIO2
0417 *
0418 ** TRACKBALL 5
0419 *
0420 SCA7 LDA IDSC      INTERRUPTING DEVICE SELECT CODE TO A
0421      CPA DSC      IS INTERRUPTING SELECT CODE THE DSCA CARD?
0422      RSS          SKIP IF YES?
0423      JMP SPUR2    JUMP TO SPURIOUS INTERUPT IF NOT
0424      LDA =D4
0425      STA EQT12,I  SET STATE CONTROL VARIABLE
0426 GET2 LIA DAT      DATA FROM DATA LINES TO A
0427      LDB EQT7,I  GET BUFFER ADDRESS
0428      INB          POINT TO WORD TWO
0429      STA B,I      STORE Y-POSITION IN BUFFER+1
0430      JMP TBDUN
0431 *
0432 TBDUN LDA =B3
0433      IOR CLBIT    THIS ADDED P0ER RWB 6/15/81
0434      IOR ENBIT
0435      JMP CIO3
0436 *
0437 ** TRACK BALL TRANSMIT END
0438 *
0439 SCA4 LDA IDSC      INTERRUPTING DEVICE SELECT CODE TO A
0440      CPA SSC      IS INTERRUPTING SELECT CODE THE STATUS CARD?
0441      RSS          SKIP IF YES?
0442      JMP SPUR2    JUMP TO SPURIOUS INTERUPT IF NOT
0443      LDB =B2      SET TRANS. LOG FOR WRITE
0444      LDA EQT6,I  GET CONTROL WORD
0445      AND =B3      ISOLATE FUNCTION CODE
0446      CPA =B1      IS IT A READ?
0447      LDB =B4      SET TRANS. LOG TO 4 FOR READ
0448      JMP CEND1
0449 *
0450 *
0451 *
0452 CIO1 NOP
0453      OTB DAT
0454 CIO3 OTA CST
0455      STC CST,C
0456      ISZ C.41
0457      JMP C.41,I
0458 *

```

```

0459 *
0460 *
0461 C102 NOP
0462 CLC DAT.C
0463 STC DAT.C
0464 ISZ C.41
0465 JMP C.41.I
0466 *
0467 ** DMA TRANSFER 1
0468 *
0469 SCAB LDA IDSC INTERRUPTING DEVICE SELECT CODE TO A
0470 CPA SSC IS INTERRUPTING SELECT CODE THE STATUS CARD?
0471 RSS SKIP IF YES?
0472 JMP SPUR2 JUMP TO SPURIOUS INTERUPT IF NOT
0473 LDA =D9
0474 STA EQT12.I SET STATE CONTROL VARIABLE
0475 *
0476 *
0477 *
0478 ***** THIS ROUTINE MODIFIES THE DCPC INITIALIZATION INSTRUCTIONS
0479 * TO SPECIFY THE SELECT CODES OF THE ASSIGNED DCPC
0480 *
0481 STDMA LDB EQT11.I GET LOW SELECT CODE FROM EQT
0482 LDA OTA "OTA INSTRUCTION" TO A
0483 IOR B OR IN THE LOWER SELECT CODE
0484 STA D3 STORE AT LABEL D3
0485 STA D5 STORE AT LABEL D5
0486 ADA =B4 ADD 4 TO INSTR. TO CHANGE TO HIGHER SELECT CODE
0487 STA D1
0488 *
0489 LDA CLC "CLC INSTR." TO A
0490 IOR B OR IN LOWER SELECT CODE
0491 STA D2
0492 LDA STC
0493 IOR B
0494 STA D4
0495 ADA =B4
0496 STA D6
0497 STA D7
0498 LDA CLC
0499 IOR DSC
0500 STA D6+1
0501 LDA STF
0502 IOR DSC
0503 STA D6+2
0504 LDA STC

```

```

0505      IOR DSC
0506      STA D7+1
0507      *
0508      *
0509      *
0510      ***** THIS ROUTINE INITIATES THE DCPC DATA TRANSFERS *****
0511      *** THE INSTRUCTIONS BELOW WITH D LABELS ARE MODIFIED BY THE
0512      * THE ROUTINE "SETIO" TO SPECIFY THE CORRECT SELECT CODES
0513      * FOR THE DCPC CHANNEL ASSIGNED BY THE SYSTEM
0514      *
0515      GOIO LDA EQT4,I
0516          AND =B77      MASK FOR COMMAND SELECT CODE
0517          INA          INCREMENT FOR DATA SELECT CODE
0518          IOR BIT15   TURN ON HANDSHAKE BIT (STC)
0519      D1  OTA 6B      PUT CONTROL WORD 1 TO DCPC CHANNEL (SELECT 6 OR 7)
0520      D2  CLC 2B      PREPARE MEM. ADDR. REGISTER FOR CW2 (SELECT CODE 2 OR 3)
0521          LDA EQT6,I  GET CONTROL WORD
0522          AND =B3      MASK FOR FUNCTION
0523          STA B        STORE FUNCTION IN B
0524          LDA EQT7,I  ADDRESS OF BUFFER TO A
0525          CPB =B1      IS FUNCTION A READ?
0526          IOR BIT15   TURN ON "IN" BIT FOR A READ, IF YES
0527      D3  OTA 2B      CONTROL WORD 2 TO DCPC CHANNEL (SC 2 OR 3)
0528      D4  STC 2B      PREPARE MEM. ADDR. REGISTER FOR CW3 (SC 2 OR 3)
0529          LDA EQT8,I  GET LENGTH OF TRANSFER
0530          CMA,INA     MAKE TWO'S COMPLEMENT
0531      D5  OTA 2B      CONTROL WORD 3 TO DCPC CHANNEL (SELECT CODE 2 OR 3)
0532          CPB =B1      IS IT A READ?
0533          JMP D7       JUMP IF YES
0534      D6  STC 6B      ACTIVATE DCPC CHANNEL (SELECT CODE 6 OR 7)
0535          CLC DAT      CLEAR DEVICE
0536          STF DAT      ACTIVATE DEVICE?
0537          JMP GOEND
0538      D7  STC 6B      ACTIVATE DCPC CHANNEL (SELECT CODE 6 OR 7)
0539          STC DAT,C    ACTIVATE DEVICE
0540      GOEND ISZ C.41  NOW RETURN TO CIC WITH
0541          JMP C.41,I   CONTINUATION
0542      *
0543      ** DMA TRANSFER 2
0544      *
0545      SCA9 LDA EQT11,I GET LOW SELECT CODE OF DCPC CHANNEL
0546          ADA =B4      ADD 4 TO CONVERT LOW DCPC SELECT CODE TO HIGH S.C.
0547      CHKDF CPA IDSC  IS INTERRUPTING SELECT CODE THAT OF THE
0548      *              ASSIGNED DCPC CHANNEL?
0549          RSS          SKIP IF YES
0550          JMP SPUR2    JUMP TO SPURIOUS INTERUPT IF NOT

```

```

0551      LDA =D10      SET CONTROL STATE VARIABLE
0552      STA EQT12,I
0553      *
0554      *
0555      *
0556      *
0557      LDA TRAN
0558      IOR CLBIT      * SET CLEAR BIT
0559      IOR ENBIT      * ADD END BIT
0560      JMP CIO3      * GO TO IT
0561      *
0562      ** DMA TRANSFER 3
0563      *
0564      *
0565      SCA10 LDA IDSC      INTERRUPTING DEVICE SELECT CODE TO A
0566      CPA SSC      IS INTERRUPTING SELECT CODE THE STATUS CARD?
0567      RSS      SKIP IF YES?
0568      JMP SPUR2      IF NOT GO TO SPURIOUS INTERRUPT
0569      LDB EQT11,I      RETRIEVE LOWER SELECT CODE OF ASSIGNED DCPC CHANNEL
0570      LDA CLC      "CLC INSTRUCTION" TO A
0571      IOR B      OR IN THE CHANNELS LOWER SELECT CODE
0572      STA CD1      STORE INSTRUCTION AT LABEL CD1
0573      ADA =B4      ADD 4 TO INSTRUCTION IN A TO ADJUST TO HIGHER SELECT CODE
0574      STA CD2      STORE INSTRUCTION AT LABEL CD2
0575      *
0576      CD1 CLC 2B,C      CLEAR LOW DCPC SELECT CODE
0577      CD2 CLC 6B,C      CLEAR HIGH DCPC SELECT CODE
0578      LDB EQT8,I      LENGTH OF BUFFER TO B (TRANSMISSION LOG)
0579      CEND1 LIA CST      GET STATUS WORD FROM COMTAL
0580      AND =B37      STRIP OFF UNUSED BITS
0581      STA SAVE1      SAVE IN SAVE1 TEMPORARILY
0582      LDA EQT5,I      REMOVE PREVIOUS STATUS
0583      AND =B177400      BITS IN EQT WORD 5
0584      IOR SAVE1      OR IN NEW BITS
0585      STA EQT5,I      AND RESET INTO EQT WORD 5
0586      *
0587      CEND2 CLC DAT,C      CLEAR DEVICE DATA SELECT CODE
0588      CLC CST,C      CLEAR DEVICE COMMAND SELECT CODE
0589      *
0590      *
0591      CLA      SET A = 0 = OK RETURN CODE
0592      IOR =B100000      SET BIT TO RETURN DCPC CHANNEL
0593      JMP C.41,I      MAKE COMPLETION RETURN TO C1C
0594      *
0595      *
0596      *
0596      ***** SETIO SUBROUTINE FOR COMPLETION/CONTINUATION SECTION *****

```

```

0597 *
0598 CSIO NOP
0599 STA IDSC SAVE SELECT CODE OF INTERRUPTING DEVICE
0600 LDA EQT4,I
0601 AND =B77 MASK FOR STATUS SELECT CODE
0602 STA SSC SAVE STATUS SELECT CODE AT SSC
0603 INA ADD 1 TO A FOR DATA SELECT CODE
0604 STA DSC SAVE DSCA SELECT CODE AT DSC
0605 LDA OTA "OTA INSTRUCTION" TO A
0606 IOR SSC OR IN STATUS SELECT CODE
0607 STA CIO3 STORE INSTRUCTION IN A AT LABEL CIO3
0608 LDA STC
0609 IOR SSC
0610 STA CIO3+1
0611 INA
0612 * ADD 1 TO INSTRUCTION IN A TO CHANGE
STATUS SELECT CODE TO DATA SELECT CODE
0613 STA CIO2+2
0614 LDA CLC
0615 IOR SSC
0616 STA CEND2+1
0617 INA
0618 STA CEND2
0619 STA CIO2+1
0620 LDA LIA
0621 IOR SSC
0622 STA CEND1
0623 INA
0624 STA GET1
0625 STA GET2
0626 LDA OTB
0627 IOR DSC
0628 STA CIO1+1
0629 JMP CSIO,I RETURN
0630 *
0631 *
0632 *
0633 *
0634 *
0635 *
0636 ***** CONSTANTS AND STORAGE AREA *****
0637 *
0638 RFLAG BSS 1
0639 FUNC BSS 1
0640 TRAN BSS 1
0641 SAVE1 BSS 1
0642 SAVE2 BSS 1

```

```

0643 CW1 BSS 1
0644 CW2 BSS 1
0645 CW3 BSS 1
0646 TMASK BSS 1
0647 IDSC BSS 1
0648 SSC BSS 1
0649 DSC BSS 1
0650 BIT13 OCT 20000
0651 BIT14 OCT 40000
0652 BIT15 OCT 100000
0653 RMDTE OCT 2
0654 GOBIT OCT 4
0655 ENBIT OCT 10
0656 NBSY OCT 10
0657 CLBIT OCT 20
0658 TOUT DEC -30000
0659 ERNUM DEC 3
0660 **
0661 ** THE FOLLOWING VALUES (SUFFIX V) DETERMINE WHICH DRIVER
0662 ** SUBFUNCTIONS ARE INVOKED. PARAMETER IFUNC IN SAMPLE CALL ABOVE.
0663 **
0664 DSPLV OCT 0
0665 ILTRV OCT 100
0666 FMTRV OCT 200
0667 PCMTV OCT 300
0668 TBALV OCT 400
0669 SUPTV OCT 500
0670 LINKV OCT 600
0671 **
0672 CST EQU 24
0673 DAT EQU 25
0674 A EQU 0
0675 B EQU 1
0676 OTA OTA 0
0677 CLC CLC 0.C
0678 STC STC 0.C
0679 LIA LIA 0
0680 OTB OTB 0
0681 SFS SFS 0
0682 STF STF 0
0683 *****BASE PAGE COMMUNICATIONS AREA DEFINITIONS
0684 . EQU 1650B
0685 *
0686 **
0687 ** THE FOLLOWING CONSTANTS ARE USED TO READ THE EQUIPMENT TABLE
0688 ** ASSOCIATED WITH THIS DRIVER. SEE THE HP-1000 QUICK REFERENCE

```

USED ONLY FOR DOCUMENTATION. ACTUAL INSTRUCTION IS SET UP BY A SETIO ROUTINE.



0689 \*\* MANUAL FOR DIAGRAMS AND TABLES ON WHICH THESE CONSTANTS ARE BASED.

0690 \*\*

0691 INTBA EQU .+4

0692 EQT1 EQU .+8

0693 EQT2 EQU .+9

0694 EQT3 EQU .+10

0695 EQT4 EQU .+11

0696 EQT5 EQU .+12

0697 EQT6 EQU .+13

0698 EQT7 EQU .+14

0699 EQT8 EQU .+15

0700 EQT9 EQU .+16

0701 EQT10 EQU .+17

0702 EQT11 EQU .+18

0703 EQT12 EQU .+81

0704 EQT13 EQU .+82

0705 EQT14 EQU .+83

0706 EQT15 EQU .+84

0707 \*

0708 \*

0709 \*

0710 \*\*\*\*\* STATE CONTROL ADDRESS TABLE \*\*\*\*\*

0711 CTABA DEF CTAB

0712 CTAB NOP

0713 JMP SCA1 DISPLAY COMMAND COMPLETION

0714 JMP SCA2 TRACK BALL 1

0715 JMP SCA3 TRACK BALL 2

0716 JMP SCA4 TRACK BALL TRANSMIT END

0717 JMP SCA5 TRACK BALL 3

0718 JMP SCA6 TRACK BALL 4

0719 JMP SCA7 TRACK BALL 5

0720 JMP SCAB DMA TRANSFER 1

0721 JMP SCA9 DMA TRANSFER 2

0722 JMP SCA10 DMA TRANSFER 3

0723 \* BSS 10

0724 END

Standard Bibliographic Page

1. Report No. NASA TM-87646		2. Government Accession No.		3. Recipient's Catalog No.	
4. Title and Subtitle IPLIB (Image Processing Library) User's Manual				5. Report Date December 1985	
				6. Performing Organization Code 505-61-01-05	
7. Author(s) Nettie D. Faulcon, James H. Monteith, and Keith Miller				8. Performing Organization Report No.	
				10. Work Unit No.	
9. Performing Organization Name and Address NASA Langley Research Center Hampton, VA 23665-5225				11. Contract or Grant No.	
				13. Type of Report and Period Covered Technical Memorandum	
12. Sponsoring Agency Name and Address National Aeronautics and Space Administration Washington, DC 20546				14. Sponsoring Agency Code	
				15. Supplementary Notes	
16. Abstract  IPLIB is a collection of HP FORTRAN 77 subroutines and functions that facilitate the use of a COMTAL image processing system driven by an HP-1000 computer. It is intended for programmers who want to use the HP 1000 to drive the COMTAL Vision One/20 system. It is assumed that the programmer knows HP 1000 FORTRAN 77 or at least one FORTRAN dialect. It is also assumed that the programmer has some familiarity with the COMTAL Vision One/20 system.					
17. Key Words (Suggested by Authors(s)) COMTAL Image Processing System HP 1000 Computer FORTRAN 77			18. Distribution Statement  Unclassified - Unlimited  Subject Category - 61		
19. Security Classif.(of this report) Unclassified		20. Security Classif.(of this page) Unclassified		21. No. of Pages 216	22. Price A10

For sale by the National Technical Information Service, Springfield, Virginia 22161



