T H E U N I V E R S I T Y O F M I C H I G A N

Memorandum 23

THE DF ROUTINES USER'S GUIDE

Alfred B. Cocanower

CONCOMP: Research in Conversational Use of Computers
ORA Project 07449
F.H. Westervelt, Director

supported by:

DEPARTMENT OF DEFENSE
ADVANCED RESEARCH PROJECTS AGENCY
WASHINGTON, D.C.

CONTRACT NO. DA-49-083 OSA-3050
ARPA ORDER NO. 716

administered through:

OFFICE OF RESEARCH ADMINISTRATION ANN ARBOR

May 1969

ABSTRACT

The DF routine package allows a IBM 360/67 programmer to assemble DEC 338 display files, send them to the DEC 338, control the displaying of these files, and enable light-pen service. The DF routines provide a set of elementary display file operations only; they do not provide complex operations such as producing an entire graph with an axis, labels, and curve with one subroutine call. However, subroutines that allow the displaying of graphs with one subroutine call can be written using the DF routines.

Topics discussed in this report include: elementary example programs, DF routine summaries, 338 display file organization and transmission format, display file assembly parameters and characteristics, and instructions for debugging programs from a TTY. A typical $RUN command for a program containing calls on the DF routines and pseudo-octal numbers are also discussed.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

1. INTRODUCTION

The DF routine package allows an IBM 360/67 programmer to assemble DEC 338 display files, send them to the DEC 338, control the displaying of these files, and enable light-pen service. The DF routines provide a set of elementary display file operations only; they do not provide complex operations such as producing an entire graph with an axis, labels, and curve with one subroutine call. However, subroutines that allow the displaying of graphs with one subroutine call can be written using the DF routines.

Most of the remaining topics for this report are discussed in the appendices. They are: five elementary example programs (Appendix A), DF routine summaries (Appendix B), 338 display file organization and transmission format (Appendix C), display file assembly parameters and characteristics (Appendix D), and instructions for debugging programs from a TTY (Appendix E). In the remaining paragraphs, a typical $RUN command for a program containing calls on the DF routines and pseudo octal numbers will be discussed.

2. $RUN EXAMPLE

Before considering the $RUN of an example program with calls on DF routines, consider the diagram of the hardware for the 360/67 and 338 (Figure 1). The 338 terminal with a TTY connected responds like any other TTY connected to the 360/67. The essential difference is the variety of input and output devices that is available to augment the TTY at the 338, for example, the display and light pen.

Now consider the execution of Example 1 in Appendix A. Assume that the object module for Example 1 is in a file named EX#1. The object modules for the DF routines are available in SAVE:DF. The $RUN is as follows:

```
$RUN EX#1+SAVE:DF
```

The default values for the assignment of the logical I/O devices are used, that is, SCARDS=*SOURCE* and SPRINT=*SINK*. The DF routines transmit display files from the 360/67 to the 338 via sink *SINK*, and receive information from the 338 via source, *SOURCE*.

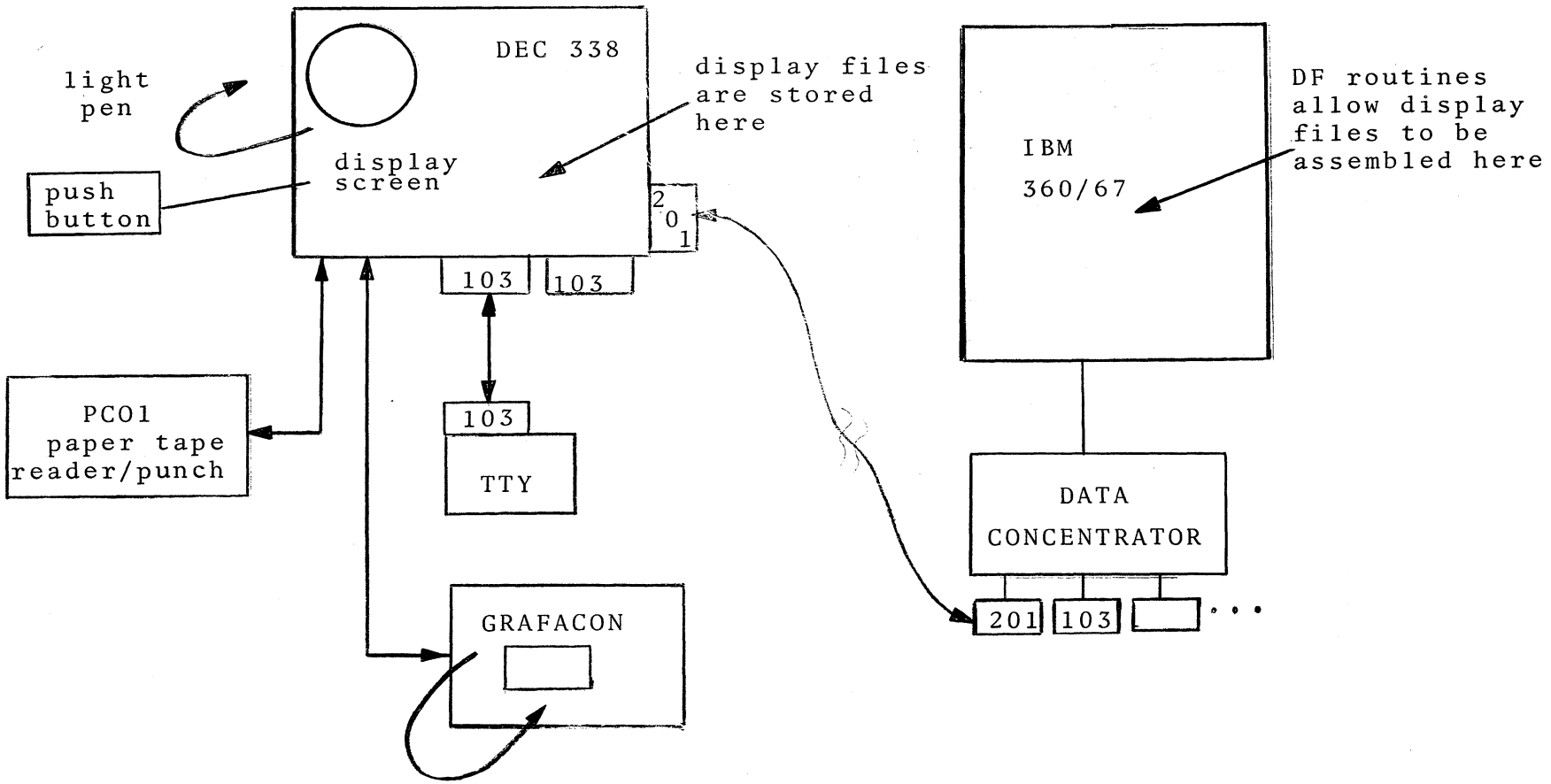

Figure 1. 338/67 Configuration

3. PSEUDO OCTAL NUMBERS

Since the structure of display files (see Appendix C) is based on 12-bit words, with each word represented by four octal digits, the natural number base for the arguments of some DF routines is octal.

To simplify programming for the user, these arguments are to be expressed as if the number base for these arguments were octal (e.g., 0417_8) and not decimal (e.g., $0417_8 = 271_{10}$). While FORTRAN treats these arguments as if they were decimal during their conversion to binary at compile time, the DF routines have been coded to expect this undesired decimal conversion and the number is reprocessed to obtain the correct value.

ACKNOWLEDGMENTS

R. Johnston deserves special recognition for his pioneering work in the DF routine concept and coding. The continuing efforts of W.S. Gerstenberger in providing a communication link between the 360/67 and the 338 are much appreciated.

APPENDIX A. FIVE ELEMENTARY EXAMPLE PROBLEMS

Five example programs using DF routines are given in this appendix. Photographs of the display screen are included for each example.

```
>   FORTRAN IV G COMPILER        MAIN            03-04-69        15:55.21        PAGE 0001

>                C       DISPLAY EXAMPLE #1, DRAW A 2" BY 2" SQUARE ON THE DISPLAY SCREEN.
>                C
>                C         INITIALIZE THE DISPLAY FILE CONSTRUCTION BUFFER IN THE IBM/360
>                C
>                C       AND START THE DISPLAY FILE WITH THE STANDARD PAR AND MODE.
>                C
>   0001               CALL DFINI(0,0)
>                C
>                C         MOVE THE LIGHT BEAM FROM THE LOWER LEFT HAND CORNER OF THE
>                C
>                C       DISPLAY SCREEN (STANDARD INITIAL POSITION OF THE LIGHT BEAM) TO
>                C
>                C       THE LOWER LEFT HAND CORNER OF THE SQUARE.  THAT IS, DRAW AN
>                C
>                C       INVISIBLE VECTOR TO (1.0,1.0), X AND Y, IN INCHES RELATIVE TO
>                C
>                C       THE LOWER LEFT HAND CORNER OF THE DISPLAY SCREEN; THE THIRD
>                C
>                C       ARGUMENT BEING ZERO IMPLIES THAT THE VECTOR IS TO BE INVISIBLE
>                C
>                C       OR UNINTENSIFIED.
>                C
>   0002               CALL DFXYC(1.0,1.0,0)
>                C
>                C         NOW DRAW THE LEFT SIDE OF THE SQUARE; THIRD ARGUMENT NOT ZERO
>                C
>                C       IMPLIES VISIBLE OR INTENSIFIED VECTOR.
>                C
>   0003               CALL DFXYC(1.0,3.0,1)
>                C
>                C         NOW DRAW THE TOP, RIGHT SIDE, AND BOTTOM OF THE SQUARE.
>                C
>   0004               CALL DFXYC(3.0,3.0,1)
>   0005               CALL DFXYC(3.0,1.0,1)
>   0006               CALL DFXYC(1.0,1.0,1)
>                C
>                C         NOW SEND THE DISPLAY FILE TO THE 338.  THE FIRST ARGUMENT FOR
>                C
>                C       DF201 IS A SET OF FOUR BINARY SWITCHES; THE ZERO MEANS:
>                C
>                C               THE DISPLAY FILE IS UNBLANKED OR IS TO BE DISPLAYED,
>                C
>                C               THE DISPLAY FILE IS NOT A SUBROUTINE DISPLAY FILE,
>                C
>                C               THE DISPLAY FILE IS NOT TO BE ERASED, AND
>                C
>                C               THE DISPLAY FILE IS NOT LIGHT PEN SENSITIVE.
>                C
>                C       THE SECOND ARGUMENT IS THE NAME (AN INTEGER) OF THE DISPLAY FILE.
>                C
>   0007               CALL DF201(0,1)
>   0008               CALL SYSTEM
>   0009               END
```

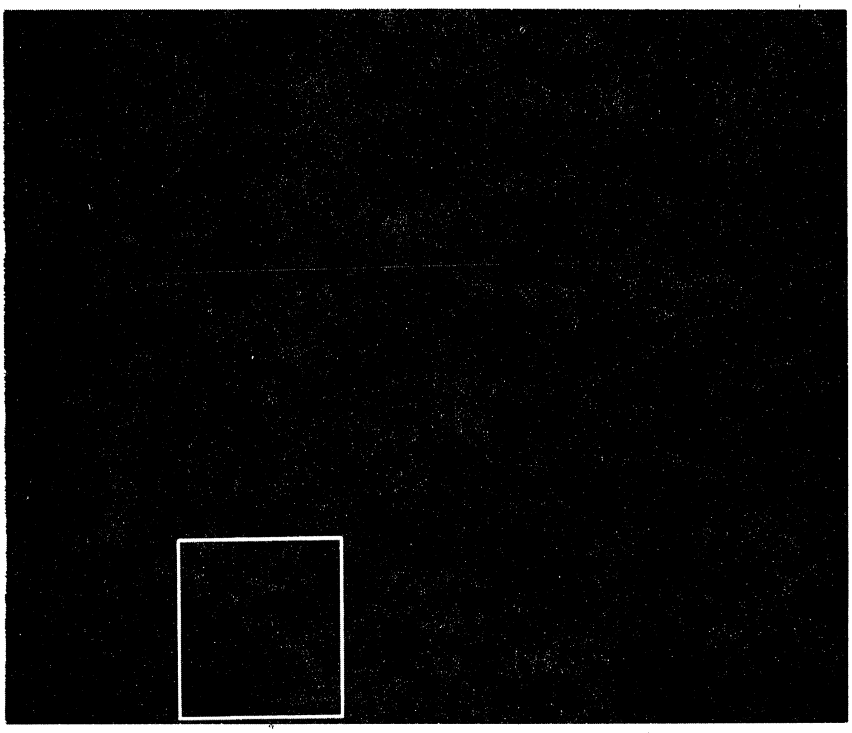

Figure 1A. Output for Example 1.

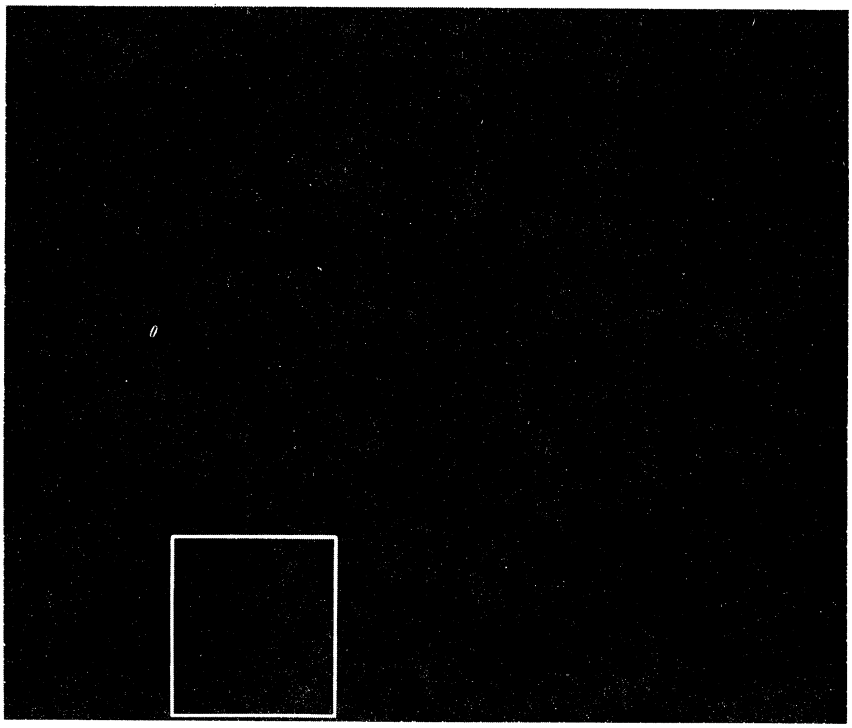

Figure 2A. Output for Example 2.
(Identical to Figure 1A.)

```
>   FORTRAN IV G COMPILER        MAIN            03-04-69        15:55.23        PAGE 0001

>                C       DISPLAY EXAMPLE #2, DRAW A 2" BY 2" SQUARE USING DFXYV.
>                C
>                C         DEFINE ALL THE X AND Y COORDINATES AND THE INTENSIFICATION
>                C
>                C       VECTOR.  NOTE THAT THE I VECTOR IS ONE WORD LONGER THAN THE X
>                C
>                C       OR Y VECTORS; THE LAST ENTRY IN THE I VECTOR IS LESS
>                C
>                C       THAN ZERO AND IMPLIES THAT THE PREVIOUS X, Y, I TRIPLE WAS THE
>                C
>                C       FINAL DATA VALUE TO BE PLOTTED.
>                C
>    0001              REAL X(5)/1.0,1.0,3.0,3.0,1.0/
>    0002              REAL Y(5)/1.0,3.0,3.0,1.0,1.0/
>    0003              INTEGER I(6)/0,1,1,1,1,-1/
>                C
>                C         INITIALIZE-
>                C
>    0004              CALL DFINI(0,0)
>                C
>                C         DRAW SQUARE-
>                C
>    0005              CALL DFXYV(X,Y,I)
>                C
>                C         SEND TO 338-
>                C
>    0006              CALL DF201(0,2)
>    0007              CALL SYSTEM
>    0008              END
```

```
FORTRAN IV G COMPILER        MAIN            03-04-69        15:55.24        PAGE 0001

>                C       DISPLAY EXAMPLE #3, PUT A LABEL ON THE DISPLAY SCREEN.
>                C
>                C         THE FIRST TWO ARGUMENTS IN DFLAB ARE THE X AND Y COORDINATES
>                C
>                C       OF THE LOWER LEFT HAND CORNER OF THE FIRST CHARACTER IN THE
>                C
>                C       STRING RELATIVE TO THE LOWER LEFT HAND CORNER OF THE DISPLAY
>                C
>                C       SCREEN.
>                C
>                C         THE NEXT ARGUMENT IS THE CHARACTER STRING TO BE DISPLAYED;
>                C
>                C       NOTE THAT THE STRING IS POSTFIXED WITH AN "@".
>                C
>                C         THE FOURTH ARGUMENT IS A RELATIVE CHARACTER SIZE INDICATOR, IN
>                C
>                C       THIS CASE, THE LARGEST SIZE.
>                C
>                C         THE LAST ARGUMENT IS THE HORIZONTAL/VERTICAL ORIENTATION SWITCH
>                C
>                C       FOR THE STRING; ZERO IMPLIES HORIZONTAL, NOT ZERO IMPLIES VERTICAL.
>                C
>    0001              CALL DFLAB(3.0,2.0,'HI THERE@',3,0)
>                C
>                C       NOW SEND THE LABEL TO THE 338.
>                C
>    0002              CALL DF201(0,3)
>                C
>                C          NOW FOR A VERTICAL LABEL; THE CHARACTER DATA IS IN AN ARRAY
>                C
>    0003              INTEGER LABEL(2)/'CONC','OMP@'/
>                C
>    0004              CALL DFLAB(2.0,8.0,LABEL,2,1)
>    0003              CALL DF201(0,30)
>    0006              CALL SYSTEM
>    0007              END
```

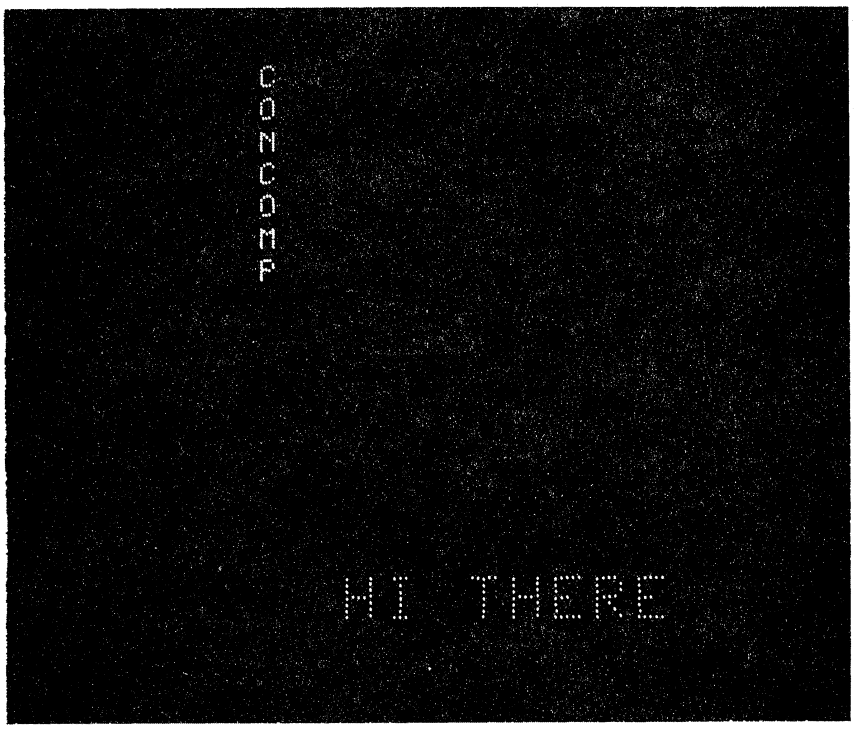

Figure 3A. Output for Example 3.

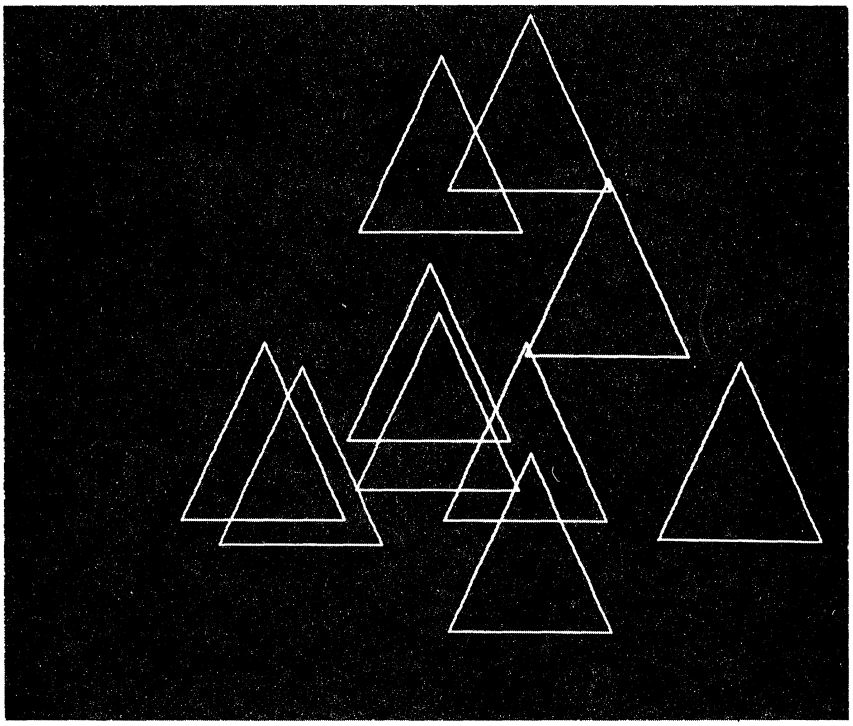

Figure 4A. Output for Example 4.

```
FORTRAN IV G COMPILER        MAIN            03-04-69        15:55.25        PAGE 0001
>                C       DISPLAY EXAMPLE #4, USING SUBROUTINE DISPLAY FILES AND DFPJM.
>                C
>                C         THE OBJECTIVE IS TO PLACE TEN IDENTICAL TRIANGLES AT RANDOM
>                C
>                C       LOCATIONS ON THE DISPLAY SCREEN.  ONE SOLUTION WOULD BE TO
>                C
>                C       ASSEMBLE TEN DISPLAY FILES OF TRIANGLES AND RANDOMLY LOCATE
>                C
>                C       THEM ON THE DISPLAY SCREEN.  THE SOLUTION GIVEN HERE WILL DEFINE
>                C
>                C       ONE SUBROUTINE DISPLAY FILE OF THE TRIANGLE AND ONE DISPLAY FILE
>                C
>                C       OF CALLS OR PUSH JUMPS TO THIS TRIANGLE.
>                C
>                C         CONSTRUCT THE TRIANGLE.
>                C
>    0001              REAL X(3)/1.0,2.0,0.0/,Y(3)/2.0,0.0,0.0/
>    0002              INTEGER I(4)/1,1,1,-1/
>                C
>                C         THE STANDARD SECOND ARGUMENT IN DFINI (MODE) DOES MANY THINGS
>                C
>                C       INCLUDING THE REPOSITIONING OF THE LIGHT BEAM TO THE LOWER LEFT
>                C
>                C       HAND CORNER OF THE DISPLAY SCREEN.  THIS CONDITION IS NOT WANTED
>                C
>                C       FOR THIS EXAMPLE.  IN FACT, THE SAME DISPLAY FILE IS TO BE
>                C
>                C       DISPLAYED AT TEN DIFFERENT LOCATIONS ON THE DISPLAY SCREEN.  THE
>                C
>                C       VALUE, 1121, FOR THE MODE DOES NOT ALLOW THE LIGHT BEAM TO BE
>                C
>                C       REPOSITIONED AT THE LOWER LEFT HAND CORNER OF THE DISPLAY SCREEN
>                C
>                C       FOR EACH CALL ON THE TRIANGLE AMONG OTHER THINGS.
>                C
>    0003              CALL DFINI(0,1121)
>    0004              CALL DFXYV(X,Y,I)
>                C
>                C         THE FIRST ARGUMENT IN DF201 IMPLIES THAT THE DISPLAY FILE OF
>                C
>                C       THE TRIANGLE IS TO BE A SUBROUTINE.
>                C
>    0005              CALL DF201(2,4)
>                C
>                C         NOW TO CONSTRUCT THE CALLS ON THE TRIANGLE.
>                C
>                C         FIRST RE-INITIALIZE THE DISPLAY FILE CONSTRUCTION BUFFER.
>                C
>    0006              CALL DFINI(0,0)
>                C
>    0007              DATA LAST/192837465/
>                C
>    0008              DO 100 J=1,10
>                C
>                C         GET RANDOM VALUES FOR X AND Y.
```

```
>    FORTRAN IV G COMPILER        MAIN            03-04-69        15:55.25        PAGE 0002

>                 C
>     0009              CALL RANDU(LAST,IRAND,XR)
>     0010              LAST = IRAND
>     0011              CALL RANDU(LAST,IRAND,YR)
>     0012              LAST = IRAND
>                 C
>                 C         PUT THE X AND Y COORDINATES OF THE TRIANGLE LOCATION INTO THE
>                 C
>                 C       DISPLAY FILE CONSTRUCTION BUFFER.
>                 C
>     0013              CALL DFXYC(7.0*XR,7.0*YR,0)
>                 C
>                 C         SINCE THE SUCESSIVE TRIANGLES ARE PLACED RELATIVE TO THE LAST
>                 C
>                 C       TRIANGLE AND NOT THE LOWER LEFT HAND CORNER OF THE DISPLAY SCREEN,
>                 C
>                 C       THE LIGHT BEAM IS NOT TO BE PLACED AT THE LOWER LEFT HAND CORNER
>                 C
>                 C       OF THE DISPLAY SCREEN AFTER THE PUSH JUMP TO EACH TRIANGLE.  SO
>                 C
>                 C       THE MODE IS CHANGED WITH DFAPM.
>                 C
>     0014              IF (J .EQ. 1) CALL DFAPM(0,1121)
>                 C
>                 C         PUSH JUMP TO THE TRIANGLE.
>                 C
>     0015         100  CALL DFPJM(4)
>                 C
>                 C         SEND DISPLAY FILE TO THE 338.
>                 C
>     0016              CALL DF201(0,5)
>     0017              CALL SYSTEM
>     0018              END
```

```
>   FORTRAN IV G COMPILER        RANDU          03-04-69        15:55.27       PAGE 0001

>    0001              SUBROUTINE RANDU(IX,IY,YFL)
>    0002              IY = IX*65539
>    0003              IF (IY)3,6,6
>    0004         5    IY = IY + 2147483647 + 1
>    0005         6    YFL = IY
>    0006              YFL = YFL * 0.465661E-9
>    0007              RETURN
>    0008              END
```

```
FORTRAN IV G COMPILER        MAIN            03-04-69        15:55.27        PAGE 0001

>                C       DISPLAY EXAMPLE #5, USING THE LIGHT PEN AND DFPM.
>                C
>                C         IN THIS LIGHT PEN DEMONSTRATION PROGRAM, THREE DISPLAY FILES
>                C
>                C       (ONE DESCRIBES A SQUARE, ANOTHER A TRIANGLE, AND THE LAST A LABEL
>                C
>                C       'LIGHT PEN ENABLED') ARE CONSTRUCTED AND SENT TO THE 338.  AFTER
>                C
>                C       THE LIGHT PEN HAS BEEN ENABLED, THE USER CAN POINT AT EITHER THE
>                C
>                C       LINES THAT DESCRIBE THE SQUARE OR THE TRIANGLE WITH THE LIGHT
>                C
>                C       PEN (SHUTTER OPEN) AND THE FOLLOWING SEQUENCE OF EVENTS WILL
>                C
>                C       OCCUR:
>                C
>                C              THE 'LIGHT PEN ENABLED' LABEL WILL BE TEMPORARILY BLANKED
>                C                   THAT IS, MADE INVISIBLE,
>                C
>                C              THE FIGURE POINTED AT WILL BE BLANKED THEN UNBLANKED,
>                C                   THAT IS, BLINKED,
>                C
>                C              THE LIGHT PEN WILL BE ENABLED AGAIN,
>                C
>                C              THE 'LIGHT PEN ENABLED' LABEL WILL BE UNBLANKED.
>                C
>    0001              REAL XS(5)/2.0,2.0,4.0,4.0,2.0/, XT(4)/6.0,7.0,8.0,6.0/
>    0002              REAL YS(5)/2.0,4.0,4.0,2.0,2.0/, YT(4)/2.0,4.0,2.0,2.0/
>    0003              INTEGER IS (6)/0,1,1,1,1,-1/,    IT(5)/0,1,1,1,-1/
>                C
>                C         CONSTRUCT SQUARE, MAKE LIGHT PEN SENSITIVE, AND SEND TO THE 338.
>                C
>    0004              CALL DFINI(0,0)
>    0005              CALL DFXYV(XS,YS,IS)
>    0006              CALL DF201(8,0)
>                C
>                C         CONSTRUCT TRIANGLE, MAKE LIGHT PEN SENSITIVE AND SEND TO THE
>                C
>                C       338.
>                C
>    0007              CALL DFINI(0,0)
>    0008              CALL DFXYV(XT,YT,IT)
>    0009              CALL DF201(8,1)
>                C
>                C         CONSTRUCT LABEL AND SEND TO THE 338.
>                C
>    0010              CALL DFLAB(2.5,6.0,'LIGHT PEN ENABLED@',2,0)
>    0011              CALL DF201(0,2)
>                C
>                C         ENABLE LIGHT PEN FOR POINTING MODE.
>                C
>    0012          100 CALL DFPM(NAME,IGNORE,FORGET,ALSO,&999)
>                C
>                C         NOW BLANK THE 'LIGHT PEN ENABLED' LABEL.  THIS SUBROUTINE CALL
>                C
```

```
>   FORTRAN IV G COMPILER        MAIN            03-04-69        15:55.27       PAGE 0002

>                C       WILL NOT BE MADE UNTIL A LIGHT PEN HIT HAS OCCURRED.  IF AN END-
>                C
>                C       OF-FILE IS GIVEN INSTEAD OF A LIGHT PEN HIT, CONTROL WILL BE
>                C
>                C       TRANSFERRED TO STATEMENT 999, I.E.  CALL SYSTEM.
>                C
>    0013              CALL DFCHG(1,2)
>                C
>                C          BLANK THE DISPLAY FILE WHICH WAS POINTED AT BUT DO NOT CHANGE
>                C
>                C       LIGHT PEN SENSITIVITY.
>                C
>    0014              CALL DFCHG(9,NAME)
>                C
>                C          UNBLANK THE DISPLAY FILE WHICH WAS POINTED AT WITH THE LIGHT
>                C
>                C       PEN.
>                C
>    0015              CALL DFCHG(8,NAME)
>                C
>                C          UNBLANK THE LABEL.
>                C
>    0016              CALL DFCHG(0,2)
>                C
>                C          NOW RE-ENABLE THE LIGHT PEN.
>                C
>    0017              GO TO 100
>                C
>    0018         999  CALL SYSTEM
>    0019              END
```

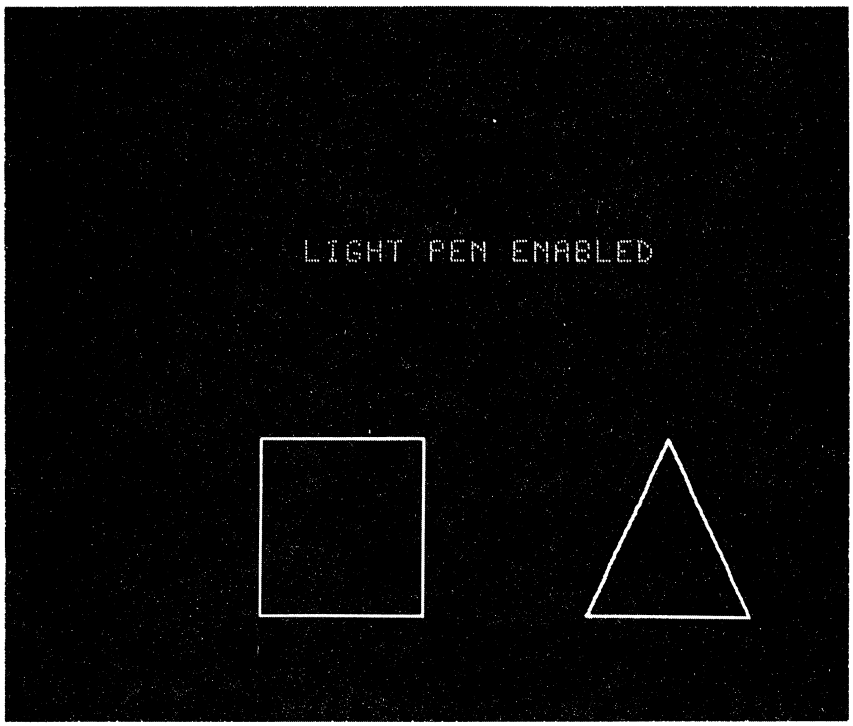

Figure 5A. Output for Example 5.

APPENDIX B. DF SUBROUTINE DESCRIPTIONS

Each DF subroutine is summarized in this appendix. The summary includes: subroutine name, subroutine function, FORTRAN IV calling sequence, argument descriptions, and return code descriptions. The following table gives the object module sizes and entry points:

Obj.Mod.	Other Entry Points	Size (bytes)
DFINI	DFESC, DFOCT, DFDESC, DFAPM	855
DFXYV	DFXYC, DFXYR	456
DF201	DFCHG, DFSN	770
DFSAVE	DFGET	682
DFORG	DFRAS	312
DFPJM		382
DFLAB		904
DFPM, DFPMR, DFLP, DFLPR, DFLT, DFLTR		854
DFAVL		108
PSECT1		1948
PSECT2 (up to 5 dynamically acquired)		1072

Note: The default FORTRAN mode (integer or floating point) conventions are used to indicate the mode for all arguments in the following subroutine descriptions.

NAME: DFINI

FUNCTION: Initializes the display file construction buffer and starts the display file with a PAR and a MODE. Also, several internal switches and pointers are checked and/or reset.

CALLING SEQUENCE: FORTRAN IV:

CALL DFINI(IPAR, MODE,&1,&2)

ARGUMENTS: IPAR - Parameters for the 338 display controller, allows scale change, light-pen status change, and light-beam intensity change [default 0417_8—enables intensity change to 7].

MODE — More parameters for the 338 display controller, allows mode (type of display file) change and specifies three conditions [default 1127_8 — enables mode change to vector mode, clears coordinate and sector bits, and enters data state]. The user cannot specify that the internal stop bit be set.

RETURN CODES: &1 - control is transferred to this statement if the user tries to acquire more than five display file construction buffers. Normally a buffer is reused once it has been acquired unless the

user "saves" it with DFSAVE.

(***DF** BUFFER TABLE FULL).

&2 - if either argument is negative, control is transferred to this statement (***DF** ARGUMENT NEGATIVE).

COMMENTS: The default value of either argument is specified by making the value of that argument zero in the call, CALL DFINI(0,0).

See pseudo octal number comment regarding the values of the arguments IPAR and MODE.

NAME: DFESC

FUNCTION: Puts an escape bit in the high-order position of the last word of the current display file in the display file construction buffer if the display file is in data state. If not in data state, nothing happens.

CALLING SEQUENCE: FORTRAN IV:

```
CALL DFESC(&1)
```

ARGUMENTS: None

RETURN CODES: &1 - if no display file construction buffer is active, control returns to this statement (***DF** NO BUFFER).

COMMENTS: None.

NAME:	DFDESC
FUNCTION:	Removes the escape bit in the high-order position of the last word of the current display file in the display file construction buffer if the display file is in command state. If not in command state, nothing happens.
CALLING SEQUENCE:	FORTRAN IV: CALL DFESC(&1)
ARGUMENTS:	None
RETURN CODES:	&1 - if no display file construction buffer is active, control is returned to this statement (***DF** NO BUFFER).
COMMENTS:	None

NAME: DFOCT

FUNCTION: Inserts a user-specified, four-digit octal number into the display file construction buffer and assigns the (internal to the DF subroutines) "in data state" switch a value one or zero depending on the value of the argument, ISW.

CALLING SEQUENCE: FORTRAN IV:

```
CALL DFOCT(NBR,ISW,&1,&2,&3)
```

ARGUMENTS: NBR - a four-digit octal number that is to be placed into the display file (see pseudo octal number comment).

ISW - value for the "in data state" switch; if ISW = 0, then the switch is assigned 0 (no), for all other values of ISW the switch is assigned 1 (yes).

RETURN CODES: &1 - if no display file construction buffer is active, control is returned to this statement (***DF** NO BUFFER).

&2 - if the argument is negative, control is returned to this statement (***DF** NEGATIVE ARGUMENT).

&3 - if the display file construction buffer is full, control is returned to this statement (***DF** BUFFER FULL).

COMMENTS: This subroutine is to be used only by knowledgeable users. The essential background is a working understanding of how display files are assembled for the 338. The problem is that this subroutine will put any four-digit octal number into the display file without any check to see if that number is legal since error checking is impossible. The end result can be a display file that snarks RAMP.

NAME: DFAPM

FUNCTION: Allows the values of IPAR and MODE to be changed from that specified in DFINI and used when necessary in the assembly of the remainder of the display file. All previous values of IPAR and MODE already inserted into the display file construction buffer are not changed.

CALLING SEQUENCE: FORTRAN IV

CALL DFAPM(IPAR,MODE,&1)

ARGUMENTS: IPAR - Parameters for the 338 display controller allows scale change, light-pen status change, and light-beam intensity change.

MODE - More parameters for the 338 display controller, allows mode change and specifies three conditions. The user cannot specify that the internal stop bit be set.

RETURN CODES: &1 - control is returned to this statement if no display file construction buffer is active (***DF** NO BUFFER).

COMMENTS: 1. If either IPAR or MODE equal zero, then that argument is given the default value (see DFINI for default value).

2. If either IPAR or MODE is less than zero, then the current value of that argument is not changed. For example, if the call on DFINI were: CALL DFINI (0457,1126), the values for IPAR and MODE will be IPAR = 0457_8, MODE = 1126_8. After one or more 338 commands were put into the display file construction buffer and a call on DFAPM were made, CALL DFAPM (-1,0), then the values of IPAR and MODE will be IPAR = 0457_8 (-1 means don't change), MODE = 1127_8 (0 means use standard value).

3. See the pseudo octal number comment regarding values of IPAR and MODE.

NAME: DFXYV

FUNCTION: Assembles a series of entries in a vector mode display file (i.e., the picture will be a sequence of straight lines) from arrays of input data, the x and y coordinates of the endpoints of each vector and an intensify switch (intensified/unintensified; visible/invisible) for each vector.

CALLING SEQUENCE: FORTRAN IV

```
CALL DFXYV(X,Y,I,&1,&2,&3)
```

ARGUMENTS: X,Y - arrays of the x and y coordinates of the endpoints of vectors. The x and y values are in inches relative to the origin (0,0) of the display screen coordinate system (default origin at the lower lefthand corner of the screen; the user can specify another origin for the display screen coordinate system anywhere on the screen). The screen is 9.375 inches by 9.375 inches.

I - an array of intensify switches, if the value of I=0, the vector is unintensified (invisible); I>0, the vector is intensified (visible).

I<0 implies that the previous X,Y,I triple were the final data values in the current data set.

RETURN CODES: &1 - control is returned to this statement when the display file construction buffer is full; a maximum of 255 vectors can be described in one display file (***DF** BUFFER FULL).

&2 - control is returned to this statement if a vector end point is off the screen (***DF** OFF THE SCREEN).

Note: All vectors are plotted up to the one that causes either the first or second error return.

&3 - control is returned to this statement if no display file construction buffer is available (***DF** NO BUFFER).

COMMENTS: None

NAME: DFXYC

FUNCTION: Assembles one entry in a vector mode display file.

CALLING SEQUENCE: FORTRAN IV

CALL DFXYC(X,Y,I,&1,&2,&3)

ARGUMENTS: X,Y - the x and y coordinates of the endpoint of a vector in inches.

I - intensify switch. If the value of

I=0 the vector is unintensified (invisible);

I≠0 the vector is intensified (visible).

RETURN CODES: &1 - control is returned to this statement when the display file construction buffer is full; a maximum of 255 vectors can be described in one display file (***DF** BUFFER FULL).

&2 - control is returned to this statement if the vector endpoint is off the screen (***DF** OFF THE SCREEN).

&3 - control is returned to this statement if no display file construction buffer is available (***DF** NO BUFFER).

COMMENTS: See DFXYV for more details.

NAME: DFXYR

FUNCTION: Construct one entry in a vector mode display file with the X and Y coordinates given in raster points instead of inches.

CALLING SEQUENCE: FORTRAN IV

CALL DFXYR(IX,IY,I,&1,&2,&3)

ARGUMENTS: IX,IY - the x and y coordinates of the endpoint of the vector given in raster points, 1023 raster points = 9.375 inches

I - intensify switch. If the value of

I=0 the vector is unintensified (invisible);

I≠0 the vector is intensified (visible).

RETURN CODES: &1 - control is returned to this statement when the display file construction buffer is full; a maximum of 255 vectors can be described in one display file (***DF** BUFFER FULL).

&2 - control is returned to this statement if a vector endpoint is off the screen (***DF** OFF THE SCREEN).

&3 - control is returned to this statement if no display file construction is available (***DF** NO BUFFER).

COMMENTS: See DFXYV for more details.

NAME: DF201

FUNCTION: Associates a set of user-specified switches with the display file, associates a user-specified integer name with the display file, and transmits the display file to the terminal computer, a DEC 338.

CALLING SEQUENCE: FORTRAN IV

CALL DF201(ISW,NAME,&1,&2,&3)

ARGUMENTS: ISW - a set of binary switches (in one word)

01 - if ISW=1, the display file is blanked (off, not displayed).

02 - if ISW=2, the display file is a subroutine file, another display file can have a push-jump to this file.

04 - if ISW=4, the display file is to be erased and the memory locations occupied in the 338 reclaimed.

08 - if ISW=8, the display file is to be light-sensitive.

Any combination of the above can be specified by setting ISW equal to the sum of the individual switch values. (Some switch combinations are meaningless

and are ignored by DF201.)

NAME - an integer in the range 0-127 that the user wants to associate with a particular display file as its name. A user can have up to 128 different display files at any given time in the 338.

RETURN CODES: &1 - control is returned to this statement if the value of ISW<0 or ISW>15 (***DF** ILLEGAL SWITCHES).

&2 - control is returned to this statement if the value of NAME<0 or NAME>127 (***DF** ILLEGAL NAME).

&3 - control is returned to this statement if no display file construction buffer is available (***DF** NO BUFFER).

COMMENTS: Some switch values imply particular types of PARs and MODEs for the display file; while the user should be aware of these and specify the appropriate values in DFINI, DF201 or RAMP will check for these cases and make corrections if necessary.

NAME: DFCHG

FUNCTION: Allows the user to specify changes in the switch values of a previously defined display file.

CALLING SEQUENCE: FORTRAN IV

CALL DFCHG(ISW,NAME,&1,&2,&3)

ISW - a set of binary switches.

01 - if ISW=1, the display file is blanked (off, not displayed).

02 - if ISW=2, this display file is a subroutine file, another display file can have a push-jump to this file.

04 - if ISW=4, this display file is to be erased and the space it occupied reclaimed.

08 - if ISW=8, this display file is to be light-pen sensitive.

Any combination of the above can be specified by setting ISW equal to the sum of the individual switch values. (Some switch combinations are meaningless and are ignored by DFCHG.)

NAME - the integer name of the display file whose switches are to be changed.

RETURN CODES:

&1 - control is returned to this statement if the value of ISW<0 or >15 (***DF** ILLEGAL SWITCHES).

&2 - control is returned to this statement if the value of NAME<0 or >127 (***DF** ILLEGAL NAME).

&3 - control is returned to this statement if no display file construction buffer is available (***DF** NO BUFFER).

NAME:	DFSN
FUNCTION:	Issues the RAMP command SN(START AGAIN - erase all display files currently defined in the 338) and reinitializes some tables and pointers in a region common to all the DF routines.
CALLING SEQUENCE:	FORTRAN IV CALL DFSN
ARGUMENTS:	None
RETURN CODES:	None
COMMENTS:	None

NAME: DFLAB

FUNCTION: Allows the user to put a label on the display screen.

CALLING SEQUENCE: FORTRAN IV

```
CALL DFLAB(X,Y,'STRING@', ISIZE,IHV,
&1,&2,&3,&4)
```

ARGUMENTS: X,Y - coordinate (inches) of the lower left-hand corner of the first character in the string relative to display screen origin.

"STRING@' - the string of characters that are to be displayed. The end of the string is indicated by "@". Two "@"s (@@) mean the literal "@" and not the end of the string.

ISIZE - a size parameter for the character, range 0-3.

0 gives characters ~3/40" high

1 gives characters ~3/20" high

2 gives characters ~3/10" high

3 gives characters ~3/5" high

IHV - is a binary switch indicating either horizontal or vertical orientation of the label. IHV=0 means horizontal; IHV≠0 means vertical.

RETURN CODES:

&1 - control is returned to this statement if the buffer table is full (***DF** BUFFER TABLE FULL).

&2 - control is returned to this statement if the label goes off the screen (***DF** OFF THE SCREEN).

&3 - control is returned to this statement if the character size is illegal (***DF** ILLEGAL CHARACTER SIZE).

&4 - control is returned to this statement if the display file construction buffer is full (***DF** BUFFER FULL).

COMMENTS:

1. The maximum number of characters for a label depends both on the size of the characters and the position of the first character relative to the right-hand side or lower edge of the screen. Characters will never wrap around these two boundaries; any effort to do so will cause the off-the-screen comment to be printed and only those characters up to the overflow to be plotted. The user need not call DFINI before DFLAB as this is done in DFLAB if necessary. The user must call DF201 to send the label to the 338.

2. Successive calls can be made to DFLAB so that more than one label can be put into the same display file. A group of vectors may precede or follow any label as long as the display file buffer is not full.

3. If the value for x is less than 0, then the label will continue from the current light-beam location on the display screen.

4. The following information will allow calculation of character size and string length: (a) The distance between raster points is 0.0091642 inches, (b) the standard character format is a 5x7 grid located in the lower left-hand corner of a 7x10 grid, (c) when n is the number of characters in a string, its horizontal length is given by

    n*7*0.0091642*(ISIZE+1) inches

and its vertical length is given by

    n*10*0.0091642*(ISIZE+1).

NAME: DFPM

FUNCTION: Allows the user to point at a visible, light-pen sensitive display file with the light pen and returns to the calling program in the 360, the name of the display file which was pointed at, a displacement into the display file, and the x,y coordinates (in inches) of the light-pen hit. Uses the RAMP command PM (POINTING MODE).

CALLING SEQUENCE: FORTRAN IV

```
CALL DFPM(NAME,IDISP,X,Y,&1,&2,&3)
```

ARGUMENTS: NAME - this variable will be assigned the value of the integer name associated with the display file which was pointed at with the light pen.

IDISP - this variable will be assigned an integer value which is the displacement from the beginning of the display file (or display file instruction index number) at the time of the light-pen hit (for knowledgeable users).

X,Y - these variables will be assigned the values of the x and y coordinate of the light-pen hit respectively, in

inches relative to the lower left-hand corner of the display screen.

RETURN CODES: &1 - control will be returned to this statement if an EOF (end-of-file) is given instead of pointing at a display file with the light pen (no comment printed).

&2 - control will be returned to this statement if no display files are both light-pen enabled and unblanked (***DF** NO DFSLP ENABLED AND UNBLANKED).

&3 - control will be returned to this statement if the table lookup for the display file number fails (***DF** FAILED TO MATCH NAME).

COMMENTS: DFINI must be called at least once in the calling program before DFPM is called since DFINI establishes vital internal pointers.

NAME: DFPMR

FUNCTION: Allows the user to point at a visible, light-pen-sensitive display file with the light pen and returns to the calling program in the 360 the name of the display which was pointed at, a displacement into the display file, and the x,y coordinates (in raster points) of the light-pen hit. Uses the RAMP command PM (POINTING MODE).

CALLING SEQUENCE: FORTRAN IV

```
CALL DFPMR(NAME,IDISP,IX,IY,&1,&2,&3)
```

ARGUMENTS: NAME - this variable will be assigned the value of the integer name associated with the display file which was pointed at with the light pen.

IDISP - this variable will be assigned an integer value which is the displacement from the beginning of the display file (or display file instruction index number) at the time of the light-pen hit (for knowledgeable users).

IX,IY - these variables will be assigned the values of the x and y coordinates of the light-pen hit, respectively, in raster points relative to the lower left-hand

corner of the display screen.

RETURN CODES: &1 - control will be returned to this statement if an EOF (end-of-file) is given instead of pointing at a display file with the light pen (no comment printed).

&2 - control will be returned to this statement if no display files are both light-pen enabled and unblanked (***DF** NO DFSLP ENABLED AND UNBLANKED).

&3 - control will be returned to this statement if the table lookup for the display file number fails (***DF** FAILED TO MATCH NAME).

COMMENTS: DFINI must be called at least once in the calling program before DFPMR is called since DFINI establishes vital internal pointers.

NAME: DFLP

FUNCTION: Allows the user to do light-pen tracking with rubber-banding via the RAMP command LP. The values of the final tracking cross position (in inches) relative to the lower left-hand corner of the display screen are returned to the calling program.

CALLING SEQUENCE: FORTRAN IV

CALL DFLP(XI,YI,X,Y,&1,&2)

ARGUMENTS: XI,YI - the initial x,y coordinates respectively of the light-pen tracking cross in inches relative to the lower left-hand corner of the display screen.

X,Y - these variables will be assigned the final x,y coordinates respectively of the tracking cross in inches relative to the lower left-hand corner of the display screen.

RETURN CODES: &1 - control will be returned to this statement if an EOF (end-of-file) is given instead of the normal termination of light-pen tracking (no comment printed).

&2 - control is returned to this statement if DFINI has not been called (***DF** DFINI MUST BE CALLED FIRST).

COMMENTS: None

NAME: DFLPR

FUNCTION: Allows the user to do light-pen tracking with rubber-banding via the RAMP command LP. The values of the final tracking-cross position (in raster points) relative to the lower left-hand corner of the display screen are returned to the calling program.

CALLING SEQUENCE: FORTRAN IV

CALL DFLPR(IXI,IYI,IX,IY,&1,&2)

ARGUMENTS: IXI,IYI - the initial x,y coordinates, respectively, of the light-pen tracking cross in raster points relative to the lower left-hand corner of the display screen.

IX,IY - these variables will be assigned the final x,y, coordinates respectively of the tracking cross in raster points relative to the lower left-hand corner of the display screen.

RETURN CODES: &1 - control wlll be returned to this statement if an EOF (end-of-file) is given instead of the normal termination of light-pen tracking (no comment printed).

&2 - control is returned to this statement if DFINI has not been called

(***DF** DFINI MUST BE CALLED FIRST).

COMMENTS: None

NAME: DFLT

FUNCTION: Allows the user to do light-pen tracking without rubber-banding via the RAMP command LT. The values of the final tracking-cross position (in inches) relative to the lower left-hand corner of the display screen are returned to the calling program.

CALLING SEQUENCE: FORTRAN IV

CALL DFLT(IX,YI,X,Y,&1,&2)

ARGUMENTS: XI,YI - the initial x,y coordinates respectively of the light-pen tracking cross in inches relative to the lower left-hand corner of the display screen.

X,Y - these variables will be assigned the final x,y coordinates respectively of the tracking cross in inches relative to the lower left-hand corner of the display screen.

RETURN CODES: &1 - control will be returned to this statement if an EOF (end-of-file) is given instead of the normal termination of light-pen tracking (no comment printed).

&2 - control is returned to this statement if DFINI has not been called

(***DF** DFINI MUST BE CALLED FIRST).

COMMENTS: None

NAME: DFLTR

FUNCTION: Allows the user to do light-pen tracking without rubber-banding via the RAMP command LT. The values of the final tracking-cross position (in raster points) relative to the lower left-hand corner of the display screen are returned to the calling program.

CALLING SEQUENCE: FORTRAN IV

CALL DFLTR(IXI,IYI,IX,IY,&1,&2)

ARGUMENTS: IXI,IYI - the initial x,y coordinates respectively of the light-pen tracking cross in raster points relative to the lower left-hand corner of the display screen.

IX,IY - these variables will be assigned the final x,y coordinates respectively of the tracking cross in raster points relative to the lower left-hand corner of the display screen.

RETURN CODES: &1 - control will be returned to this statement if an EOF (end-of-file) is given instead of the normal termination of light-pen tracking (no comment printed).

&2 - control is returned to this statement if DFINI has not been called

(***DF** DFINI MUST BE CALLED FIRST).

COMMENTS: None

NAME: DFPJM

FUNCTION: Allows the user to insert a push-jump to another display file (required to be defined and of subroutine type) in the display file currently being constructed.

CALLING SEQUENCE: FORTRAN IV

CALL DFPJM(NAME,&1,&2,&3,&4,&5)

ARGUMENTS: NAME - the integer name of the display file to be push-jumped to.

RETURN CODES: &1 - control is returned to this statement if the buffer is full (***DF** BUFFER FULL).

&2 - control is returned to this statement if the NAME display file is not a subroutine file or has not been defined (***DF** NOT A SUB FILE).

&3 - control is returned to this statement if the NAME is illegal (not in the range 0-127) (***DF** ILLEGAL NAME).

&4 - control is returned to this statement if no display file construction buffer is available (***DF** NO BUFFER).

&5 - control is returned to this statement if the endpoint of the display file

to be push-jumped to will go off the screen (***DF** OFF THE SCREEN).

NAME: DFSAVE

FUNCTION: Allows the user to save the contents of the current display file construction buffer by moving the pointer to the buffer from the current buffer-pointer table to the save buffer-pointer table and associates a four-character user-supplied name with that buffer pointer.

CALLING SEQUENCE: FORTRAN IV

CALL DFSAVE('NAME',&1,&2,&3)

ARGUMENTS: 'NAME' - a string of up to four characters that the user wants to associate with save buffer; all alphanumeric and graphic characters are legal.

RETURN CODES: &1 - control will be returned to this statement if an illegal character occurs in a name (***DF** ILLEGAL CHARACTERS IN NAME).

&2 - control will be returned to this statement if the save-buffer-pointer table is full, maximum five (***DF** BUFFER TABLE FULL).

&3 - control will be returned to this statement if no display file construction buffer is available (***DF** NO BUFFER).

COMMENTS: None

NAME: DFGET

FUNCTION: Allows the user to return a saved display file construction buffer to the current or active buffer status.

CALLING SEQUENCE: FORTRAN IV

CALL DFGET('NAME',&1,&2,&3)

ARGUMENTS: 'NAME' - a string of up to four characters that a user has associated with a saved file.

RETURN CODES: &1 - control will be returned to this statement if an illegal character occurs in a name (***DF** ILLEGAL CHARACTERS IN NAME).

&2 - control will be returned to this statement if the buffer name is not found in the table (***DF** NO SUCH NAME).

&3 - control will be returned to this statement if the save buffer table is empty (***DF** BUFFER TABLE EMPTY).

COMMENTS: None

NAME: DFORG

FUNCTION: Allows the user to change the origin for the display screen coordinate system to any other point on the screen.

CALLING SEQUENCE: FORTRAN IV

CALL DFORG(X,Y,&1,&2,&3)

ARGUMENTS: X,Y - the coordinates (inches) of the new origin relative to (0,0) the lower left-hand corner of the display screen.

RETURN CODES: &1 - control will be returned to this statement if the display file construction buffer is full (***DF** BUFFER FULL).

&2 - control will be returned to this statement if the new origin is off the screen (***DF** OFF THE SCREEN).

&3 - control will be returned to this statement if no display file construction buffer is available (***DF** NO BUFFER).

COMMENTS: An invisible vector is drawn to the origin of the new coordinate system from the last beam location.

NAME:	DFRAS
FUNCTION:	Allows the user to assign different values (in raster points) to locations where the coordinates of the current light-beam position is saved.
CALLING SEQUENCE:	FORTRAN IV CALL DFRAS(IX,IY,,,&3)
ARGUMENTS:	IX,IY - the new values for the coordinates of the light beam in raster points.
RETURN CODES:	&3 - control will be returned to this statement if no display file construction buffer is available (***DF** NO BUFFER).
COMMENTS:	This subroutine is available for knowledgeable users.

NAME: DFAVL

FUNCTION: Returns an unused display file name (an integer from 0-127). Starts at zero and proceeds to higher numbers.

CALLING SEQUENCE: FORTRAN IV

CALL DFAVL(NAME,&1)

ARGUMENT: NAME - the value of the available display file name will be assigned to this variable.

RETURN CODES: &1 - control will be returned to this statement if no names are available.

APPENDIX C. 338 DISPLAY FILE ORGANIZATION AND TRANSMISSION HEADER.

This appendix provides some insight into the structures of 338 display files assembled when using the DF routines. For a more complete discussion of 338 display commands and display file structure see the 338 manual (DEC-08-G61A-D).

All display files assembled by the DF routines begin with a PAR and a MODE. (See the sample display file in Figure C1a; this display file is the result of executing Example 1 in Appendix A.) Calling DFINI inserts the PAR and MODE into the display file construction buffer. The purpose of PAR and MODE is to specify some parameters for the 338 display controller, for example, the light-beam intensity, the display file data format, and transfer from command state to data state. The next five pairs of words in the display file are x and y coordinate data, in this case, for the endpoints of vectors drawn on the display screen. Calling DFXYC inserts one x,y pair into the display file. (Note: The x and y data are represented with incremental values in the display file but they are specified relative to an absolute coordinate system in the calling sequence.) Each pair of words contains other information in addition to coordinate data, i.e., intensify switch, escape bit, and algebraic signs for the coordinate

values (see Figure C2 and note that a 338 word contains 12 bits.) The final entry in the display file is a POP. This is an end-of-display-file command and it is added to the display file by the 338 executive, RAMP.

Since the word length in the 338 is 12 bits and the basic memory element in the 360 is the byte or 8 bits, the display file storage allocation during assembly in the 360 has the format indicated in Figure C3, that is, the bit format is the standard column binary format. While this format may seem wasteful (75% utilization of bits), it is justified by the resulting simplicity in assembling the display file. Prior to transmitting the display file to the 338 from the 360, the display file is packed in that four bytes (two 338 words) of the assembled display file are placed into 3 bytes. This reduces transmission time by approximately 25% since the unit of transmission to the 338 is the 8-bit byte.

The subroutine DF201 adds the header to the display file, packs the display file (as indicated above), and transmits the display file from the 360 to the 338. The purpose of the header is to provide the 338 executive, RAMP, with necessary information about the display file. The first byte contains $9C_{16}$; this is an internal code for RAMP meaning that the following information is related to a display file. The next byte contains the value of

SW specified in the argument list of DF201 (see Figure C4 for current bit meanings in SW). The last two bytes of the header contain the 338's internal name of the display file which is mapped via a table to or from the user-supplied file name. (The 338's internal name is actually an address of a location in the display file control table in the 338. This name (address) is sent from the 338 to DF201 whenever DF201 sends a display file to the 338 that has an external (user) name for which no corresponding 338 name currently exists. DF201 indicates this condition to the 338 by using zero for the internal name which causes the 338 to assign a new name which is then returned to DF201. DFCHG will delete the internal 338 file name whenever the "erase" bit is set.) The display file immediately follows the header in transmission. The numerical contents (in octal) of the display file specified in Example 1 in Appendix A and discussed above are shown in Figure C1b. (Note, there are slightly more than $109_{10}=155_{8}$ raster points per inch.)

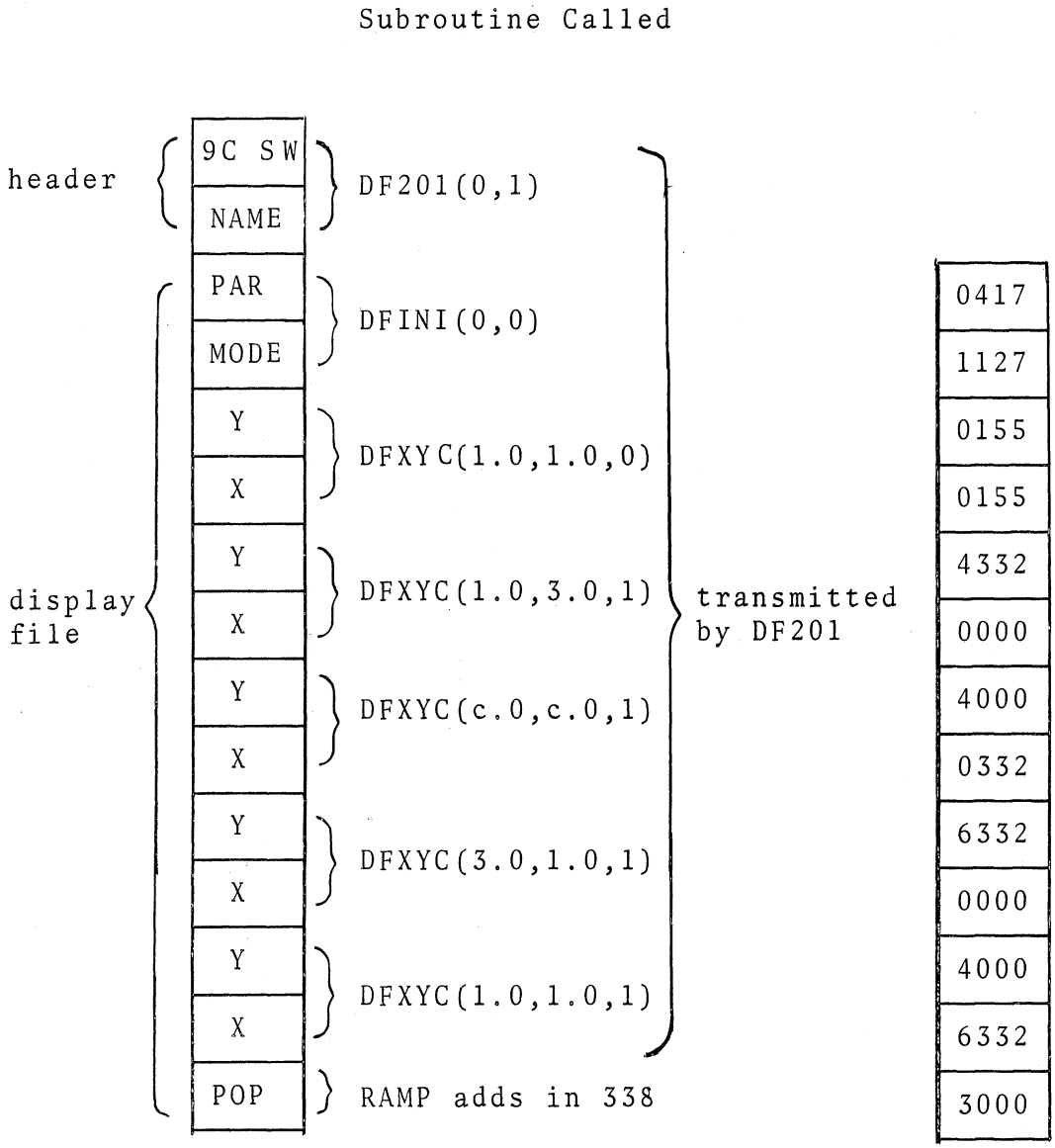

Figure C1a. The Basic Parts of a Display File and the Subroutine Calls That Produce the Various Parts.

Figure C1b. The Assembled (Octal Numbers) Display File Described in Figure C1a. The x and y data are in incremental coordinates.

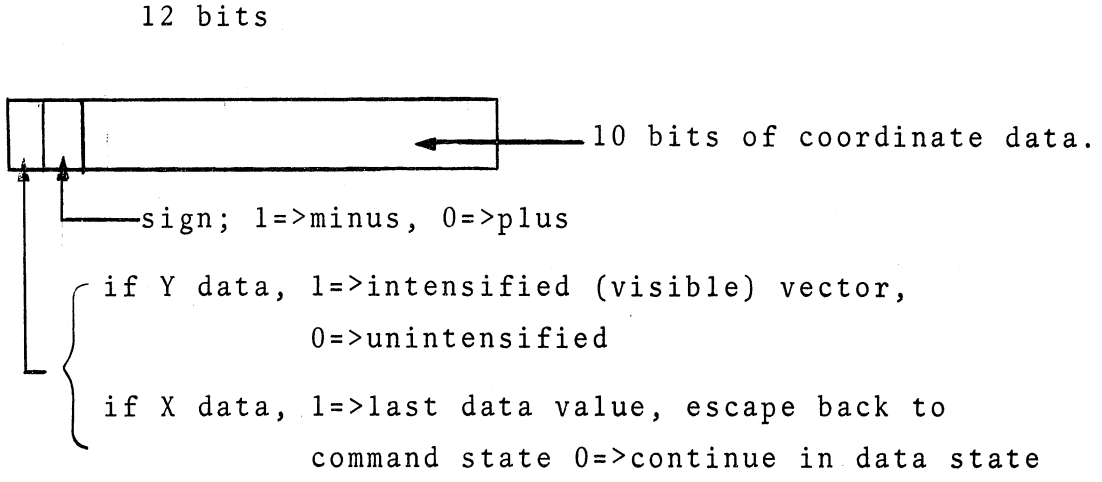

Figure C2. X and Y Display File Word Format (Vector Mode).

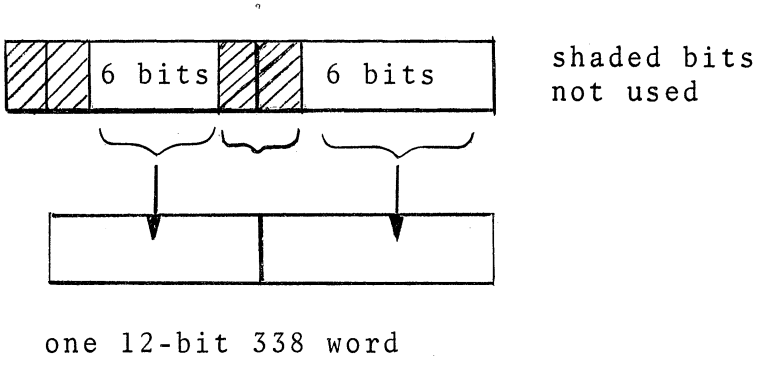

Figure C3. Format for Storing One 12-bit 338 Word into Two Bytes.

RAMP does not examine the 01 or 08 bits unless the corresponding enable bit is a one. A particular characteristic is specified by making the appropriate bit a one (yes).

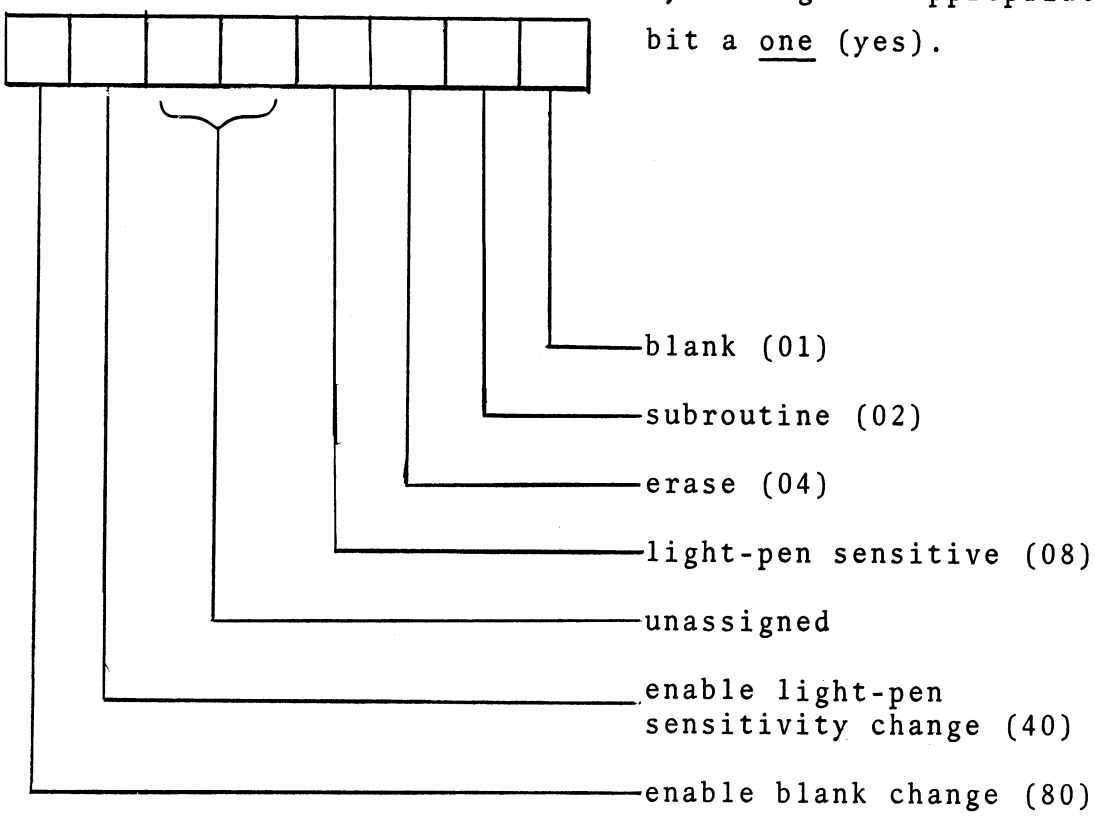

Figure C4. Bit Assignments in SW.

APPENDIX D. DISPLAY FILE ASSEMBLY PARAMETERS AND CHARACTERISTICS

The following is a list of display file assembly parameters and characteristics:

1. A maximum of 128 (names from 0 to 127) display files can be defined at any given time and stored in the 338.

2. The maximum display file length is 512-338 words; for example, a maximum of 255 vectors can be described in one display file.

3. While 128 display files could be defined at one time and the length of any of them could be 512 PDP-8 words, the number and/or size of display files defined and in core memory in the 338 is limited since only one core bank (4096 PDP-8 words) is available for display files. Allowing for system overhead in the display file core bank, seven display files with maximum length can be defined at one time or 128 display files with average length of 26 PDP-8 words can be defined at one time.

4. A maximum of five display file construction buffers (in the 360/67) can be acquired (currently the contents of a construction buffer cannot be saved on a file).

5. Display files can be sensed with the light pen.

6. The following information about a display file

is saved in the buffer during the assembly of that display file:

(a) values of 32 internal switches (only five have assigned meanings),

(b) the current light-beam location in raster points,

(c) the current display screen origin in inches relative to the lower left-hand corner of the display screen,

(d) the current length of the display file,

(e) the current values of IPAR and MODE.

APPENDIX E. DEBUGGING FROM THE TTY

Debugging programs containing calls on DF routines require that either the debugging be done at the 338 terminal or at a TTY with appropriate responses provided by the user (these responses are normally provided automatically by the 338). The user indicates that a particular run of his program is to be a debugging run by specifying LDN9 on the $RUN command. LDN9 is assigned a file where the output produced by the DF routines and normally sent to the 338 is to be stored. This must be a file. The DF routines that transmit information to the 338 from the 360 and the nature of the information sent are listed in the following table.

Table E2. Summary of Information Transmitted to the Terminal

Routine	Information Sent
DF201	header and display file
DFCHG	header
DFPM, DFPMR	RAMP command PM
DFLP, DFLPR	" " LP
DFLT, DFLTR	" " LT
DFSN	" " SN

The DF routines that expect information from the 338 are listed in the following table.

Table E2. Summary of Information Expected from the Terminal

Routine	Information Expected
DF201	Internal name assigned by 338, if this is first time a particular display file name has been used; otherwise nothing. Format - NNNN.
DFPM, DFPMR	Internal name, displacement, X and Y coordinates, light-pen hit. Format - NNNNØDDDDØXXXXØYYYY.
DFLP, DFLPR DFLT, DFLTR	X and Y coordinates of final light-pen tracking-cross position. Format - XXXXØYYYY.

Two problems remain: (1) How does the user know when his program has made a call on a DF routine that expects a response? (2) How does the user interpret the output produced by the DF routines? The first problem can be resolved by the inclusion of print statements in the user program that prompts the user for DF routine responses. The prompting could be made a part of the DF routines if there is sufficient user demand. The second problem is more difficult to resolve unless the user is knowledgeable 338 programmer. Those DF routines that issue RAMP commands, write those commands in eight-bit ASCII code. This allows each byte to be translated to letters and numbers. DF201

transmits a header and display file as described in Appendix C. DFCHG transmits the header only.

UNCLASSIFIED

Security Classification

DOCUMENT CONTROL DATA - R & D

(Security classification of title, body of abstract and indexing annotation must be entered when the overall report is classified)

1. ORIGINATING ACTIVITY *(Corporate author)*

THE UNIVERSITY OF MICHIGAN
CONCOMP PROJECT

2a. REPORT SECURITY CLASSIFICATION

Unclassified

2b. GROUP

3. REPORT TITLE

THE DF ROUTINES USER'S GUIDE

4. DESCRIPTIVE NOTES *(Type of report and inclusive dates)*

Memorandum 23

5. AUTHOR(S) *(First name, middle initial, last name)*

ALFRED B. COCANOWER

6 REPORT DATE	7a. TOTAL NO. OF PAGES	7b. NO. OF REFS
May 1969	69	None

8a. CONTRACT OR GRANT NO.	9a. ORIGINATOR'S REPORT NUMBER(S)
DA-49-083 OSA-3050	Memorandum 23
b. PROJECT NO.	
c.	9b. OTHER REPORT NO(S) *(Any other numbers that may be assigned this report)*
d.	

10. DISTRIBUTION STATEMENT

Qualified requesters may obtain copies of this report from DDC

11. SUPPLEMENTARY NOTES	12. SPONSORING MILITARY ACTIVITY
	Advanced Research Projects Agency

13. ABSTRACT

The DF routine package allows a IBM 360/67 programmer to assemble DEC 338 display files, send them to the DEC 338, control the displaying of these files, and enable light-pen service. The DF routines provide a set of elementary display file operations only; they do not provide complex operations such as producing an entire graph with an axis, labels, and curve with one subroutine call. However, subroutines that allow the displaying of graphs with one subroutine call can be written using the DF routines.

Other topics discussed in this report include: elementary example programs, DF routine summaries, 338 display file organization and transmission format, display file assembly parameters and characteristics, and instructions for debugging programs from a TTY. A typical $RUN command for a program containing calls on the DF routines and pseudo octal numbers are also discussed.

DD FORM 1 NOV 65 1473

Unclassified

Security Classification

Unclassified
Security Classification

14. KEY WORDS	LINK A		LINK B		LINK C	
	ROLE	WT	ROLE	WT	ROLE	WT
display file						
terminal computer (DEC 338)						
central Computer (IBM 360/67						
CRT						
assemble						
computer communications						