

H
QC
874.3
U68
no.18

AA Western Region Computer Programs and
blems NWS WRCP - No. 18



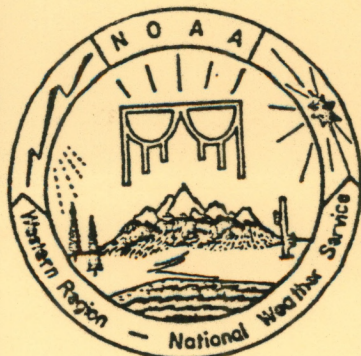
AFOS GRAPHICS CREATION FROM FORTRAN

Salt Lake City, Utah
March 1981

**U.S. DEPARTMENT OF
COMMERCE**

/ National Oceanic and
Atmospheric Administration

/ National Weather
Service



This Western Region publication series is considered as a subset of our Technical Memorandum series. This series will be devoted exclusively to the exchange of information on and documentation of computer programs and related subjects. This series was initiated because it did not seem appropriate to publish computer program papers as Technical Memoranda; yet, we wanted to share this type of information with all Western Region forecasters in a systematic way. Another reason was our concern that in the developing AFOS-era there will be unnecessary and wasteful duplication of effort in writing computer programs in National Weather Service (NWS). Documentation and exchange of ideas and programs envisioned in this series hopefully will reduce such duplication. We also believe that by publishing the programming work of our forecasters, we will stimulate others to use these programs or develop their own programs to take advantage of the computing capabilities AFOS makes available.

We solicit computer-oriented papers and computer programs from forecasters for us to publish in this series. Simple and short programs should not be prejudged as unsuitable.

The great potential of the AFOS-era is strongly related to local computer facilities permitting meteorologists to practice in a more scientific environment. It is our hope that this new series will help in developing this potential into reality.

NOAA Western Region Computer Programs and Problems NWS WRCP

- 1 Standardized Format for Computer Series.
- 2 AFOS Crop and Soil Information Report Programs. Ken Mielke, July 1979.
- 3 Decoder for Significant Level Transmissions of Raobs. John A. Jannuzzi, August 1979.
- 4 Precipitable Water Estimate. Elizabeth Morse, October 1979.
- 5 Utah Recreational Temperature Program. Kenneth M. Labas, November 1979.
- 6 Normal Maximum/Minimum Temperature Program for Montana. Kenneth Mielke, December 1979.
- 7 Plotting of Ocean Wave Energy Spectral Data. John R. Zimmerman, December 1979.
- 8 Raob Plot and Analysis Routines. John Jannuzzi, January 1980.
- 9 The SWAB Program. Morris S. Webb, Jr., April 1980. (PB80-196041)
- 10 Flash-Flood Procedure. Donald P. Laurine and Ralph C. Hatch, April 1980. (PB80-298658)
- 11 Program to Forecast Probability of Summer Stratus in Seattle Using the Durst Objective Method. John Zimmerman, May 1980.
- 12 Probability of Sequences of Wet and Dry Days. Hazen H. Bedke, June 1980. (PB80-223340)
- 13 Automated Montana Hourly Weather Roundup. Joe L. Johnston, July 1980. (PB81-102576)
- 14 Lightning Activity Levels. Mark A. Mollner, July 1980. (PB81-108300)
- 15 Two Fortran Applications of Wind-Driven Ekman Water Transport Theory: Upwelling Index and Storm Tide. Kent S. Short, July 1980. (PB81-102568)
- 16 AFOS System Local Database Save and Rebuild Procedures or A Master Doomsday Program. Brian W. Finke, July 1980. (PB81-108342)
- 17 AFOS/RDOS Translator Subroutine. Morris S. Webb, Jr., August 1980. (PB81-108334)

H
QC
874.3
U68
no. 18

NOAA Western Region Computer Programs and Problems NWS WRCP - No. 18

AFOS GRAPHICS CREATION FROM FORTRAN

"

Alexander E. MacDonald

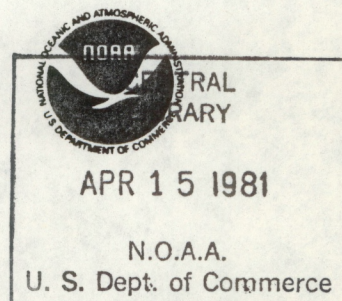
National Weather Service Western Region Headquarters
Salt Lake City, Utah
March 1981

UNITED STATES
DEPARTMENT OF COMMERCE

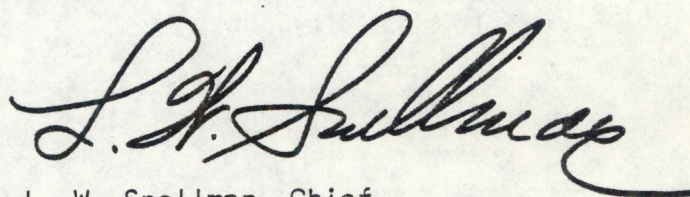
NATIONAL OCEANIC AND
ATMOSPHERIC ADMINISTRATION

National Weather
Service
Richard E. Hallgren, Director

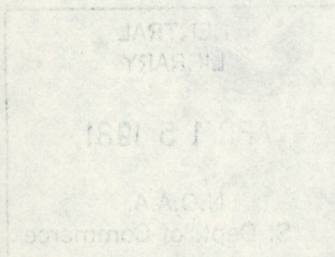
81 1323



This Technical Memorandum has been
reviewed and is approved for
publication by Scientific Services
Division, Western Region.



L. W. Snellman, Chief
Scientific Services Division
Western Region Headquarters
Salt Lake City, Utah



CONTENTS

	<u>Page</u>
Prologue	iv
I. General Information	1
A. Introduction	1
1. Simple Explanation of Graphics Creation . . .	1
2. Display Space	2
3. CLS Memory	4
B. Environment	4
C. References	4
II. Program Reference	5
A. CART	5
1. Program Description	5
2. Call Statement and External Variables	5
3. Machine Requirements	6
B. LINES	6
1. Program Description	6
2. Call Statement and External Variables	6
3. Machine Requirements	7
C. TEXT	7
1. Program Description	7
2. Call Statement and External Variables	8
3. Machine Requirements	10
D. UTF	10
1. Program Description	10
2. Call Statement and External Variables	10
3. Machine Requirements	10
4. Caution and Restriction	10
Acknowledgment	11
Appendix A - Examples of Graphics Creation	12
Appendix B - Compiling and Loading a FORTRAN Program . . .	15
Appendix C - Running and Displaying a Graphics File . . .	16
Appendix D - Universal Transmission Format and Debugging .	17

PROLOGUE

This publication may be used both as a beginner's manual and a reference to the AFOS Graphics developed by Western Region Scientific Services Division. The beginner should study the entire manual; after experience, the programmer will only need Chapter II as a reference.

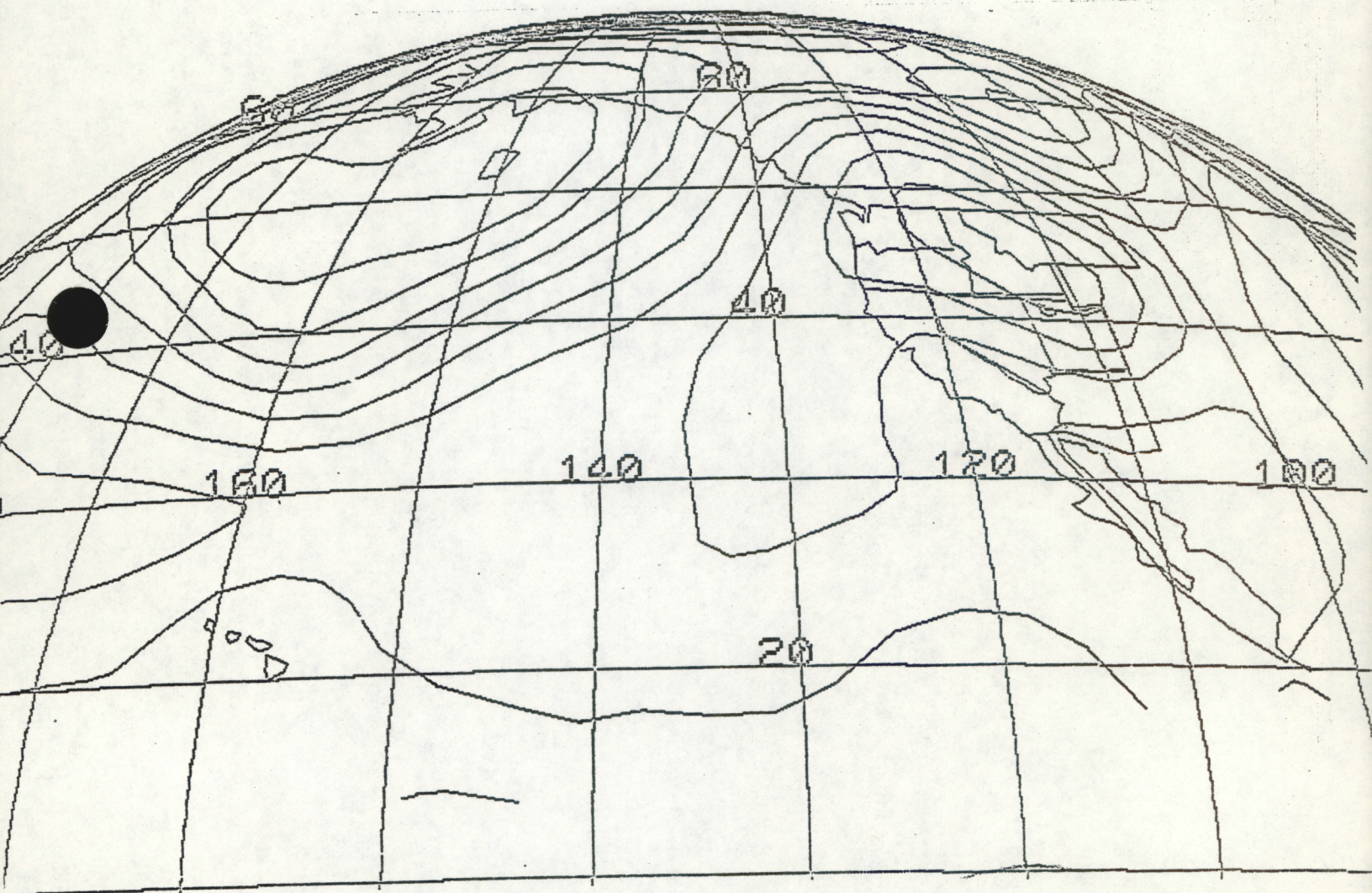


Figure 1. Graphics can be useful! This experimental program, written by the author, takes any AFOS graphic and puts it on a GOES projection.

AFOS GRAPHICS CREATION FROM FORTRAN

Alexander E. MacDonald*
National Weather Service Western Region
Salt Lake City, Utah

I. General Information

A. Introduction:

This publication describes a family of programs which allows creation of AFOS graphics simply and efficiently from FORTRAN. They represent a powerful capability which can be exploited by National Weather Service offices to present information in new and innovative ways, and to automate tasks and ease workload within offices. The use of standard Data General software allows applications and graphics to be developed and run at any AFOS-equipped office.

This software was developed primarily by Scientific Services Division of the Western Region. All programs are labeled Version 1.10 and as changes, corrections, and improvements are made, new versions will be issued. This is the first graphics software which has been documented and will be maintained and supported; those offices with previous releases should use this instead. The current release has several improvements and corrections to the earlier undocumented graphics software.

This graphics package is available to all National Weather Service offices. Coincidental with the release of this CP, the SMCC and all regional SSDs will be supplied with a floppy disk that contains the source code and a library for all programs described in the CP. Field offices may obtain the floppy disk and help in software development from their regional SSD.

This package has been designed to be simple to use but with the flexibility and generality to allow functions such as zoom thresholding, reverse block characters, and other sophisticated graphic capabilities. For complete freedom for more experienced programmers, we have made the source code of the programs available so that they may be improved and tailored to specific applications. For ease of use, all the subroutines are available in an "AFOS Graphics" library titled AG.LB.

The CP is meant to be both a beginner's manual and a reference for AFOS graphics. In the remainder of this chapter, we describe in simple terms how the software works, and present some concepts that are needed for graphic creation. Chapter II is primarily a reference, with each program and its call variables discussed in detail.

1. Simple Explanation of Graphics Section

To create a graphic, the user (that's you!) must write a FORTRAN program which uses the subroutines of this package to create the graphic. The two most important subroutines are:

*Dr. MacDonald's present affiliation PROFS Program Office, Environmental Research Laboratories, Boulder, Colorado.

LINES - A subroutine which draws lines and points, and

TEXT - A subroutine to plot alphanumeric text and special symbols.

These lines and text are drawn in a "display space" which is described in Part 2 below. The whole operation is similar to having a blank sheet (the display space) and a pencil to draw "lines" and print characters "text".

The "driver" program that creates the graphic has two parts. The actual drawing consists of a number of calls to LINES or TEXT:

```
CALL LINES (      )  
CALL TEXT (      )  
      :  
CALL LINES (      )
```

The graphics portion of the program must always be closed with a call to the subroutine UTF:

```
CALL UTF (      )
```

The subroutine UTF converts the display file to "Universal Transmission Format" and makes it storable and displayable on AFOS.

The Appendices A through D detail the techniques necessary for graphics creation. Appendix A gives two examples of FORTRAN "driver" programs to create graphics. Each illustrates different aspects of graphics creation. Appendix B shows the user how to compile and load the program.

The output of a program to create a graphic is an RDOS file. This file may be displayed directly on AFOS or stored in the AFOS database as described in Appendix C.

Appendix D details the internals of a UTF graphics file that the sophisticated programmer may need for changes or debugging.

2. Display Space

The graphics are "drawn" in a coordinate system referred to in this CP as a "display space". As shown in Figure 1, this is a cartesian grid with the abscissa (X) direction domain extending from 0 to 4095, and the ordinate (Y) direction with a domain from 0 to 3071. If the subroutine LINES was used to draw a vector from the point (0,0) to (2000,3000), a line would appear on the AFOS screen as depicted in Figure 2.

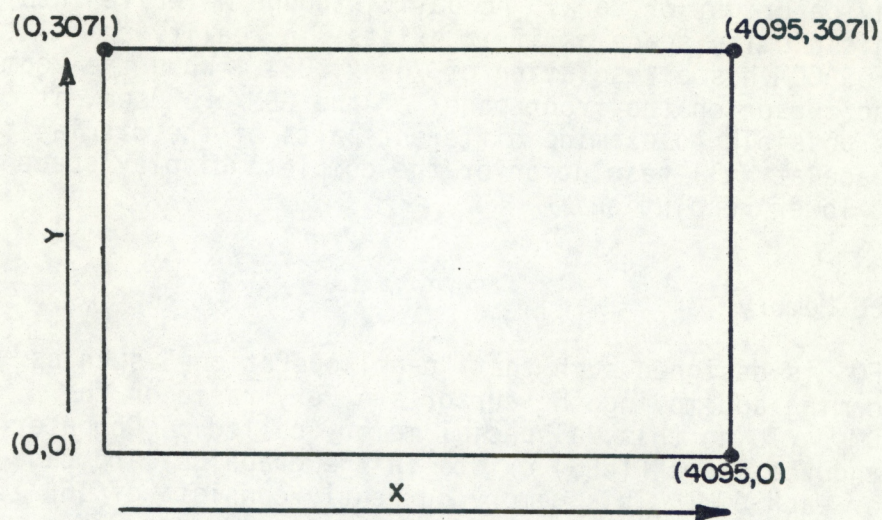


Figure 1. The AFOS display space is a cartesian grid with $0 \leq X \leq 4095$ and $0 \leq Y \leq 3071$.

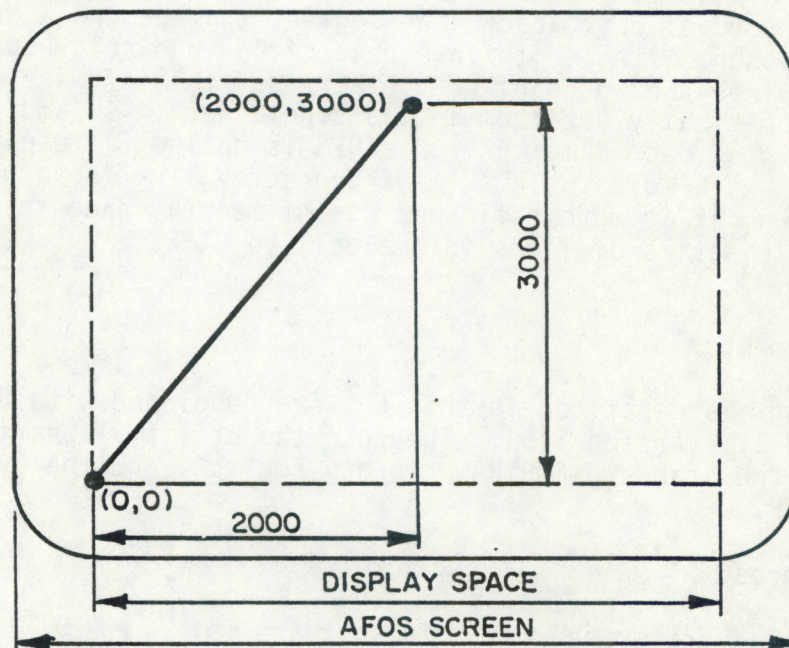


Figure 2. A line drawn on the AFOS screen from (0,0) to (2000,3000) by the subroutine "LINES". The dashed line represents the boundary of the display space.

At the most basic level, the AFOS graphic is composed of $3072 \times 4096 = 12,582,912$ pixels (picture cells) which may be bright or dark. Actually, though we write into this display space as if it exists, in reality the AFOS GDM has a resolution of 768×1024 . When the zoom and cursor on the front panel of the GDM are used, it is possible to examine different parts of the display space at full resolution or the complete display space at lower resolution.

3. CLS Memory

AFOS is designed such that "panel operations" such as zooming and moving the cursor are very rapid on the GDMs. To do this, a special memory called a "Computer Language Store" (CLS) exists in the graphics card cage for each GDM. This memory currently consists of 16K 16 bit words, and may at some time be expanded to 32K. The 16K words allow 4096 words for each of three overlays, plus 4096 for a map background. Actually, the boundaries between the overlays in CLS are not absolute, so an individual graphic can be longer than 4096, but clearly the sum of all CLS memory used for a GDM must not exceed 16K words.

When creating graphics with large numbers of lines or characters, the programmer must be conscious of the limited amount of CLS memory available. In general, each two alphanumeric characters (or special symbols) will use one word of CLS. Each line segment that has an X or Y component with length in excess of 63 in display space requires two words of CLS; otherwise, the software will automatically put them into a single word. In addition to this, each time TEXT or LINES is called, it uses a few extra words. Thus, if a connected line is to be drawn, it is more efficient use of memory space to do the whole series of lines in one call to LINES.

B. Environment:

The programs described in this CP were developed with Data General's FORTRAN IV, Version 5.54. They use the utility libraries UTIL.LB, BG.LB, the standard FORTRAN Library FORT.LB, and the system library AFOSE.LB.

C. References:

The primary reference for this software is:

"Universal Transmission Format Version 2.0", October 1, 1977. Systems Development Office, National Weather Service, Silver Spring, Maryland.

"AFOS Displays Programmers Reference", Ford Aerospace Corporation, WDL-TR7676A, Palo Alto, California.

II. Program Reference

In this section each of the programs which are part of the graphics family is discussed individually in alphabetical order.

A. CART:

1. Program Description

CART is a subroutine which, given a latitude and longitude, computes a location in AFOS display space for any of the basic polar stereographic map backgrounds. For example, if one wished to put a symbol on an LFM map background at the location of a city, they would call "CART" with the latitude and longitude of the station, and the routine would return the X and Y location of that city on the LFM map background. Example 2 of Appendix A gives an example of the use of the subroutine CART.

2. Call Statement and External Variables

CALL CART (LAT, LON, MAP, X, Y, IER)

The variables are:

LAT - An input variable that is the latitude of the point to be plotted. It is a real variable, and should be checked to assure it is within the limits of the map background being used (e.g., the United States map background should not be passed a LAT greater than 50, nor less than 20).

LON - An input variable that is the longitude of point to be plotted. It is a real variable and should be checked to assure it is in the domain of map background being plotted.

MAP - This is the map background selection, an input integer variable. Its allowable values correspond to the AFOS map background numbers.

MAP = 1; Northern hemisphere

MAP = 2; North America

MAP = 3; United States

X - AFOS display space X coordinate. It is an output integer variable.

Y - AFOS display space Y coordinate. It is an output integer variable.

IER - Output integer error code:

IER = 1 No error, normal exit.

IER = 2 Map selected not defined.

IER = 3 Illegal latitude.

IER = 4 Illegal longitude.

3. Machine Requirements

The subroutine CART requires 2000 locations in core. It does not access the disk, nor use any overlays.

B. LINES:

1. Program Description

LINES is a subroutine which creates connected lines from a sequence of absolute display space coordinates. The lines may be actual ("pen down") or blank ("pen up") and can use the "declutter" feature of AFOS (i.e., only appear at certain zooms). The zoom feature may also be disabled.

Typically, a programmer will call LINES numerous times, once for each continuous sequence of lines to be drawn. Alternately, all lines to be drawn could be done in a single call, with blank lines drawn between sequences of displayed vectors.

2. Call Statement and External Variables

CALL LINES (LX, LY, N, IZT, IZD)

Variables:

LX - An integer array containing the X display coordinates of a sequence of points. The absolute value of these points should be between 0 and 4095. If the programmer wants to draw a blank segment, the X coordinate of the destination point should be the negative of its actual display location:

LX(27) = -LX(27); makes line to point 27 blank.

LY - An integer array containing the Y display coordinates of a sequence of points. The value of these points should be between 0 and 3075.

N - The number of points to be drawn between.

NOTE: To dimension LX and LY, the programmer must estimate the largest number of points he will have in a single string; in other words, what is the maximum value that N will take on. The programmer may want an "IF" statement to check if the indices of LX and LY get larger than their dimensioned value.

IZT - An integer variable whose value determines zoom threshold:

IZT = 1; Display at all zooms.
IZT = 2; Display at 4X or higher.
IZT = 3; Display at 9X or higher.
IZT = 4; Display at 16X or higher.

IZD - An integer variable which can disable the zoom. If IZD = 1, then the line will be anchored at the location LX (1) and LY (1). Otherwise (e.g., IZD=0), it will zoom normally.

EXAMPLE We want to draw a line from point A (529,367) to point B (1927,3010). The following is an excerpt of a FORTRAN program to do this:

```
DIMENSION LX(2), LY(2)
LX(1) = 529
LX(2) = 1927
LY(1) = 367
LY(2) = 3010
N = 2
IZT = 1; All zooms
IZD = 0; Zoom enabled
CALL LINES (LX, LY, N, IZT, IZD)
```

3. Machine Requirements

The subroutine LINES uses about 2300 locations in core. Along with TEXT, it requires about 2K locations (total) of buffer memory. On first call to LINES or TEXT, a temporary file, GINS is created, which is deleted by the subroutine UTF. The size of GINS is similar to the size of the output graphic file.

C. TEXT:

1. Program Description

TEXT allows the display of ASCII characters and special fonts (such as the hurricane symbol). The symbols are passed in an array (two characters per word) called SCRIPT, with beginning points IP and JP in display space.

2. Call Statement and External Variables

CALL TEXT (SCRIPT, IP, JP, ISIZ, IZT, IXOF, IYOF)

Variables:

SCRIPT - An integer array which contains the characters and symbols to be output. Each sixteen bit word can contain two eight-bit characters.

An example of a sequence of characters to display the thunderstorm symbol is (see Table 1):

```
SCRIPT (1) = 22K; "Start of Symbols"
SCRIPT (2) = 41 ; Thunderstorm Symbol
SCRIPT (3) = 21K; "End of Symbols"
```

(Note: The K signifies octal in Data General FORTRAN.)

The maximum size of SCRIPT is 50, which allows 100 characters to be plotted. The character sequence must be terminated with a word equal to binary 0, e.g.:

```
SCRIPT (22) = 0
```

IP,JP - Input integer variables. These are the X and Y display locations of the lower left corner of the first character in the display string. These should be checked to assure they are in the display space domain, i.e., $0 \leq IP \leq 4095$, and $0 \leq JP \leq 3071$.

ISIZ - An input integer variable that specifies the size and type of character. If ISIZ equals 0, 1, or 2, the character will be standard size. If ISIZ = 3, it will be large size:

```
ISIZ = 0; Standard size, unblocked
ISIZ = 1; Standard size, blocked
ISIZ = 2; Standard size, reverse video blocked
ISIZ = 3; Large size.
```

Note: "Blocked" refers to the clearing of a rectangular area beneath the character. "Reverse video" reverses black and white on the character display block.

IZT - An integer variable whose value determines the zoom threshold. See IZT in LINES for values.

IXOF, IYOF - Integer input variables which are used for "offsets" from the points IP and JP. An offset allows the programmer to place a character a certain distance from a point, independent of the zoom level. This is useful for tasks such as station model plotting where the characters should maintain actual display pixel distance for visual purposes.

STANDARD CHARACTER SET

	00	10	20	30	40	50	60	70	80	90	100	110	120
0	nul	down space			(2	<	F	P	Z	d	n	x
1		up space)	3	=	G	Q	[e	o	y
2		1.5 new line		space	*	4	>	H	R	\	f	p	z
3		new line		!	+	5	?	I	S]	g	q	{
4				"	,	6	@	J	T	^	h	r	
5				#	-	7	A	K	U	_	i	s	}
6				\$.	8	B	L	V	'	j	t	~
7		reset flag		%	/	9	C	M	W	a	k	u	
8	back space	set flag		&	0	:	D	N	X	b	l	v	
9	fwd space			'	1	;	E	O	Y	c	m	w	

SPECIAL CHARACTER SET

	00	10	20	30	40	50	60	70	80	90	100	110	120
0	nul	down space	☉	S	(.)	7	⚡	~	▲	H	M	Σ	5
1	☉	up space	☉	\$	℔	J	⚡	∞]·	L	↙	↘	5
2	☉	1.5 new line	↙	℔	∇	△	⚡	(≡)]:	⚡	△	↘	^
3	☉	new line	↙	(S))(≡	≡	;	≡)	∠	△	↘	^
4	⊗	○	↙	=	,	∇	≡	↔]×	∠	△	↘	^
5	⊗	①	↙	==	•	⚡	≡	⚡]·	∞	○	↘	ψ
6	□	⊗	✓	=	*		≡	*]×	∠	∞	↘	ψ
7	★	reset flag	↙	↙	⚡		;	△]·	∠	—	↘	ψ
8	back space	set flag	↘	↘	~	⚡	∞	∇	℔	×	—	↘	
9	fwd space	☉	∞)(]·	⚡	⚡	:	⚡	∞	∞	↘	

Table 1. The decimal value of a character is obtained by adding the column and row values. For example, the thunderstorm has decimal value 40+1=41.

The absolute value of the offsets must be less than 64, and if they are zero, the offset is disabled.

3. Machine Requirements

The subroutine TEXT uses about 500 locations in core. See "Machine Requirements" for LINES.

D. UTF:

1. Program Description

The subroutine UTF converts the buffer files created by LINES and TEXT to "Universal Transmission Format" so that they may be displayed on the AFOS GDM and stored in the AFOS database. It must always be called after the last call to LINES or TEXT in a program. It also stores the product into the AFOS database, if the product name exists (i.e., is in either the PIL or the WISH list).

The graphics format of the files used by AFOS are described in Appendix D.

2. Call Statement and External Variables

CALL UTF (AFOS NAME, RDOSNAME)

AFOSNAME - An integer array containing the nine character AFOS product identifier.

RDOSNAME - An integer array containing the RDOS filename (up to 10 characters plus a period and two-character extension).

Example:

CALL UTF("SLCMISGP1", "GRAPHIC.01")

3. Machine Requirements

UTF uses about 1200 words of memory.

IMPORTANT: The RDOS file, (GRAPHIC.01 in the example above) must be a random or contiguous file. Sequential files are forbidden for graphics.

4. Caution and Restriction

Program must be run on DPØ or be linked from DPØ or else graphic will not automatically store in the database.

ACKNOWLEDGMENT

I would particularly like to thank Len Snellman, Chief, Scientific Services Division Western Region Headquarters, for his foresighted and perspicacious approach to applied meteorology, including support for such as this.

Thanks to Linda Miller and Evelyn Allan for being able to convert my hand-writing to neatly typed pages.

Programming credits:

CART - D. Laurine.

LINES - J. Fors, A. MacDonald.

TEXT - A. MacDonald, J. Fors.

UTF - A. MacDonald, J. Wakefield.

APPENDIX A

EXAMPLES OF GRAPHICS CREATION

The best method to illustrate the procedures required for creating graphics is to show some examples. In these Appendices are two complete examples showing different aspects of graphics creation. Beginners in graphic creation should have little trouble copying the technique to create simple graphics similar to those in the examples. Once this has been done, the user can proceed to graphics that are more interesting and useful.

In each of the examples three steps are followed:

- STEP 1: A main program is presented which creates the graphic using the graphics subroutines described in this CP. (Appendix A.)
- STEP 2: The program is compiled and loaded. (Appendix B.)
- STEP 3: The program is run, creating an RDOS graphic file, which may be displayed directly using the DSP: command, or may be called from the AFOS database for display. (Appendix C.)

To create FORTRAN source programs like the examples which follow, the programmer would use an ASCII editor, such as Data General's EDIT or SPEED.

TYPE EXAMPLE1.FR

C PROGRAM: EXAMPLE1.FR

C

INTEGER LX(4),LY(4),SCRIPT(2)
COMMON /D/ LX,LY,SCRIPT

DATA LX/100,300,200,100/

DATA LY/100,100,300,100/

DATA SCRIPT/"AFOS"/

C***** USE SUBROUTINE LINES TO DRAW TRIANGLE

N=4 ; FOUR POINTS
IZT=1 ; DISPLAY AT ALL ZOOM THRESHOLDS
IZD=0 ; ZOOMABLE LINE

CALL LINES(LX,LY,N,IZT,IZD)

C***** USE SUBROUTINE TEXT TO WRITE "AFOS"

IP=193 ; I COORDINATE OF SCRIPT START
JP=180 ; J COORDINATE OF SCRIPT START
ISIZ=2 ; REVERSE VIDEO BLOCKED
IZT=1

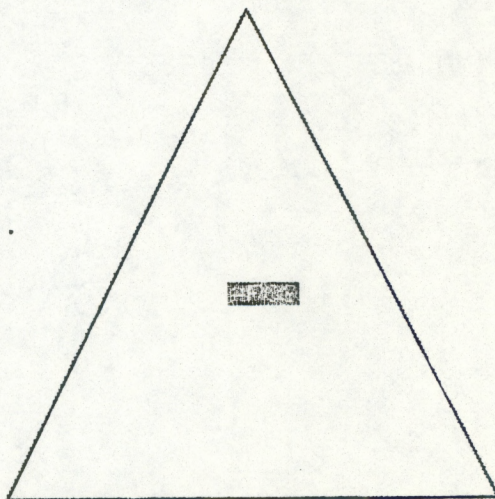
IXOF=0 ; WILL NOT USE OFFSET FEATURE
IYOF=0

CALL TEXT(SCRIPT,IP,JP,ISIZ,IZT,IXOF,IYOF)

C***** CONVERT TO UNIVERSAL TRANSMISSION FORMAT

CALL UTF("AFOSPROD1","RDOSFILE")

STOP
END




```

TYPE EXAMPLE2.FR
C  PROGRAM:  EXAMPLE2.FR

C  PURPOSE:  THE PROGRAM PLOTS A HURRICANE SYMBOL OVER THE GREAT
C  SALT LAKE.  WHEN HALLGREN SEES THIS HE'LL GIVE UTAH AN OCEAN SERVICES
C  UNIT.

```

```

      INTEGER SCRIPT(4)

```

```

      REAL LAT,LON

```

```

C  THE ROUTINE CART WILL BE USED TO FIND THE DISPLAY SPACE LOCATION,
C  (IP,JP), OF THE SALT LAKE.  THE LATITUDE IS  41 AND THE LONGITUDE
C  IS 112.5.  THE UNITED STATES MAP BACKGROUND WILL BE USED.

```

```

      LAT=41
      LON=112.5
      MAP=3

```

```

      CALL CART(LAT,LON,MAP,IP,JP,IER)
      IF(IER.NE.1) TYPE "EX2 PROB 1", IER

```

```

C*****

```

```

C  TO LOAD THE HURRICANE SYMBOL, WE START SPECIAL CHARACTERS,
C  LOAD THE SYMBOL, AND STOP SPECIAL CHARACTERS.

```

```

      SCRIPT(1)=22K  ; START OF SYMBOLS
      SCRIPT(2)=121  ;  HURRICANE SYMBOL
      SCRIPT(3)=21K  ;  END OF SYMBOLS
      SCRIPT(4)=0    ;  END OF TEXT STRING

```

```

      ISIZ=3  ; DOUBLE SIZE FONT
      IZT=1  ; DISPLAY AT ALL ZOOMS

```

```

      IXOF=0  ; NO OFFSET
      IYOF=0

```

```

      CALL TEXT(SCRIPT,IP,JP,ISIZ,IZT,IXOF,IYOF)

```

```

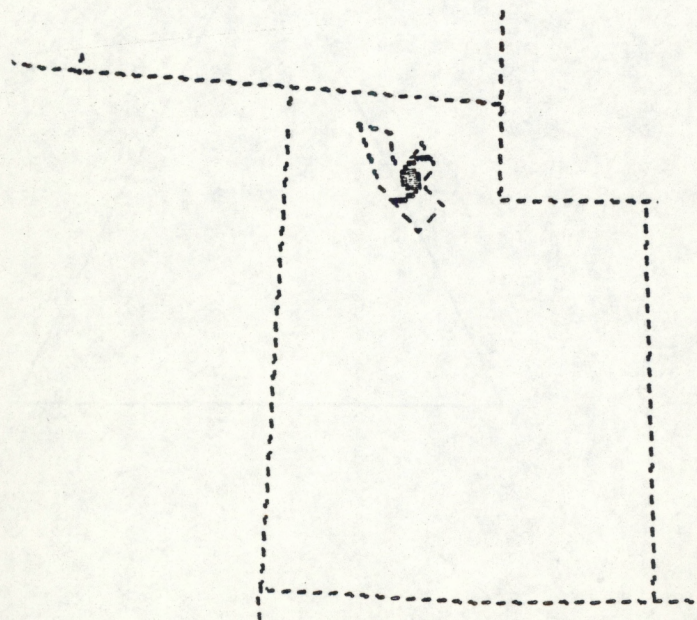
      CALL UTF("NMCGPHT20","HUR.GP")

```

```

      STOP
      END

```



APPENDIX B

COMPILING AND LOADING A FORTRAN PROGRAM

To compile a program such as EXAMPLE1.FR, the FORT command is used:

```
FORT EXAMPLE1.FR
```

```
PROGRAM IS RELOCATABLE
```

```
.TITL .MAIN
```

(Note: In the Appendices, user input commands will be underlined for clarity.)

The relocatable loader is used to make the "core image" or "save file" which will create the graphic.

The command line that creates a program is generally referred to as the "load line". It is good programming technique to create an ASCII file whose contents are the load line, for a couple of reasons. First, during the debugging process the program is ordinarily loaded many times; therefore, a single word command is easier to use. Second, the file exists as a record of what is in the load line for future reference.

For the program EXAMPLE1, the load line "macro" is called LLEXL.MC and its contents are

```
RLDR/P/N EXAMPLE1 AG.LB BG.LB UTIL.LB FORT.LB AFOSE.LB
```

The extension .MC tells RDOS that this is a "macro" or indirect file, which means execute the contents of the file. In this case the programmer would do the relocatable load as follows:

```
LLEX1
```

After a successful load, a "save" or "executable" file exists called EXAMPLE1.SV.

APPENDIX C

RUNNING AND DISPLAYING A GRAPHICS FILE

To run a program once it has been created by the relocatable loader, one merely types the name of the program at the system console (the Dasher) or uses the RUN: at any AK to do the equivalent:

At system console:

EXAMPLE1

At the AK:

RUN: EXAMPLE1

Once the program has been run, it should create a graphic file. In the case of EXAMPLE1, the RDOS graphic file was titled RDOSFILE.

To display this RDOS file directly from the disk, the DSP: command from the AK may be used. For illustrative purposes, assume the file is on DP2. Then at the AK, one sets one of the GDM output switches, and types:

DSP:DP2:RDOSFILE

It can also be called as a standard AFOS product from the AK:

AFOSPROD1

APPENDIX D

UNIVERSAL TRANSMISSION FORMAT AND DEBUGGING

Part 1. Graphic Primitive Format:

The UTF graphics file created by EXAMPLE1 is shown below. It is printed by FPRINT in two ways, octal word and octal byte:

```

FPRINT/Z RDOSFILE;FPRINT/Z/B RDOSFILE
 0 040506 047523 050122 047504 030460 030060 177777 177777 AFOSPROD1000....
10 002400 140400 000000 010020 000014 000000 000000 000303 ..A.....C
20 000000 062000 062000 003000 144000 000037 116000 144037 .....H.....H.
30 116037 034305 020000 140400 132101 043117 051603 000000 ..8E .A.4AFOS...
40 000000 000000 000000 000000 000000 000000 000000 000000 .....

***
 0 101 106 117 123 120 122 117 104 061 060 060 060 377 377 377 377
10 005 000 301 000 000 000 020 020 000 014 000 000 000 000 000 000
20 000 000 144 000 144 000 006 000 310 000 000 037 234 000 310 037
30 234 037 070 305 040 000 301 000 254 101 106 117 123 203 000 000
40 000 000 000 000 000 000 000 000 000 000 000 000 000 000 000 000

***

```

We now describe the graphic file in detail. Each section of the description below is referenced to a circled number in the byte FPRINT above:

- (1) First nine bytes, words 0_8 to 4.5_8 : This is the AFOS product identifier,

AFOSPROD1.

- (2) Next three bytes, words 4.5_8 to 6.0_8 above: The AFOS address, which should always be internal,

"000".

- (3) Words 6_8 and 7_8 : No-ops, set to -1, Octal 177777.

- (4) Word 10_8 : 2400K

- (5) The next six words are the Graphic Product Definition.
(Note: In this example, an extra byte has been stuffed in as described in Part 2 of this Appendix.)

Note: To understand the discussions of the remaining graphics primitives, the reader should refer to the figures in Part 4 of this Appendix.

The GPD must always appear once per graphic product as the first subset. It serves to define the properties of the product for the AFOS graphics system. The subset is always 6 words long. Assignments for PI are as follows (all numbers given decimal);

- 1: Northern hemisphere (master of entire hemisphere)
- 2-74: Mesocale northern hemisphere (high resolution master of a section of hemisphere).

IMAX (word 3) defines the horizontal width of the graphic area in pixels (resolution points). JMAX (word 4) is the vertical height of the plot area in pixels. YEAR (word 5) appears in units and tens. TIME (word 6) is given to the nearest minute of the 24-hour clock.

- (6) Next, beginning in the above file at location 17.5, is a string of relative vectors.

This graphic information subset is comprised of an arbitrary length string of x and y coordinate modifiers (Δx , Δy) based on the string starting point in the I and J coordinate field (words 2 and 3). Each successive Δx , Δy is added algebraically to the last computed I, J coordinate position to produce a graphic line formed from these successive points. System support software will, of course, join adjacent points with straight line segments.

Positive Δx entries modify I coordinates to the right, negative to the left. Positive Δy values move J coordinates up, negative down. Negative values are entered in 2s complement notation.

The detail of the relative vector format is found in Part 4 of the Appendix.

- (7) The text code begins in the above graphics file at location 31.5.

This subset provides the user with the ability to enter character strings which start at some desired distance (given by Δx , Δy in word 4) from a coordinate reference point (in words 2 and 3). This Δx , Δy offset points to the lower left corner of the 1st character in the string and remains a constant distance in pixels, regardless of any hardware zoom selection executed on the device by the operator.

- (8) Every UTF file is terminated with an "End of Text" byte, 203_8 , such as the right byte in word 36₈ above. If the 203_8 byte is not present, the file will not display on AFOS.

Part 2. Byte Stuffing:

There is one more important facet of UTF that the programmer should understand. Since the 203_8 byte always terminates the file, there must be a way to substitute for this byte when it naturally appears in the data. UTF uses the following substitutions.

Whenever _____ appears	it substitutes
203 ₈	020 ₈ 014 ₈
020 ₈	020 ₈ 020 ₈

Part 3. Debugging and Changes:

Knowledge of UTF outlined in this Appendix is useful in two ways. First, the debugging process is greatly aided. If a programmer's file will not display, they may do an FPRINT of the display file to trace down the problem. Many problems, such as a missing "End of Text" are diagnosed in this way.

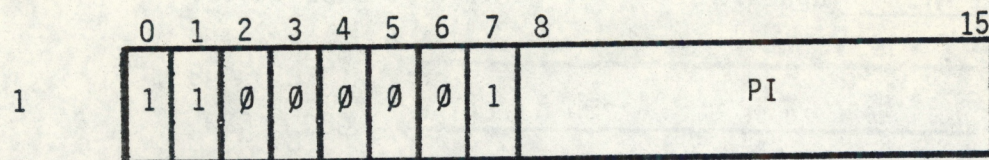
The second useful tool is that sophisticated users may make use of other capabilities of UTF not explicitly available in this set of programs. For example, the programmer may redefine the maximum display space coordinates by changing words IMAX and JMAX in the Graphic Product Definition. Partly for this reason the source code for the graphics programs is included on the AFOS Graphics floppy.

Part 4. UTF Detail:

The following four pages, taken from "Universal Transmission Format" (see the first reference), describe the UTF primitive formats in detail.

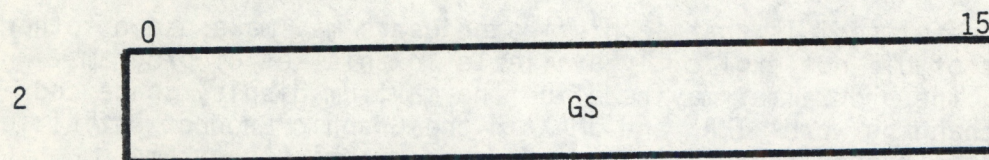
GRAPHIC PRODUCT DEFINITION

WORD

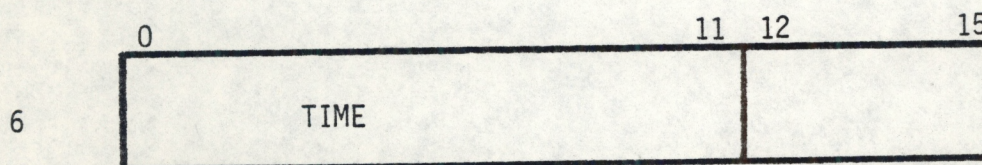
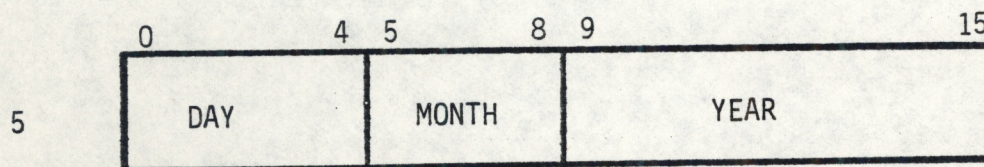
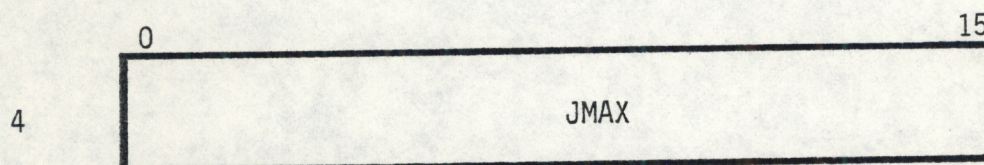
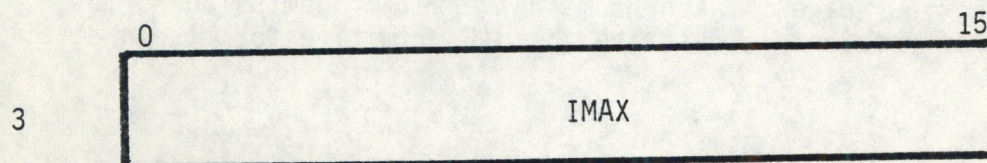


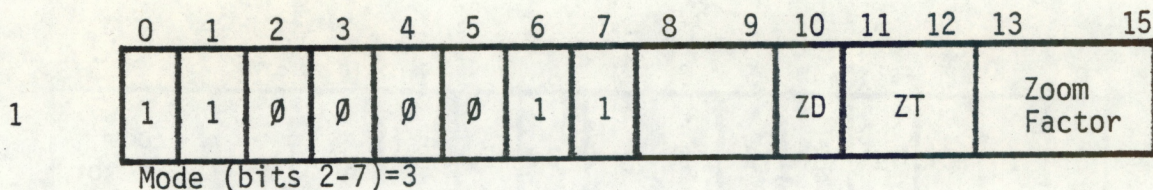
Mode (bits 2-7) = 1

PI - Projection Indicator. If this product is not associated with any background, then PI=0. If a background is to be used, then PI holds the identification number of the cartesian master background file from which the background was extracted. These numbers are assigned in such a way that they identify not only the master file, but also the type of geographic projection used (mercator, polar, etc.). For further information, see the AFOS GRAPHIC SYSTEM USER GUIDE, Section 1.



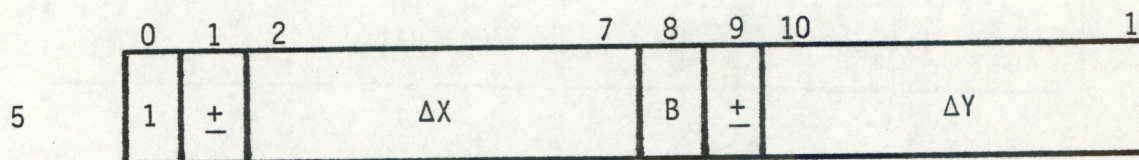
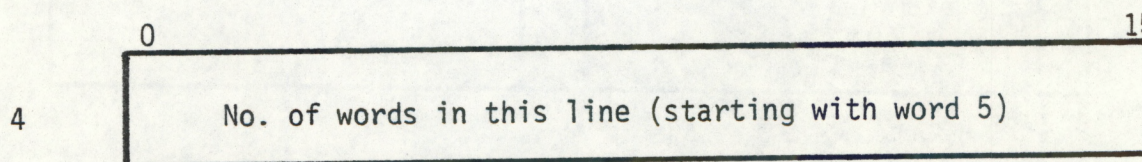
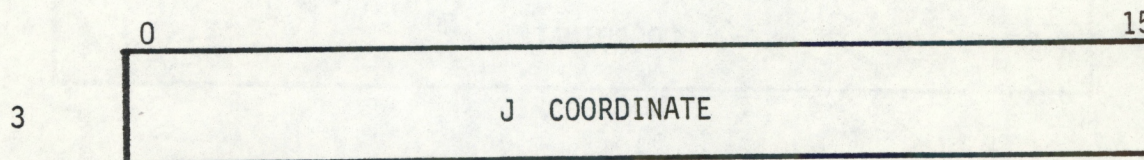
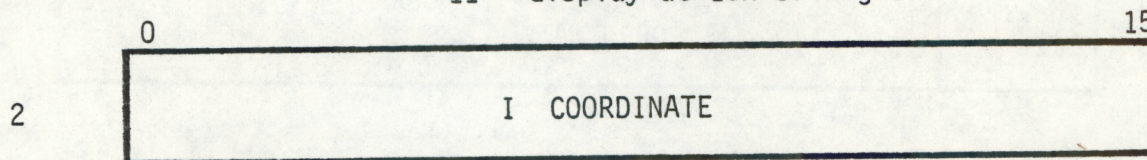
GS - Geography Scale. If PI is nonzero, GS is set to the numeric ratio of the projection in ten thousands. For example, if the projection is at a scale of 1:60 million, GS=13560₈ (6000₁₀). For a scale of 1:500,000 (1/2 million), GS=62₈ (50₁₀).



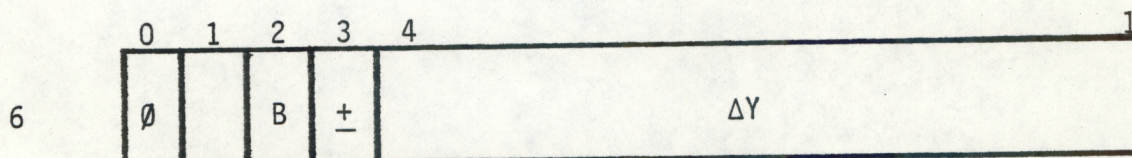
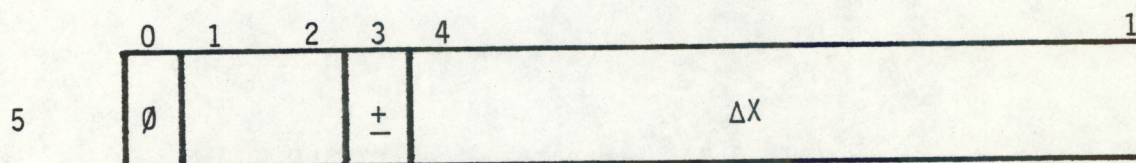


ZD:Zoom Disable. If ZD=1, vector lengths upon display will remain at the basic product zoom size as determined by Zoom Factor. If the zoom hardware/software on the device does not support this function, the option becomes a NOP.

ZT:Zoom Threshold. 00 - display at all zoom levels.
 01 - display at 4X or higher.
 10 - display at 9X or higher.
 11 - display at 16X or higher.



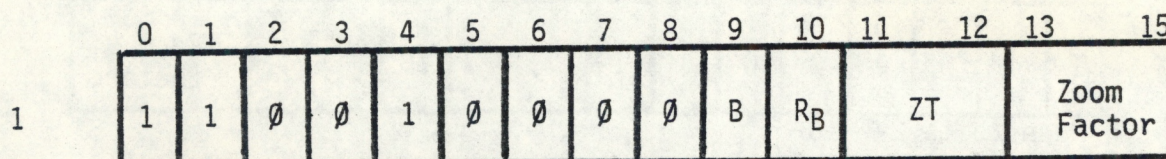
OR



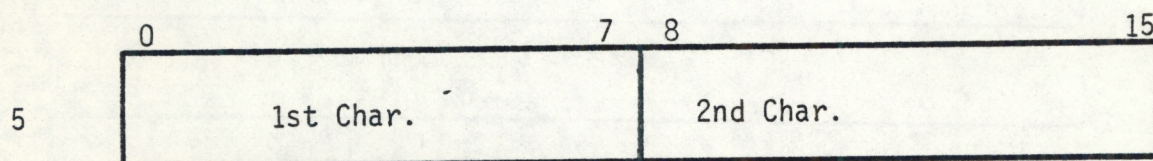
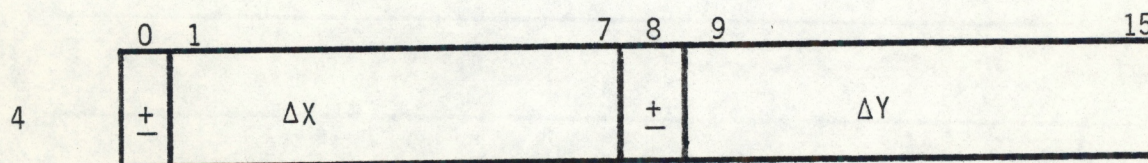
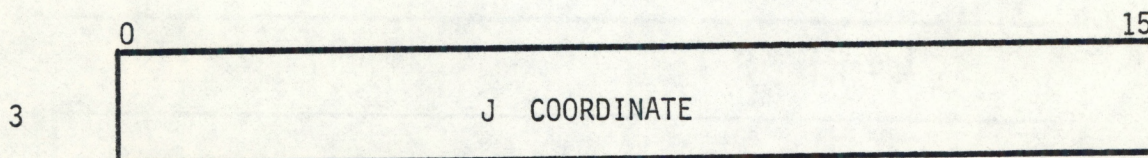
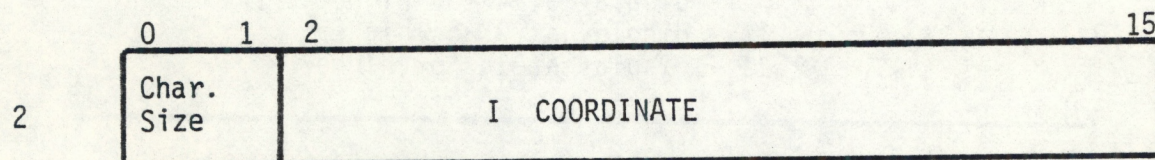
B:Blank Vector Indicator. If B=1, cursor is moved to the specified location but no line is drawn. If B=0, line is drawn.

WORD

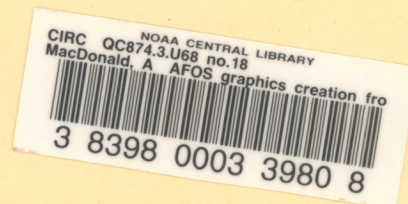
OFFSET ALPHA-NUMERIC Chars.



Mode (bits 2-7) = 10

For B, R_B, and Char. Size subfields, see Mode 5.Zoom Factor: applies only to the I and J coordinates, not to the ΔX , ΔY field or the character font.

Next mode byte (Bit 0=1) terminates ASCII string.



NOAA SCIENTIFIC AND TECHNICAL PUBLICATIONS

The National Oceanic and Atmospheric Administration was established as part of the Department of Commerce on October 3, 1970. The mission responsibilities of NOAA are to assess the socioeconomic impact of natural and technological changes in the environment and to monitor and predict the state of the solid Earth, the oceans and their living resources, the atmosphere, and the space environment of the Earth.

The major components of NOAA regularly produce various types of scientific and technical information in the following kinds of publications:

PROFESSIONAL PAPERS — Important definitive research results, major techniques, and special investigations.

CONTRACT AND GRANT REPORTS — Reports prepared by contractors or grantees under NOAA sponsorship.

ATLAS — Presentation of analyzed data generally in the form of maps showing distribution of rainfall, chemical and physical conditions of oceans and atmosphere, distribution of fishes and marine mammals, ionospheric conditions, etc.

TECHNICAL SERVICE PUBLICATIONS — Reports containing data, observations, instructions, etc. A partial listing includes data serials; prediction and outlook periodicals; technical manuals, training papers, planning reports, and information serials; and miscellaneous technical publications.

TECHNICAL REPORTS — Journal quality with extensive details, mathematical developments, or data listings.

TECHNICAL MEMORANDUMS — Reports of preliminary, partial, or negative research or technology results, interim instructions, and the like.



Information on availability of NOAA publications can be obtained from:

**ENVIRONMENTAL SCIENCE INFORMATION CENTER (D822)
ENVIRONMENTAL DATA AND INFORMATION SERVICE
NATIONAL OCEANIC AND ATMOSPHERIC ADMINISTRATION
U.S. DEPARTMENT OF COMMERCE**

**6009 Executive Boulevard
Rockville, MD 20852**