terak
CORPORATION

FORTRAN IV V2.1
RELEASE GUIDE

TERAK P/N 60-0048-001

REV 1

14405 NORTH SCOTTSDALE ROAD • SCOTTSDALE, ARIZONA 85254 • (602) 991-1580

CONTENTS

PREFACE

This document is an introduction to the TERAK FORTRAN IV V2.1 Software Kit. It lists the contents of the software kit, provides suggestions for user protection, and defines the functional capabilities of the "READY TO USE" FORTRAN IV V2.1 compiler and system library. All sections should be read. Particular attention should be given to section 3, as it provides a step by step procedure for backing up the diskettes. Following this procedure will insure that the user is protected from accidental destruction of the FORTRAN IV V2.1 system diskettes.

SECTION 1

FORTRAN IV V2.1 DOCUMENT/DISKETTE LIST

The following documents and disks are contained in the Terak version of the DEC FORTRAN IV V2.1 system software. If any items are missing, contact the Terak Software Service Group.

DOCUMENTS: (F4/RT-11 V2.1 Documentation Kit, TERAK # 95-0007-001)

1. FORTRAN IV V2.1 RELEASE GUIDE, (this document)	TERAK # 60-0048-001
2. F4/RT-11/RSTS/E V2.1 Release Notes, AA-H273A-TC	TERAK # 60-0053-001
3. RT-11/FORTRAN IV INSTALLATION GUIDE, AA-5240-TC	TERAK # 60-0054-001
4. PDP-11 FORTRAN IV LANGUAGE REFERENCE MANUAL, DEC-11-LFLRA-C-D	TERAK # 60-0013-001
5. RT-11/FORTRAN IV USER'S GUIDE, DEC-11-LRRUB-A-D	TERAK # 60-0055-001
6. F4 SIGGRAPH CORE-79 GRAPHICS RELEASE GUIDE	TERAK # 60-0049-001
7. F4 LOW LEVEL GRAPHICS RELEASE GUIDE	TERAK # 60-0050-001
8. 2 each 8510/a SYSTEM REFERENCE CARD	TERAK # 60-0022-001

DISKS: (F4 V2.1 Diskette Distribution Kit, TERAK # 94-0012-001).

1. FORTRAN IV V2.1 COMPILER OBJECTS (NO-BOOT)	TERAK # 61-0009-001
2. FORTRAN IV V2.1 LIBRARY OBJECTS (NO-BOOT)	TERAK # 61-0009-003
4. "READY TO USE" FORTRAN IV V2.1 COMPILER	TERAK # 61-0009-004
5. "READY TO USE" SYSTEM LIBRARY	TERAK # 61-0009-005
6. FORTRAN IV V2.1 GRAPHICS DEMOS	TERAK # 61-0009-006

SECTION 2

MINIMUM FORTRAN IV V2.1 SYSTEM CONFIGURATION

The minimum hardware configuration consists of the TERAK 8510/a GRAPHICS COMPUTER PROCESSOR SYSTEM plus either a TERAK 8512 or TERAK 8515 disk drive. This configuration is predicated by the diskette storage requirements for the FORTRAN IV compiler and the system library. The compiler plus a minimal support operating system consumes the storage capacity of a diskette. Similarly, the system library plus a minimal support operating system consumes a diskette. Although the user may see "free" blocks available on the diskette, the amount of free space would allow for the development of only the most trivial of FORTRAN programs. (See discussion in section iv, SPACE PROBLEMS and SOLUTIONS in "RT-11/85a OPERATING SYSTEM V3B RELEASE GUIDE", TERAK P/N (60-0029-001).

The minimum software support that the user must have is a functional RT-11 V3B Operating System. This requirement is established by DEC ("RT-11 FORTRAN IV Installation Guide", AA-5240C-TC section 1.3 page 2).

SECTION 3

GUIDELINES FOR DISKETTE INSPECTIONS AND BACKUP

Upon receiving the software distribution disks, the manager of the installation site should :

1) scan each disk for damaged data,
2) verify the directory contents of each disk, and
3) make a backup copy of each disk.

These procedures are documented in the RT-11 System User's Guide, and are summarized, below for convenience.

NOTE: That these procedures apply for RT-11 V3b and systems with at least two disk drives.

NOTE: In the following procedures, character sequences to be typed or observed are enclosed in single quotes. The expression <Control-X>, where X is a letter, indicates that the control key is to be depressed, then the indicated letter key is to be typed. The expression <XXX> indicates that the XXX key is to be typed; for example, the expression <return> indicates that the return key is to be typed.

INSPECTION OF DISKS FOR DAMAGED DATA

The demonstration disk (61-0009-006) may be bootstrapped by pressing the power switch upward, momentarily, and then inserting the disk into drive QX0. Type 'R DUP <return>'. The response '*' will be displayed. To scan the demonstration disk for damaged data, type 'QX0:/K <return>'. To scan for damaged data on the other disks in the software kit, insert the selected disk into drive QX1 and type 'QX1:/K <return>'. In either case, after about a half minute, the response '*' should be displayed. If the scanning does not complete, or if any message such as:

```
'BAD BLOCKS  TYPE  FILENAME  REL BLK
     XXX     HARD'
```

is displayed, data on the disk may have been damaged. Contact the Terak Software Service Group.

INSPECTION OF DIRECTORY CONTENTS OF DISKS

The demonstration disk (61-0009-006) may be bootstrapped by pressing the power switch upward, momentarily, and then inserting the disk into drive QX0. To display the directory of the demonstration disk, type 'DIR QX0: <return>'. To display the directory of the other disks in the software kit, insert the selected disk into drive QX1 and type 'DIR QX1: <return>'. In either case, the disk directory will be displayed on the screen. To hold the scrolling of the display, type <Control-S>; to release scrolling, type <Control-Q>. The directory of each disk is provided later in this document, and should be compared with the directory displayed. If any differences are found, contact the Terak Software Service Group.

DISK BACKUP PROCEDURE FOR MULTIDRIVE SYSTEM

CAUTION: The system disk (any disk in QX0:) should not be swappped with any other disk without performing the re-boot operation. Failure to re-boot can cause the RT-11 operating system, to perform a "swap" between memory and the disk. This "swap" will write over a portion of the disk destroying the data content of the disk blocks involved. To avoid any such damage to disk content, swapping of system disks without re-booting, is discouraged unless explicitly stated in the documentation.

To perform the backup process you:
(1) turn the system on (or press power switch upward and release);

(2) insert into the bottom drive (QX0) the "FORTRAN IV GRAPHICS LIBRARIES" distribution disk. The system should boot and respond with

```
RT-11    V03B-00  .SGM

.LOAD TT:

.SET USR NOSWAP

.SET TT:SCOPE

.
```

(3) Now take a blank disk, label it 'Copy 1 of "FORTRAN IV GRAPHICS LIBRARIES" and insert it into the second drive, QX1.

(4) Respond to the moniter prompt (.) with the command:

'COPY/DEVICE QX0: QX1:<Return>'

If you make an error in typing a command, you may either (a), backspace over your error to correct it, or (b), use the delete key. The system usually does not act on a command until you hit the Return key. (Important exceptions to this are the control commands executed by simultaneously pressing the CTRL key and the appropriate letter key; for instance: <CTRL>C, which is a command to exit the current activity and return to the (.) prompt.) The system will respond with the statement

```
QX1:/Copy are you sure?
```

At this point, you must be absolutely certain that the statement says QX1: rather than QX0:. If it does say QX0: hit the Return key only, and the system will respond with another monitor prompt (.); then reenter the COPY/DEVICE command exactly as shown above.

(5) Once the system has responded with exactly the statement above, type

'Y<Return>'

After about one minute, the system should return with the (.) prompt. This indicates that the disk in drive QX0: has been successfully backed up onto the disk in drive QX1:. If any other messages occur, attempt the procedure again. If failure persists, contact the Terak Software Service Group. The above procedure should be used to back up the following disks: (a) "READY TO USE" FORTRAN IV V2.1 COMPILER: (b) "READY TO USE" SYSTEM LIBRARY. Repeat the above steps (1 - 5) with the following modification at step 1: you need not turn the system off and then back on, but should, instead reboot by pushing the top half of the toggle switch on the 8510 momentarily.

Next, use the following procedure to back up the NON-BOOTABLE disks: a) FORTRAN IV V2.1 COMPILER OBJECTS
b) FORTRAN IV V2.1 LIBRARY OBJECTS

To start the process:

(1) Boot any one of the three disks you have created above.

(2) In response to the monitor prompt(.), type:

'R DUP<return>'

To run the program DUP.SAV. DUP will respond with:

*

(3) In response, type:

'QX1:A=QX0:/I/W<Return>'

DUP will respond with:

continue?

(4) DO NOT type a response yet. Remove the multidrive system disk from drive QX0: and insert the "COMPILER OBJECTS DISK" into the bottom disk drive (QX0) and an appropriately labeled blank disk in the upper drive (QX1).

CAUTION

Before you proceed, check that your command was entered exactly as shown above and that the distribution drive disk is in QX0 and a blank disk is in QX1.

(5) Once you are absolutely sure that everything is correct, type:

'Y<Return>'

After about one minute, the system should return with the message:

Insert system disk,Are you Ready?

This indicates that the disk in drive QXØ: has been successfully backed up onto the disk in drive QX1:. If any other messages occur, attempt the procedure again. If failure persists, contact the Terak Software Service Group.

(6) Remove the distribution disk from drive QXØ:. Insert into the bottom drive the original system disk and type:

'Y<Return>'

(7) and then to DUP's prompt (*) type

'<Ctrl>C'

At this point, you will have completed the backup of all the RT- 11 system disks. Put the original disks you received in this software kit in a safe place, and use the disks that you have just created for the remainder of this procedure.

SECTION 4

USER DEVELOPMENT ENVIRONMENT

The "READY TO USE" FORTRAN IV V2.1 Compiler and the "READY TO USE" System Library have been organized to function in the following manner. The user's application program is resident on the disk in drive QX1. The "READY TO USE" Compiler is on the disk inserted into drive QXØ. The application program is compiled with the compiler object being stored on the user's disk in QX1. The compiler is then removed from QXØ and the "READY TO USE" System Library disk is inserted (Remember to re-boot after inserting the disk). The user's application program is then linked with the system library and the run-time program image is stored on QX1.

An alternate approach would be to create a NO BOOT disk and copy the compiler and system library onto it. The NO BOOT restriction is because the compiler and system library consume the storage capacity of the disk. The mode of operation would then be for the user to create a disk which contains the operating system and application programs. The compiler and system library would be inserted into QX1 and the program development disk would be inserted into QXØ. The creation of this compiler and system library configuration is left to the user.

SECTION 5

UTILIZING THE GRAPHICS DEMO DISKETTE

The purpose of this diskette is to provide the user with operational experience and examples for utilizing the FORTRAN IV V2.1 Compiler, System library, and TERAK graphics libraries. Each demonstration is provide as sources which can be studied, compiled, linked, and executed. In addition a ready to run version is provided to quickly put graphics images on to the screen. For those users just beginning with FORTRAN, a review of the FORTRAN Language Reference Manual is suggested. For more advanced programmers, the FORTRAN User's Guide should be studied for a review of system and I/O extensions. For graphics programmers, studying the Graphics Release Guides is advised.

There are two different levels of demonstrations provided. The first deals with a set of graphic functions that are very primitive in capability. The second level of demonstration deals with the SIGGRAPH CORE-79 compatible graphics capabilities.

To execute the demonstrations, bootstrap the copy of the Demonstration Disk (60-0009-006) by pressing the power switch upward, momentarily, and then insert the disk into drive QX0. Enter the Date and Time, e.g. 'DATE 12-JUL-79 <return>' and 'TIME 21:25 <return>'.

To execute the LOW LEVEL graphics demonstration, type 'R LOWDEM <return>'. At that time a dot will appear on the screen to the left of the cursor. The cursor will continue to move left to right, leaving a trail of dots behind. This time is utilized by the program to calculate the surface for display. When the image has been calculated, the screen will clear and a sine surface will be drawn on the screen. When the image is complete the program will wait until a '<return>' is typed. Typing a '<return>' will allow the program to exit back to the RT-11 operating system.

Two programs are provided for the SIGGRAPH CORE-79 demonstrations. To run the first program type 'R SGDEM1 <return>'. This will display a sine wave and a set of axes. Typing a '<return>' will exit to the RT-11 operating system. To run the second program, type 'R SGDEM2 <return>'. This will draw a graph similar to figure 2 in the FORTRAN IV SIGGRAPH CORE-79 GRAPHICS RELEASE GUIDE. To exit the demonstration and return to RT-11, type '<return>'.

The procedure for compiling and linking each demonstration program is contained on the diskette under the same file name as the demonstration program, but with the extension ".DOC" (listings are also included in the Graphics Release Guides). These files not only show how the program was created, but also provide examples on how to use the FORTRAN IV Compiler and System Library as distributed by TERAK.

SECTION 6

BUILDING VARIATIONS OF THE FORTRAN IV COMPILER

The FORTRAN IV Compiler can be built in many different ways depending upon the need of the user. Generally, one variation is selected based upon a tradeoff of memory capacity, mass storage capacity, execution speed, and compiler defaults. A detailed discussion of the selectable options and their impact on the installation of the compiler is contained in sections 2.1 - 2.3.2.1 (pages 3-5) of the RT-11 FORTRAN IV INSTALLATION GUIDE.

The options selected for the "READY TO USE" FORTRAN IV V2.1 compiler include:

1. 56 lines maximum per listing page
2. 136 characters maximum for ASCII records
3. 6 active I/O channels maximum
4. FIS hardware
5. inline code "SIZE" optimization
6. inline and threaded code expansion.

The RT-11 V3B command sequence used to install the compiler on the TERAK distribution diskette is contained in APPENDIX A and on the diskette in file FORTRA.DOC . A differently configured compiler could be installed by duplicating the documented sequence and changing the appropriate option selection.

When selecting options to be installed in the compiler, the user should remember that the 8510/a supports the FIS instruction sets.

SECTION 7

BUILDING VARIATIONS OF THE SYSTEM LIBRARY

The system library can be built in many different ways depending on the needs of the user. Generally, one variation is selected based upon a tradeoff of memory capacity, mass storage capacity, execution speed, and compiler configuration. A brief discussion of the library capabilities and document references is included below.

LOW LEVEL GRAPHICS (LOWGRF.OBJ)	Routines which manipulate individual dots on the TERAK 8510/a graphics display. See the F4 LOW LEVEL GRAPHICS RELEASE GUIDE for details.
SIGGRAPH CORE-79 GRAPHICS (SIGGRF.OBJ)	Routines which provide graphics capabilities compatible with the SIGGRAPH CORE-79 graphics standard. See F4 SIGGRAPH CORE-79 GRAPHICS RELEASE GUIDE.

SYSTEM SUBROUTINE LIBRARY (SYSF4.OBJ)	A collection of subroutines which provide specialized processing capabilities not within the normal scope of FORTRAN. See the RT-11 ADVANCED PROGRAMMER'S GUIDE, chapter 4 for details.
FORTRAN OTS LIBRARY (FORLIB.OBJ)	This is the Object Time System (OTS) library that supports the FORTRAN language. See RT-11/RSTS/E FORTRAN IV USER'S GUIDE, chapter 2 for details.

The primary question regarding choice of library configuration, is whether the system library should be fragmented into individual functional libraries or be combined into one contigous library. There are arguments for and against each approach. In general, if enough disk storage is available, those library routines which are frequently used should be included in the system library. This simplifies the general program linking process. However, if the amount of disk storage allocated to the system library must be optimized or divided among several disks, then the fragmented approach may be the only choice. The user's enviroment, application and experience will be the deciding factors when building the system libraries. SECTIONS 2.4.1 and 2.4.1.1 in "RT-11 FORTRAN IV Installation Guide" provide some additional comments concerning the library configuration question. Regardless of the decisions made, the libraries must be either created or linked in a specific sequence. The order of the library files must appear to the LINKER in the following sequence:

```
LOWGRF.OBJ
SIGGRF.OBJ
SYSF4.OBJ
FORLIB.OBJ
```

The "READY TO USE" system library was configured using:

1. FIS hardware
2. no virtual arrays
3. threaded code subscript checking
4. no stand alone support
5. Low Level graphics
6. SIGGRAPH graphics
7. SYSF4

The RT-11 V3B command sequence used to install the system library (SYSLIB.OBJ) on the TERAK distribution diskette is contained in APPENDIX B and on the diskette in file SYSLIB.DOC A different library organiztion can be created by referring to the example and the system generation procedures.

SECTION 8

RESTRICTIONS AND ERRORS

Several DEC documentation corrections and compiler limitations are discussed in the RSTS/E and RT-11 FORTRAN IV RELEASE NOTES. The user is encouraged to read that information.

APPENDIX C of this document should be reviewed for additional discussion of restrictions and errors.

SECTION 9

DISKETTE DIRECTORIES

FORTRAN IV V2.1 COMPILER OBJECTS TERAK P/N 61-0009-001

```
FORGEN.SAV    20 05-Sep-77      FORTRA.HLP     5 13-Aug-77
DEMO  .FOR     2 01-Oct-78      F4LINK.COM     1 01-Oct-78
F4LINL.COM     1 01-Oct-78      F4LTHR.COM     1 01-Oct-78
FROOT .OBJ     4 01-Oct-78      F2     .OBJ   10 01-Oct-78
F3    .OBJ    11 01-Oct-78      F6     .OBJ   10 01-Oct-78
F7    .OBJ    12 01-Oct-78      F9     .OBJ   14 01-Oct-78
F11   .OBJ     9 01-Oct-78      F12    .OBJ    9 01-Oct-78
F14   .OBJ    16 01-Oct-78      F20    .OBJ   35 01-Oct-78
F15   .OBJ    16 01-Oct-78      F16    .OBJ   15 01-Oct-78
F17   .OBJ    15 01-Oct-78      F0     .OBJ   13 01-Oct-78
F1    .OBJ     9 01-Oct-78      F4     .OBJ    9 01-Oct-78
F5    .OBJ    18 01-Oct-78      F8     .OBJ   13 01-Oct-78
F10   .OBJ     6 01-Oct-78      F19    .OBJ   21 01-Oct-78
F21   .OBJ    16 01-Oct-78      F18    .OBJ   30 01-Oct-78
F13   .OBJ    17 01-Oct-78      OBJGSD.OBJ    24 01-Oct-78
INLINE.OBJ     1 01-Oct-78      THREAD.OBJ     1 01-Oct-78
LOOP  .OBJ    18 01-Oct-78      REGALO.OBJ    27 01-Oct-78
CDUMP .OBJ     8 01-Oct-78      CONVRT.OBJ    25 01-Oct-78
PEEP  .OBJ    17 01-Oct-78
37 Files, 479 Blocks
7 Free blocks
```

FORTRAN IV V2.1 LIBRARY OBJECTS TERAK P/N 61-0009-002

```
V2NS  .OBJ     5 01-Oct-78      V2S    .OBJ    6 01-Oct-78
VIRP  .OBJ    12 01-Oct-78      VIRNP  .OBJ   12 01-Oct-78
NHD   .OBJ    42 01-Oct-78      EAE    .OBJ   43 01-Oct-78
EIS   .OBJ    43 01-Oct-78      FPU    .OBJ   26 01-Oct-78
FIS   .OBJ    41 01-Oct-78      OTSCOM.OBJ   102 01-Oct-78
UNI   .OBJ     6 01-Oct-78      NOVIR  .OBJ    1 01-Oct-78
SYSF4 .OBJ    39 21-Mar-78
13 Files, 378 Blocks
108 Free blocks
```

FORTRAN IV V2.1 GRAPHICS LIBRARY TERAK P/N 61-0009-003

```
SWAP  .SYS     24 03-Dec-78       QX    .SYS      3 06-Jan-79
TT    .SYS      2 03-Dec-78       QXMNSJ.SYS     62 05-Mar-79
PIP   .SAV     16 03-Dec-78       DIR   .SAV     17 03-Dec-78
DUP   .SAV     21 06-Jan-79       STARTS.COM      1 11-Jan-78
CHRSET.SYS      4 05-Dec-78       GRFASC.MAC      5 20-Dec-77
HIDE  .MAC      5 20-Dec-77       SETBIT.MAC      4 20-Dec-77
CLRBIT.MAC      4 20-Dec-77       ANDFOR.MAC      3 20-Dec-77
ORFOR .MAC      3 20-Dec-77       XORFOR.MAC      3 20-Dec-77
TSTBIT.MAC      5 20-Dec-77       GRFPNT.FOR      1 20-Dec-77
LOWGRF.OBJ     12 12-Oct-79       LOWRLD.COM      2 12-Oct-79
SIGGRF.OBJ    128 17-Oct-79
21 Files, 325 Blocks
161 Free blocks
```

FORTRAN IV V2.1 GRAPHICS LIBRARY TERAK P/N 61-0009-004

```
SWAP  .SYS     24 03-Dec-78       QX    .SYS      3 06-Jan-79
TT    .SYS      2 03-Dec-78       QXMNSJ.SYS     62 05-Mar-79
PIP   .SAV     16 03-Dec-78       DIR   .SAV     17 03-Dec-78
DUP   .SAV     21 06-Jan-79       LINK  .SAV     29 05-Dec-78
EDIT  .SAV     19 05-Dec-78       STARTS.COM      1 11-Jan-78
CHRSET.SYS      4 05-Dec-78       FORTRA.DOC      8 15-Oct-79
FORTRA.SAV    203 05-OCT-79
15 Files, 411 Blocks
75 Free blocks
```

FORTRAN IV V2.1 "READY TO USE" SYSTEM LIBRARY TERAK P/N 61-0009-005

```
SWAP  .SYS     24 03-Dec-78       QX    .SYS      3 06-Jan-79
TT    .SYS      2 03-Dec-78       QXMNSJ.SYS     62 05-Mar-79
DUP   .SAV     21 06-Jan-79       LINK  .SAV     29 05-Dec-78
LIBR  .SAV     18 05-Dec-78       STARTS.COM      1 11-Jan-78
CHRSET.SYS      4 05-Dec-78       SYSLIB.DOC     11 17-Oct-79
SYSLIB.OBJ    254 17-Oct-79       DIR   .SAV     17 03-Dec-78
PIP   .SAV     16 03-Dec-78
13 Files, 462 Blocks
24 Free blocks
```

FORTRAN IV V2.1 GRAPHICS DEMO TERAK P/N 61-0009-006

```
SWAP  .SYS     24 03-Dec-78       QX    .SYS      3 06-Jan-79
TT    .SYS      2 03-Dec-78       QXMNSJ.SYS     62 05-Mar-79
PIP   .SAV     16 03-Dec-78       DIR   .SAV     17 03-Dec-78
DUP   .SAV     21 06-Jan-79       STARTS.COM      1 11-Jan-78
CHRSET.SYS      4 05-Dec-78       SGDEM1.DOC      7 17-Oct-79
LOWDEM.FOR      7 16-Oct-79       LOWDEM.DOC      7 16-Oct-79
SGDEM1.FOR      2 17-Oct-79       SGDEM2.DOC      7 17-Oct-79
SGDEM2.FOR      5 17-Oct-79       LOWDEM.SAV     99 17-Oct-79
SGDEM2.SAV     23 17-Oct-79       SGDEM1.SAV     16 17-Oct-79
18 Files, 323 Blocks
163 Free blocks
```

APPENDIX A

"FORTRA.DOC" LISTINGS

```
!*****************************************************************************
!*********************  NOTES DESCRIBING THE INSTALLATION  *******************
!*********************  PROCEDURE USED FOR "READY TO USE"  *******************
!*********************       FORTRAN IV V2.1 COMPILER      *******************
!*****************************************************************************
!
!FORTRA.DOC
!CREATED 9-28-79          UPDATED 9-28-79
!
!THE PURPOSE OF THE FILE IS TO DOCUMENT THE EXACT RT-11 V3B COMMAND
!     SEQUENCE USED TO INSTALL THE FORTRAN IV COMPILER CONTAINED
!     ON THE DISTRIBUTION DISKETTE, TERAK PART NUMBER 61-0009-004.
!-----------------------------------------------------------------------------
!
!** STEP #1 **
!PLACE THE "FORTRAN IV V2.1 COMPILER OBJECT" DISKETTE,
!     TERAK PART NUMBER 61-0009-001,
!     INTO QX1:
!-----------------------------------------------------------------------------
!
!** STEP #2 **
!PLACE THE TARGET DISKETTE FOR THE COMPILER INTO QX0:
!     THIS DISKETTE MUST CONTAIN THE MINIMUM OPERATING SYSTEM FILES:
!          SWAP.SYS
!          QX.SYS
!          TT.SYS
!          QXMNSJ.SYS
!          LINK.SAV
!          STARTS.COM
!          CHRSET.SYS
!
!THE FOLLOWING FILES (WHICH ARE INCLUDED ON THE DISTRIBUTED "READY
!     TO USE COMPILER DISKETTE, TERAK PART NUMBER 61-0009-004)
!     INCREASE THE FUNCTIONALITY AND EASE OF OPERATION:
!          PIP.SAV
!          DIR.SAV
!          DUP.SAV
!          EDIT.SAV
!-----------------------------------------------------------------------------
!** STEP #3 **
!BOOT THE TARGET DISKETTE, QX0:.
!
!-----------------------------------------------------------------------------
!
!** STEP #4 **
!ASSIGN THE LOGICAL DEVICES REQUIRED FOR COMPILER INSTALLATION TO THE
!     APPROPRIATE PHYSICAL DEVICES.
ASSIGN QX1: INP
ASSIGN QX0: OUP
!-----------------------------------------------------------------------------
```

```
!-----------------------------------------------------------------------
!
!** STEP #5 **
!EXECUTE THE COMPILER DEFAULT SETTING PROGRAM "FORGEN" USING
!     THE FOLLOWING DIALOG.
RUN INP:FORGEN
!ACCEPT THE "FORGEN" DEFAULT FOR MAX LINE PER LISTING PAGE
Y
!ACCEPT THE "FORGEN" DEFAULT FOR MAX RECORD LENGTH
Y
!ACCEPT THE "FORGEN" DEFAULT FOR MAX SIMULTANEOUS OPEN I/O CHANNELS
Y
!THIS COMPILER IS TO EXECUTE ON RT-11 V3B
Y
!DO NOT SELECT EAE FLOATING POINT EXPANSION AS THE AVAILABLE FIS IS FASTER
N
!DO NOT SELECT EIS FLOATING POINT EXPANSION AS THE AVAILABLE FIS IS FASTER
N
!DO SELECT FIS FLOATING POINT EXPANSION AS IT IS THE FASTEST FORM
Y
!MAKE THE COMPILER OPTIMIZE INLINE CODE FOR SIZE NOT SPEED
N
!FOR FLEXIBILITY INCLUDE BOTH INLINE AND THREADED CODE
N
!-----------------------------------------------------------------------
!
!** STEP #6 **
!EXECUTE THE INDIRECT COMMAND FILE TO LINK A COMPILER THAT SUPPORTS
!     BOTH INLINE AND THREADED CODE EXPANSION.
!     <NOTE: THE DEFAULT IS SET TO INLINE, BUT CAN BE OVER RIDDEN
!           WITH THE COMPILER OPTION /CODE:THR>
@INP:F4LINK
!-----------------------------------------------------------------------
!
!A VERSION OF "FORTRA.SAV" IS NOW RESIDENT ON QXØ: WITH THE
!     SPECIFIC CHARACTERISTICS SELECTED IN THE INSTALLATION COMMANDS.
!
!*************************************************************************
!*********************  NOTES DESCRIBING THE INSTALLATION  ***************
!*********************  PROCEDURE USED FOR "READY TO USE"  ***************
!*********************       FORTRAN IV V2.1 COMPILER      ***************
!*************************************************************************
```

APPENDIX B

"SYSLIB.DOC" LISTINGS

```
!*****************************************************************************
!*********************  NOTES DESCRIBING THE INSTALLATION  *******************
!*********************  PROCEDURE USED FOR "READY TO USE"  *******************
!*********************           SYSTEM LIBRARY            *******************
!*****************************************************************************
!
!SYSLIB.DOC
!CREATED 9-28-79          UPDATED 9-28-79
!
!THE PURPOSE OF THE FILE IS TO DOCUMENT THE EXACT RT-11 V3B COMMAND
!     SEQUENCE USED TO INSTALL THE SYSTEM LIBRARY CONTAINED
!     ON THE DISTRIBUTION DISKETTE, TERAK PART NUMBER 61-0009-005.
!-----------------------------------------------------------------------------
!
!** STEP #1 **
!PLACE THE TERAK FORTRAN IV GRAPHICS LIBRARY DISKETTE,
!     TERAK PART NUMBER 61-0009-003,
!     INTO QX1:
!-----------------------------------------------------------------------------
!
!** STEP #2 **
!PLACE THE TARGET DISKETTE FOR THE LIBRARY INTO QX0:
!     THIS DISKETTE MUST CONTAIN THE MINIMUM OPERATING SYSTEM FILES:
!          SWAP.SYS
!          QX.SYS
!          TT.SYS
!          QXMNSJ.SYS
!          LIBR.SAV
!          STARTS.COM
!          CHRSET.SYS
!
!THE FOLLOWING FILES (WHICH ARE INCLUDED ON THE DISTRIBUTED "READY
!     TO USE" SYSTEM LIBRARY DISKETTE, TERAK PART NUMBER 61-0009-005)
!     INCREASE THE FUNCTIONALITY AND EASE OF OPERATION:
!          PIP.SAV
!          DIR.SAV
!          DUP.SAV
!          LINK.SAV
!-----------------------------------------------------------------------------
!
!** STEP #3 **
!BOOT THE TARGET DISKETTE, QX0:.
!
!-----------------------------------------------------------------------------
!
!** STEP #4 **
!ASSIGN THE LOGICAL DEVICES REQUIRED FOR LIBRARY INSTALLATION TO THE
!     APPROPRIATE PHYSICAL DEVICES.
ASSIGN QX1: INP
ASSIGN QX0: OUP
!-----------------------------------------------------------------------------
```

```
!-----------------------------------------------------------------------
!
!** STEP #5 **
!BUILD THE SYSTEM LIBRARY ROOT
!     WHICH INCLUDES BOTH TERAK GRAPHICS LIBRARIES
!          (1) F4 LOW LEVEL GRAPHICS "LOWGRF.OBJ"
!          (2) F4 SIGGRAPH CORE-79 COMPATIBLE GRAPHICS "SIGGRF.OBJ"
R LIBR
OUP:SYSLIB[-1]=INP:LOWGRF,SIGGRF
!-----------------------------------------------------------------------
!
!** STEP #6 **
!REMOVE THE DISKETTE IN QX1: (TERAK PART NUMBER 61-0009-003)
!-----------------------------------------------------------------------
!
!** STEP #7 **
!PLACE THE "FORTRAN IV V2.1 LIBRARY OBJECT" DISKETTE,
!     TERAK PART NUMBER 61-0009-002,
!     INTO QX1:
!-----------------------------------------------------------------------
!
!** STEP #8 **
!INSTALL THE DEC SUPPLIED SYSTEM LIBRARY ROUTINES INTO
!     THE SYSTEM LIBRARY GENERATED
!          (1)EXTENDED FORTRAN OPERATIONAL LIBRARY, "SYSF4.OBJ"
!          (2)FORTRAN IV V2.1 OTS LIBRARY, "FORLIB.OBJ"
!THE FORTRAN IV V2.1 LIBRARY AS INSTALLED HERE
!     (1)MAKES USE OF THE AVAILABLE FIS HARDWARE
!     (2)DOES NOT ALLOW FOR VIRTUAL ARRAYS AS MEMORY MANAGEMENT IS NOT
!        AVAILABLE
!     (3)INCLUDES ARRAY BOUND CHECKING FOR THREADED CODE EXPANSION
!     (4)DOES NOT INCLUDE STAND ALONE FORTRAN
OUP:SYSLIB[-1]=OUP:SYSLIB,INP:SYSF4,FIS,OTSCOM,NOVIR,V2S/G
!FIRST GLOBAL REQUEST
$ERRS
!SECOND GLOBAL REQUEST
$ERRTB
!THIRD GLOBAL RERUEST
$VRINT
!FOURTH GLOBAL REQUEST
!(THERE ARE NONE TO ENTER SO JUST RETURN)
<RETURN>
!-----------------------------------------------------------------------
!
!** STEP #9 **
!REMOVE DISKS FROM BOTH QX0: AND QX1:
!-----------------------------------------------------------------------
!
!** STEP #10 **
!PLACE THE SYSTEM LIBRARY DISK
!     (THE ONE JUST CREATED)
!     INTO QX1:
!-----------------------------------------------------------------------
```

```
!-----------------------------------------------------------------------
!
!** STEP #11 **
!PLACE THE "READY TO USE" FORTRAN IV COMPILER DISKETTE,
!     TERAK PART NUMBER 61-0009-004,
!     INTO QX0:
!
!-----------------------------------------------------------------------
!
!** STEP #12 **
!
!BOOT THE COMPILER DISKETTE, QX0:.
!
!-----------------------------------------------------------------------
!
!** STEP #13 **
!SQUEEZE THE SYSTEM LIBRARY DISKETTE
!     (THE ONE JUST CREATED) IN QX1:
SQUEEZE QX1:
!QX1: /ARE YOU SURE?
Y
!
!-----------------------------------------------------------------------
!
!** STEP #14 **
!COPY "PIP" ONTO THE SYSTEM LIBRARY DISKETTE
!     (THE ONE JUST CREATED) IN QX1:
COPY
!FROM
PIP.SAV
!TO
QX1:PIP.SAV
!
!-----------------------------------------------------------------------
!
!** STEP #15 **
!COPY "DIR" ONTO THE SYSTEM LIBRARY DISKETTE
!     (THE ONE JUST CREATED) IN QX1:
COPY
!FROM
DIR.SAV
!TO
QX1:DIR.SAV
!
!-----------------------------------------------------------------------
!
!
!A VERSION OF THE RICH "SYSLIB.OBJ" IS NOW RESIDENT ON QX1: WITH THE
!     SPECIFIC CHARACTERISTICS SELECTED IN THE INSTALLATION COMMANDS.
!
!***********************************************************************
!**********************  NOTES DESCRIBING THE INSTALLATION  ****************
!**********************  PROCEDURE USED FOR "READY TO USE"  ****************
!**********************            SYSTEM LIBRARY           ****************
!***********************************************************************
```

terak
CORPORATION

F4 SIGGRAPH CORE-79
GRAPHICS RELEASE GUIDE

REV 1

TERAK P/N 60-0049-001

14405 NORTH SCOTTSDALE ROAD • SCOTTSDALE, ARIZONA 85254 • (602) 991-1580

****
********** F4 SIGGRAPH CORE-79 **********
********** GRAPHICS RELEASE GUIDE **********Rev 1*
********** Pub no. 60-0049-001*

INTRODUCTION

This package provides RT-11 FORTRAN IV with graphics support for the TERAK 8510/a Graphics Computer System. The routines described herein provide the FORTRAN programmer with a powerful and easy- to-use 'tool kit' for producing visual displays. Graphics are programmed using the FORTRAN 'CALL' statement. These routines are a derivative of the SIGGRAPH CORE-79 proposed standard. Additional information concerning the TERAK implementation and the proposed standard may be found in Appendix A.

TERAK 8510/A GRAPHICS COMPUTER SYSTEM HARDWARE OVERVIEW

The graphics display is presented on the 8532 Video Monitor as a 320 dot wide by 240 dot high matrix. The graphics display, when active, illuminates or blanks each dot according to the contents of a memory buffer. Sixteen hundred (16 bit) words are required for each third of the screen; forty eight hundred for an entire 320 by 240 dot display. This can be reduced, with simultaneous reduction of the graphics area of the monitor displayed, by the zone blanking feature. Each third of the graphics display may be blanked or displayed.

The character display is presented on the Video Monitor as a matrix 80 characters wide by 24 characters high. The display presented to the user is a video overlay of the graphics and character displays. Except for the conventions established by this and other system software, the two displays are independent. Like the graphics display, the character display can be selectively blanked.

For blanking control, the display matrix is divided into three horizontal zones. Each graphics zone is 80 dots high by 320 dots wide; each character zone is 8 characters high by 80 characters wide. Blanking of any one character display zone is mutually independent of the blanking of any one graphics display zone.

GRAPHICS OVERVIEW

The graphics routines provide conventions for the description of a graphics display by the user program. Clipping and Window to Viewport translations are performed to allow the user program the simplest means of expressing the desired result. Although effectively instantaneous, the processing of graphics can be viewed as separate operations upon 'virtual' pictures. Figure 1 illustrates an analogy to the process, and should be referred to as the graphics extensions are presented.

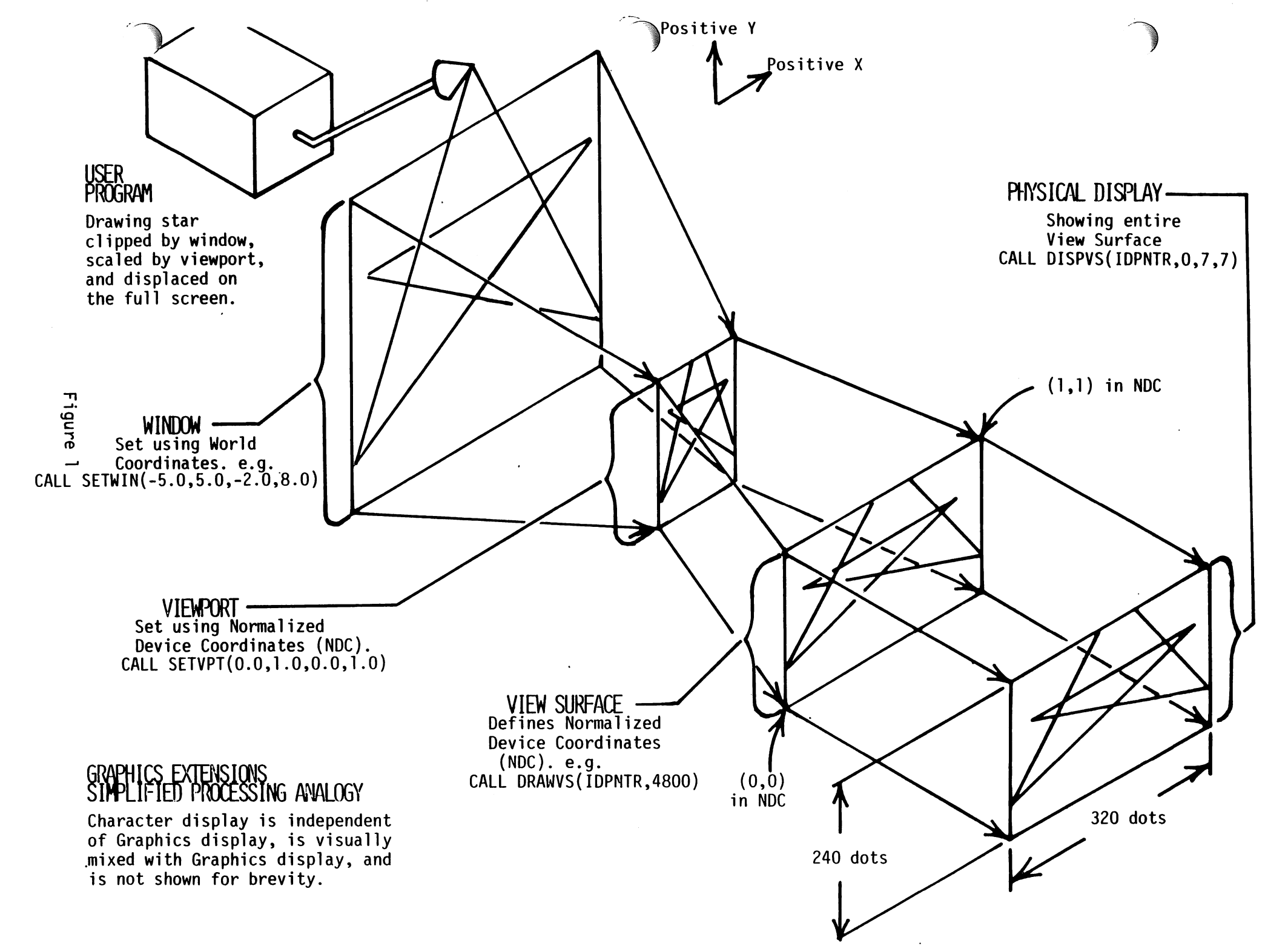

Figure 1

GRAPHICS EXTENSIONS
SIMPLIFIED PROCESSING ANALOGY

Character display is independent of Graphics display, is visually mixed with Graphics display, and is not shown for brevity.

WORLD COORDINATES

Graphics are employed when the user program wishes to represent relationships of data pictorially. For example, one such relationship would be SALES vs TIME for a particular product history. The graph of such a relationship would typically consist of two axes and special markers connected by lines. Such a graph is presented in Figure 2.

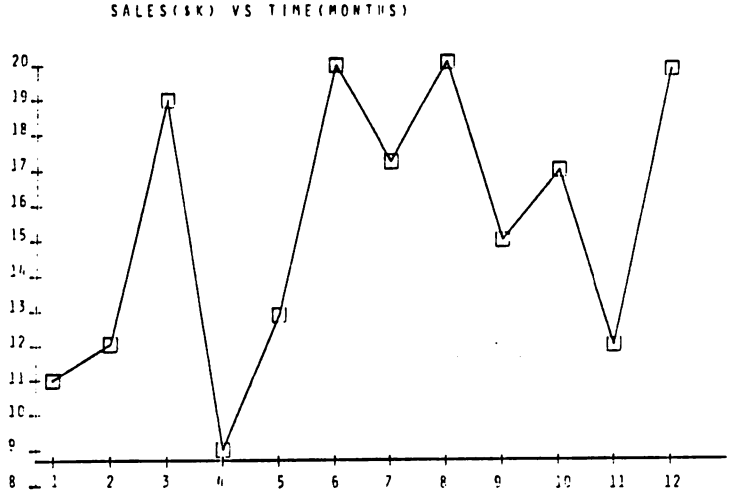

FIGURE 2

To produce such a graph , the user program must describe the various display elements in terms of positions and measurements in a Cartesian coordinate system. The coordinates used by the program may represent any unit of measure. In Figure 2, SALES is measured in the vertical direction, in dollars, and TIME is measured in the horizontal direction, in months. The parameters passed to the graphics routines must be floating point constants or variables, and may range throughout the world of coordinate values, including values not representable within the defined window. These coordinates are called WORLD COORDINATES.

CURRENT POSITION

Complete displays on the TERAK 8510/a are generated by output primitives which generate lines, markers and text on the display. Intrinsic to this graphics support is the concept of the current position of a cursor at a position in the World Coordinate space. An analogy to the current position would be the position of an artist's pencil after each pencil stroke. Each output primitive affects the value of the current position in a way defined herein. The current position may be obtained by the routine INQCPN. For example the call:

```
CALL INQCPN ( x , y )
```

within a user program returns the horizontal and vertical current position of the graphic cursor into the variables x and y respectively.

LINES, MARKERS, AND POLYLINES

A user program produces a display by invoking output primitives to specify a series of operations using positions in the World Coordinate space. For example, to draw a line from the current position to the point with World Coordinates (10.0, 20.5), the user program would issue the call:

```
CALL LINABS ( 10.0 , 20.5 )
```

Following the execution of the LINABS statement the current position then becomes the end point of the line just drawn. The name of the routine LINABS means to draw a line to the 'absolute' position specified. Lines may also be drawn 'relative' to the current position via the LINREL routine.

The current position may be changed without drawing lines by issuing calls to the MOVABS and MOVREL routines. For example the call:

```
CALL MOVREL ( 1.4 , -0.5 )
```

will move the current position 1.4 units in the positive X direction, and 0.5 units in the negative Y direction.

Routines are provided to draw predefined markers centered at a point. The size of any marker will be a maximum of 7 dots high by 7 dots wide. Markers are useful to distinguish certain points in the display. The marker routine moves the current postion to the point where the marker is to be centered and draws the current marker symbol there. For example:

```
CALL SETMKS ( 8 )
CALL MRKABS ( 100.0 , 200.0 )
```

will move the current position to the World Coordinate point (100.0 , 200.0) and then draw a 7 by 7 diamond shaped figure at that point. A table of all marker patterns available is listed on page 12 of this document.

Arrays of points may be defined and then passed to a polyline routine which will then draw lines to connect the points. If the arrays x and y were dimensioned at 20 elements, the following call would connect all the points defined by (x(n) , y(n)) for n=1 to 20.

```
CALL PLYLNA ( x(1) , y(1),20 )
```

TEXT

Character text may be drawn into the display by using the TEXT routine. Two variations of text may be displayed: either low precision or medium precision. Low precision text is placed into the Character Page buffer overlaying the graphics display. Medium precision text is placed into the graphics display. The routine SETCPR is used to set the character precision for all subsequent calls to TEXT. When the precision of the text is medium, the characters drawn onto the graphics space also have the attributes of 'size' and 'spacing'. These attributes may be set from the user program using the SETCSZ and SETCSP routines, respectively. For example the calls:

```
CALL SETCSZ ( 1 , 2 )
CALL SETCSP ( 0.1 , 0.15 )
CALL SETCPR ( 2 )
CALL TEXT ( 'string of text' )
```

will output the characters 'STRING OF TEXT' starting at the current position. Characters are patterned after the 8 by 10 dot template of each character in the writeable character generator of the 8510/a. The characters in the example will be 20 screen dots (2*10) high by 8 screen dots (1*8) wide. After each character is drawn, the current position will change by 0.1 (in World Coordinates) in the X direction and by 0.15 in the Y direction. The TEXT routine outputs the string of text until it encounters a zero byte (ASCII NUL). If the argument passed to TEXT is not a quoted string, it is the user's responsibility to put at least one ASCII NUL at the end of the text.

WINDOWS

These routines allow the user program to locate and specify objects in the World Coordinate system, permiting the program to work in a coordinate system which is natural to the application. To produce a display, the graphics routines must convert the World Coordinate specifications to the absolute physical screen coordinates of the TERAK 8510/a display. To perform this conversion, the user program must specify which portion of the World Coordinate space is to be displayed on the View Surface. This region is called a WINDOW. It must be rectangular and is defined by a call to the SETWIN routine, whose arguments specify the minimums and maximums along each axis. The generic form of the SETWIN call is:

```
CALL SETWIN ( xmin , xmax , ymin , ymax )
```

When output primitves are executed to define objects either partially or entirely outside of the edges of the Window, clipping automatically takes place at the Window edge. The current position, however, may be outside the Window. If the current position does lie outside the Window and a line is drawn which exists partially within the Window, the line will be drawn with a slope natural to the connecting points with only that portion of the line inside the Window being visible.

VIEWPORTS

A Viewport is a rectangular portion of the View Surface onto which the Window is mapped. Most often the Viewport will occupy the entire View Surface. However, this is not always desired. For example, consider dividing the View Surface into four quadrants and mapping a different display onto each quadrant. To do this, it would be necessary to locate Viewports at different locations within the View Surface. These locations are best expressed with respect to the View Surface, rather than the World Coordinates. Normalizing these coordinates frees the user program from device dependent use of the physical display coordinates. This Viewport coordinate system is called Normalized Device Coordinates (NDC). The coordinates range from 0 to 1 in both the X and Y directions, with (0,0) at the lower left of the View Surface, and (1,1) at the upper right of the View Surface, as shown in Figure 3.

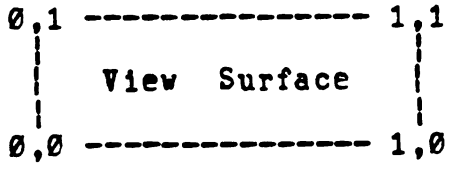

FIGURE 3

Normalized Device Coordinates locate and define the Viewport onto which the Window is mapped, as directed by the SETVPT routine. The generic form of the SETVPT call is:

```
CALL SETVPT ( xmin , xmax , ymin , ymax )
```

For example:

`CALL SETVPT ( 0.0 , 0.5 , 0.0 , 0.5 )`	MAP WINDOW ONTO THE LOWER LEFT QUADRANT OF THE VIEW SURFACE.
`CALL SETVPT ( 0.25 , 0.75 , 0.0 , 0.5 )`	MAP WINDOW ONTO LOWER HALF OF THE MIDDLE 2/4 OF THE VIEW SURFACE.
`CALL SETVPT ( 0.0 , 1.0 , 0.0 , 1.0 )`	MAP THE WINDOW ONTO THE ENTIRE VIEW SURFACE. THIS IS THE DEFAULT VIEWPORT.

Once a VIEWPORT and WINDOW have been defined, calls can then be made to the graphics routines to create the display.

ALLOCATING AND DISPLAYING THE GRAPHICS SPACE

The memory used for creation of a graphics display must be acquired by the user program via the system function IGETSP. For example:

```
I = IGETSP ( 4800 , 4800 , IDPNTR )
```

will request 4800 16-bit words of dynamic memory. Upon return, the variable I will contain the actual number of words acquired and the variable IDPNTR will contain the starting address of the acquired memory. Note that IDPNTR is only an indirect pointer to the acquired memory and is not useable within the FORTRAN program as an array name. Therefore, the user program cannot manipulate this memory like an array dimensioned in the program itself.

A graphics display can be created without necessarily viewing it. For example, a user program may create a display for later viewing. The DISPLAY_VIEW_SURFACE (DISPVS) routine directs the hardware to display an array as graphics on the video monitor and also controls the zone blanking of the display. The independence of the View Surface defined by DRAWVS, and the View Surface defined by DISPVS, provides the capability of drawing on one View Surface, while displaying another.

The following program fragment would be used to allocate memory for a full screen of graphics and display all three zones of graphics and characters:

```
C ALLOCATE AND DISPLAY A FULL SCREEN OF GRAPHICS
        I = IGETSP ( 4800 , 4800 , IDPNTR )
        IF ( I .NE. 4800 ) STOP 'INSUFFICIENT MEMORY'
        CALL GRFINI
        CALL DRAWVS ( IDPNTR , 4800 )
        CALL DISPVS ( IDPNTR , 0 , 7 , 7 )
        .
        .
        .
```

The above program fragment defines the view surface and view surface-to-display mapping most commonly desired. Users desiring to control the view surface size and mapping onto the physical surface should review the detailed descriptions of View Surfaces, DRAWVS and DISPVS later in this document.

GRAPHICS ROUTINES SUMMARY

The full complement of routines are summarized below. Note that underscores (_) are used to concatenate phrases from which mnemonics were extracted.

CALL	ARGUMENT LIST	EFFECT
GRFINI		GRAPHICS_INITIALIZATION intializes the graphics control environment (see the section on intial values). GRFINI must be called before making calls to the other graphics routines. It may be called again if a return to the initial default values is desired.
DRAWVS	(pointer , size)	DRAW_VIEW_SURFACE defines which space to use as graphics memory . Pointer is the indirect pointer variable which was specified on a previous reference to the IGETSP function. Size must be a multiple of 1600. Both parameters are type INTEGER.
DISPVS	(pointer , screen-displacement , graf-zones , char-zones)	DISPLAY_VIEW_SURFACE defines which space is to be mapped into the screen display. Pointer is as defined under DRAWVS. Screen-displacement defines how mapping is to occur, and graf-zones and character-zones define graphics and character blanking. All arguments are type INTEGER. (see page 14 for further detail)
SETVPT	(left , right , bottom , top)	SET_VIEWPORT defines which portion of the View Surface (defined by DRAWVS) is to receive the Window to Viewport mapping. All arguments are in Normalized Device Coordinates, i.e., 0 <= argument <= 1 for all arguments. Also, 'left < right' and 'bottom < top' must be true. All arguments are type REAL.
SETWIN	(xmin , xmax , ymin , ymax)	SET_WINDOW defines the X,Y World Coordinate space mapped onto the Viewport. Lines, markers, and text are clipped per the arguements in SETWIN. All arguments are type REAL.

CALL	ARGUMENT LIST	EFFECT
SETLST	(linestyle)	SET_LINESTYLE defines how lines, text, and markers are to be drawn, where: Linestyle = 1 draw white 2 same as 1 3 draw black 4 draw complement- (valid only for lines) Linestyle is type INTEGER.
NEWFRM		NEWFRAME resets the View Surface defined by DRAWVS according to the current linestyle. If linestyle = 1 or 2 then the background will become BLACK. If linestyle = 3 or 4 then the background will become WHITE.
INQCPN	(xpos , ypos)	INQUIRE_CURRENT_POSITION returns the current x and y position in World Coordinates into the variables xpos and ypos. Both arguments are type REAL.
MOVABS	(x , y)	MOVE_ABSOLUTE moves the current position to absolute World Coordinates x,y. The View Surface is not affected. Both arguments are type REAL.
MOVREL	(dx , dy)	MOVE_RELATIVE moves the current position to relative World Coordinates dx,dy. That is, the new currrent position is computed from the old current position plus the displacements dx and dy. Both arguments are type REAL.
LINABS	(x , y)	LINE_ABSOLUTE draws a line, per the current linestyle, from the current position to absolute World Coordinates x,y. The current position becomes x,y. Both arguments are type REAL.
LINREL	(dx , dy)	LINE_RELATIVE draws a line, per the current linestyle, from the current position to relative World Coordinates dx,dy. That is, the new currrent position is computed from the old current position plus the displacements dx and dy. Both arguments are type REAL.

CALL	ARGUMENT LIST	EFFECT
PLYLNA	(xarray(sub) , yarray(sub) , n)	POLYLINE_ABSOLUTE draws n lines connecting the coordinate points xarray(i) , yarray(i) for i indexed from sub to n. Dimensions of xarray and yarray must be at least sub + n. Xarray and yarray must be single dimension arrays. Xarray and yarray are type REAL and n is type INTEGER.
PLYLNR	(dxarry(sub) , dyarry(sub) , n)	POLYLINE_RELATIVE draws n lines connecting the coordinate points dxarry(i) , dyarry(i) for i indexed from sub to n. Dimensions of dxarry and dyarry must be at least sub + n. Dxarry and dyarry must be single dimension. Note that each new current position is computed relative to the last current position. Dxarry and dyarry are type REAL and n is type INTEGER.
SETMKS	(n)	SET_MARKER_SYMBOL sets the current marker symbol to pattern n. This pattern will be used on subsequent calls to the marker routines. N is type INTEGER.
MRKABS	(x , y)	MARKER_ABSOLUTE moves the current position to World Coordinates x,y and draws the current marker symbol centered there. Both arguments are type REAL.
MRKREL	(dx , dy)	MARKER_RELATIVE moves the current position to relative World Coordinates dx,dy, and draws the current marker symbol centered there. That is, the new currrent position is computed from the old current position plus the displacements dx and dy. Both arguments are type REAL.

CALL	ARGUMENT LIST	EFFECT
PLYMKA	(xarray(sub) , yarray(sub) , n)	POLYMARKER_ABSOLUTE draws n current marker symbols. The coordinates xarray(i) , yarray(i) for i indexed from sub to n are used as the absolute World Coordinates for each marker. Xarray and yarray must be single dimension arrays and their size must be at least sub+n. Xarray and yarray are type REAL and n is type INTEGER.
PLYMKR	(dxarry(sub) , dyarry(sub) , n)	POLYMARKER_RELATIVE draws n current marker symbols. The coordinates dxarry(i), dyarry(i) for i indexed from sub to n are used as the relative World Coordinates for each marker, i.e., the current position is computed from the old position plus the next pair of displacements in the arrays. Dxarry and dyarry must be single dimension arrays and their size must be at least sub+n. Dxarry and dyarry are type REAL and n is type INTEGER.
SETCPR	(charprecision)	SET_CHAR_PRECISION specifies the type of text output by the TEXT routine. Charprecision = 1 will cause TEXT to produce low precision text output into the character display, starting at a row and column closest to the current position. Charprecision = 2 or 3 will cause TEXT to produce medium precision text output into the graphics display, starting with the lower left corner of the first character at the current position. Charprecision =2 will cause TEXT to generate characters at a slow rate, for visual impact. Charprecision is type INTEGER.
SETCSZ	(width , height)	SET_CHAR_SIZE specifies the size of characters output by the TEXT routine when charprecision = 2 or 3. The character height is height*10 , and the character width is width*8. Width and height are in physical screen dots. Width and height are type INTEGER.

CALL	ARGUMENT LIST	EFFECT
SETCSP	(dx , dy)	SET_CHAR_SPACING defines the movement of the current position after each character is drawn by TEXT when charprecision is 2 or 3 (graphics text). The spacing is in relative World Coordinates, That is, after each character, the new currrent position is computed from the old current position plus the displacements dx and dy. Both arguments are type REAL.
TEXT	(string expression)	TEXT outputs the string defined by string expression per the current values of charprecision, charspace, and charsize. Characters are output until a zero byte (ASCII NUL) is encountered.

INITIAL VALUES

When the routine GRFINI is called , the default values of the control attributes are set as if the following calls had been made:

```
CALL SETMKS ( 1 )
CALL SETVPT ( 0.0 , 1.0 , 0.0 , 1.0 )
CALL SETWIN ( 0.0 , 1.0 , 0.0 , 1.0 )
CALL MOVABS ( 0.0 , 0.0 )
CALL SETLST ( 1 )
CALL SETCPR ( 1 )
CALL SETCSP ( 0.0 , 0.0 )
CALL SETCSZ ( 1 , 1 )
```

MARKERS AVAILABLE

For the SET_MARKER_SYMBOL (SETMKS) routine, the following eleven markers are available:

N	MARKER PATTERN
1	single dot
2	plus sign (+) 7 by 7 dots
3	asterisk (*) 7 by 7 dots
4	circle 7 dots in diameter
5	cross (X) 7 by 7 dots
6	vertical bar 5 dots high
7	horizontal bar 5 dots wide
8	diamond 7 by 7 dots
9	square 7 by 7 dots
10	square 5 by 5 dots
11	block (filled square) 7 by 7 dots

VIEW SURFACES

A View Surface is that portion of the graphics array which is manipulated (drawn on and displayed) by calls to graphics routines. Typically, the 'portion' is the entire array. A View Surface is always the same width as the physical display, but the height of a View Surface may be less than, equal to, or greater than the physical display height.

The DRAW_VIEW_SURFACE (DRAWVS) routine defines the View Surface which is to be drawn on by the graphics routines. The 'size' parameter of DRAWVS defines its View Surface height, which may be one or more zones. (A zone corresponds to 1600 16-bit words or 1/3 of the physical screen.)

The DISPLAY_VIEW_SURFACE (DISPVS) routine directs the Hardware to the View Surface which is to be displayed on the physical display. The 'displacement' parameter defines the View Surface to physical display zone mapping of the View Surface on the display. The zone blanking parameters define the zones in the character and graphics display which are visible.

The View Surface defined by DRAWVS and the View Surface displayed by DISPVS are completely independent. Normally, they are coincident.

DRAWVS

For reference, the generic DRAWVS call is:

```
CALL DRAWVS ( pointer , size )
```

Memory space for graphics is allocated in increments of 1600 16- bit words, corresponding to the size of one graphics blanking zone (1/3 of graphics display). This allows the user program to define a View Surface which is smaller, equal to, or larger than the physical display. Normalized Device Coordinates of the View Surface always remain from 0 to 1 in both the X and Y directions, although the physical mapping of the device coordinates may vary. Thus, the Normalized Device Coordinate range 0 to 1 in the Y direction may correspond to one, two, three, etc zones of physical display, as defined by the 'size' parameter. The Normalized Device Coordinate range 0 to 1 in the X direction always corresponds to the full width of the physical display. The size parameter in DRAWVS must be a multiple of 1600.

DISPVS

For reference, the generic DISPVS call is:

CALL DISPVS (pointer , displacement , graph-zones , char-zones)

A View Surface may be displayed on the physical display of the Video Monitor at a displacement corresponding to any of the physical display zones. The 'displacement' parameter in DISPVS specifies the number of zones of offset required by the user program. The normal value is zero, which will display the View Surface starting at pointer in the upper zone of the physical display. A displacement of -1 will start the display in the upper zone at pointer+1600, -2 at pointer+3200, etc. Decreasing the displacement has the visual effect of moving the display upward on the physical display. The displacement may also be positive. Increasing the displacement has the visual effect of moving the display downward on the screen.

Blanking of the graphics and character displays is controlled by the DISPVS parameters 'graf-zones' and 'char-zones'. These zone parameters are computed by assigning the value 4, 2, and 1 respectively, to the upper, middle, and lower physical display zones (either graphics or character), and then adding up the values of the zones to be displayed. The following table summarizes the eight combinations:

PARAMETER VALUE	EFFECT ON GRAPHICS OR CHARACTER BLANKING
0	BLANK ALL THREE ZONES
1	DISPLAY LOWER, BLANK MIDDLE AND UPPER ZONES
2	DISPLAY MIDDLE, BLANK LOWER AND UPPER ZONES
3	DISPLAY LOWER AND MIDDLE, BLANK UPPER ZONE
4	DISPLAY UPPER, BLANK LOWER AND MIDDLE ZONES
5	DISPLAY UPPER AND LOWER, BLANK MIDDLE ZONE
6	DISPLAY UPPER AND MIDDLE, BLANK LOWER ZONE
7	DISPLAY ALL ZONES

Note that the DISPVS displacement parameter may cause the display of memory outside of the View Surface. Any such superfluous memory in the physical display may be blanked by setting the 'graphics-zone' blanking parameter to correspond to the 'displacement' parameter. A generalized analogy to the TERAK graphics display process is presented in Figure 4.

EXAMPLE PROGRAM LISTINGS

The following two programs are on the Graphics Demos Diskette. The file names are SGDEM1.FOR and SGDEM2.FOR .

```
    PROGRAM  SGDEM1
C
C THIS PROGRAM DRAWS A PAIR OF AXES AND PLOTS A SINE WAVE
C
C
C ALLOCATE MEMORY FOR DISPLAY SURFACE AND INITIALIZE
C
      I = IGETSP ( 4800 , 4800 , IDPNTR )
C
C GRFINI MUST BE THE FIRST GRAPHICS ROUTINE CALLED
C
      CALL GRFINI
      CALL DRAWVS ( IDPNTR , 4800 )
      CALL NEWFRM
C
C TURN ON ALL GRAPHICS AND CHARACTER DISPLAYS
C
      CALL DISPVS ( IDPNTR , 0 , 7 , 7 )
C
      CALL SETWIN ( -10.0 , 10.0 , -1.2 , 1.2 )
      CALL SETLST ( 1 )
      CALL MOVABS ( 0.0 , -1.5 )
      CALL LINABS ( 0.0 , 1.5 )
      CALL MOVABS ( -10.0 , 0.0 )
      CALL LINABS ( 10.0 , 0.0 )
      Y = SIN ( -10.0 )
      CALL MOVABS ( -10.0 , Y )
      X = -10.0
10    CONTINUE
      IF ( .NOT. (X .LE. 10.0) ) GO TO 20
         Y = SIN ( X )
         CALL LINABS ( X , Y )
         X = X + 0.2
         GO TO 10
20    CONTINUE
      PAUSE 'TYPE RETURN TO EXIT'
      CALL DISPVS ( IDPNTR , 0 , 0 , 7 )
      STOP
      END
```

```
!*****************************************************************************
!*********************    NOTES DESCRIBING THE CREATION    *******************
!*********************  PROCEDURE USED FOR "READY TO RUN"  *******************
!*********************       SIGGRAPH  GRAPHICS DEMO       *******************
!*********************            SGDEM1.FOR               *******************
!*****************************************************************************
!
!SGDEM1.DOC
!CREATED 10-17-79          UPDATED 10-17-79
!
!THE PURPOSE OF THE FILE IS TO DOCUMENT THE EXACT RT-11 V3B COMMAND
!     SEQUENCE USED TO CREATE THE SIGGRAPH  GRAPHICS DEMO CONTAINED
!     ON THE DISTRIBUTION DISKETTE, TERAK PART NUMBER 61-0009-006.
!
!THIS COMMAND SEQUENCE MAKES USE OF THE "READY TO USE" FORTRAN COMPILER ON
!     DISTRIBUTION DISKETTE, TERAK PART NUMBER 61-0009-004
!THIS COMMAND SEQUENCE MAKES USE OF THE "READY TO USE" SYSTEM LIBRARY ON
!     DISTRIBUTION DISKETTE, TERAK PART NUMBER 61-0009-005
!-----------------------------------------------------------------------------
!
!** STEP #1 **
!PLACE THE "FORTRAN IV GRAPHICS DEMO" DISKETTE,
!     TERAK PART NUMBER 61-0009-006,
!     INTO QX1:
!-----------------------------------------------------------------------------
!
!** STEP #2 **
!PLACE THE "READY TO USE" FORTRAN IV COMPILER DISKETTE,
!     TERAK PART NUMBER 61-0009-004,
!     INTO QX0:
!-----------------------------------------------------------------------------
!
!** STEP #3 **
!BOOT THE COMPILER DISKETTE, QX0:.
!-----------------------------------------------------------------------------
!
!** STEP #4 **
!COMPILE THE FORTRAN SOURCE FOR THE SIGGRAPH GRAPHICS DEMO
R FORTRA
QX1:SGDEM1,QX1:SGDEM1=QX1:SGDEM1
!-----------------------------------------------------------------------------
!
!** STEP #5 **
!REMOVE THE COMPILER DISKETTE IN QX0:
!-----------------------------------------------------------------------------
```

```
!-----------------------------------------------------------------------
!
!** STEP #6 **
!PLACE THE "READY TO USE" SYSTEM LIBRARY DISKETTE,
!     TERAK PART NUMBER 61-0009-005,
!     INTO QX0:
!-----------------------------------------------------------------------
!
!** STEP #7 **
!BOOT THE SYSTEM LIBRARY DISKETTE, QX0:
!-----------------------------------------------------------------------
!
!** STEP #8 **
!LINK THE COMPILED FORTRAN OBJECT WITH THE SYSTEM LIBRARY
R LINK
QX1:SGDEM1,QX1:SGDEM1=QX1:SGDEM1
!-----------------------------------------------------------------------
!
!THE COMPILATION AND LINKING OF THE SIGGRAPH GRAPHICS DEMO IS COMPLETE
!     THE FOLLOWING FILES SHOULD NOW EXIST ON QX1:
!          (1)"SGDEM1.LST" , COMPILER LISTING
!          (2)"SGDEM1.OBJ" , COMPILER OBJECT
!          (3)"SGDEM1.MAP" , LINKER MAP
!          (4)"SGDEM1.SAV" , RT-11 EXECUTABLE MODULE
!
!THE DEMONSTRATION CAN BE EXECUTED BY THE FOLLOWING RT-11 COMMAND SEQUENCE
SET USR SWAP
RUN QX1:SGDEM1
!
!*************************************************************************
!**********************     NOTES DESCRIBING THE CREATION    ***************
!**********************  PROCEDURE USED FOR "READY TO RUN"   ***************
!**********************       SIGGRAPH  GRAPHICS DEMO        ***************
!**********************            SGDEM1.FOR                ***************
!*************************************************************************
```

```
      PROGRAM SGDEM2
C
C THIS PROGRAM DRAWS A GRAPH SIMILAR TO FIGURE 2
C
      DIMENSION   SALES(12),MTHS(2),KSALES(2)
C
      DATA                  SALES   /11000.0 , 12053.0 ,
     *                              18987.0 , 9000.0 ,
     *                              12889.0 , 20000.0 ,
     *                              17234.0 , 20100.0 ,
     *                              15002.0 , 17000.0 ,
     *                              12000.0 , 19876.0/
C
      DATA             MTHS ( 2 )      /0/
      DATA                 KSALES(2)       /0/
C
C ACQUIRE MEMORY FOR GRAPHICS
C
      I = IGETSP ( 4800 , 4800 , IDPNTR )
      IF ( I .NE. 4800 ) STOP 'NO DYNAMIC MEMORY'
C
C INITIALIZE VIEW SURFACE AND TURN ON ALL GRAPHICS AND
C CHARACTER ZONES.  GRFINI MUST BE THE FIRST GRAPHICS ROUTINE TO BE
C CALLED.
C
      CALL GRFINI
      CALL DRAWVS ( IDPNTR , 4800 )
      CALL NEWFRM
      CALL DISPVS ( IDPNTR , 0 , 7 , 7 )
C
C VIEWPORT DEFAULTS TO ENTIRE SCREEN
C
      CALL SETWIN ( 0.0 , 13.0 , 8000.0 , 23000.0 )
C
C LINE STYLE IS WHITE ON BLACK
C
      CALL SETLST ( 1 )
C
C DRAW THE AXES
C
      CALL MOVABS ( 0.7 , 20000.0 )
      CALL LINABS ( 0.7 , 8000.0 )
      CALL MOVABS ( 0.7 , 8700.0 )
      CALL LINABS ( 13.0 , 8700.0 )
```

```
C
C SET CHARACTER QUALITY FOR GRAPHICS DISPLAY
C
      CALL SETCPR ( 2 )
C
C SET CHARACTER SPACING
C
      CALL SETCSP ( 0.3 , 0.0 )
C
C PLACE TICK MARKS FOR THE MONTHS
C
      CALL SETMKS ( 6 )
      DO 10 J = 1,12
         CALL MRKABS ( FLOAT(J) , 8700.0 )
         CALL MOVABS ( FLOAT(J) , 8000.0 )
         IF ( .NOT. (J .LT. 10) ) GO TO 2
            ENCODE ( 2 , 1 , MTHS(1) ) J
1           FORMAT ( I1 )
            GO TO 4
2        CONTINUE
            ENCODE ( 2 , 3 , MTHS(1) ) J
3           FORMAT ( I2 )
4        CONTINUE
         CALL TEXT ( MTHS )
10    CONTINUE
C
C PLACE TICK MARKS FOR SALES
C
      CALL SETMKS ( 7 )
      Y = 8000.0
20    CONTINUE
      IF ( .NOT. (Y .LE. 20000.0) ) GO TO 30
         CALL MRKABS ( 0.7 , Y )
         CALL MOVABS ( 0.0 , Y )
         K = INT( Y/1000.0 )
         ENCODE ( 6 , 22 , KSALES(1) ) K
22       FORMAT ( I2 )
         CALL TEXT ( KSALES )
         Y = Y + 1000.0
         GO TO 20
30       CONTINUE
C
C PLACE LEGEND
C
      CALL MOVABS ( 1.0 , 21000.0 )
      CALL TEXT ( 'SALES (K$) VS TIME (MONTHS)' )
C
C DRAW THE INFORMATION
C
      CALL SETMKS ( 9 )
      DO 40 J = 1,12
         IF ( J .EQ. 1 ) CALL MOVABS ( 1.0 , SALES(J) )
         CALL LINABS ( FLOAT(J) , SALES(J) )
         CALL MRKABS ( FLOAT(J) , SALES(J) )
40    CONTINUE
C
      PAUSE 'TYPE RETURN TO EXIT'
      CALL DISPVS ( IDPNTR , 0 , 0 , 7 )
      END
```

```
!*****************************************************************************
!*********************    NOTES DESCRIBING THE CREATION    *******************
!*********************  PROCEDURE USED FOR "READY TO RUN"  *******************
!*********************       SIGGRAPH  GRAPHICS DEMO       *******************
!*********************                SGDEM2.FOR           *******************
!*****************************************************************************
!
!SGDEM2.DOC
!CREATED 10-17-79          UPDATED 10-17-79
!
!THE PURPOSE OF THE FILE IS TO DOCUMENT THE EXACT RT-11 V3B COMMAND
!     SEQUENCE USED TO CREATE THE SIGGRAPH  GRAPHICS DEMO CONTAINED
!     ON THE DISTRIBUTION DISKETTE, TERAK PART NUMBER 61-0009-006.
!
!THIS COMMAND SEQUENCE MAKES USE OF THE "READY TO USE" FORTRAN COMPILER ON
!     DISTRIBUTION DISKETTE, TERAK PART NUMBER 61-0009-004
!THIS COMMAND SEQUENCE MAKES USE OF THE "READY TO USE" SYSTEM LIBRARY ON
!     DISTRIBUTION DISKETTE, TERAK PART NUMBER 61-0009-005
!-----------------------------------------------------------------------------
!
!** STEP #1 **
!PLACE THE "FORTRAN IV GRAPHICS DEMO" DISKETTE,
!     TERAK PART NUMBER 61-0009-006,
!     INTO QX1:
!-----------------------------------------------------------------------------
!
!** STEP #2 **
!PLACE THE "READY TO USE" FORTRAN IV COMPILER DISKETTE,
!     TERAK PART NUMBER 61-0009-004,
!     INTO QX0:
!-----------------------------------------------------------------------------
!
!** STEP #3 **
!BOOT THE COMPILER DISKETTE, QX0:.
!-----------------------------------------------------------------------------
!
!** STEP #4 **
!COMPILE THE FORTRAN SOURCE FOR THE SIGGRAPH GRAPHICS DEMO
R FORTRA
QX1:SGDEM2,QX1:SGDEM2=QX1:SGDEM2
!-----------------------------------------------------------------------------
!
!** STEP #5 **
!REMOVE THE COMPILER DISKETTE IN QX0:
!-----------------------------------------------------------------------------
!
!** STEP #6 **
!PLACE THE "READY TO USE" SYSTEM LIBRARY DISKETTE,
!     TERAK PART NUMBER 61-0009-005,
!     INTO QX0:
!-----------------------------------------------------------------------------
```

```
!-------------------------------------------------------------------------
!
!** STEP #7 **
!BOOT THE SYSTEM LIBRARY DISKETTE, QX0:
!-------------------------------------------------------------------------
!
!** STEP #8 **
!LINK THE COMPILED FORTRAN OBJECT WITH THE SYSTEM LIBRARY
R LINK
QX1:SGDEM2,QX1:SGDEM2=QX1:SGDEM2
!-------------------------------------------------------------------------
!
!THE COMPILATION AND LINKING OF THE SIGGRAPH GRAPHICS DEMO IS COMPLETE
!     THE FOLLOWING FILES SHOULD NOW EXIST ON QX1:
!          (1)"SGDEM2.LST" , COMPILER LISTING
!          (2)"SGDEM2.OBJ" , COMPILER OBJECT
!          (3)"SGDEM2.MAP" , LINKER MAP
!          (4)"SGDEM2.SAV" , RT-11 EXECUTABLE MODULE
!
!THE DEMONSTRATION CAN BE EXECUTED BY THE FOLLOWING RT-11 COMMAND SEQUENCE
SET USR SWAP
RUN QX1:SGDEM2
!
!*************************************************************************
!**********************    NOTES DESCRIBING THE CREATION    **************
!**********************  PROCEDURE USED FOR "READY TO RUN"  **************
!**********************       SIGGRAPH  GRAPHICS DEMO        **************
!**********************              SGDEM2.FOR              **************
!*************************************************************************
```

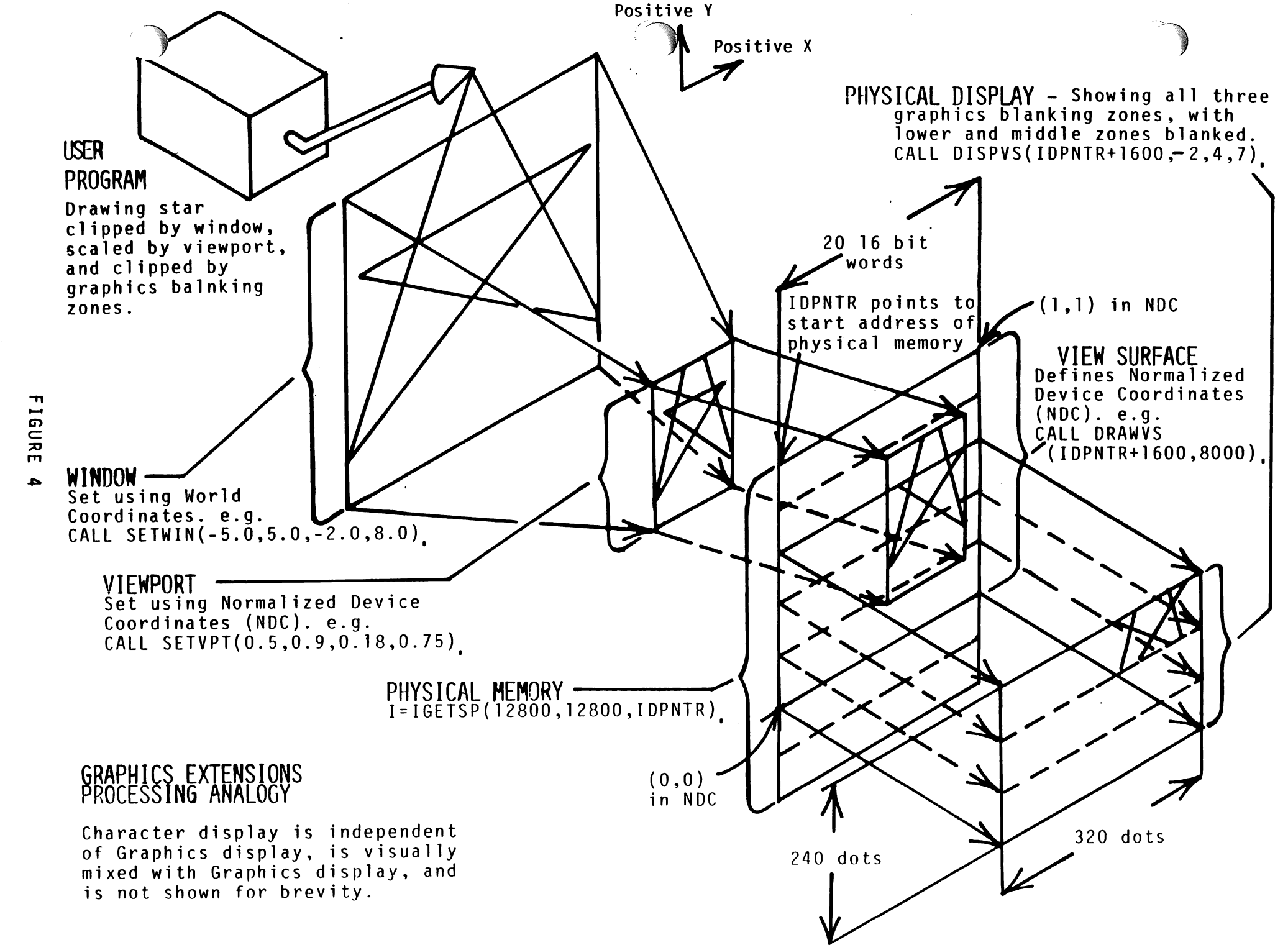

FIGURE 4

GRAPHICS EXTENSIONS PROCESSING ANALOGY

Character display is independent of Graphics display, is visually mixed with Graphics display, and is not shown for brevity.

APPENDIX A

The Graphics Extensions package provided by TERAK for RT-11 FORTRAN IV is a derivative of the basic output, no input, 2D proposed standard as developed by the ACM/SIGGRAPH Graphic Standards Planning Committee. Not all features of the SIGGRAPH CORE-79 standard have been implemented. However, those features which have been implemented comply with the functional and semantic specifications for that feature. Even though all features of the standard have not been implemented, the user is still provided with a powerful, functional, integrated graphics library. The complete proposed graphics standard is contained in

Computer Graphics
A Quarterly Report of SIGGRAPH-ACM
Vol 13, Number 3, August 1979
Status Report of the Graphics Standards
Planning Committee.

Copies may be purchased by contacting

ACM
P.O. Box 12105
Church Street Station
New York, New York 10249

The proposed standard has been specifically designed to avoid the limitations of computer language binding. Language binding considerations have been left to the implementor. The approach taken in the TERAK implementation has been to apply generally accepted FORTRAN IV programming techniques. All capabilities have been implemented using the FORTRAN subroutine call with conventional subroutine argument sequences. The variable names have been compressed to meet FORTRAN limitations, while still maintaining a high degree of readability.

The TERAK implementation of the CORE-79 standard is compatible with output level 1: Basic Output, input level 1: no Input and dimension level 1: 2D as described below. It should be noted that the only error checking that exists is that which is provided by the FORTRAN IV runtime system supported by DEC RT-11 version 3B.

OUTPUT PRIMITIVES

All of the output primitives, as summarized in the CORE-79 standard Appendix A, are supported. They provide a functional implementation of the features defined for each routine as defined in section 2, of the standard. The only feature that is not included is the routine "INQUIRE_TEXT_EXTENT_2".

PICTURE SEGMENTATION AND NAMING

No support is provided for explicit temporary picture segmentation and naming as described in section 3 of the standard.

ATTRIBUTES

Only a subset of the attributes, as specified in section 4 of the standard are supported. They are:

SET_LINESTYLE
SET_CHARSIZE
SET_CHARSPACE
SET_CHARPRECISION
SET_MARKER_SYMBOL.

Some additional attribute values have been provided which are defined as implementation dependent and exceed the minimum standard requirements. The user is referred to the appropriate discussion in the GRAPHICS ROUTINES SUMMARY section of this document for additional details.

VIEWING OPERATIONS AND COORDINATE TRANSFORMATIONS

Two of the viewing and coordinate routines, as defined in section 5 of the standard, are supported. These are:

SET_WINDOW
SET_VIEWPORT

The SET_VIEWPORT routine functions as if the default viewport specification were the entire normalized device coordinate space, (as set by a nonsupported graphics routine SET_NDC_SPACE_2). All routines in the TERAK implementation function as if SET_WINDOW_CLIPPING is "ON".

CONTROL

Currently, none of the graphics control routines, as defined in section 7 of the standard, are supported by the TERAK implementation. The functional intent of that section has, however, been provided in three routines: GRFINI (graphics initialization), DRAWVS (draw view surface) and DISPVS (display view surface). These routines provide control of the graphics display space for the TERAK implementation. It should be noted that these routines are not a part of the CORE-79 standard. They do however, provide access to the full power of the TERAK 8510a graphics hardware. Support of these specialized hardware features are not supported by the standard.

FUTURE CONSIDERATIONS

Future releases of this graphics library will continue support of the CORE-79 standard and will move towards a greater degree of compliance and compatibility.