TEKTRONIX®

PLOT 10

4010A01

(4010A10, 4010A11, 4010A12)

TERMINAL CONTROL SYSTEM

SYSTEM MANUAL

TEKTRONIX®

PLOT 10

4010A01

(4010A10, 4010A11, 4010A12)

TERMINAL CONTROL SYSTEM

SYSTEM MANUAL

Tektronix, Inc.
P.O. Box 500
Beaverton, Oregon 97077

070-2242-00

First Printing FEB 1977

TERMINAL CONTROL SYSTEM REFERENCE MATERIAL

The 4010A01 PLOT 10 Terminal Control System (TCS) Manual supports the following PLOT 10 packages. Please place all orders through your Tektronix Sales Engineer.

4010A01 PLOT 10 Terminal Control System

4010A10 PLOT 10 Terminal Control System for IBM with TSO

4010A11 PLOT 10 Terminal Control System for CDC SCOPE/Intercom with Opt. 20

4010A12 PLOT 10 Terminal Control System for DEC PDP-11 with DOS

PREFACE

This manual is organized as a continuation of the 4010A01 PLOT 10 Terminal Control System User Manual. Routines described in the User Manual are not discussed in detail here. This format is based on the assumption that anyone using the System Manual has access to a User Manual.

This manual supports Release 3.0 or later of TCS. If you have been using Release 2.0 of TCS, see the Appendix of this manual for information on updating your programs to run with Release 3.0 or later, including Level 1.

TABLE OF CONTENTS

SECTION 1

INTRODUCTION

This manual describes all the 4010A01, 4010A10, 4010A11, and 4010A12 PLOT 10 Terminal Control System routines not covered in the User Manual. These are the internal subroutines, which in most cases need not be called by the user. Flow diagrams are provided for the package as a whole and individually for the more complicated routines.

This manual also describes the Terminal Status Area, a common block of variables named /TKTRNX/, which represents the current state of the Terminal.

For user who wish to modify the system, this manual provides discussions of parameter modifications, user-written transformation routines and deletion of unwanted features.

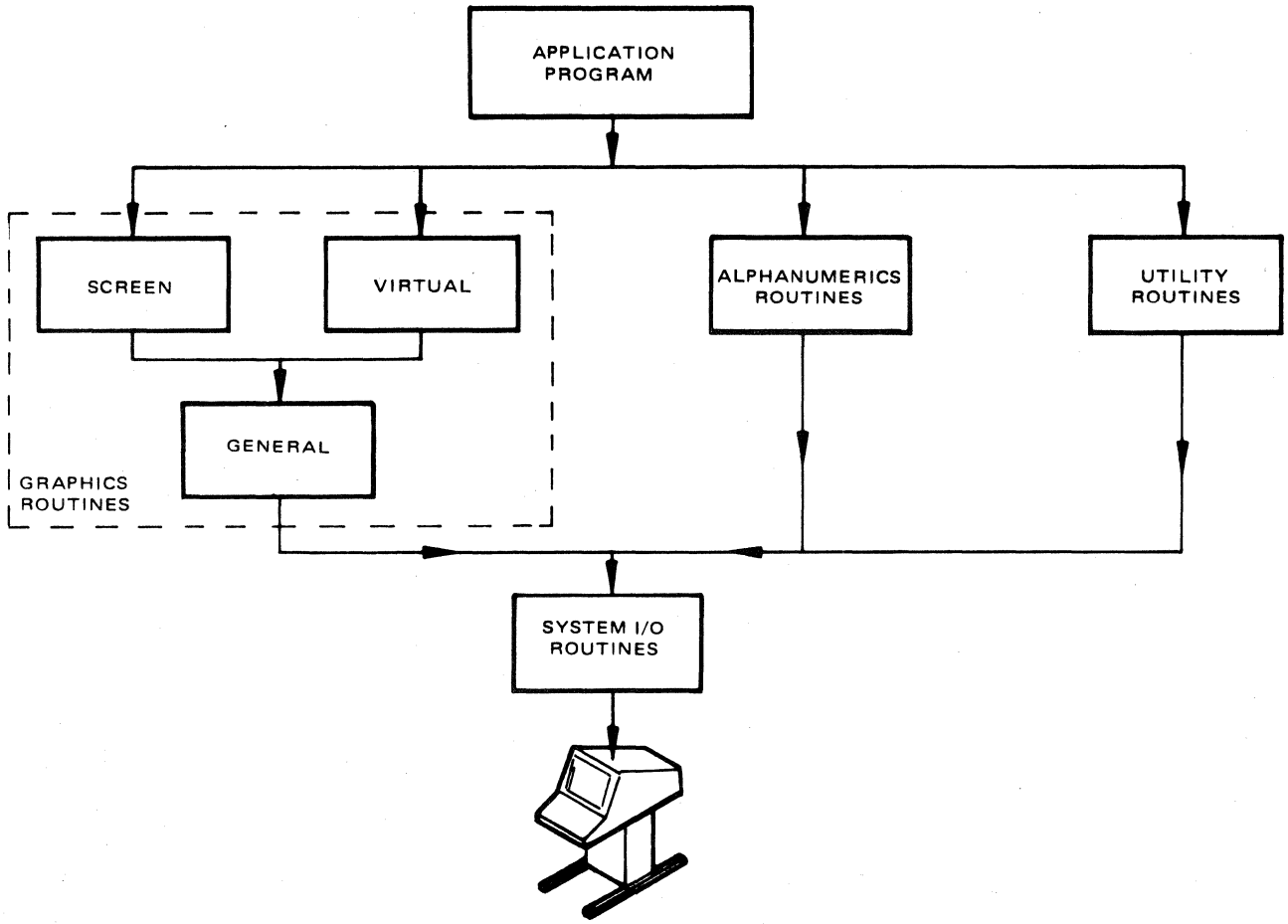
SECTION 2 TERMINAL CONTROL SYSTEM STRUCTURE

2.1. Flowchart of Subroutine Groups

The Terminal Control System contains 115 routines which can be divided into 4 functional groups:

1. Graphics Routines
   A. Screen Routines
   B. Virtual Routines
   C. General Routines
2. Alphanumerics Routines
3. Utility Routines
4. System Input/Output Routines

The following flowchart shows the functional relationships of the 4 groups. The relationships of all individual routines are shown in the TCS Flow Chart, Section A4 of this manual.



Subroutine Group Structure Flow Chart.

2.2 Subroutine Group Descriptions

This section provides a brief description of each of the routines in the 4 functional groups, including those described in detail in the TCS User Manual.

2.2.1 Graphics Routines

The graphics routines can be subdivided into 3 groups:

A. Screen Routines

B. Virtual Routines

C. General Routines

A. SCREEN ROUTINES

These routines allow the user to perform screen-level graphics. Each routine places the Terminal in the proper mode and sends the character(s) necessary to perform the desired operation with a minimum of overhead.

THE FOLLOWING ROUTINES ARE DESCRIBED IN THE TCS USER'S MANUAL:

Graphic Output Routines

DRWABS	performs a screen level draw to absolute coordinates.
DRWREL	performs a draw to coordinates relative to the current beam position.
DSHABS	draws a dashed line to absolute coordinates.
DSHREL	draws a dashed line to coordinates relative to the current beam position.
*INCPLT	plots points incrementally in the desired direction.
MOVABS	performs a screen level move to absolute coordinates.
MOVREL	performs a move to coordinates relative to the current beam position.
PNTABS	draws a point at absolute coordinates.
PNTREL	draws a point at coordinates relative to the current beam position.

Graphic Input Routines

** DCURSR	activates the crosshair for screen graphic input and accepts graphic input characters.
** SCURSR	performs the same function as DCURSR.

Measurement Conversion Routines

KCM	function which converts centimeters to raster units.
KIN	function which converts inches to raster units.

*Applies only to the 4014 or 4015 Terminal with Enhanced Graphics Module.

**Does not apply to the 4006 Terminal. See the Appendix for more information

THE FOLLOWING ROUTINE IS DESCRIBED IN THE SYSTEM MANUAL:

*IPMOD	places the Terminal in incremental plot mode.

B. VIRTUAL ROUTINES

The following routines allow the user to specify moves and draws in any given coordinate system. The Terminal Control System converts these moves and draws into screen coordinates. The virtual routines also include those routines which establish the data to screen relationship and those which access the previously established relationship. THESE ROUTINES ARE DESCRIBED IN THE USER MANUAL:

Graphic Output Routines

DASHA	draws a dashed line in absolute, virtual coordinates.
DASHR	draws a dashed line in virtual coordinates, relative to the current beam position.
DASHSA	draws a segmented dashed line to absolute coordinates defined by a polar transformation.
DASHSR	draws a segmented dashed line to relative coordinates defined by a polar transformation.
DRAWA	draws to absolute, virtual coordinates.
DRAWR	draws to relative, virtual coordinates.
DRAWSA	draws a segmented line to absolute coordinates defined by a polar transformation.
DRAWSR	draws a segmented line to relative coordinates defined by a polar transformation.
MOVEA	moves to absolute, virtual coordinates.
MOVER	moves to relative, virtual coordinates.
POINTA	draws a point at absolute, virtual coordinates.
POINTR	draws a point at relative, virtual coordinates.

Relationship Establishing Routines

DWINDO	sets the corners of the virtual window. Performs the same function as VWINDO.
LINTRN	sets the transformation to linear.
LOGTRN	sets the transformation to log or semi-log.
POLTRN	sets the transformation to polar.
RROTAT	sets the rotation factor for relative virtual graphics.
RSCALE	sets the scaling factor for relative virtual graphics.
SWINDO	sets the corners of the screen window. Performs the same function as TWINDO.
TWINDO	sets the corners of the screen window. Performs the same function as SWINDO.
VWINDO	sets the corners of the virtual window. Performs the same function as DWINDO.

*Applies only to the 4014 or 4015 Terminal with Enhanced Graphics Module.

Graphic Input Routines

* VCURSR	activates the crosshair cursor for virtual graphic input and accepts graphic input characters.

THESE ROUTINES ARE DESCRIBED IN THE TCS SYSTEM MANUAL:

Scaling Routines

RESCAL	calculates all transformation parameters.
PSCAL	called by RESCAL to calculate the polar transformation parameters.

Conversion and Clipping Routines

CLIPT	checks for the need to clip vectors in virtual space; clips the vectors or calls PARCLT.
LVLCHT	checks for the need to match virtual and screen coordinates.
PARCLT	clips lines parallel to a screen window edge.
PCLIPT	determines whether coordinates are inside or outside the virtual window.
REL2AB	converts relative coordinates to absolute coordinates.
REVCOT	transforms screen coordinates into virtual coordinates.
V2ST	converts virtual coordinates to screen coordinates and moves to the clipped starting coordinate if necessary.
WINCOT	transforms virtual coordinates into screen coordinates.

C. GENERAL ROUTINES

The following routines are used by the screen and virtual routines to set Status Variables, place the Terminal in a particular mode and/or output appropriate graphics.

THESE ROUTINES ARE DESCRIBED IN THE TCS SYSTEM MANUAL:

DSHMOD	sets the Terminal for outputting a dashed line.
PLTCHR	computes the ADE characters needed to address a screen location.
** PNTMOD	places the Terminal in point plot mode.
TKDASH	constructs and outputs dashed lines.
TKPNT	outputs a point.
VECMOD	places the Terminal in vector mode.
XYCNVT	produces an optimized set of plot characters.

*Does not apply to the 4006 Terminal. See Appendix for more information.
**Applies only to the 4014 or 4015 Terminal with Enhanced Graphics Module.

2.2.2. Alphanumerics Routines

These routines control and execute alphanumeric input and output in one of three formats:

A1 FORTRAN format;

Am FORTRAN format, where m is the number of characters per word available on a particular system, as defined at implementation;

ADE (ASCII Decimal Equivalent) format, where each ASCII character is represented by an integer from 0 to 127 (see the USASCII Code Functions Charts at the end of this manual).

THE FOLLOWING ROUTINES ARE DESCRIBED IN THE TCS USER MANUAL:

Input/Output Routines

A1IN	allows the user to input an array in A1 FORTRAN format.
AINST	accepts an array in Am FORTRAN format.
A1OUT	outputs an array of characters in A1 FORTRAN format.
ANCHO	outputs a non-control ADE (ASCII Decimal Equivalent) character.
ANSTR	outputs an array of non-control ADE characters.
AOUTST	outputs an array of characters in Am FORTRAN format.

Terminal Controlling Routines

ANMODE	places the Terminal in alphanumeric mode and dumps the output buffer.
BAKSP	causes the A/N cursor to move back one space.
CARTN	moves the A/N cursor to the left margin.
*CHRSIZ	changes the current character size.
HOME	returns the cursor to the Home position (0,767).
LINEF	moves the A/N cursor down one line (line feed).
NEWLIN	calls CARTN and LINEF.
RSTTAB	selectively removes tabs.
SETMRG	sets the Terminal screen margins.
SETTAB	sets tabs in user-defined tab tables.
TABHOR	moves to the next value in the horizontal tab table.
TABVER	moves to the next value in the vertical tab table.
TTBLSZ	notifies Terminal Status Area of the user-defined dimensions of a tab table.

Information Returning Routines

*CSIZE	provides the current character height and width in raster units.
LEFTIO	function which returns the remaining space in the output buffer or the number of characters remaining in the input buffer.

*Applies only to the 4014 or 4015 Terminal.

LINWDT	function which returns in raster units the width of a given number of adjacent characters.
LINHGT	function which returns in raster units the height of a given number of lines.

THE FOLLOWING ROUTINES ARE DESCRIBED IN THE TCS SYSTEM MANUAL:

FORTRAN-ADE Translation Routines**

KA12AS	converts A1 characters to ADE characters.
KAM2AS	converts Am characters to ADE characters.
KAS2A1	converts ADE characters to A1 characters.
KAS2AM	converts ADE characters to Am characters.

Terminal Controlling Routines

ALFMOD	places the Terminal in alphanumeric mode.

2.2.3. Utility Routines

These routines allow the user to have direct control of features of the Terminal and the Terminal Control System not related to graphics or alphanumerics.

THE FOLLOWING ROUTINES ARE DESCRIBED IN THE TCS USER MANUAL:

Terminal Controlling Routines

BELL	causes the Terminal bell to ring.
*CZAXIS	changes the Z-Axis mode.
ERASE	erases the Terminal screen without changing the beam position.
FINITT	terminates the program in which it appears.
HDCOPY	causes a hardcopy to be generated.
INITT	initializes the Terminal Control System.
NEWPAG	erases the screen and returns the cursor to the Home position.
RECOVR	updates the Terminal hardware to match the Status Variables.
RESET	initializes the Terminal Control System without a page erase.
RESTAT	restores the Status Variable values which were saved by SVSTAT.

**See page 2-5 for an explanation of terms.

*Applies only to the 4014 or 4015 Terminal.

SETBUF	specifies an output buffer type.
SVSTAT	saves the current Status Variable values.
*TERM	specifies the Terminal type and addressing (1024 or 4096 addressable points) in use.

Information Returning Routines

SEEBUF	returns the current format of the output buffer.
SEEDW	returns the current values of the virtual window limits.
SEELOC	returns the last position of the graphic beam.
*SEEMOD	returns the current hardware dash type, Z-Axis mode and Terminal mode.
SEEMRG	returns the values of the current screen margins.
SEEREL	returns the scaling and rotation variable values.
SEETRM	returns the type of Terminal and addressing which has been specified.
SEETRN	returns the type of transformation in use.
SEETW	returns the current values of the screen window limits.

Input/Output Routines

TINPUT	accepts an input of one ADE character.
TINSTR	accepts an array of ADE characters.
TOUTPT	outputs a single ADE character.
TOUTST	outputs an array of ADE characters.

THE FOLLOWING ROUTINES ARE DESCRIBED IN THE TCS SYSTEM MANUAL:

Terminal Controlling Routines

IOWAIT	causes the system to wait while the Terminal is busy.
*CWSEND	sets the hardware dash type and Z-Axis mode.

Information Returning Routines

GENFLG	checks the general condition flag, KGNFLG.
TCSLEV	returns the software release number and the date of the last modification.

*Applies only to the 4014 or 4015 Terminal.

2.2.4. System I/O Routines

These routines provide the I/O interface between the Terminal Control System and the user's computer system.

THE FOLLOWING ROUTINES ARE DESCRIBED IN THE TCS SYSTEM MANUAL:

Output Buffering Routines

BUFFPK	packs the TCS output buffer.
TSEND	dumps the output buffer.

User-Written I/O Routines*

**ADEIN	accepts input, usually from the Terminal, in system-dependent format and converts it to an array in ADE format.
**ADEOUT	converts characters from ADE to system-dependent format and outputs them, usually to the Terminal.

*Supplied by Tektronix for TSO, PDP-11 and CDC-Synchronous versions of TCS.

**ADE (ASCII Decimal Equivalent) is the ASCII character set represented in integers from 0 to 127. See the USASCII Functions Charts at the end of this manual.

SECTION 3 SYSTEM SUBROUTINE DESCRIPTIONS

The subroutines described in this section are NOT described in the TCS User Manual. They are system routines which in most cases need not be called by the user. Flow charts are included for the more complex routines. The following routines are described in alphabetical order:

ALFMOD
BUFFPK
CLIPT
CWSEND
DSHMOD
GENFLG
IOWAIT
IPMOD
LVLCHT
PARCLT
PCLIPT
PLTCHR
PNTMOD
PSCAL
REL2AB
RESCAL
REVCOT
TKDASH
TKPNT
TSEND
V2ST
VECMOD
WINCOT
XYCNVT

See the Appendix for descriptions of the six user-written, system-dependent subroutines:*

ADEIN
ADEOUT
KA12AS
KAM2AS
KAS2A1
KAS2AM

3.1. ALFMOD – Enter Alphanumeric Mode

ALFMOD outputs an ASCII US character which places the Terminal in alphanumeric mode. Subsequent data sent to the Terminal will be interpreted as alphanumeric characters rather than as graphic vectors. This routine always sends a US, since the Terminal mode is not checked. ALFMOD is different from ANMODE in that it does not dump the output buffer.

Calling Sequence:

```
CALL ALFMOD
```

*Supplied by Tektronix for TSO, PDP-11 and CDC-Synchronous versions of TCS.

3.2. BUFFPK – Pack the Buffer

BUFFPK loads the ADE characters it receives into an output buffer. When the buffer is filled or when a buffer dump is requested, BUFFPK calls ADEOUT to perform the output. For buffer types 1, 2, or 3, if NCHAR is larger than MAXLEN, the size of the buffer (see page A2), the extra characters are truncated and lost. Buffer type 4 assumes ADEOUT can handle any size buffer array. If the buffer is type 1 or 2, extra characters are added to counteract the effects of CR, LF, etc. between outputs. (For more information, see SETBUF in the User Manual and page A2 of this System Manual.)

Calling Sequence:

```
CALL BUFFPK (NCHAR,IARRAY)
```

Parameters Entered:

NCHAR — The number of characters to be added to the buffer. NCHAR = 0 is a request to dump the buffer.

IARRAY — The array containing the characters to be added to the buffer in ADE format.

3.3. CLIPT – Clip Virtual Vectors

CLIPT checks for the need to clip virtual vectors and clips those in need before they are converted to screen coordinates. Horizontal and vertical lines are handled separately. Calling this routine will affect the Status Variable KGNFLG in the Terminal Status Area as follows:

KGNFLG = 0 — if any part of the vector is inside the window

= 1 — if the vector is entirely outside the window

Calling sequence:

```
CALL CLIPT(BUFIN,BUFOUT)
```

Parameter Entered:

BUFIN — An array containing the end points of the line segment (vector) before clipping.

Parameter Returned:

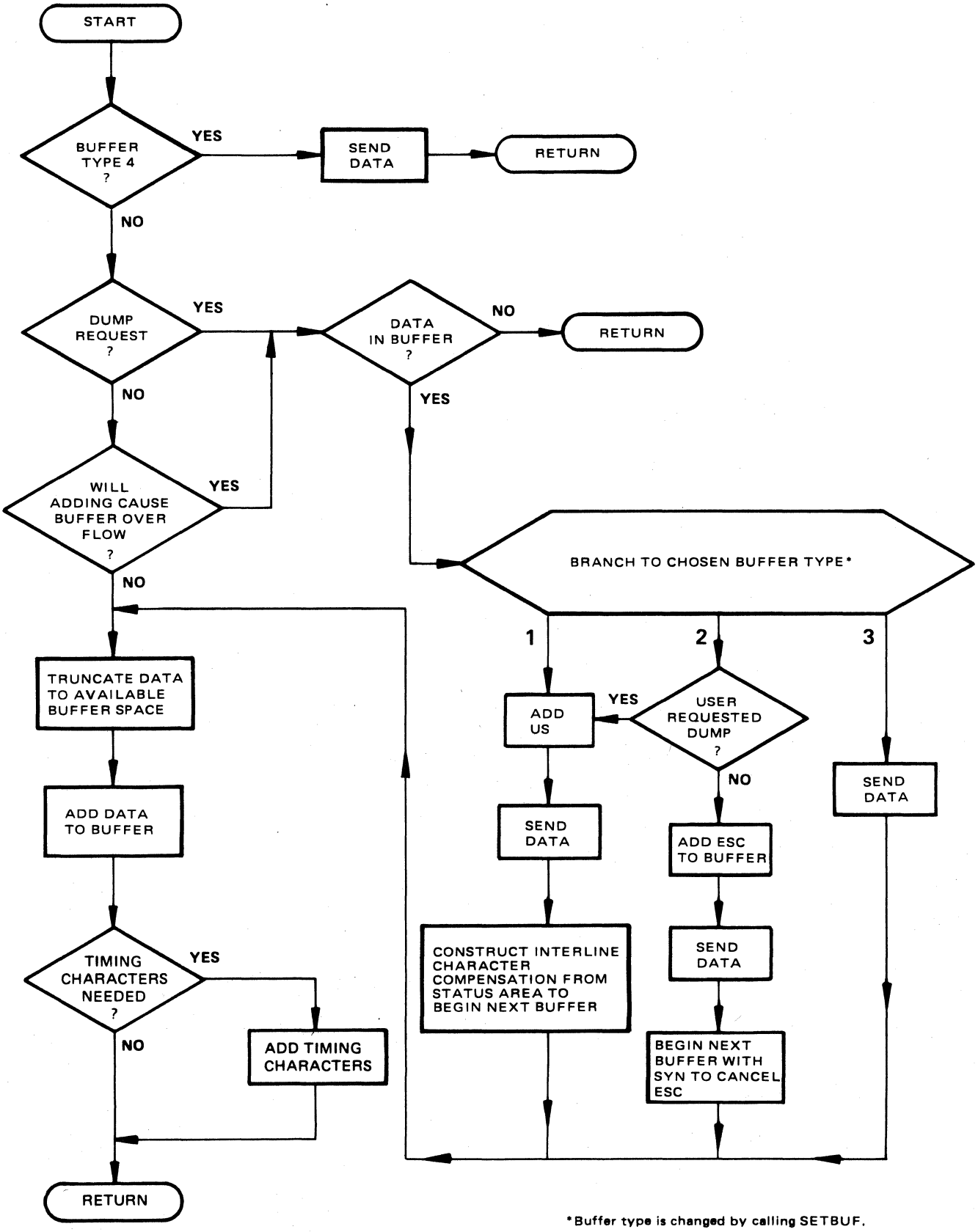
BUFOUT — An array containing the endpoints of the clipped line segment.

NOTE

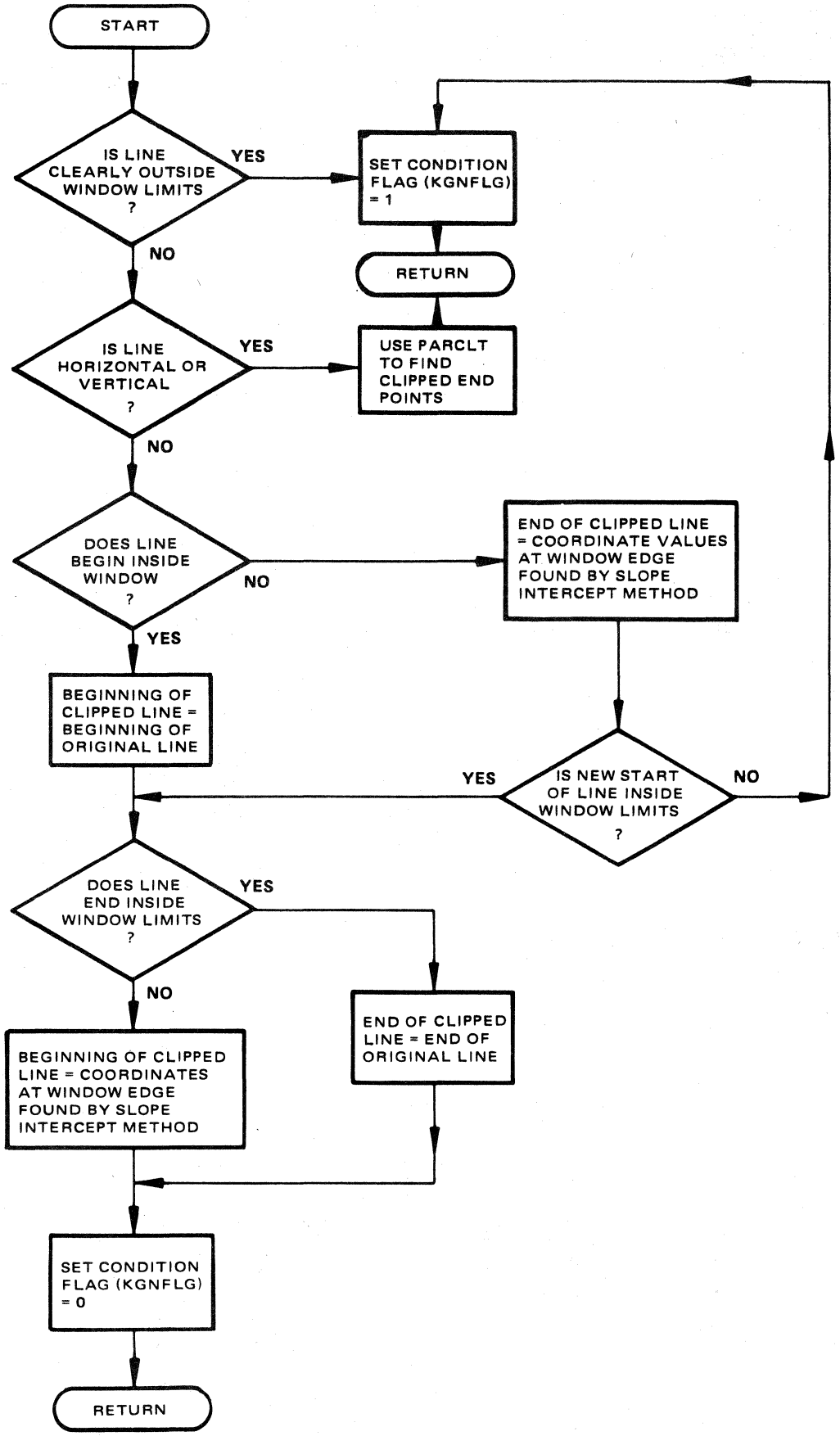
The format of both the above arrays is:

1. *beginning X*
2. *beginning Y*
3. *ending X*
4. *ending Y*

BUFFPK



*Buffer type is changed by calling SETBUF.


START
IS LINE CLEARLY OUTSIDE WINDOW LIMITS ?
YES
SET CONDITION FLAG (KGNFLG) = 1
RETURN
NO
IS LINE HORIZONTAL OR VERTICAL ?
YES
USE PARCLT TO FIND CLIPPED END POINTS
NO
DOES LINE BEGIN INSIDE WINDOW ?
NO
END OF CLIPPED LINE = COORDINATE VALUES AT WINDOW EDGE FOUND BY SLOPE INTERCEPT METHOD
YES
BEGINNING OF CLIPPED LINE = BEGINNING OF ORIGINAL LINE
YES
IS NEW START OF LINE INSIDE WINDOW LIMITS ?
NO
DOES LINE END INSIDE WINDOW LIMITS ?
YES
END OF CLIPPED LINE = END OF ORIGINAL LINE
NO
BEGINNING OF CLIPPED LINE = COORDINATES AT WINDOW EDGE FOUND BY SLOPE INTERCEPT METHOD
SET CONDITION FLAG (KGNFLG) = 0
RETURN

3.4. CWSEND – Send a Control Word

CWSEND* is called when the user changes the Status Variables KLINE and KZAXIS through CZAXIS or any of the dashed line routines. CWSEND outputs the ESC sequence necessary to compensate for interline characters and sets the hardware dash type and Z-Axis mode.

Calling Sequence:

```
CALL CWSEND
```

3.5. DSHMOD – Enter Dashed Line Mode

DSHMOD outputs a US to reset the Terminal, then a GS to place the Terminal in graphics mode if the Terminal is not already in dashed line mode. DSHMOD enters the dash type into the Status Variable KDASHT and cancels graphic output optimization.

Calling Sequence:

```
CALL DSHMOD (L)
```

Parameter Entered:

L The dash type for the next dashed line (see the User Manual, Section 3.12).

3.6. GENFLG – Check the General Condition Flag

GENFLG allows the user to reference Status Variable KGNFLG in the Terminal Status Area. (See Section 5.1 for a list of KGNFLG values.) This variable is set in CLIPT, PCLIPT, SETTAB and RESCAL. The user may call GENFLG with no effect on the Terminal Status Area.

Calling Sequence:

```
K = GENFLG (ITEM)
```

Parameter Entered:

ITEM The value (0 or 1) for which the user is checking.

Parameter Returned:

K True if ITEM = KGNFLG; otherwise false.

3.7. IOWAIT – Wait During I/O

IOWAIT sends a series of SYN characters so that no data will be sent while the Terminal is busy. The number of SYN characters sent is determined by multiplying the desired wait time (ITIME) by the number of characters transmitted per second (as determined by INITT). The user may call IOWAIT with no effect on the Terminal Status Area.

*Applies only to the 4014 or 4015 Terminal with Enhanced Graphics Module.

Calling Sequence:

CALL IOWAIT (ITIME)

Parameter Entered:

ITIME — The wait time in tenths of a second.

3.8. IPMOD — Enter Increment Mode

IPMOD* outputs as US to cancel any previous mode and sets the Terminal to alphanumeric mode. It then outputs an RS to put the Terminal in incremental plot mode.

Calling Sequence:

CALL IPMOD

3.9. LVLCHT — Check Graphic Level

LVLCHT checks the Status Variable KGRAFL to determine whether it is necessary to update the virtual coordinates to match the screen coordinates. An update is needed when KGRAFL has been set to 0 by any screen level graphic routine. LVLCHT calls subroutine REVCOT to update the virtual coordinates.

Calling Sequence:

CALL LVLCHT

3.10. PARCLT — Clip Lines Parallel to Window Edge

PARCLT is used to clip a line which is parallel to the window edge. The routine checks to see if the end points of the clipped line are within the range of a pair of given limits (usually the window limits). It returns a pair of values inside the limit range. The user may call PARCLT with no effect on the Terminal Status Area.

Calling Sequence:

CALL PARCLT (RL1,RL2,RM1,RM2,RN1,RN2)

Parameters Entered:

RL1	The variable No. 1 to be checked.
RL2	The variable No. 2 to be checked.
RM1	The minimum limit of the desired range.
RM 2	The maximum limit of the desired range.

Parameters Returned:

RN1	The variable No. 1 with a value inside the desired range.
RN2	The variable No. 2 with a value inside the desired range.

*This routine applies only to the 4014/4015 Terminals with Enhanced Graphics Module.

3.11. PCLIPT – Clip a Point Outside the Virtual Window

PCLIPT determines whether a given point is inside the virtual window. The routine sets Status Variable KGNFLG to = 0 if the point is inside.

Calling Sequence:

```
CALL PCLIPT(X,Y)
```

Parameters Entered:

X	The virtual X coordinate being checked.
Y	The virtual Y coordinate being checked.

3.12. PLTCHR – Convert X,Y Plot Characters

PLTCHR returns an array containing the ADE (ASCII Decimal Equivalent) characters which are needed to address a given point on the Terminal screen. The order in which this array is returned is:

HiY, LSBYX*, LoY, HiX, LoX.

This routine sets variable KPADV in the Terminal Status Area. KPADV contains the number of timing SYN characters needed.

Calling Sequence:

```
CALL PLTCHR(IX,IY,ICHAR)
```

Parameters Entered:

IX	The X-coordinate of the point.
IY	The Y-coordinate of the point.

Parameter Returned:

ICHAR	The array containing the plot characters.

3.13. PNTMOD – Enter Point Plot Mode

PNTMOD outputs a US to set the Terminal to alphanumeric mode, without checking for the previous Terminal mode. It then cancels the optimization of plot characters and sets Status Variable KKMODE to 2. If the Terminal is a 4014 or 4015 with Enhanced Graphics Module, PNTMOD also outputs an FS to place the Terminal in hardware point plot mode.

Calling Sequence:

```
CALL PNTMOD
```

*Least Significant Bit(Y,X); this bit is used for 12-bit (4096) addressing on the 4014 or 4015 Terminal with the Enhanced Graphics Module. For other terminals or for regular 10-bit (1024) addressing, this character is ignored.

3.14. PSCAL – Scale the Polar Transformation

PSCAL calculates the information needed for a polar transformation. The limits of a polar window of the shape requested are determined from the angle minimum (TRPAR1), angle maximum (TRPAR2), radius suppression (TRPAR5), and the virtual radius minimum and maximum (TMINVX and TMAXVX). The calculated limits are used to determine the angle scale factor (TRFACY), the X and Y screen offsets (TRPAR3 and TRPAR4), and the angle offset (TRPAR6).

Calling Sequence:

```
CALL PSCAL
```

3.15. REL2AB – Convert Relative to Absolute

REL2AB computes and returns an absolute virtual coordinate specified by the displacement requested, scale and rotation factors (supplied by RSCALE and RROTAT stored in the Terminal Status Area) and the present virtual location stored in the Terminal Status Area. The present virtual location is used as the origin for rotation and scaling. REL2AB calls LVLCHT to update the virtual coordinates before performing the calculation.

Calling Sequence:

```
CALL REL2AB (XIN, YIN,XOUT,YOUT)
```

Parameters Entered:

XIN	The virtual X displacement.
YIN	The virtual Y displacement.

Parameters Returned:

XOUT	The updated X absolute coordinate.
YOUT	The updated Y absolute coordinate.

3.16. RESCAL – Set the Transformation Scale

RESCAL calculates the linear and logarithmic transformation parameters used by REVCOT and WINCOT. RESCAL uses the Status Variables set by VWINDO, SWINDO, DWINDO and TWINDO and the transformation routines POLTRN, LINTRN and LOGTRN. After the calculation, the transformation parameters are stored in the Terminal Status Area. RESCAL calls PSCAL to calculate polar transformation.

RESCAL sets KGNFLG = 1 if the transformation requested is an invalid one, such as a logarithmic transformation on an axis with negative limits. Otherwise, KGNFLG = 0.

Provision for a user-defined transformation is included (see Section 4.2).

Calling Sequence;

```
CALL RESCAL
```

3.17. REVCOT – Transform Window Coordinates

REVCOT transforms screen coordinates into virtual coordinates. The transformation parameters have different meanings depending on whether linear, logarithmic or polar transformation is in effect. The routine branches to a different section for each type of transformation. Provision for a user-defined transformation is included (see Section 4.2). Calling REVOCT has no effect on the Terminal Status Area.

Calling Sequence:

```
CALL REVCOT(IX,IY,X,Y)
```

Parameters Entered:

IX	The screen X coordinate.
IY	The screen Y coordinate.

Parameters Returned:

X	The virtual X coordinate.
Y	The virtual Y coordinate.

3.18. TKDASH

TKDASH constructs and outputs dashed lines. The dash type is determined by the Status Variable KDASHT set by subroutine DSHMOD. If the dash type is a software type, TKDASH constructs a table which gives the length of each segment in raster units. This table is used to determine the destination of each light or dark segment drawn until the end point of the line is reached. If the starting point of this line is the same as the end point of the last line drawn and the dash type is the same, the pattern is continued and not restarted.

If the dash type is 1, 2, 3 or 4 and the Terminal is a 4014 or 4015 with Enhanced Graphic Module, the dash type is set in the Terminal Status Area and CWSEND is called to output the control sequence needed to place the Terminal in the correct state. A vector to the destination is then output.

If another model Terminal is used, the hardware dash types are simulated by software dash types (see the User Manual, Section 3.12).

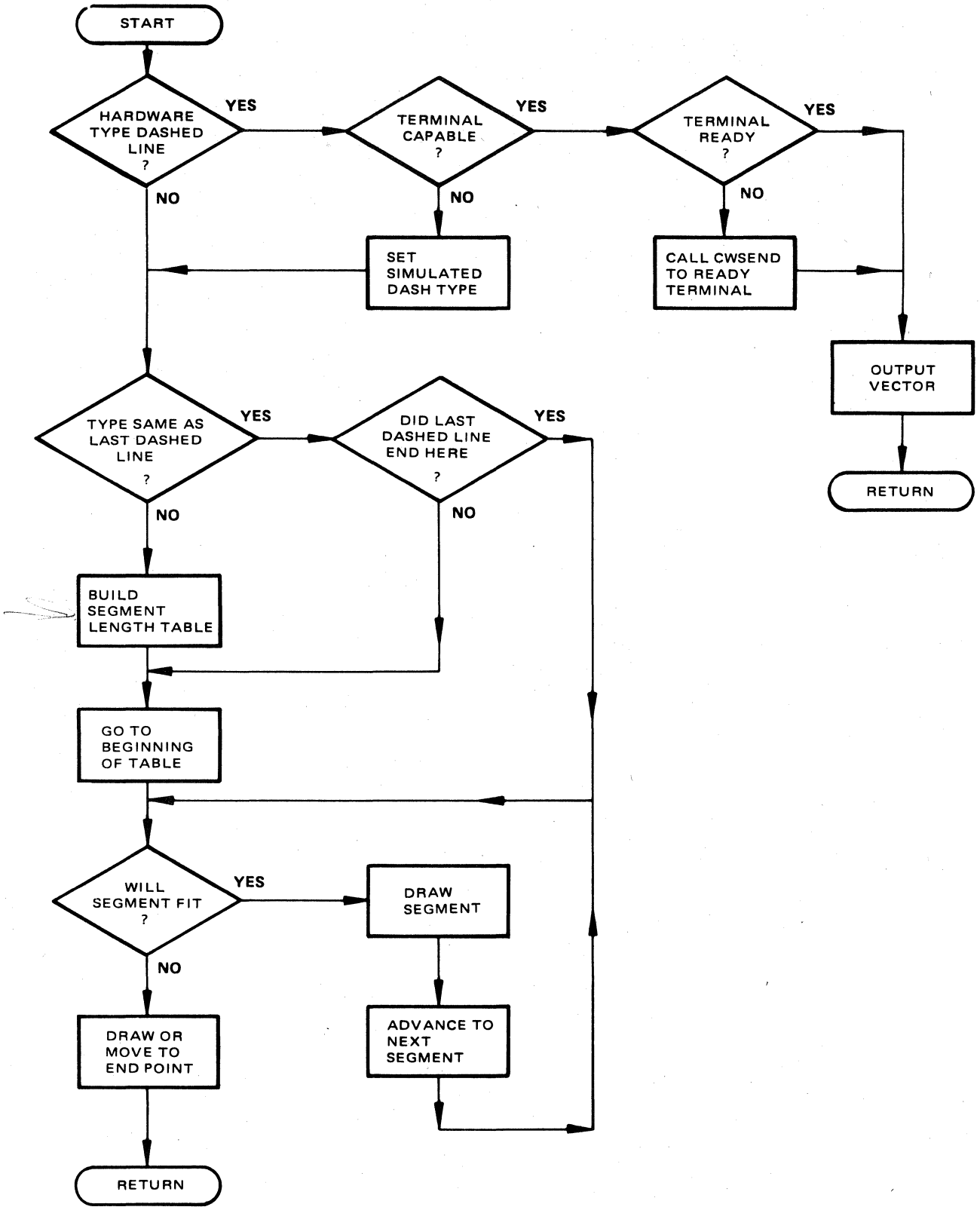
Calling Sequence:

```
CALL TKDASH(IX,IY)
```

Parameters Entered:

IX	The X screen coordinate of the dashed line destination.
IY	The Y screen coordinate of the dashed line destination.

TKDASH


START
HARDWARE TYPE DASHED LINE ?
YES
NO
TERMINAL CAPABLE ?
YES
NO
SET SIMULATED DASH TYPE
TERMINAL READY ?
YES
NO
CALL CWSEND TO READY TERMINAL
OUTPUT VECTOR
RETURN
TYPE SAME AS LAST DASHED LINE ?
YES
NO
DID LAST DASHED LINE END HERE ?
YES
NO
BUILD SEGMENT LENGTH TABLE
GO TO BEGINNING OF TABLE
WILL SEGMENT FIT ?
YES
NO
DRAW SEGMENT
ADVANCE TO NEXT SEGMENT
DRAW OR MOVE TO END POINT
RETURN

3.19. TKPNT – Output a Point

TKPNT plots the point specified in hardware point plot mode, for a 4014 or 4015 Terminal with Enhanced Graphics Module. If the Terminal does not have the hardware capabilities, the routine causes a move to the point and the drawing of the point to simulate point plot mode. The user can set the Terminal type in subroutine TERM.

Calling Sequence:

```
CALL TKPNT(IX,IY)
```

Parameters Entered:

IX	The X coordinate of the point.
IY	The Y coordinate of the point.

3.20. TSEND – Dump the Buffer

TSEND calls BUFFPK with the length parameter = 0, causing the output buffer to be dumped.

Calling Sequence:

```
CALL TSEND
```

3.21. V2ST – Transform Virtual to Screen Coordinates

V2ST converts coordinates from virtual space to screen space and creates a move to the clipped starting coordinates if it is appropriate. This routine returns the screen coordinates for subroutines MOVEA, DRAWA, POINTA or DASHA. V2ST updates both the graphic and imaginary beams. The imaginary beam accounts for a point addressed in virtual space which cannot be represented on the terminal screen. V2ST references the Status Variable KGNFLG; if KGNFLG = 1, the entire line is outside the window, and neither V2ST nor the four virtual absolute routines take any action.

Calling Sequence:

```
CALL V2ST(I,X,Y,IX,IY)
```

Parameters Entered:

I	An integer flag which equals 0 if the routine is called for a move or a point plot and does not equal 0 if the routine is called for a draw or a dashed line.
X	The virtual space X coordinate.
Y	The virtual space Y coordinate.

Parameters Returned:

IX	The screen X coordinate.
IY	The screen Y coordinate.

3.22. VECMOD – Enter Graphics Mode

VECMOD outputs a US to set the Terminal to alphanumeric mode, if the Terminal is not already in graphics mode. The routine then replaces the plot characters with an invalid value (−1) so that they will be updated by the next vector. The routine then outputs a GS to place the Terminal in graphics mode and cause the next vector to be dark.

Calling Sequence:

```
CALL VECMOD
```

3.23. WINCOT – Transform Window Coordinates

WINCOT transforms virtual coordinates into the appropriate screen coordinates. A branch is made to a different section of the routine, depending on whether the transformation in effect is linear, logarithmic or polar. The user can define his own transformation (see Section 4.2). Calling this routine does not affect the Terminal Status Area.

Calling Sequence:

```
CALL WINCOT(X,Y,IX,IY)
```

Parameters Entered:

X	The virtual X coordinate.
Y	The virtual Y coordinate.

Parameters Returned:

IX	The screen X coordinate.
IY	The screen Y coordinate.

3.24. XYCNVT – Convert and Output X,Y

XYCNVT compares the plot characters needed to draw to a specified location with the last set of plot characters sent. It then produces an optimized set of plot characters to draw the vector. Reducing the number of plot characters has two advantages:

1. There is less chance of transmission errors.
2. Less transmission time is required to draw the vector.

The routine is designed so that bright vectors are not drawn repeatedly to the same screen location. This saves time and avoids damaging the screen. A vector is drawn if any one of the following conditions is true:

1. The endpoint of the vector is different from that of the last vector drawn.
2. The previous vector was a dark vector to the same location.

3. The desired vector is dark (i.e., a move).

Calling Sequence:

```
CALL XYCNVT(IX,IY)
```

Parameters Entered:

IX	The screen X coordinate.
IY	The screen Y coordinate.

SECTION 4 MODIFYING THE SYSTEM

This section describes ways in which the Terminal Control System can be modified. Three types of changes are described:

1. Changing the I/O and translate parameters to fit individual computer system requirements.

2. Adding user-written transformation routines.

3. Reducing the size of the package. The user can eliminate:

   . unused routines;
   . polar and/or logarithmic transformations;
   . unnecessary I/O routines.

   The actual "pruning" of the package can occur on either of two levels:

   A—LEVEL PRUNING removes unused routines. Internally called routines are replaced by smaller dummy routines.

   B—LEVEL PRUNING removes all the code which supports unused features.

CAUTION

Routines should be eliminated from the system only after careful consideration. Removal should be well documented by the user. If the software is pruned as indicated, however, it will be fully supported by Tektronix.

4.1. Changing I/O and Translate Parameters

The input and output buffers and translate arrays of the Terminal Control System are based on a line length of 72 characters (the longest line possible on some computer systems*). The user can change these values to the limits allowed by his computer system.

Routine	Parameter	Use	Present Value
A1IN	IADE	Translate array	Dimensioned to 72
	MAXLEN	Maximum data length	72 characters
A1OUT	IADE	Translate array	Dimensioned to 72
AINST	IADE	Translate array	Dimensioned to 72
	MAXLEN	Maximum data length	72 characters
ANSTR	MAXLEN	Maximum characters sent to TOUTST	Set in KACHAR
AOUTST	IADE	Translate array	Dimensioned to 72
	MAXLEN	Maximum data length	72 characters
BUFFPK	IDATA	Data array	Dimensioned to 72
	MAXLEN	Maximum size of data array	72 characters

*132 on TSO and PDP-11 systems; 80 on CDC-Synchronous systems.

Routine	Parameter	Use	Present Value
INITT RESET	Common default assignments; see Section 5. Be very careful when making changes.		
TINSTR	INBUFF	Input data array	Dimensioned to 72
TOUTST	IUSE	Data transfer array	Dimensioned to 72
	MAXLEN	Maximum data length	72 characters

4.2. Adding User-Written Transformations

In addition to linear, logarithmic and polar transformations, the Terminal Control System allows the user to add his own transformation. To do this, he must write the following four routines:

Subroutine USETRN

This routine allows the program to perform a user-defined transformation. Status Variable KEYCON should be set to 4. The routine should also set any other Status Variables necessary for the transformation calculations (see Section 5).

Calling Sequence:

```
CALL USETRN [user-defined arguments]
```

Subroutine URSCAL

This routine uses the Status Variables set by USETRN to calculate the parameters needed to perform the transformation.

Calling Sequence:

```
CALL URSCAL
```

Subroutine USECOT

This routine converts virtual coordinates (X,Y) into screen coordinates (IX,IY) through the use of the Status Variables set by URSCAL.

Calling Sequence:

```
CALL USECOT(X,Y,IX,IY)
```

Subroutine UREVCT

This routine converts screen coordinates (IX,IY) into virtual coordinates (X,Y).

Calling Sequence:

```
CALL UREVCT(IX,IY,X,Y)
```

In addition to writing the above routines, the user must change subroutine RESCAL to allow the calling of URSCAL when KEYCON = 4. Similar changes should be made to allow the calling of UREVCT from subroutine REVCOT and USECOT from subroutine WINCOT.

If the user wishes to define segmented vectors for his transformation, he should write subroutine USDRAW(X,Y) and subroutine USDASH(X,Y,L) to perform these functions. Subroutines DRAWSA and DASHSA should be modified accordingly to allow the calling of USDRAW and USDASH. The parameters of USDRAW and USDASH should correspond respectively to those of DRAWSA and DASHSA. The following is an example of a user-written transformation:

```
C * SAMPLE PROGRAM TO USE USER TRANSFORMATION
      CALL INITT(30)
      CALL DWINDO(0.,10.,0.,10.)
      CALL MOVEA(0.,0.)
      DO 10 I=1,10
C * SHOW THE DEFAULT TRANSFORMATION
      CALL DRAWA(FLOAT(I),FLOAT(I))
10    CONTINUE
C * INVOKE THE USER TRANSFORMATION
      CALL USETRN
      CALL MOVEA(0.,0.)
      DO 20 I=1,10
C * SHOW THE USER TRANSFORMATION
      CALL DASHA(FLOAT(I),FLOAT(I),2)
20    CONTINUE
      CALL MOVEA(0.,0.)
      DO 30 I=1,10
C * SHOW THE USER SEGMENTED AND TRANSFORMED LINE
      CALL DRAWSA(FLOAT(I),FLOAT(I))
30    CONTINUE
      CALL FINITT(0,700)
      END
C * SUBROUTINE TO INVOKE USER TRANSFORMATION *
      SUBROUTINE USETRN
      COMMON /TKTRNX/ TMINVX,TMINVY,TMAXVX,TMAXVY,TREALX,TREALY,
     1 TIMAGX,TIMAGY,TRCOSF,TRSINF,TRSCAL,TRFACX,TRFACY,
     2 TRPAR1,TRPAR2,TRPAR3,TRPAR4,TRPAR5,TRPAR6,KMOFLG(8),KPAD2,
     3 KBAUDR,KGNFLG,KGRAFL,KHOMEY,KKMODE,KHORSZ,KVERSZ,KTBLSZ,
     4 KSIZEF,KLMRGN,KRMRGN,KFACTR,KTERM,KLINE,KZAXIS,KBEAMX,KBEAMY,
     5 KMOVEF,KPCHAR(5),KDASHT,KMINSX,KMINSY,KMAXSX,KMAXSY,KEYCON,
     6 KINLFT,KOTLFT,KUNIT
      KEYCON=4
      CALL RESCAL
      RETURN
      END
C
C * SUBROUTINE TO CALCULATE USER TRANSFORMATION PARAMETERS
      SUBROUTINE URSCAL
      COMMON /TKTRNX/ TMINVX,TMINVY,TMAXVX,TMAXVY,TREALX,TREALY,
     1 TIMAGX,TIMAGY,TRCOSF,TRSINF,TRSCAL,TRFACX,TRFACY,
     2 TRPAR1,TRPAR2,TRPAR3,TRPAR4,TRPAR5,TRPAR6,KMOFLG(8),KPAD2,
```

```
     3 KBAUDR,KGNFLG,KGRAFL,KHOMEY,KKMODE,KHORSZ,KVERSZ,KTBLSZ,
     4 KSIZEF,KLMRGN,KRMRGN,KFACTR,KTERM,KLINE,KZAXIS,KBEAMX,KBEAMY,
     5 KMOVEF,KPCHAR(5),KDASHT,KMINSX,KMINSY,KMAXSX,KMAXSY,KEYCON,
     6 KINLFT,KOTLFT,KUNIT
C * CALCULATE THE MINIMUM TRANSFORMED VALUE OF X AND Y
      TRPAR1=TMINVX**3
      TRPAR2=TMINVY**5
C * CALCULATE SCALE FACTORS X AND Y AS  SCREEN RANGE / TRANSFORMED RANGE
      TRFACX=FLOAT(KMAXSX-KMINSX)/(TMAXVX**3-TRPAR1)
      TRFACY=FLOAT(KMAXSY-KMINSY)/(TMAXVY**5-TRPAR2)
      RETURN
      END
C
C * SUBROUTINE TO CALCULATE USER TRANSFORMATION SCREEN COORDINATES
      SUBROUTINE USECOT(X,Y,IX,IY)
      COMMON /TKTRNX/ TMINVX,TMINVY,TMAXVX,TMAXVY,TREALX,TREALY,
     1 TIMAGX,TIMAGY,TRCOSF,TRSINF,TRSCAL,TRFACX,TRFACY,
     2 TRPAR1,TRPAR2,TRPAR3,TRPAR4,TRPAR5,TRPAR6,KMOFLG(8),KPAD2,
     3 KBAUDR,KGNFLG,KGRAFL,KHOMEY,KKMODE,KHORSZ,KVERSZ,KTBLSZ,
     4 KSIZEF,KLMRGN,KRMRGN,KFACTR,KTERM,KLINE,KZAXIS,KBEAMX,KBEAMY,
     5 KMOVEF,KPCHAR(5),KDASHT,KMINSX,KMINSY,KMAXSX,KMAXSY,KEYCON,
     6 KINLFT,KOTLFT,KUNIT
C * TRANSFORM X AND Y
      XTEMP=X**3
      YTEMP=Y**5
C * SUBTRACT THE MINIMUM TRANSFORMED VALUES
      XTEMP=XTEMP-TRPAR1
      YTEMP=YTEMP-TRPAR2
C * SCALE TO FIT SCREEN WINDOW
      XTEMP=XTEMP*TRFACX
      YTEMP=YTEMP*TRFACY
C * ADD THE SCREEN ORIGIN
      IX=KMINSX+IFIX(XTEMP)
      IY=KMINSY+IFIX(YTEMP)
      RETURN
      END
C * SUBROUTINE TO USER SEGMENT LINES
      SUBROUTINE USDRAW(X,Y)
      COMMON /TKTRNX/ TMINVX,TMINVY,TMAXVX,TMAXVY,TREALX,TREALY,
     1 TIMAGX,TIMAGY,TRCOSF,TRSINF,TRSCAL,TRFACX,TRFACY,
     2 TRPAR1,TRPAR2,TRPAR3,TRPAR4,TRPAR5,TRPAR6,KMOFLG(8),KPAD2,
     3 KBAUDR,KGNFLG,KGRAFL,KHOMEY,KKMODE,KHORSZ,KVERSZ,KTBLSZ,
     4 KSIZEF,KLMRGN,KRMRGN,KFACTR,KTERM,KLINE,KZAXIS,KBEAMX,KBEAMY,
     5 KMOVEF,KPCHAR(5),KDASHT,KMINSX,KMINSY,KMAXSX,KMAXSY,KEYCON,
     6 KINLFT,KOTLFT,KUNIT
C * MAKE SURE CURRANT VIRTUAL BEAM IS CORRECT
      CALL LVLCHT
C * DRAW HORIZONAL
      CALL DRAWA(X,TIMAGY)
C * VERTICAL DRAW TO END POINT WILL BE DONE IN DRAWSA
      RETURN
      END
```

The "C" which indicates a comment line is then removed from the appropriate CALL statements in DRAWSA, RESCAL and WINCOT, as marked by an arrow at the left margin.

DRAWSA

```
      C * USER SEGMENTATION
      400   CONTINUE
      ----> CALL USDRAW(X,Y)
            GO TO 100
            END
```

RESCAL

```
      C * USER FUNCTION
      400   CONTINUE
      ----> CALL URSCAL
            GO TO 600
      C * NO SCALE
      500   TRFACX=1.
            TRFACY=1.
      600   RETURN
            END
```

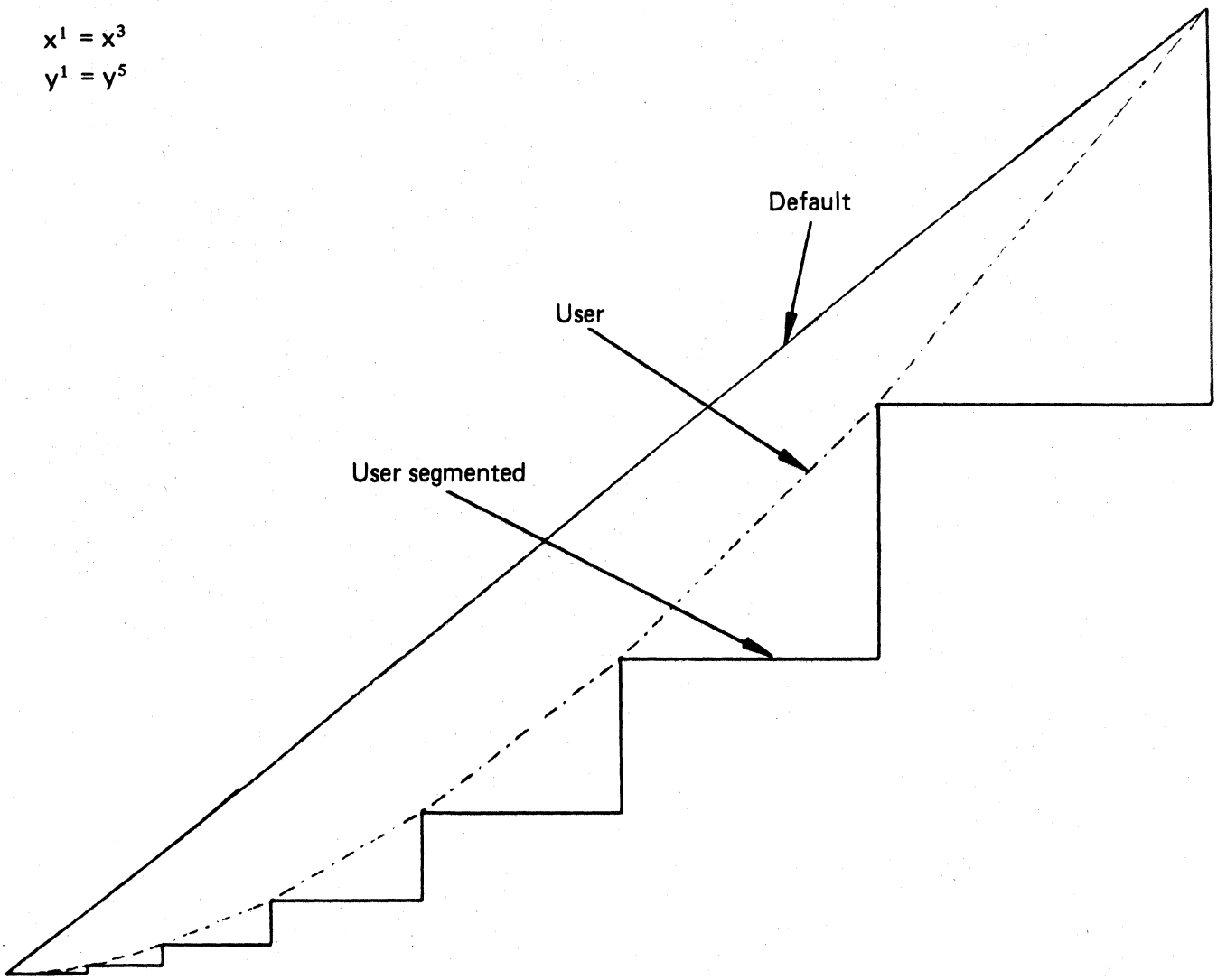
WINCOT

```
      C * USER TRANSFORMATION IN USE
      700   CONTINUE
      ----> CALL USECOT(X,Y,IX,IY)
      C * EXIT POINT
      800   RETURN
            END
```

This program produces the following graph:

USETRN

$x^1 = x^3$
$y^1 = y^5$



4.3. Reducing Package Size

4.3.1. Removing Unused Routines

NOTE

The savings resulting from the removal of routines will vary from system to system. If the user's link-load process does not load unused routines, savings in storage costs from the deletion of these routines may be negligible.

Routines may be removed from the Terminal Control System if two conditions are met by ALL programs using TCS:

1. None of the programs directly calls the routine to be removed.

2. None of the programs accesses any code that calls the routine.

The deletion process may extend to routines called only by previously deleted routines. The Subroutine Calling Reference Chart in Section 6 of this manual will help to determine which routines may be deleted.

Before deleting any routine, however, check to be sure it is not needed by other Tektronix software and that no other user intends to use it. The removal of any routine should be documented for future reference.

Routines Not Called Internally By The Terminal Control System

A1IN	DRWREL	LOGTRN	SEETRM
A1OUT	DSHREL	MOVER	SEETRN
AINST	DWINDO	MOVREL	SEETW
ANCHO	ERASE	PNTREL	SETBUF
ANMODE	FINITT	POINTR	SETMRG
AOUTST	GENFLG	POLTRN	SETTAB
BAKSP	HDCOPY	RESTAT	SVSTAT
BELL	HOME	RROTAT	SWINDO
CARTN	INCPLT	RSCALE	TABHOR
CSIZE	INITT	RSTTAB	TABVER
CZAXIS	KCM	SEEBUF	TERM
DASHR	KIN	SEEDW	TINPUT
DASHSR	LINEF	SEELOC	TTBLSZ
DCURSR	LINTRN	SEEMOD	TWINDO
DRAWR	LINHGT	SEEMRG	VCURSR
DRAWSR	LINWDT	SEEREL	VWINDO

4.3.2. A-Level Pruning

A-level pruning is the removal of features through the deletion of the user-called routines supporting those features. Internally called routines must be replaced by dummy routines having the same name and arguments as the routines they replace. A dummy routine must contain an executable statement, a RETURN statement and an END statement (on some computer systems the executable statement may be omitted). See the Subroutine Calling Reference Chart in Section 6 of this manual for the arguments of each subroutine.

A dummy function needs an assignment statement: [function name] = 0.

Feature Eliminated	Routines Eliminated
4014/4015 Support	
Changing Character Size	CHRSIZ*
Enhanced Graphics Option	
Incremental Plotting	INCPLT IPMOD
Hardware Dashed Lines & Z-Axis Control	CZAXIS CWSEND*
Special Vector Types	
Incremental Plotting	INCPLT IPMOD
Software or Hardware Produced Point Plotting Plotting	PNTABS PNTREL POINTA POINTR TKPNT PNTMOD*
Dash Plotting	DASHSA DASHSR TKDASH DSHABS DSHREL DSHMOD DASHA DASHR
Segmented Vectors	DRAWSA DRAWSR DASHSA .DASHSR

*A dummy routine replacement is required.

Feature Eliminated	Routines Eliminated
Relative Vectors	
Relative Virtual Vectors	DRAWR POINTR DASHR MOVER DASHSR DRAWSR REL2AB RSCALE RROTAT
Relative Screen Vectors	DRWREL DSHREL PNTREL MOVREL
User Alphanumeric Output	A1OUT AOUTST ANCHO ANSTR
User Input	
Crosshair	DCURSR SCURSR VCURSR
Keyboard	A1IN AINST TINPUT TINSTR*
Transformation Support	
Polar Plotting	POLTRN PSCALL** PSCAL

**Warning: TINSTR must be present for any input through the Terminal Control System.*

***A dummy routine replacement is required.*

4.3.3. B-Level Pruning

B-level pruning is the removal of all code used by an unwanted feature. Entire routines which support that feature are eliminated; in other routines, statements which support that feature are removed or modified.

If a feature is eliminated by B-level pruning, comparable changes need not be made at the A level.

B-level pruning may be used to eliminate the following features from the Terminal Control System:

1. Polar Plotting
2. Logarithmic Plotting
3. Multiple Character Sizes
4. Point Plotting
5. Hardware Point Plotting
6. Z-Axis Mode Changes AND Hardware Dashed Lines
7. Z-Axis Mode Changes
8. Hardware Dashed Lines
9. Interline Character Effect Supression
10. Software Dashed Lines*

Code to be modified or removed is marked at the left margin.

Each feature is described separately. If the same line is to be changed for several features, the changes should be made accumulatively.

CAUTION

Eliminate only that code which is truly unnecessary for your operation. Document all changes for future reference.

*If you have the four hardware dash types of the Enhanced Graphics Module, you may wish to eliminate software dashes to save storage.

Feature 1. Polar Plotting

Remove:	POLTRN	PSCAL	
Modify:	DASHSA	RESCAL	REVCOT
	DRAWSA	WINCOT	

DASHSA

```
C * LINEAR LOG POLAR USER ERROR
      GO TO (100, 100, 200, 400, 100),KEY
C * ERROR LINEAR LOG
100   CALL DASHA(X,Y,L)
      XOLD=X
      YOLD=Y
      RETURN
C * POLAR
200   DX=X-XOLD
      DY=Y-YOLD
      YSTEP=SIGN(5.0/TRFACY,DY)
      IF(ABS(DY*.75) .LT. ABS(YSTEP)) GO TO 100
      FRAC=DX/DY
      NSEG=(DY/YSTEP)+.9999
      YOUT=YOLD
300   IF(NSEG .LT. 2) GO TO 100
      YOUT=YOUT+YSTEP
      XOUT=XOLD+(YOUT-YOLD)*FRAC
      CALL DASHA(XOUT,YOUT,L)
      NSEG=NSEG-1
      GO TO 300
C * USER SEGMENTATION
400   CONTINUE
C      CALL USDASH(X,Y,L)
      GO TO 100
      END
```

Change 200 to 100

Remove

DRAWSA

```
C * LINEAR LOG POLAR USER ERROR
      GO TO (100, 100, 200, 400, 100),KEY
C * ERROR LINEAR LOG
100   CALL DRAWA(X,Y)
      XOLD=X
      YOLD=Y
      RETURN
C * POLAR
200   DX=X-XOLD
      DY=Y-YOLD
      YSTEP=SIGN(5.0/TRFACY,DY)
      IF(ABS(DY*.75) .LT. ABS(YSTEP)) GO TO 100
      FRAC=DX/DY
      NSEG=(DY/YSTEP)+.9999
      YOUT=YOLD
300   IF(NSEG .LT. 2) GO TO 100
      YOUT=YOUT+YSTEP
      XOUT=XOLD+(YOUT-YOLD)*FRAC
      CALL DRAWA(XOUT,YOUT)
      NSEG=NSEG-1
      GO TO 300
C * USER SEGMENTATION
400   CONTINUE
C      CALL USDRAW(X,Y)
      GO TO 100
      END
```

Change 200 to 100

Remove

RESCAL

```
C * BRANCH TO PROPER SECTION AND RETURN
C * LINEAR LOG POLAR USER ERROR
      GO TO (100,200,300,400,500),KEY
C * BOTH AXES LINEAR
100   TRPAR1=0.
C * SEMI LOG OR LOG LOG
200   KEYL=TRPAR1+1.001
```

Change 300 to 100

```
C * POLAR SCALING
300   CALL PSCAL
      GO TO 600
C * USER FUNCTION
400   CONTINUE
C      CALL URSCAL
      GO TO 600
```

Remove

WINCOT

```
C * BRANCH TO PROPER SECTION
C * LINEAR LOG POLAR USER ERROR
      GO TO(500,300,600,700,100),KEY
C       ERROR
100   IX=X
      IY=Y
      GO TO 800
```

Change 600 to 500

```
C * CONVERT LINEAR
500   IX=IFIX(DX*TRFACX+.5)+KMINSX
      IY=IFIX(DY*TRFACY+.5)+KMINSY
C * GO TO EXIT
      GO TO 800
C * POLAR TRANSFORMATION
600   A=(Y-TRPAR6)*TRFACY
      R=(X-TRPAR5)*TRFACX
      IX=R*COS(A*DE2RAD)+TRPAR3
      IY=R*SIN(A*DE2RAD)+TRPAR4
C * GO TO EXIT
      GO TO 800
C * USER TRANSFORMATION IN USE
700   CONTINUE
C      CALL USECOT(X,Y,IX,IY)
C * EXIT POINT
800   RETURN
      END
```

Remove

REVCOT

```
C *       LINEAR LOG POLAR USER ERROR
      GO TO(300, 400, 500, 600, 100  ),KEY
C * ERROR
100   X=IX
      Y=IY
      GO TO 700
```

Change 500 to 300

Remove

```
C * POLAR
500   DX=FLOAT(IX)*TRPAR3
      DY=FLOAT(IY)*TRPAR4
      Y=ATAN2(DY,DX)*57.2957795131
      X=SQRT(DY*DY+DX*DX)/TRFACX+TRPAR5
C * ADJUST ANGLE MOD 2 PI TO VALUE WITHIN WINDOW
      DEC=.FALSE.
510   IF(Y .GT. TRPAR1) GO TO 530
C * INCREMENT ANGLE
      Y=Y+360.0
      GO TO 510
530   IF(Y .LE. TRPAR2) GO TO 550
C * DECREMENT ANGLE
      Y=Y-360.0
      DEC=.TRUE.
      GO TO 530
550   IF(DEC .AND. Y .LT. TRPAR1)Y=Y+360.0
      IF(TMINVX .GE. 0.)GO TO 560
      TR1A=AMOD(TRPAR1+180.,360.)
      TR2A=AMOD(TRPAR2+180.,360.)
      IF(Y.GT.AMAX1(TR1A,TR2A).OR.Y.LT.AMIN1(TR1A,TR2A))GO TO 560
      Y=AMOD(Y+180.,360.)
      X=-X
560   Y=Y/TRFACY+TRPAR6
      GO TO 700
C * USER CONVERSION
600   CONTINUE
```

Feature 2. Logarithmic Transformations

Remove: LOGTRN

Modify: RESCAL
WINCOT
REVCOT

RESCAL

```
      C * BRANCH TO PROPER SECTION AND RETURN
      C * LINEAR LOG POLAR USER ERROR
            GO TO (100,200,300,400,500),KEY
      C * BOTH AXES LINEAR
      100   TRPAR1=0.
      C * SEMI LOG OR LOG LOG
      200   KEYL=TRPAR1+1.001
Remove C * X AXIS -- LINEAR OR LOG
            GO TO (210,215,210,215),KEYL
      C * LINEAR
      210   TRFACX=FLOAT(KMAXSX-KMINSX)/(TMAXVX-TMINVX)
            GO TO 250
      C * PREVENT INVALID TRANSFORMATION
      215   IF(TMINVX .GT. 0.0 .AND. TMAXVX .GT. 0.0)GO TO 220
Remove      KGNFLG=1
            TRPAR1=TRPAR1-1.0
            GO TO 210
      C * SEMI LOG X AXIS
      220   TRPAR2=ALOG(TMINVX)
            TRFACX=FLOAT(KMAXSX-KMINSX)/(ALOG(TMAXVX)-TRPAR2)
      C * Y AXIS -- LINEAR OR LOG
      250   GO TO (260,260,270,270),KEYL
      C * LINEAR
      260   TRFACY=FLOAT(KMAXSY-KMINSY)/(TMAXVY-TMINVY)
            GO TO 600
```

Change 200 to 100

WINCOT

```
C * BRANCH TO PROPER SECTION
C * LINEAR LOG POLAR USER ERROR
      GO TO(500,300,600,700,100),KEY
C       ERROR
100   IX=X
      IY=Y
      GO TO 800
C * LOG TRANSFORM
300   KEYL=TRPAR1+.001
      IF(KEYL .EQ. 2) GO TO 400
C * SETUP X LOG TRANSFORM
      DX=ALOG(X)*TRPAR2
400   IF(KEYL .EQ. 1) GO TO 500
C * SETUP Y LOG TRANSFORM
      DY=ALOG(Y)*TRPAR3
C * CONVERT LINEAR
500   IX=IFIX(DX*TRFACX+.5)+KMINSX
      IY=IFIX(DY*TRFACY+.5)+KMINSY
```

Change 300 to 500

Remove

REVCOT

```
C *       LINEAR LOG POLAR USER ERROR
      GO TO(300, 400, 500, 600, 100  ),KEY
C * ERROR
100   X=IX
      Y=IY
      GO TO 700
C * LINEAR
300   X=DX+TMINVX
      Y=DY+TMINVY
      GO TO 700
C * LOG SCALES
400   KEYL=TRPAR1
      X=DX+TMINVX
      Y=DY+TMINVY
      IF(KEYL .NE. 2)X=E**(DX+TRPAR2)
      IF(KEYL .NE. 1)Y=E**(DY+TRPAR3)
      GO TO 700
C * POLAR
500   DX=FLOAT(IX)*TRPAR3
```

Change 400 to 300

Remove

Feature 3. Multiple Character Sizes

Remove: CHRSIZ

Modify: RESET
RESTAT

RESET

```
      C * SET 4014 ENHANCED FOR SOLID LINES
            IF(KTERM .GE. 3)CALL CWSEND
      C * PLACE 4014 IN LARGE CHARACTER SIZE
Remove      IF(KTERM .GE. 2)CALL CHRSIZ(1)
      C * PLACE THE TERMINAL IN A/N MODE
            CALL ALFMOD
            RETURN
            END
```

RESTAT

```
      100   RB(I)=RARRAY(I)
            DO 101 I=1,41
      101   IB(I)=RARRAY(I+19)
      C * RESTORE CHARACTER SIZE
Remove      IF(KTERM .GT. 1)CALL CHRSIZ(KSIZEF)
      C * RESTORE ZAXIS AND DASH LINE
            IF(KTERM .GT. 2)CALL CWSEND
      C * CALL TO RECOVER POSITION AND MODE
            CALL RECOVR
            RETURN
            END
```

Feature 4. Point Plotting

Remove:	PNTABS	PNTREL	PNTMOD
	POINTA	POINTR	TKPNT

Modify: RECOVR BUFFPK

RECOVR

```
C * PLACE IN THE PROPER MODE
      IF(MODE .LT. 1)MODE=1
      IF(MODE .GT.5)MODE=5
      GO TO (100,200,120,100,200),MODE
100   CALL ALFMOD
      GO TO 200
120   CALL PNTMOD
C * RESTORE THE GRAPHIC LEVEL FLAG
200   KGRAFL=IFLAG
      RETURN
      END
```

Remove

Change 120 to 200

BUFFPK

```
C * MODE IS A/N,VEC,PNT,INC,DSH
      GO TO (21, 22, 23, 24, 22),KEY
C * ENTER A/N MODE
21    IDATA(LENOUT)=31
      GO TO 50
C * IF READY FOR A MOVE, THEN REMOVE FIXUP CHARS
22    IF(KMOVEF .EQ. 1) LENOUT=2
      LENOUT=LENOUT-1
C * CHECK IF DASHED LINE OR Z AXIS MUST BE RESTORED
      IF(KLINE .EQ. 0 .AND. KZAXIS .EQ. 0) GO TO 50
      IDATA(LENOUT+1)=27
      LENOUT=LENOUT+2
      IDATA(LENOUT)=96+KZAXIS*8+KLINE
      GO TO 50
C * ENTER POINT MODE
23    IF(KTERM .LT. 3)GO TO 22
      IDATA(LENOUT)=28
      LENOUT=LENOUT+1
      GO TO 22
C * ENTER INCREMENTAL PLOT MODE
24    IDATA(LENOUT)=30
```

Change 23 to 22

Remove

Feature 5. Hardware Point Plotting

Modify: TKPNT PNTMOD BUFFPK

TKPNT

Remove

```
C * THIS SECTION IS FOR 4014 ENHANCED
      IF(KTERM .GE. 3)GO TO 10
C ***************************************************
C * PUT OUT A GS FOR SIMULATED POINT PLOT MODE
      CALL TOUTPT(29)
      KMOVEF=1
```

PNTMOD

Remove

```
C * FOR HARDWARE POINT PLOT OUTPUT AN (FS)
      IF(KTERM .GE. 3)CALL TOUTPT(28)
      RETURN
      END
```

BUFFPK

```
C * MODE IS A/N,VEC,PNT,INC,DSH
      GO TO (21, 22, 23, 24, 22),KEY
C * ENTER A/N MODE
21    IDATA(LENOUT)=31
      GO TO 50
C * IF READY FOR A MOVE, THEN REMOVE FIXUP CHARS
22    IF(KMOVEF .EQ. 1) LENOUT=2
      LENOUT=LENOUT-1
C * CHECK IF DASHED LINE OR Z AXIS MUST BE RESTORED
      IF(KLINE .EQ. 0 .AND. KZAXIS .EQ. 0) GO TO 50
      IDATA(LENOUT+1)=27
      LENOUT=LENOUT+2
      IDATA(LENOUT)=96+KZAXIS*8+KLINE
      GO TO 50
C * ENTER POINT MODE
```

Change 23 to 22

Remove

```
23    IF(KTERM .LT. 3)GO TO 22
      IDATA(LENOUT)=28
      LENOUT=LENOUT+1
      GO TO 22
C * ENTER INCREMENTAL PLOT MODE
24    IDATA(LENOUT)=30
```

Feature 6. Z-Axis Mode Changes AND Hardware Dashed Lines

Remove:	CWSEND	CZAXIS		
Modify:	RESET	CARTN*	DRAWA**	TKDASH**
	RESTAT	LINEF*	DRAWABS**	

RESET

Remove

```
C * SET 4014 ENHANCED FOR SOLID LINES
      IF(KTERM .GE. 3)CALL CWSEND
C * PLACE 4014 IN LARGE CHARACTER SIZE
      IF(KTERM .GE. 2)CALL CHRSIZ(1)
C * PLACE THE TERMINAL IN A/N MODE
      CALL ALFMOD
      RETURN
      END
```

RESTAT

Remove

```
C * RESTORE ZAXIS AND DASH LINE
      IF(KTERM .GT. 2)CALL CWSEND
C * CALL TO RECOVER POSITION AND MODE
      CALL RECOVR
      RETURN
      END
```

*Described in Feature 7

**Described in Feature 8

Feature 7. Z-Axis Mode Changes

Remove: CZAXIS*

Modify: CARTN LINEF CWSEND*

CARTN

```
Remove    C * RESTORE ZAXIS STATE ON 4014
          C     IF(KTERM .GE. 2)CALL CWSEND
          30    RETURN
                END
```

LINEF

```
Remove    C * RESTORE ZAXIS MODE IF APPROPRIATE
          200   IF(KTERM .GE. 2)CALL CWSEND
                RETURN
                END
```

CWSEND*

```
                DIMENSION ICODE(2)
                DATA ICODE(1)/27/
Delete          ICODE(2)=96+KZAXIS*8+KLINE
KZAXIS*8        CALL TOUTST(2,ICODE)
                RETURN
                END
```

*Not required if Feature 6 has been eliminated.

Feature 8. Hardware Dashed Lines

Modify: DRAWA DRWABS TKDASH CWSEND*

DRAWA

Remove

```
C * SET TERMINAL TO DRAW SOLID LINES IF NEEDED
C * THIS SECTION IS NEEDED FOR 4014 ENHANCED ****
      IF(KLINE .EQ. 0)GO TO 5
      KLINE=0
      CALL CWSEND
5     CONTINUE
C ************************************************
```

DRWABS

Remove

```
C * THIS SECTION IS FOR 4014 ENHANCED *****
      IF(KLINE .EQ. 0)GO TO 5
      KLINE=0
      CALL CWSEND
5     CONTINUE
C ******************************************
```

TKDASH

Remove

```
C * THIS SECTION IS FOR 4014 ENHANCED ***************
      IF(KDASHT .GT. 4)GO TO 101
      IF(KTERM .GE. 3)GO TO 103
      IF(KDASHT .EQ. 0)GO TO 330
C ********************************************************
C * HARDWARE DASH SIMULATION FOR TYPE 1 + 2 TERMINALS
      KDASHT=ISIMHD(KDASHT)
C * THIS SECTION IS ALSO FOR 4014 ENHANCED **********
101   IF(KLINE .EQ. 0)GO TO 104
      KLINE=0
      CALL CWSEND
      GO TO 104
C * SET AND TRANSMIT HARDWARE DASH CODE
103   IF(KLINE .EQ. KDASHT)GO TO 330
      KLINE=KDASHT
      CALL CWSEND
      GO TO 330
104   CONTINUE
C ********************************************************
```

CWSEND*

Delete +KLINE

```
      DATA ICODE(1)/27/
      ICODE(2)=96+KZAXIS*8+KLINE
      CALL TOUTST(2,ICODE)
      RETURN
      END
```

*Not required if Feature 6 has been eliminated.

Feature 9. Interline Character Effect Suppression

Remove: SETBUF

Modify: INITT BUFFPK

INITT

```
C * SET THE OUTPUT BUFFER FORMAT
      KUNIT=1
      KINLFT=0
      KOTLFT=1
      CALL RESET
      CALL NEWPAG
      RETURN
      END
```

Change 1 to 3 or 4

BUFFPK

```
10    IF(NODATA .EQ. 1)GO TO 50
      NODATA=1
C * DETERMINE THE FORMAT THE USER WANTS BUFFER DUMPED IN
      GO TO (20,30,40,45),KUNIT
C * OUTPUT BUFFER FORMAT IS (GS),PLTCHRS,DATA,(US)
20    LENOUT=LENOUT+1
C * APPEND (US) TO END OF BUFFER
      IDATA(LENOUT)=31
      CALL ADEOUT(LENOUT,IDATA)
C * RESTORE THE BEAM POSITION AT FIRST OF THE NEXT BUFFER
      ISUB=1
      IF(KTERM .GE. 3) ISUB=2
      CALL PLTCHR(KBEAMX,KBEAMY,IDATA(ISUB))
      IDATA(2)=IDATA(ISUB)
      LENOUT=5+ISUB
      IDATA(1)=29
C * AND NOW THE MODE BEFORE THE OUTPUT WAS ASKED FOR
      DO 19 I=2,KPAD2
      IDATA(LENOUT)=22
19    LENOUT=LENOUT+1
      KEY=KKMODE+1
      IF(KEY .LT. 1)KEY=1
      IF(KEY .GT. 5)KEY=1
C * MODE IS A/N,VEC,PNT,INC,DSH
      GO TO (21, 22, 23, 24, 22),KEY
C * ENTER A/N MODE
21    IDATA(LENOUT)=31
      GO TO 50
C * IF READY FOR A MOVE, THEN REMOVE FIXUP CHARS
22    IF(KMOVEF .EQ. 1) LENOUT=2
      LENOUT=LENOUT-1
C * CHECK IF DASHED LINE OR Z AXIS MUST BE RESTORED
      IF(KLINE .EQ. 0 .AND. KZAXIS .EQ. 0) GO TO 50
      IDATA(LENOUT+1)=27
      LENOUT=LENOUT+2
      IDATA(LENOUT)=96+KZAXIS*8+KLINE
      GO TO 50
C * ENTER POINT MODE
```

Remove

Remove

```
23      IF(KTERM .LT. 3)GO TO 22
        IDATA(LENOUT)=28
        LENOUT=LENOUT+1
        GO TO 22
C * ENTER INCREMENTAL PLOT MODE
24      IDATA(LENOUT)=30
C * RAISE OR LOWER PEN AS NEEDED
C * THE FOLLOWING 3 LINES ARE NOT NEEDED ON SOME PLOTTERS *****
        LENOUT=LENOUT+1
        IDATA(LENOUT)=80
        IF(KMOVEF .EQ. 1)IDATA(LENOUT)=32
C ***************************************************************



        GO TO 50
C * OUTPUT BUFFER FORMAT IS (SYN),DATA,(ESC)
30      IF(NCHAR .LE. 3 .AND. KGNMOD .NE. 1)GO TO 25
        LENOUT=LENOUT+1
C * APPEND (ESC) TO END OF BUFFER
        IDATA(LENOUT)=27
        CALL ADECUT(LENOUT,IDATA)
        IDATA(1)=22
        LENOUT=1
        GO TO 50
C * OUTPUT BUFFER FORMAT IS DATA ONLY
40      CALL ADECUT(LENOUT,IDATA)
```

Feature 10. Software Dashed Lines*

Modify: TKDASH

Remove *all but* the following lines:

```
      SUBROUTINE TKDASH(IX,IY)
      COMMON /TKTRNX/ TMINVX,TMINVY,TMAXVX,TMAXVY,TREALX,TREALY,
     1 TIMAGX,TIMAGY,TRCOSF,TRSINF,TRSCAL,TRFACX,TRFACY,
     2 TRPAR1,TRPAR2,TRPAR3,TRPAR4,TRPAR5,TRPAR6,KMOFLG(8),KPAD2,
     3 KBAUDR,KERROR,KGRAFL,KHOMEY,KKMODE,KHORSZ,KVERSZ,KTBLSZ,
     4 KSIZEF,KLMRGN,KRMRGN,KFACTR,KTERM,KLINE,KZAXIS,KBEAMX,KBEAMY,
     5 KMOVEF,KPCHAR(5),KDASHT,KMINSX,KMINSY,KMAXSX,KMAXSY,KEYCON,
     6 KINLFT,KOTLFT,KUNIT

        IF(KDASHT .LT. 0)GO TO 320

        IF(KDASHT .GT. 4)GO TO 101
        IF(KTERM .GE. 3)GO TO 103

C * THIS SECTION IS ALSO FOR 4014 ENHANCED **********
101     IF(KLINE .EQ. 0)GO TO 104
        KLINE=0
        CALL CWSEND
        GO TO 104
C * SET AND TRANSMIT HARDWARE DASH CODE
103     IF(KLINE .EQ. KDASHT)GO TO 330
        KLINE=KDASHT
        CALL CWSEND
        GO TO 330


320     CALL TOUTPT(29)
        KMOVEF=1
330     CALL XYCNVT(IX,IY)


340     RETURN
        END
```

Change 104 to 330

Change 104 to 330

*If you have the four hardware dash types of the Enhanced Graphics Module, you may wish to eliminate software dashes to save storage.

SECTION 5 STATUS VARIABLES

5.1. Description of Variables

The Terminal Status Area is the common area named /TKTRNX/. It allows routines in the Terminal Control System a quick reference to the current condition of both the software and the Terminal. This reduces the number of control characters and routine linkages necessary to place the Terminal in the user requested condition. The following Status Variables comprise the Terminal Status Area:

KACHAR — The number of characters available to the user in the input buffer.

KBAUDR — The number of characters transmitted per second.

KBEAMX — The beam X coordinate.

KBEAMY — The beam Y coordinate.

KDASHT — User requested dashed line type:

- 1 through 4 — hardware dash or software-simulated hardware dash
- 10 or greater — a software dash

KEYCON — The transformation type:

1. linear
2. logarithmic
3. polar
4. user-defined

KFACTR — The addressing factor:

1. 4096 addressable points
4. 1024 addressable points

KGNFLG — The general condition flag:

Routine	Meaning of KGNFLG Value	
	0 Action Completed	1 Action Cannot Be Completed
SETTAB	all OK	no room in tab table
PCLIPT	point inside virtual window limits	point outside virtual window limits
CLIPT	line partly inside window limits	line entirely outside window limits
RESCAL	valid transformation	requested transformation has a negative window limit

KGNMOD — The graphic crosshair cursor flag:

0. not set
1. set

KGRAFL — The graphic level flag:

0. screen level graphics
1. virtual graphics

KHOMEY — The home Y value:
- 767 in 1024 point system
- 3068 in 4096 point system

KHORSZ — The character width in 4096-space raster units.

KINLFT — The number of characters left in the TCS input buffer.

KKMODE — The Terminal mode for both hardware and software:
- 0 alphanumeric mode
- 1 solid vector mode
- 2 point plot mode
- 3 incremental plot mode
- 4 dashed vector mode

KLEVEL — Not used.

KLINE — The hardware dashed line type:
- 0 solid line
- 1 dotted line
- 2 dash-dot line
- 3 short-dashed line
- 4 long-dashed line

KLMRGN — The left margin for alphanumeric operations.

KMAXSX — The screen window maximum X value.

KMAXSY — The screen window maximum Y value.

KMINSY — The screen window minimum X value.

KMINSY — The screen window minimum Y value.

KMOFLG (1,2) — Not used

KMOVEF — The move flag:
- 0 bright vector next
- 1 [ready for] dark vector next

KOBLEN — The output buffer length.

KOTLFT — The number of available spaces left in the TCS output buffer.

KPAD2 — The minimum number of pad characters needed to allow enough time for the Terminal to perform mode changes or draw vectors.

KPADV — The number of pad characters needed to allow for next vector.

KPCHAR — The last set of plot characters; used in drawing a vector.

KRMRGN — The right margin for alphanumeric operations.

KSIZEF — The character size in effect:

Value	Characters/Line	Number of Lines
1	74	35
2	81	38
3	121	58
4	133	64

KTBLSZ — The tab table size.

KTERM — The type of Terminal in use:

1 4006-1 Releases 2.0 through 3.3 require modification to BAKSP 4010, 4012, 4013
2 4014, 4015, 4014EGM or 4015EGM
3 4014 EGM or 4015EGM only

KTRAIL — The number of system character positions needed at the end of output buffer for interline characters.

KUNIT — The output buffer format (see SETBUF, User's Manual, Section 7.11.1.).

KVERSZ — The height of a character in 4096-space raster units.

KZAXIS — The Z-Axis mode type:

0 normal
1 defocused
2 write-through

TIMAGX, TIMAGY — The position of the imaginary beam anywhere in virtual space (may be outside the virtual window and screen limits).

TMAXVX, TMAXVY, TMINVX, TMINVY — The virtual window limits; used for clipping routines.

TRCOSF — The cosine for the relative virtual vector rotation.

TREALX, TREALY — The position of the real beam in virtual coordinates (must be inside the virtual window).

TRFACX, TRFACY — The scale factors used in converting virtual to screen coordinates.

TRSCAL — The scale factor used in converting relative virtual to absolute virtual coordinates.

TRSINF — The sine for relative virtual vector rotation.

Logarithmic Transformation

TRPAR1 — The axis type in effect:

Value	X Axis	Y Axis
0	linear	linear
1	log	linear
2	linear	log
3	log	log

TRPAR2 — The log of minimum virtual X.

TRPAR3 — The log of minimum virtual Y.

Polar Transformation

TRPAR1 — The beginning screen angle.

TRPAR2 — The ending screen angle.

TRPAR3 — The screen X coordinate of the virtual origin.

TRPAR4 — The screen Y coordinate of the virtual origin.

TRPAR5 — The radius suppression sum; to be subtracted from the virtual radius (X coordinate) before transformation.

TRPAR6 — The virtual screen angle offset.

5.2. Status Variable Setting and Reference Charts

5.2.1. Variables

Status Variable	Initial Setting (INITT)	Set By	Referenced By
KACHAR	*	SETBUF	ANSTR TOUTST
KBAUDR	*	INITT	INITT IOWAIT SEETRM
KBEAMX	0	ANSTR ANCHO BAKSP CARTN INCPLT NEWPAG RESET XYCNVT	ANCHO ANSTR BAKSP BUFFPK DRAWA DRWABS DRWREL DSHMOD DSHREL INCPLT LINEF LVLCHT MOVREL PLTCHR PNTREL SEELOC TABHOR TABVER TKDASH RECOVR
KBEAMY	767	ALFMOD INCPLT LINEF NEWPAG RESET XYCNVT	ALFMOD BUFFPK CARTN DRAWA DRWABS DRWREL DSHMOD DSHREL INCPLT LINEF LVLCHT MOVREL BAKSP

*Depends on the Baud rate entered.

Status Variable	Initial Setting (INITT)	Set By	Referenced By
			PLTCHR PNTREL RECOVR SEELOC TABHOR TABVER TKDASH
KDASHT		DSHMOD TKDASH	TKDASH
KEYCON	1	LINTRN LOGTRN POLTRN RESET	DRAWSA DASHSA RESCAL REVCOT SEETRN WINCOT
KFACTR	4	INITT TERM	ANCHO ANSTR BAKSP CSIZE KCM KIN LINEF LINHGT LINWDT PLTCHR RESET SCURSR SEETRM
KGNFLG	0	CLIPT PCLIPT RESCAL SETTAB	DASHA DRAWA GENFLG MOVEA POINTA V2ST
KGNMOD	0	INITT SCURSR	BUFFPK
KGRAFL	0	ALFMOD DRWABS DSHABS IPMOD LVLCHT MOVABS	LVLCHT RECOVR

Status Variable	Initial Setting (INITT)	Set By	Referenced By
		PNTABS PSCAL RECOVR RESCAL	
KHOMEY	767	RESET	ALFMOD ANSTR HOME LINEF NEWPAG RESET
KHORSZ	56	CHRSIZ RESET	ANCHO ANSTR BAKSP CSIZE LINWDT
KINLFT	0	INITT SCURSR TINSTR	LEFTIO
KKMODE	0	ALFMOD DSHMOD IPMOD PNTMOD V2ST VECMOD	ANCHO ANSTR BAKSP BUFFPK CARTN DRAWA DRWABS DSHMOD INCPLT LINEF NEWPAG POINTA PNTABS RECOVR SEEMOD VECMOD V2ST
KLINE	0	DRAWA DRWABS RESET TKDASH	CWSEND DRAWA DRWABS SEEMOD TKDASH

Status Variable	Initial Setting (INITT)	Set By	Referenced By
KLMRGN	0	RESET SETMRG	ANSTR CARTN HOME NEWPAG RESET SEEMRG
KMAXSX KMAXSY	1023 780	RESET SWINDO TWINDO	PSCAL RESCAL SEETW
KMINSX KMINSY	0 0	RESET SWINDO TWINDO	PSCAL RESCAL REVCOT SEETW WINCOT
KMOFLG			SUSTAT RESTAT
KMOVEF	0	BELL DSHMOD INCPLT RECOVR TKDASH TKPNT VECMOD XYCNVT	BUFFPK DRAWA DRWABS XYCNVT
KOBLEN	72	INITT	SETBUF
KOTLFT	*	BUFFPK INITT	BUFFPK LEFTIO
KPAD2	*	INITT	BUFFPK PLTCHR SETBUF
KPADV		BUFFPK INITT PLTCHR	BUFFPK
KPCHAR	55,0,127,32,64	DSHMOD PNTMOD VECMOD XYCNVT	XYCNVT
KRMRGN	1022	RESET SETMRG	ANCHO ANSTR SEEMRG TABHOR

*Depends on the Baud rate entered.

Status Variable	Initial Setting (INITT)	Set By	Referenced By
KSIZEF	1	RESET CHRISZ	SEETRM RESTAT
KTBLSZ	10	RESET TTBLSZ	RSTTAB SETTAB TABHOR TABVER
KTERM	1	INITT TERM	BUFFPK CARTN CHRSIZ CZAXIS KCM KIN LINEF PNTMOD RESET SEETRM TKDASH TKPNT XYCNVT RECOVR
KTRAIL**	1	SETBUF	BUFFPK
KUNIT	1	INITT SETBUF	BUFFPK SEEBUF
KVERSZ	88	CHRISZ RESET	ANSTR CSIZE LINEF LINHGT
KZAXIS	0	CZAXIS RESET	BUFFPK CWSEND CZAXIS SEEMOD
TIMAGX TIMAGY		LVLCHT V2ST	DASHSA DRAWSA REL2AB V2ST
TMAXVX TMAXVY	1023 780	DWINDO RESET VWINDO	CLIPT PCLIPT PSCAL RESCAL SEEDW
TIMVX TIMNVY	0 0	DWINDO RESET	CLIPT PCLIPT

**Not used in the TSO version of TCS.

Status Variable	Initial Setting (INITT)	Set By	Referenced By
		VWINDO	PSCAL RESCAL REVCOT RESTAT SEEDW SVSTAT WINCOT
TRCOSF	1	RESET RROTAT	REL2AB SEEREL
TREALX TREALY		V2ST	LVLCHT V2ST
TRFACX TRFACY	1 1	PSCAL RESCAL RESET	DASHSA(TRFACY only) DRAWSA(TRFACY only) PSCAL REVCOT SEETRN WINCOT
TRSCAL	1	RESET RSCALE	REL2AB SEEREL
TRSINF	0	RESET RROTAT	REL2AB SEEREL
TRPAR1		LOGTRN POLTRN RESCAL	PSCAL RESCAL REVCOT WINCOT
TRPAR2		POLTRN RESCAL	PSCAL RESCAL REVCOT WINCOT
TRPAR3		PSCAL RESCAL	RESCAL REVCOT WINCOT
TRPAR4		PSCAL	REVCOT WINCOT
TRPAR5		POLTRN	REVCOT WINCOT
TRPAR6		PSCAL	REVCOT WINCOT
ALL COMMON VARIABLES		SVSTAT	RESTAT

5.2.2 Routines Which Set and Reference Variables

Routine	Sets	References
ALFMOD	KBEAMY KGRFL KKMODE	KBEAMY KHOMEY
ANCHO	KBEAMX	KKMODE KBEAMX KHORSZ KFACTR KRMRGN
ANSTR	KBEAMX KBEAMY	KACHAR* KKMODE KBEAMX KBEAMY KFACTR KLMRGN KRMRGN KHORSZ KVERSZ
BAKSP	KBEAMX	KBEAMX KHORSZ KFACTR KKMODE KTERM KBEAMY
BELL	KMOVEF	
BUFFPK**	KOTLFT KPADV	KOTLFT KUNIT KBEAMX KBEAMY KKMODE KMOVEF KTERM KGNMOD KLINE KPAD2 KPADV KTRAIL KZAXIS

*Not used in the TSO version of TCS.
**Not present in PDP-11 version of TCS.

Routine	Sets	References
CARTN	KBEAMX	KLMRGN KBEAMY KKMODE KTERM
CHRSIZ	KHORSZ KSIZEF KVERSZ	KTERM
CLIPT	KGNFLG	TMAXVX TMAXVY TMINVX TMINVY
CSIZE		KHORSZ KVERSZ KFACTR
CWSEND		KZAXIS KLINE
CZAXIS	KZAXIS	KTERM
DASHA		KGNFLG
DASHSA		TIMAGX TIMAGY KEYCON TRFACY
DRAWA	KLINE	KLINE KGNFLG KKMODE KMOVEF KBEAMX KBEAMY
DRAWSA		KEYCON TIMAGX TIMAGY TRFACY
DRWABS	KLINE KGRAFL	KLINE KKMODE KMOVEF KBEAMX KBEAMY
DRWREL		KBEAMX KBEAMY
DSHABS	KGRAFL	

Routine	Sets	References
DSHMOD	KKMODE	KKMODE
	KMOVEF	KBEAMX
	KDASHT	KBEAMY
	KPCHAR	
DSHREL		KBEAMX
		KBEAMY
DWINDO	TMAXVX	
	TMAXVY	
	TMINVX	
	TMINVY	
GENFLG		KGNFLG
HOME		KHOMEY
		KLMRGN
INCPLT	KMOVEF	KKMODE
	KBEAMX	KBEAMX
	KBEAMY	KBEAMY
INITT	KBAUDR	KBAUDR
	KGNMOD	
	KOBLEN*	
	KPAD2	
	KPADV	
	KTERM	
	KFACTR	
	KINLFT	
	KOTLFT	
IOWAIT		KBAUDR
IPMOD	KKMODE	
	KGRAFL	
KCM		KFACTR
		KTERM
KIN		KFACTR
		KTERM
LEFTIO		KINLFT
		KOTLFT
LINEF	KBEAMY	KKMODE
		KBEAMY
		KVERSZ
		KFACTR
		KHOMEY
		KBEAMX
		KTERM

*Not used in the TSO version of TCS.

Routine	Sets	References
LINHGT		KFACTR KVERSZ
LINTRN	KEYCON	
LINWDT		KFACTR KHORSZ
LOGTRN	KEYCON TRPAR1	
LVLCHT	TIMAGX TIMAGY KGRAFL	KGRAFL KBEAMX KBEAMY TREALX TREALY
MOVABS	KGRAFL	
MOVEA		KGNFLG
MOVREL		KBEAMX KBEAMY
NEWPAG	KBEAMX KBEAMY	KKMODE KLMRGN KHOMEY
PCLIPT	KGNFLG	TMAXVX TMAXVY TMINVX TMINVY
PLTCHR	KPADV	KBAUDR KBEAMX KBEAMY KFACTR KPAD2 KTERM
PNTABS	KGRAFL	KKMODE
PNTMOD	KKMODE KPCHAR	KTERM
PNTREL		KBEAMX KBEAMY
POINTA		KGNFLG KKMODE
POLTRN	KEYCON TRPAR1 TRPAR2 TRPAR5	

Routine	Sets	References
PSCAL	KGRAFL	TMAXVX
	KTRFACX	TMAXVY
	KTRFACY	TMINVX
	TRPAR3	TMINVY
	TRPAR4	TRFACX
	TRPAR6	TRFACY
		TRPAR1
		TRPAR2
		KMINSX
		KMINSY
		KMAXSX
		KMAXSY
RECOVR	KMOVEF	KGRAFL
	KGRAFL	KKMODE
		KBEAMS
		KBEAMY
		KTERM
REL2AB		TRCOSF
		TRSINF
		TRSCAL
		TIMAGX
		TIMAGY
RESCAL	KGRAFL	KEYCON
	KGNFLG	KMAXSX
	TRPAR1	KMAXSY
	TRFACX	KMINSX
	TRFACY	KMINSY
	TRPAR2	TMINVX
	TRPAR3	TMINVY
		TMAXVX
		TMAXVY
		TRPAR1
		TRPAR2
		TRPAR3
RESET	KEYCON	KFACTR
	TRFACX	KHOMEY
	TRFACY	KLMRGN
	KBEAMX	KTERM
	KBEAMY	KMAXSX
	KHOMEY	KMAXSY
	KMINSX	
	KMAXSX	
	KMINSY	
	KMAXSY	
	KHORSZ	

Routine	Sets	References
	KLINE	
	KZAXIS	
	KLMRGN	
	KRMRGN	
	KSIZEF	
	KTBLSZ	
	KVERSZ	
	TMINVX	
	TMINVY	
	TMAXVX	
	TMAXVY	
	TRCOSF	
	TRSINF	
	TRSCAL	
RESTAT		ALL COMMON VARIABLES
REVCOT		KMINSX
		KMINSY
		TRFACX
		TRFACY
		KEYCON
		TMINVX
		TMINVY
		TRPAR1
		TRPAR2
		TRPAR3
		TRPAR4
		TRPAR5
		TRPAR6
RROTAT	TRSINF	
	TRCOSF	
RSCALE	TRSCAL	
RSTTAB	KTBLSZ	
SCURSR	KGNMOD	KFACTR
	KINLFT	KTERM
SEEBUF		KUNIT
SEEDW		TMAXVX
		TMAXVY
		TMINVX
		TMINVY
SEELOC		KBEAMX
		KBEAMY

Routine	Sets	References
SEEMOD		KLINE KZAXIS KKMODE
SEEMRG		KLMRGN KRMRGN
SEEREL		TRCOSF TRSINF TRSCAL
SEETRM		KBAUDR KTERM KSIZEF KFACTR
SEETRN		TRFACX TRFACY KEYCON
SEETW		KMAXSX KMAXSY KMINSX KMINSY
SETBUF	KACHAR* KTRAIL* KUNIT	KUNIT KOBLEN* KPAD2
SETMRG	KLMRGN KRMRGN	
SETTAB	KGNFLG	KTBLSZ
SVSTAT	ALL COMMON VARIABLES	
SWINDO	KMAXSX KMAXSY KMINSX KMINSY	
TABHOR		KTBLSZ KBEAMX KBEAMY KRMRGN
TABVER		KTBLSZ KBEAMX KBEAMY

*Not used in the TSO version of TCS.

Routine	Sets	References
TERM	KTERM	
	KFACTR	
TINSTR	KINLFT	KINLFT
TKDASH	KDASHT	KDASHT
	KLINE	KLINE
	KMOVEF	KTERM
		KBEAMX
		KBEAMY
TKPNT	KMOVEF	KTERM
TOUTST		KACHAR
TTBLSZ	KTBLSZ	
TWINDO	KMINSX	
	KMINSY	
	KMAXSX	
	KMAXSY	
V2ST	KKMODE	TIMAGX
	TREALX	TIMAGY
	TREALY	KGNFLG
	TIMAGX	TREALX
	TIMAGY	TREALY
		KKMODE
VECMOD	KKMODE	KKMODE
	KMOVEF	
	KPCHAR	
VWINDO	TMAXVX	
	TMAXVY	
	TMINVX	
	TMINVY	
WINCOT		TMINVX
		TMINVY
		KEYCON
		KMINSX
		KMINSY
		TRPAR1
		TRPAR2
		TRPAR3
		TRPAR4
		TRPAR5
		TRPAR6
		TRFACX
		TRFACY
XYCNVT	KPCHAR	KPCHAR
	KMOVEF	KTERM
	KBEAMX	KMOVEF
	KBEAMY	

SECTION 6 SUBROUTINE CALLING REFERENCE CHARTS

6.1. TCS Routines

Routine	Arguments	Called By	Calls
A1IN	NCHAR, IARRAY		KAS2A1 TINSTR
A1OUT	NCHAR, IARRAY		ANSTR KA12AS
ADEIN	NCHAR, IARRAY	TINSTR	
ADEOUT	NCHAR, IARRAY	BUFFPK	
AINST	NCHAR, IARRAY		KAS2AM TINSTR
ALFMOD		ANCHO ANMODE ANSTR BAKSP CARTN FINITT HOME LINEF NEWPAG RECOVR RESET TABHOR TABVER	TOUTPT
ANCHO	ICHAR		ALFMOD NEWLIN TOUTPT
ANMODE			ALFMOD TSEND
ANSTR	NCHAR, IARRAY	A1OUT AOUTST	ALFMOD NEWLIN TOUTST
AOUTST	NCHAR, IARRAY		ANSTR KAM2AS
BAKSP			ALFMOD TOUTPT MOVABS
BELL			IOWAIT TOUTPT
BUFFPK	NCHAR, IARRAY	TOUTST TSEND	ADEOUT PLTCHR

Routine	Arguments	Called By	Calls
CARTN		NEWLIN	ALFMOD CWSEND MOVABS TOUTPT
CHRSIZ	ICODE	RESET RESTAT	TOUTST
CLIPT	BUFIN, BUFOUT	V2ST	PARCLT
CSIZE	IHORZ, IVERT		
CWSEND		CARTN CZAXIS DRAWA DRWABS LINEF RESET RESTAT TKDASH RECOVR	TOUTST
CZAXIS	ICODE		CWSEND
DASHA	X, Y, L	DASHR DASHSA	DSHMOD LVLCHT TKDASH V2ST
DASHR	X, Y, L		DASHA REL2AB
DASHSA	X, Y, L	DASHSR	DASHA LVLCHT
DASHSR	X, Y, L		DASHSA REL2AB
DCURSR	ICHAR, IX, IY		SCURSR
DRAWA	X, Y	DRAWR DRAWSA	CWSEND LVLCHT V2ST VECMOD XYCNVT
DRAWR	X, Y		DRAWA REL2AB
DRAWSA	X, Y	DRAWSR	DRAWA LVLCHT
DRAWSR	X, Y		DRAWSA REL2AB

Routine	Arguments	Called By	Calls
DRWABS	IX, IY	DRWREL	CWSEND VECMOD XYCNVT
DRWREL	IX, IY		DRWABS
DSHABS	IX, IY, L	DSHREL	DSHMOD TKDASH
DSHMOD	L	DASHA DSHABS	TOUTPT XYCNVT
DSHREL	IX, IY, L		DSHABS
DWINDO	XMIN, XMAX, YMIN, YMAX		RESCAL
ERASE			IOWAIT RECOVR TOUTST
FINITT	IX, IY		ALFMOD MOVABS TSEND
GENFLG	ITEM		
HDCOPY			IOWAIT TOUTST
HOME			ALFMOD MOVABS
INCPLT	IONOFF, IDIR, NO		IPMOD TOUTPT
INITT	IBAUD		NEWPAG RESET SETBUF
IOWAIT	ITIME	ERASE BELL HDCOPY NEWPAG	TOUTPT
IPMOD		INCPLT	TOUTST
KA12AS	NCHAR, KA1, KADE	A1OUT	
KAM2AS	NCHAR, KAM, KADE	AOUTST	
KAS2A1	NCHAR, KADE, KA1	A1IN	
KAS2AM	NCHAR, KADE, KAM	AINST	
KCM	RCM		
KIN	RIN		
LEFTIO	IOBUFF		

Routine	Arguments	Called By	Calls
LINEF		NEWLIN	ALFMOD CWSEND MOVABS TOUTPT
LINHGT	NUMLIN		
LINTRN			RESCAL
LINWDT	NUMCHR		
LOGTRN	KEY		RESCAL
LVLCHT		DASHA DASHSA DRAWA DRAWSA MOVEA POINTA REL2AB	REVCOT
MOVABS	IX, IY	CARTN FINITT HOME LINEF MOVREL BAKSP NEWPAG RECOVR RESET TABHOR TABVER	VECMOD XYCNVT
MOVEA	X, Y	MOVER	LVLCHT V2ST VECMOD XYCNVT
MOVER	X, Y		MOVEA REL2AB
MOVREL	IX, IY		MOVABS
NEWLIN		ANCHO ANSTR TABHOR	CARTN LINEF
NEWPAG		INITT	ALFMOD IOWAIT MOVABS TOUTST
PARCLT	RL1, RL2, RM1, RM2, RN1, RN2	CLIPT	

Routine	Arguments	Called By	Calls
PCLIPT	X, Y	REVCOT V2ST	
PLTCHR	IX, IY, ICHAR	BUFFPK XYCNVT	
PNTABS	IX, IY	PNTREL	PNTMOD TKPNT
PNTMOD		PNTABS POINTA RECOVR	TOUTPT
PNTREL	IX, IY		PNTABS
POINTA	X, Y	POINTR	LVLCHT PNTMOD TKPNT V2ST
POINTR	X, Y		POINTA REL2AB
POLTRN	ANGMIN, ANGMAX, RSUPRS		PSCAL
PSCAL		PLTRN RESCAL	WINCOT
RECOVR		ERASE RESTAT SCURSR	ALFMOD MOVABS PNTMOD CWSEND
REL2AB	XIN, YIN, XOUT, YOUT	DASHR DASHSR DRAWR DRAWSR MOVER POINTR	LVLCHT
RESCAL		DWINDO LINTRN LOGTRN SWINDO TWINDO VWINDO	PSCAL
RESET		INITT TERM	ALFMOD CHRSIZ CWSEND MOVABS

Routine	Arguments	Called By	Calls
RESTAT	RARRAY		CHRSIZ CWSEND RECOVR
REVCOT	IX, IY, X, Y	LVLCHT VCURSR	PCLIPT
RROTAT	DEG		
RSCALE	FACTOR		
RSTTAB	ITAB, ITABLE		
SCURSR		DCURSR VCURSR	RECOVR TINSTR TOUTST
SEEBUF	KFORM		
SEEDW	XMIN, XMAX, YMIN, YMAX		
SEELOC	IX, IY		
SEEMOD	LINE, IZAXIS, MODE		
SEEMRG	MLEFT, MRIGHT		
SEEREL	RCOS, RSIN, SCALE		
SEETRM	ISPEED, ITERM, KHRSIZ, MAXADR		
SEETRN	XFAC, YFAC, KEY		
SEETW	MINX, MAXX, MINY, MAXY		
SETBUF	KFORM	INITT	
SETMRG	MLEFT, MRIGHT		
SETTAB	ITAB, ITABLE		
SVSTAT	RARRAY		
SWINDO	MINX, LENX, MINY, LENY		RESCAL
TABHOR	ITABLE		ALFMOD MOVABS NEWLIN
TABVER	ITABLE		ALFMOD MOVABS
TCSLEV	LEVEL		
TERM	ITERM, MAXADR		RESET
TINPUT	ICHAR		TINSTR
TINSTR	NCHAR, IARRAY	A1IN AINST SCURSR TINPUT	ADEIN TSEND
TKDASH	IX, IY	DASHA DSHABS	CWSEND TOUTPT XYCNVT

Routine	Arguments	Called By	Calls
TKPNT	IX, IY	PNTABS POINTA	TOUTPT XYCNVT
TOUTPT	ICHAR	ANCHO ALFMOD BAKSP BELL CARTN DSHMOD INCPLT IOWAIT LINEF PNTMOD TKDASH TKPNT VECMOD	TOUTST
TOUTST	NCHAR, IARRAY	ANSTR CHRSIZ CWSEND ERASE HDCOPY IPMOD NEWPAG SCURSR TOUTPT XYCNVT	BUFFPK
TSEND		ANMODE FINITT TINSTR	BUFFPK
TTBLSZ	ITBLSZ		
TWINDO	MINX, MAXX, MINY, MAXY		RESCAL
V2ST	I, X, Y, IX, IY	DASHA DRAWA MOVEA POINTA	CLIPT PCLIPT VECMOD WINCOT XYCNVT
VCURSR	ICHAR, X, Y		SCURSR REVCOT
VECMOD		DRAWA DRWABS MOVABS MOVEA V2ST	TOUTPT

Routine	Arguments	Called By	Calls
VWINDO	XMIN, XRANGE, YMIN, YRANGE		RESCAL
WINCOT	X, Y, IX, IY	PSCAL V2ST	
XYCNVT	IX, IY	DRAWA DRWABS DSHMOD MOVABS MOVEA TKDASH TKPNT V2ST	PLTCHR TOUTST

6.2. Standard FORTRAN Routines Called By TCS

FORTRAN Routine	Called By
ABS	DASHSA DRAWSA PSCAL TKDASH
ALOG	RESCAL WINCOT
AMAX1	PSCAL REVCOT
AMIN1	PSCAL REVCOT
AMOD	REVCOT
ATAN2	REVCOT
COS	RROTAT WINCOT
FLOAT	KCM KIN LOGTRN PSCAL RESCAL RESET REVCOT SVSTAT TKDASH

FORTRAN Routine	Called By
IABS	INCPLT
IFIX	DASHSA DRAWSA PLTCHR PSCAL RESCAL REVCOT TKDASH RESTAT WINCOT KIN KCM
MAX0	PSCAL
MIN0	BUFFPK PSCAL
MOD	INCPLT PLTCHR SCURSR TKDASH
SIGN	DASHSA DRAWSA PSCAL
SIN	RROTAT WINCOT
SQRT	REVCOT TKDASH

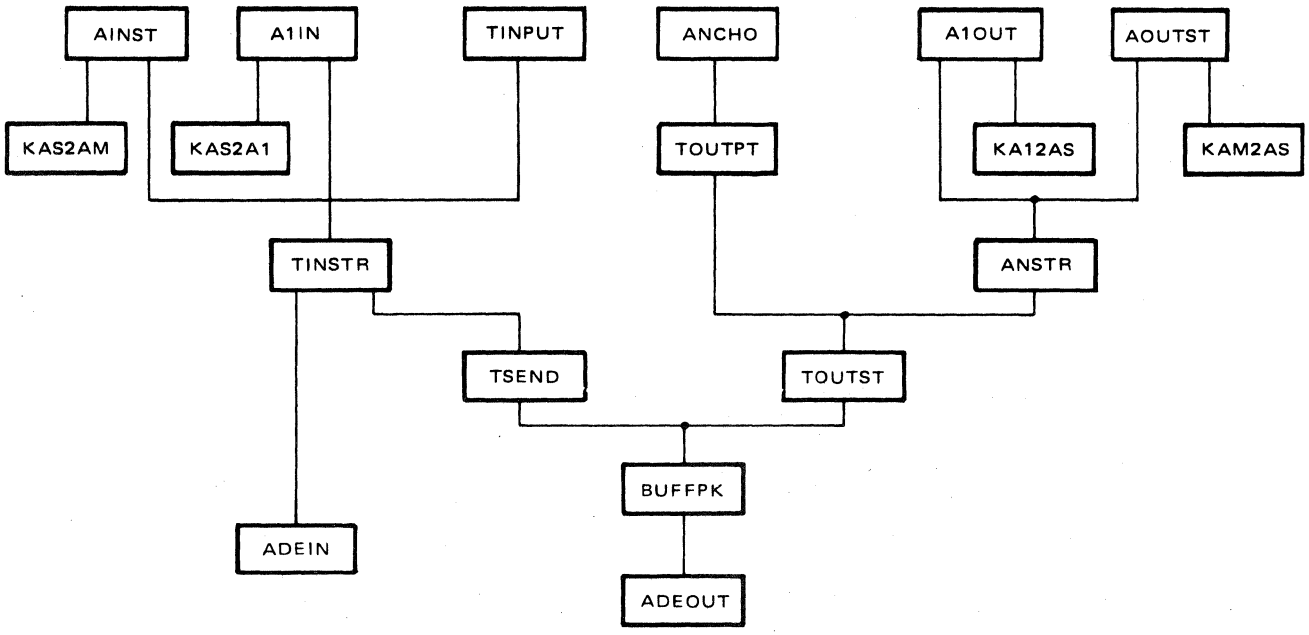
APPENDIX SYSTEM-DEPENDENT FEATURES

A 1. Terminal Control System I/O Structure

Six subroutines, ADEIN, ADEOUT, KAM2AS, KA12AS, KAS2AM and KAS2A1, are not included with the standard TCS source code and must be provided by the implementer. These routines are included in versions of TCS for TSO, PDP-11 and CDC-Synchronous systems.

If alphanumeric formatted I/O is not required, the latter four subroutines may be omitted and the subroutines that call them removed from the source file (see the TCS System Manual, Section 4.3).

The structure chart below shows the relationship of these six subroutines to the other TCS I/O subroutines. Full descriptions follow.



I/O Structure Chart

A 1.1. User-Written I/O Subroutines

ADEOUT

The routine BUFFPK assembles all the characters to be output in a buffer, takes care of any recovery needed (see below for buffer formats) and calls ADEOUT to send the contents of the buffer to the Terminal. The calling sequence for ADEOUT is:

CALL ADEOUT (NCHAR,IARRAY)

where NCHAR is the number of characters to be sent from the buffer, and IARRAY is the buffer, an integer array of ADE characters. The maximum number of characters which ADEOUT can handle should be determined by the size of the system output buffer. TCS was written with a maximum buffer size of 72, which the implementer may change to match his system's output buffer at the following locations: dimension and data statements in BUFFPK, A1OUT, and AOUTST and in the code of A1OUT and INITT.

IARRAY comes out of BUFFPK in one of four formats, depending on the Status Variable KUNIT found in /TKTRNX/ Terminal Status Area. The subroutine INITT calls SETBUF(1) and thereby sets KUNIT to 1, but the implementer may change this to 2, 3, or 4 in the source code, and the user may change KUNIT through his own call to subroutine SETBUF. The relation of IARRAY and KUNIT is as follows:

KUNIT	IARRAY
1	Recovered output, which is pure output preceded and followed by the necessary characters to return the Terminal to the condition (mode and beam position) it was in prior to the last interline sequence.
2	4014 Terminal output, which is pure output preceded by one SYN and followed by one ESC.
3	Pure output, which is only the characters given to BUFFPK by TOUTST and which assumes that interline characters are suppressed.
4	Pure output, unbuffered by BUFFPK.

NOTE

BUFFPK may add some timing characters to pure output.

Output of the following ASCII characters is not required by the Terminal Control System:*

NUL, SOH, STX, ETX, EOT, ENQ, ACK, HT, VT, SO, SI, DLE, DC1, DC2, DC3, DC4, NAK, CAN, EM.

However, other Tektronix software packages make use of the Terminal Control System I/O section and require the transmission of some of these characters, so ADEOUT should translate and output as much of the standard 128 ASCII character set as possible. The implementer may wish to use the translation subroutine KAS2A1 or KAS2AM to handle the translations required by ADEOUT.

*RUBOUT (ADE 127_{10}) is used by TCS as a graphic addressing character, so if it cannot be handled, whenever 127_{10} is found in IARRAY change it to 126_{10}.

How ADEOUT is written partly determines whether interline characters will cause any problems (see Section A 2 for more information).

ADEIN

Input to the Terminal Control System subroutines is through TINSTR, which calls ADEIN when it needs more input and buffers it for use by the other input subroutines (see the I/O Structure Chart on page A1 of this manual and the I/O Section of the User Manual). The calling sequence is:

CALL ADEIN (NCHAR,IARRAY)

where IARRAY is the integer array of the ADE characters received in the last line of input terminated by a CR but not including the CR, and NCHAR is the number of meaningful characters* in IARRAY. Since NCHAR should be limited only by the system input buffer size, and TCS was written with a maximum input buffer size of 72, the implementer should change the number 72 to match his system's input buffer size in the dimension statements of these subroutines: TINSTR, A1IN and AINST.

ADEIN should perform four functions:

1. Accept characters from the terminal
2. Translate these characters to ADE format
3. Place them into IARRAY
4. Compute NCHAR to be the number of meaningful characters returned.

As a minimum, the TTY character set should be accepted and translated, but the entire ASCII set is most desirable. The routine KA12AS or KAM2AS could be used for this translation.

ADEIN input should be essentially the same as normal monitor mode input, with identical echo and editing features. For example, when FORTRAN I/O is performed, if a Control-U is used to delete a character, ADEIN should allow for this. Note that the graphic input (GIN) mode characters, ADE 32 through 63, should not be used as editing characters.

A 1.2. User Written Translation Subroutines

The implementer must provide four simple translation subroutines, KA12AS, KAM2AS, KAS2A1, and KAS2AM, to support the alphanumeric subroutines A1OUT, AOUTST, A1IN, and AINST, respectively. The following discussion assumes that A1 and Am formats are used by the implementer in these routines, but any alphanumeric format which works is satisfactory. The "m" referred to is the number of characters per word the system supports (4 on GE and IBM, 5 on DEC PDP-10, 2 on many mini-computers, etc). A4 is recommended for compatibility with IGP.

*Trailing blanks, including any spaces entered from the keyboard immediately before the CR, are not meaningful and should not be included when computing NCHAR. TINSTR adds trailing blanks as needed when the array it is filling is longer than NCHAR.

KA12AS and KAM2AS

These routines translate characters from alphanumeric format into ADE integers. They should handle the character set required for ADEIN. KA12AS translates the first NCHAR characters from an A1 format array into an ADE integer array, while KAM2AS translates the first NCHAR characters from an Am format array into an ADE integer array. For example, an alphanumeric "A" should be translated to the integer 65_{10}. The calling sequences are:

```
CALL KA12AS (NCHAR,KA1,KADE)
CALL KAM2AS (NCHAR,KAM,KADE)
```

where NCHAR is the number of characters to be translated, KA1 and KAM are the alphanumeric arrays to be translated, and KADE is the integer array for the translated ADE characters.

KAS2A1 and KAS2AM

These routines translate characters from ADE integer form into alphanumeric format. They should handle the character set required for ADEOUT. KAS2A1 translates the first NCHAR characters from an ADE integer array into an A1 format array, while KAS2AM translates the first NCHAR characters from an ADE integer array into an Am format array. For example, the integer 66_{10} should be translated to the alphanumeric character "B". The calling sequences are:

```
CALL KAS2A1 (NCHAR,KADE,KA1)
CALL KAS2AM (NCHAR,KADE,KAM)
```

where NCHAR is the number of characters to be translated, KADE is the integer array containing the ADE characters to be translated, and KA1 and KAM are the arrays for the A1 and Am translated characters.

A 2. Interline Characters

Most computer systems are oriented to non-graphic-display teletypewriter terminals, and this causes problems for software written to drive the Tektronix graphic display terminals. The teletypewriter requires CR's, LF's and certain characters (NUL, SYN or RUBOUT) between each line of output to reposition the typing head and advance the paper. Many computer systems insert these characters automatically if they have not appeared in the last 72 (or 80 or 132) characters of the output stream to ensure that the teletypewriter does not lose data by overstriking. The interline characters CR and LF have the following effects on Tektronix graphic display terminals:

1. A CR puts the terminal into alphanumeric mode and moves the alphanumeric cursor to the left margin.

2. A LF moves the alphanumeric cursor or graphic beam position down one line height.

3. If the terminal is in graphic input (GIN) mode, a CR puts the terminal into alphanumeric mode without sending the crosshair cursor coordinates.

The 4014/4015 Terminals were designed to allow the programmer to get around these problems. No action occurs if these Terminals receive an ESC followed by one or more of these characters: CR, LF, NUL, RUBOUT. TCS takes advantage of this feature if buffer type 2 (see page A2) is chosen on 4014/4015 Terminals. A type 2 buffer ends with an ESC, so that the CR and LF which normally follow a line of output are ignored by the Terminal. This buffer begins with a SYN, otherwise a no-op character, which causes the Terminal to pay attention again. Thus interline characters cause no problem if buffer type 2 is used on a 4014 or 4015 Terminal.

For systems where all CR's and LF's can be suppressed both between lines of output and where the computer system would otherwise automatically insert them, use buffer type 3 or 4. Most systems allow the suppression of CR's and LF's between lines of program-controlled output (with carriage control characters in FORTRAN, for example), and many systems allow the suppression of the automatically inserted CR's and LF's through monitor commands (TYPE 6 on GE Mark III, TTY NO CRLF on DEC PDP-10).

For those systems which cannot suppress the automatically inserted interline characters, the interline characters between lines of output should *not* be suppressed, for they come at predictable times. Buffer type 1 is designed for use with 4006-1, 4010, 4012 and 4013 terminals on those systems which cannot otherwise overcome the interline character problems. *Graphic input mode cannot be used in this case, however, because of effect 3 above.*

For those systems which do suppress interline characters but in doing so suppress *all* CR's and LF's, including those placed in the TCS buffer for line control, the subroutines CARTN and LINEF may be changed to move the alphanumeric cursor graphically. These changes include deleting lines of code from these routines so they appear as follows:

```
C----------SUBROUTINE--CARTN-------------------------TEKTRONIX, INC.-
C
      SUBROUTINE CARTN
      COMMON /TKTRNX/ TMINVX,TMINVY,TMAXVX,TMAXVY,TREALX,TREALY,
     & TIMAGX,TIMAGY,TRCOSF,TRSINF,TRSCAL,TRFACX,TRFACY,
     & TRPAR1,TRPAR2,TRPAR3,TRPAR4,TRPAR5,TRPAR6,KMOFLG(2),
     & KGNMOD,KPADV,KACHAR,KOBLEN,KTRAIL,KLEVEL,KPAD2,
     & KBAUDR,KGNFLG,KGRAFL,KHOMEY,KKMODE,KHORSZ,KVERSZ,KTBLSZ,
     & KSIZEF,KLMRGN,KRMRGN,KFACTR,KTERM,KLINE,KZAXIS,KBEAMX,KBEAMY,
     & KMOVEF,KPCHAR(5),KDASHT,KMINSX,KMINSY,KMAXSX,KMAXSY,KEYCON,
     & KINLFT,KOTLFT,KUNIT
      CALL MOVABS(KLMRGN,KBEAMY)
      CALL ALFMOD
      RETURN
      END


C----------SUBROUTINE--LINEF-------------------------TEKTRONIX, INC.-
C
      SUBROUTINE LINEF
      COMMON /TKTRNX/ TMINVX,TMINVY,TMAXVX,TMAXVY,TREALX,TREALY,
     & TIMAGX,TIMAGY,TRCOSF,TRSINF,TRSCAL,TRFACX,TRFACY,
     & TRPAR1,TRPAR2,TRPAR3,TRPAR4,TRPAR5,TRPAR6,KMOFLG(2),
     & KGNMOD,KPADV,KACHAR,KOBLEN,KTRAIL,KLEVEL,KPAD2,
     & KBAUDR,KGNFLG,KGRAFL,KHOMEY,KKMODE,KHORSZ,KVERSZ,KTBLSZ,
     & KSIZEF,KLMRGN,KRMRGN,KFACTR,KTERM,KLINE,KZAXIS,KBEAMX,KBEAMY,
     & KMOVEF,KPCHAR(5),KDASHT,KMINSX,KMINSY,KMAXSX,KMAXSY,KEYCON,
     & KINLFT,KOTLFT,KUNIT
      KBEAMY=KBEAMY-(KVERSZ+KFACTR/2)/KFACTR
      IF(KBEAMY .GE. 0)GO TO 100
      KBEAMY=KHOMEY
100   CALL MOVABS(KBEAMX,KBEAMY)
      CALL ALFMOD
C * RESTORE ZAXIS MODE IF APPROPRIATE
200   IF(KTERM .GE. 2)CALL CWSEND
      RETURN
      END
```

A 3. Compatibility With Other Tektronix Software

A11 PLOT 10 packages of Level 1 or later are internally compatible with each other. The products listed below were originally compatible with Release 2.0 of TCS and must be updated as indicated to work properly with Level 1 TCS.

CHARACTER GENERATION SYSTEM: (all releases through 1.1)

Routine RROTAT and RSCALE in the Character Generation System contain the old TCS Release 2.0 /TKTRNX/ Terminal Status Area. Since both RROTAT and RSCALE are contained in TCS Release 3.0 through Level 1, they must be removed from the Character Generation System.

PREVIEW ROUTINES FOR CALCOMP PLOTTER: (all releases through 1.1)

Routine WHERE in the Preview package contains a reference to the old version of /TKTRNX/ Terminal Status Area. This version of /TKTRNX/ must be replaced by a copy identical to that in TCS Level 1. No other changes are required.

ADVANCED GRAPHING II: (all releases through 1.2)

The TCS extension, TCSEXT, should be deleted. See the Implementation Notes for AG-II Release 1.2 for a precise definition of TCSEXT.
One subroutine in AG-II, SETWIN, needs modification. See AG-II Implementation Notes for Release 1.2 for details.

The 4006-1 Terminal

Because the 4006-1 terminal does not generate a hardware backspace or use the GIN mode, you may wish to modify subroutine BAKSP so that it will accomplish this task. Refer to the 4010A01 PLOT 10 Terminal Control System Installation Guide.

Changes Necessary in Programs Using Release 2.0

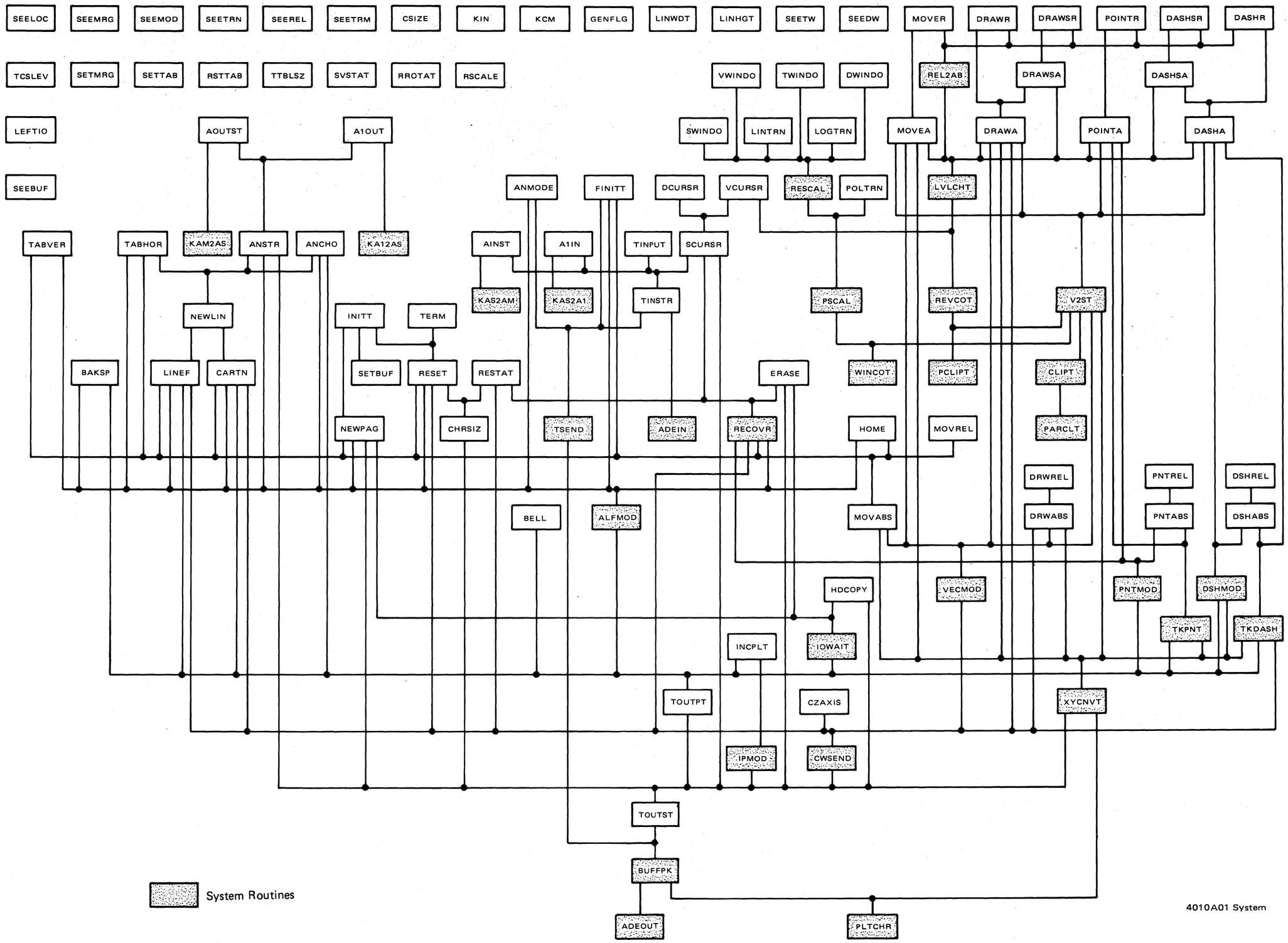
Any program referencing Status Variables in the Release 2.0 Terminal Status Area will not run with Level 1 without modification, since this common area has been changed. However, all the functions which required the Release 2.0 user to access this area are now supported by Level 1 subroutines, so conversion of these programs if fairly simple. To convert these programs, delete the /TKTRNX/ common area and change the code lines which reference the Status Variables to call the appropriate subroutines, as follows:

Release 2.0 Status Variables	Level 1 Subroutines
TRSINF, TRCOSF	RROTAT
TRSCAL	RSCALE
KLMRGN, KRMRGN	SETMRG

Since the tab tables KVERTT and KHORZT are not carried in the Release 3.3 Terminal Status Area, the user must provide a dimension statement for KHORZT and KVERTT in his program using the tab routines.

NOTE

Access to other variables in the /TKTRNX/ Terminal Status Area was not supported in Release 2.0, so it will be necessary for the user who accessed them to locate the correct subroutine or function in Release 3.3 to replace references to them.



4010A01 System

TCS System Flowchart

@

ASCII CODE CHART

CONTROL				HIGH X & Y GRAPHIC INPUT				LOW X				LOW Y			
NUL	ø	DLE	16	SP	32	ø	48	@	64	P	8ø	`	96	p	112
SOH	1	DC1	17	!	33	1	49	A	65	Q	81	a	97	q	113
STX	2	DC2	18	"	34	2	5ø	B	66	R	82	b	98	r	114
ETX	3	DC3	19	#	35	3	51	C	67	S	83	c	99	s	115
EOT	4	DC4	2ø	$	36	4	52	D	68	T	84	d	1ø0	t	116
ENQ	5	NAK	21	%	37	5	53	E	69	U	85	e	1ø1	u	117
ACK	6	SYN	22	&	38	6	54	F	7ø	V	86	f	1ø2	v	118
BEL BELL	7	ETB	23	'	39	7	55	G	71	W	87	g	1ø3	w	119
BS BACK SPACE	8	CAN	24	(	4ø	8	56	H	72	X	88	h	1ø4	x	12ø
HT	9	EM	25	)	41	9	57	I	73	Y	89	i	1ø5	y	121
LF LINE FEED	1ø	SUB	26	*	42	:	58	J	74	Z	9ø	j	1ø6	z	122
VT	11	ESC	27	+	43	;	59	K	75	[	91	k	1ø7	{	123
FF	12	FS	28	,	44	<	6ø	L	76	\	92	l	1ø8	|	124
CR RETURN	13	GS	29	-	45	=	61	M	77	]	93	m	1ø9	}	125
SO	14	RS	3ø	.	46	>	62	N	78	^	94	n	11ø	~	126
SI	15	US	31	/	47	?	63	O	79	_	95	o	111	127 RUBOUT (DEL)	

SUBJECT INDEX

SUBROUTINE AND VARIABLE INDEX

Status Variables Referenced in Text*

System Subroutines

*For a complete list, see Section 5.

User Subroutines Referenced in Text*

*For a complete list, see Section 2

TEKTRONIX®
committed to technical excellence

MANUAL CHANGE INFORMATION

PRODUCT PLOT 10
070-2242-00

CHANGE REFERENCE C1/877
DATE 8-2-77

CHANGE:	DESCRIPTION

For 4010A10 TCS for IBM 360/370 only

The changes do not effect other systems implementing TCS.

On page 3-2 in the description of BUFFPK, the third sentence in the paragraph reads: "For buffer types 1, 2, or 3, if NCHAR is larger than MAXLEN,..." The reference to MAXLEN should be changed to KOBLEN.

On the chart on pages 4-1 and 4-2 delete the references to MAXLEN under subroutines ANSTR, AOUTST, BUFFPK, and TOUTST. On page 4-1 the footnote should be changed to read:

"* 132 on PDP-11 systems; 80 on CDC-Synchronous systems, and 89 on 360/370 systems."

On page 4-7 ANMODE should be eliminated from the listing of routines not called internally by TCS.

On page 5-2 it should be noted that with Option 22 KMOFLG is used, and the text should read;

KMOFLG (1)	KMOFLG (1) = escorted mode flag 0 = disabled 1 = enabled (for use in communicating to Option 22 interfaces)
KMOFLG (2)	Not used

The chart on pages 5-4 through 5-9 should be amended as follows:

Page Number	Status Variable	Initial Setting (INITT)	Set By	Referenced By
*5-4	KACHAR	*	SETBUF	AMSTR TOUTST
**5-7	KOBLEN	89	INITT	TOUTST A1OUT AOUTST BUFFPK

*Delete this line from the chart.
**Amend this line as shown.

CHANGE:	DESCRIPTION

The following changes should be made to the chart on pages 5-10 through 5-17:

Page	Routine	Sets	References
**5-10	AlOUT		KOBLEN
**5-10	AOUTST		KOBLEN
*5-17	TOUTST		KOBLEN

**Add this line to chart.

*Amend this line as shown.

KOBLEN should be added to the list of references listed with BUFFPK on page 5-10. Also, the footnote on page 5-12 is no longer applicable.

With the 360/370 version of TCS subroutine FINITT calls ANMODE and with Option 22 ADEIN calls ADEOUT. These differences require the following changes to the subroutine charts on pages 6-1 through 6-9:

Page	Routine	Arguments	Called By	Calls
6-1	ADEIN	NCHAR, IARRAY	TINSTR	ADEOUT
6-1	ADEOUT	NCHAR, IARRAY	BUFFPK ADEIN	
6-1	ANMODE		FINITT	ALFMOD TSEND
6-3	FINITT	IX, IY		ANMODE MOVABS
6-7	TSEND		ANMODE TINSTR	BUFFPK

On page 6-1 FINITT should be deleted from the list of routines which call ALFMOD.

The I/O structure chart on page A-1 and the flowchart on page A-8 should show that ADEIN makes a call to ADEOUT. -- if Option 22 is implemented.

The last sentence in the first paragraph on page A-2 should be changed to read:

"TCS was written with a maximum buffer size of 89, which the implementer may change to match his system's output at the following locations: dimension statement in BUFFPK, AlOUT, and AOUTST and in the code of INITT."

Note, however, that it is not recommended that the change be made.

CHANGE:	DESCRIPTION

The last sentence in the first paragraph describing ADEIN on page A-3 should be changed to read:

"...and TCS was written with a maximum input buffer size of 89, the implementer should change the number 89 to match his system's input buffer size in the dimension and data statements of these subroutines: TINSTR, AlIN and AINST."

Again, it is not recommended that this change be made.

The flowchart on page A-8 should show that FINITT does not call TSEND or ALFMOD, but does call MOVABS and ANMODE.

Again, note that the above changes apply only to TCS implementations for IBM 360/370. They do not effect other versions of TCS.

TEKTRONIX®
committed to technical excellence

MANUAL CHANGE INFORMATION

PRODUCT PLOT 10 4010A01 (A10, A11, A12)

CHANGE REFERENCE C2/379

DATE 3-12-79

CHANGE: 070-2242-00 DESCRIPTION

TEXT ADDITION

AFTER APPENDIX A

ADD:

SUBROUTINES WHICH RETAIN HISTORY

The following list shows variables within the code which retain the original values through subsequent executions.

SUBROUTINE	VARIABLES
XYCWUT	IDREW
BUFFPK	LEWOUT
	NODATA
	ITEMP
	KSYWCS