EXAMPLE

This example illustrates the definition of two segments, Segment 1 and Segment 2. Segment 1 is the top-level segment, and Segment 2 is nested (or referenced) within Segment 1.

Note that the SEGDEF command changes the Model One prompt to "$" which indicates segment definition mode.

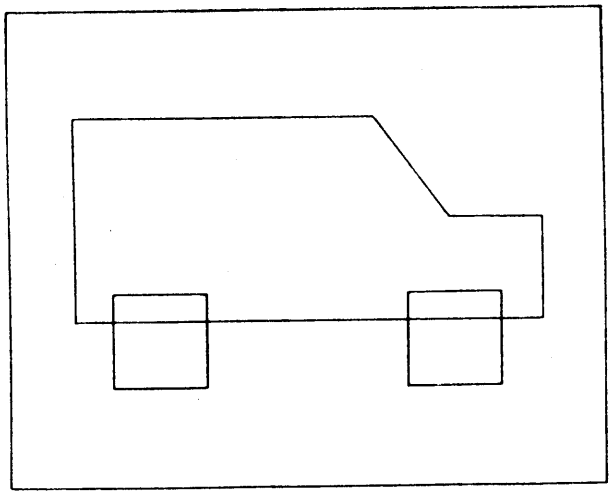
```
! SEGINI 0 256          ; Initialize display list structures with a
                          block size of 256 bytes.
! SEGDEF 1              ; Begin definition of Segment 1, which defines
                          the truck body, chassis, and the position of
                          the 2 wheels.
$ PICKID 10            ; Assign pick ID 10 label to the truck body.
$ PUSH XFORM XF2DCP    ; Push current 2-D transformation and WCS current
                          point onto the stack.
$ MOVABS -500 -160     ; Move current point to lower left corner of truck.
$ DRWREL 0 440         ; Draw 5 vectors to define truck body.
$ DRWREL 640 0
$ DRWREL 160 -220
$ DRWREL 200 0
$ DRWREL 0 -220
$ PICKID 20            ; End pick ID 10 (body); assign pick ID 20 label to
                          the chassis (bottom line) of truck.
$ DRWREL -1000 0       ; Draw chassis of truck.
$ MOVABS -320 -200     ; Move current point to position rear wheel.
$ SEGREF 2             ; Nest Segment 2, the wheel.
$ MOVABS 310 -200      ; Move current point to position front wheel.
$ SEGREF 2             ; Nest Segment 2, the wheel.
$ POP XFORM XF2DCP     ; Pop current 2-D transformation and WCS current
                          point from the stack.
$ SEGEND               ; End pick ID 20 (chassis); end definition of
                          Segment 1.

! SEGDEF 2             ; Begin definition of Segment 2, which defines the
                          shape of the wheel.
$ PICKID 30            ; Assign pick ID 30 label to the wheel.
$ PUSH CREG CURPNT     ; Save current point (center of wheel).
$ MOVREL -100 -100     ; Move current point to left corner of wheel.
$ RECREL 200 200       ; Draw wheel.
$ POP CREG CURPNT      ; Restore current point (center of wheel).
$ SEGEND               ; End pick ID 30 (wheel); end definition of
                          Segment 2, the wheel.
!                      ; Model One returns to normal command mode.
```

EXAMPLE, continued

Note that defining a segment does not draw it. To draw a segment, you execute the SEGREF command.

```
! EXMODE NORMAL DRAW     ; Set execution mode to normal draw.
! SEGREF 1               ; Execute (draw) Segment 1.
```



SEGDEL SEGDEL

SYNTAX

ASCII	SEGDEL segment	
FORTRAN Call	CALL SEGDEL (SEGID)	
Binary	[222] [segment3][segment2][segment1][segment0]	(5 bytes)
	222 decimal = 336 octal = DE hex	

FUNCTION

The SEGDEL command deletes the specified segment.

ASCII PARAMETER

segment The number of the segment to be deleted (32 bits).

FORTRAN PARAMETER

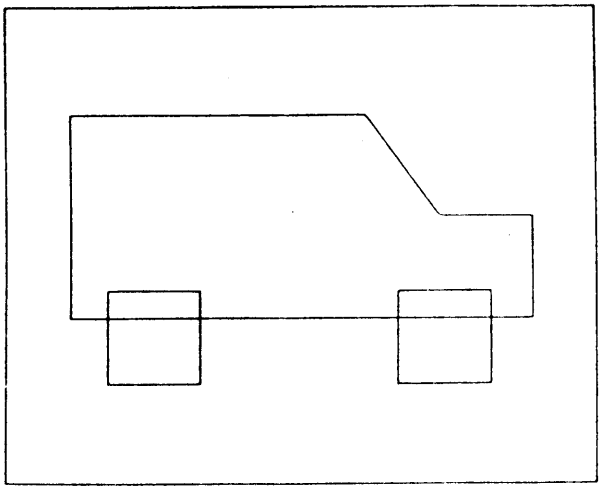
INTEGER*4 SEGID

EXAMPLE

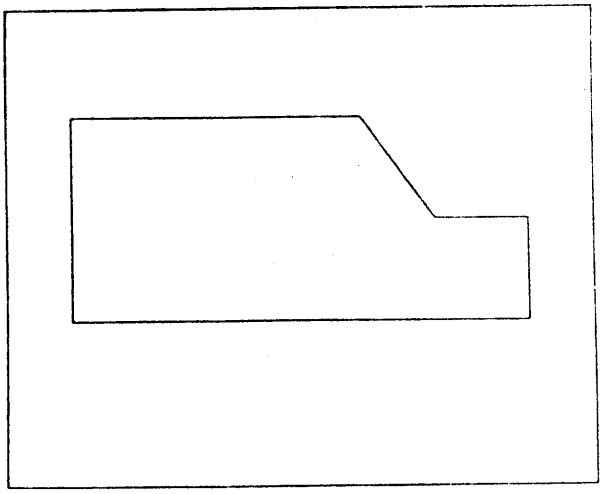
This example refers to the Segment 1 and Segment 2 which were defined in the example for the SEGDEF command. Segment 1 defines the car body, chassis, and position of the two wheels. Segment 2 defines the shape of the wheel. Segment 2 is nested within Segment 1. Note that when Segment 2 is deleted, Segment 1 is still drawn, even though it includes a reference to a segment which no longer exists.

```
! SEGREF 1          ; Execute (draw) Segment 1.
! SEGDEL 2          ; Delete Segment 2, which is nested within Segment 1.
! VAL8 0            ; Clear screen.
! FLOOD
! VAL8 63
! SEGREF 1          ; Execute (draw) Segment 1, now a truck without
                     wheels.
```

BEFORE DELETION



AFTER DELETION



SEGEND SEGEND

SYNTAX

ASCII	SEGEND
FORTRAN Call	CALL SEGEND
Binary	[221] (1 byte)
	221 decimal = 335 octal = DD hex

FUNCTION

The SEGEND command ends the definition of an open segment by closing the segment. You must also use the SEGEND command to close a segment that you append with the SEGAPP command.

EXAMPLE

```
! SEGDEF 25          ; Begin definition of Segment 25.
$ RECREL 40 40       ; Draw rectangle.
$ SEGEND             ; End definition of Segment 25.
!                    ; The Model One returns to normal command mode.
```

Note that the Model One prompt changes to "$" during the definition of a segment. SEGEND, which ends the definition, changes the prompt back to "!".

SEGINI SEGINI

SYNTAX

ASCII	SEGINI reserved, blocksize
FORTRAN Call	CALL SEGINI (BLKSIZ)
Binary	[225] [reserved = 0] [highblocksize][lowblocksize] (4 bytes)

225 decimal = 341 octal = E1 hex

FUNCTION

The SEGINI command initializes the Display List firmware. You should execute SEGINI before using any of the Display List commands.

The SEGINI command performs the following five functions:

- deletes all existing segments
- initializes the display list structures
- initializes EXMODE to NORMAL DRAW
- initializes the size of the display list blocks to the size specified by the blocksize parameter, and
- initializes the pick aperture to the default size: 20 x 20 pixels.

ASCII PARAMETERS

reserved An 8-bit reserved quantity, must be 0.

blocksize The size (in bytes) of the display list data blocks (16 bits); range is 32 to 16384. Note that an even-numbered range avoids the wasting of memory since odd block sizes are rounded down to the nearest even integer. Default is 256.

FORTRAN PARAMETER

INTEGER*2 BLKSIZ

SEGINI SEGINI

EXAMPLE

```
! SEGINI 0 128          ; Initialize display list structures with a block
                          size of 128 bytes, delete existing segments,
                          set default pick aperture size, initialize
                          EXMODE to NORMAL.
```

SEGINQ SEGINQ

SYNTAX

ASCII SEGINQ segment

FORTRAN Call CALL SEGINQ (SEGNUM, ISTAT)

Binary [229] [segment3][segment2][segment1][segment0]
(5 bytes)

229 decimal = 345 octal = E5 hex

FUNCTION

The SEGINQ command reads back the attributes of and verifies the existence of the specified segment. One 16-bit word is returned; Bit 0 (the LSB) is set if the segment is visible, bit 1 is set if the segment is pickable, and bit 2 is set if the segment is executable; all other bits are clear. If the segment does not exist, then the Model One returns -1.

ASCII PARAMETER

segment The number of the segment whose attributes are being queried.

FORTRAN PARAMETER

Input Parameter: INTEGER*4 SEGNUM
Output Parameter: INTEGER*2 ISTAT

EXAMPLE

```
! SEGINQ 1              ; Query the attributes of Segment 1.
  00007                 ; The Model One returns:
                          Segment 1 exists, visibility is ON, pickability
                          is ON, executability is ON.
! SETATR 1 PICK OFF     ; Set pickability OFF for Segment 1.
! SEGINQ 1              ; Query the attributes of Segment 1.
  00005                 ; The Model One returns:
                          Segment 1 exists, visibility is ON, pickability
                          is OFF, executability is ON.
! SETATR 1 VIS OFF      ; Set visibility OFF for Segment 1.
! SEGINQ 1              ; Query the attributes of Segment 1.
  00004                 ; The Model One returns:
                          Segment 1 exists, visibility is OFF, pickability
                          is OFF, executability is ON.
! SEGDEL 1              ; Delete Segment 1.
! SEGINQ 1              ; Query the attributes of Segment 1.
  -00001                ; The Model One returns:
                          Segment 1 does not exist.
```

SEGREF SEGREF

SYNTAX

ASCII	SEGREF segment
FORTRAN Call	CALL SEGREF (SEGID)
Binary	[216] [segment3][segment2][segment1][segment0] (5 bytes)

216 decimal = 330 octal = D8 hex

FUNCTION

The SEGREF command has two functions: to execute a top-level segment or to reference a segment within the current segment definition.

SEGREF executes a top-level segment according to the current execution mode, as set by the EXMODE command.

SEGREF can also be used to nest segments by referencing the number of a child segment from within the definition of a parent. Segments can be nested to a level of 16. Lower-level segments must be defined if they are to be executed when their parent is executed. However, they need not be defined when referenced.

Note: The SEGREF command only updates the SEGREG and PIDREG registers if you are in PICK mode and there is a pick hit.

ASCII PARAMETER

segment The number of the segment (32 bits).

FORTRAN PARAMETER

INTEGER*4 SEGID

EXAMPLE 1: Nesting a Segment

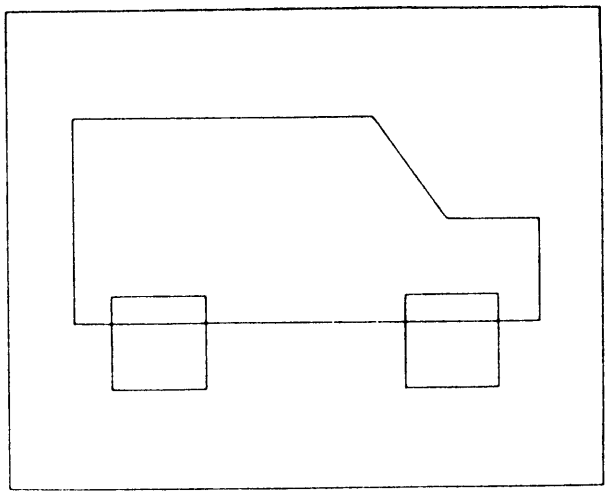
This example illustrates the use of the SEGREF command to nest one segment within another. In this example, Segment 2 (which defines the truck wheels) is nested two times within Segment 1 (which defines the truck body, chassis and position of the wheels). This is the same example that was used to illustrate the SEGDEF command.

```
! SEGINI 0 256          ; Initialize display list structures with a
                          block size of 256 bytes.
! SEGDEF 1              ; Begin definition of Segment 1, which defines
                          the truck body, chassis, and the position of
                          the 2 wheels.
$ PICKID 10            ; Assign pick ID 10 label to the truck body.
$ PUSH XFORM XF2DCP    ; Push current 2-D transformation and WCS current
                          point onto the stack.
$ MOVABS -500 -160     ; Move current point to lower left corner of truck.
$ DRWREL 0 440         ; Draw 5 vectors to define truck body.
$ DRWREL 640 0
$ DRWREL 160 -220
$ DRWREL 200 0
$ DRWREL 0 -220
$ PICKID 20            ; End pick ID 10 (body); assign pick ID 20 label to
                          the chassis (bottom line) of truck.
$ DRWREL -1000 0       ; Draw chassis of truck.
$ MOVABS -320 -200     ; Move current point to position rear wheel.
$ SEGREF 2             ; Nest Segment 2, the wheel.
$ MOVABS 310 -200      ; Move current point to position front wheel.
$ SEGREF 2             ; Nest Segment 2, the wheel.
$ POP XFORM XF2DCP     ; Pop current 2-D transformation and WCS current
                          point from the stack.
$ SEGEND               ; End pick ID 20 (chassis); end definition of
                          Segment 1.

! SEGDEF 2             ; Begin definition of Segment 2, which defines the
                          shape of the wheel.
$ PICKID 30            ; Assign pick ID 30 label to the wheel.
$ PUSH CREG CURPNT     ; Save current point (center of wheel).
$ MOVREL -100 -100     ; Move current point to left corner of wheel.
$ RECREL 200 200       ; Draw wheel.
$ POP CREG CURPNT      ; Restore current point (center of wheel).
$ SEGEND               ; End pick ID 30 (wheel); end definition of
                          Segment 2, the wheel.
!                      ; Model One returns to normal command mode.
```

EXAMPLE 2: Executing a Segment - Drawing

```
! EXMODE NORMAL DRAW     ; Set execution mode to normal draw (the default).
! SEGREF 1               ; Execute (draw) Segment 1.
```



Please refer to the EXMODE command for examples of executing segments in pick mode, highlight mode, and unhighlight mode.

SEGREN SEGREN

SYNTAX

ASCII SEGREN segment2, segment1

FORTRAN Call CALL SEGREN (SEGNEW, SEGOLD)

Binary [218] [segment23][segment22][segment21][segment20]
[segment13][segment12][segment11][segment10]
(9 bytes)

218 decimal = 332 octal = DA hex

FUNCTION

The SEGREN command renames a specified segment. Segment1 is the segment you want to rename; segment2 is the new name. If segment2 already exists, it will be overwritten.

Caution: Any references to an old segment name in SEGDEF and SEGREF commands are not automatically updated when that segment is renamed.

ASCII PARAMETERS

segment1 The number of the segment to be renamed (32 bits).

segment2 The new number for the segment (32 bits).

FORTRAN PARAMETERS

INTEGER*4 SEGNEW, SEGOLD

EXAMPLE

```
! SEGREN 21 20          ;Rename Segment 20 to Segment 21.
```

SELWIN SELWIN

SYNTAX

ASCII	SELWIN window
FORTRAN Call	CALL SELWIN (WINDOW)
Binary	[193] [window] (2 bytes)
	193 decimal = 301 octal = C1 hex

FUNCTION

The SELWIN command selects the current window for use as a scrolling text window.

If the alphanumeric emulator or VT100 emulator is in use, all alphanumeric data is sent to the currently selected window.

SELWIN redraws the entire window, for windows 1 through 8, leaving the selected window on top of the other windows.

Note: The window must be defined (see DEFWIN) before it can be selected.

ASCII PARAMETER

window Specifies the window number: range is 0 to 8.

FORTRAN PARAMETER

INTEGER*2 WINDOW

EXAMPLE

```
! DEFWIN 7 100,100 150,150 1,1 0,0
                        ; Define window number 7 using the default write
                          mask.
! SELWIN 7              ; Select window number 7 as the current window.
```

SETATR SETATR

SYNTAX

ASCII	SETATR segment, attrib, flag
FORTRAN Call	CALL SETATR (SEGID, ATTRIB, FLAG)
Binary	[230] [segment3] [segment2] [segment1] [segment0] [attrib] [flag] (7 bytes)

230 decimal = 346 octal = E6 hex

FUNCTION

The SETATR command enables or disables the visibility, pickability, or executability attributes of the specified segment. Each of these attributes is automatically set to ON upon segment definition.

A segment can have visibility enabled and pickability disabled. However, a segment which is not visible is not pickable, regardless of the current pickability attribute.

Note: When you change a segment's attributes, the new attributes are not visibly noticed until that segment is re-referenced.

Comparison of the Visibility and Executability Attributes

When visibility is on, the segment is executed and written to image memory. When visibility is off, the segment is still executed, but no writing to image memory occurs. Turning visibility off facilitates the updating of the current transformation matrix and any other user states (e.g., CREGs, VREGs, flags, etc.), while inhibiting the drawing of objects that are not desired on the screen.

When executability is on, the segment is executed and written to image memory (the same effect as setting visibility on). Turning executability off causes the segment to be trivially rejected when referenced (as if the segment were not defined). This turning off of individual segments (i.e., not drawing or traversing) improves performance (where ignoring a segment doesn't impact the functioning of the program).

SETATR SETATR

ASCII PARAMETERS

segment — The number of the segment for which you want to change an attribute (32 bits).

attrib — The attribute to be set.

0 = VIS Visibility
1 = PICK Pickability
2 = EXEC Executability

flag — Flag = 1 or ON, enable attribute; flag = 0 or OFF, disable attribute.

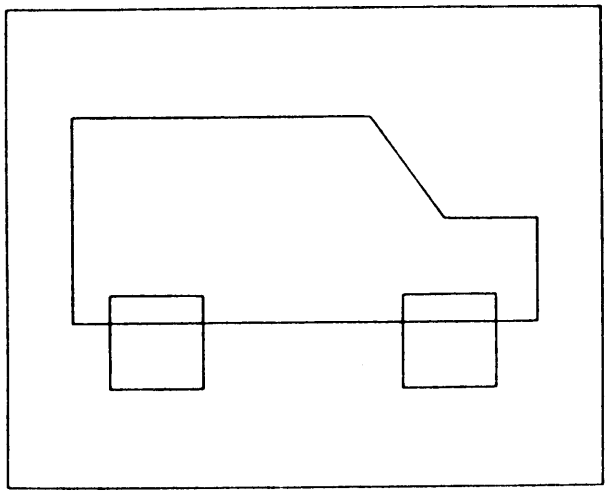
FORTRAN PARAMETERS

INTEGER*4 SEGID
INTEGER*2 ATTRIB, FLAG

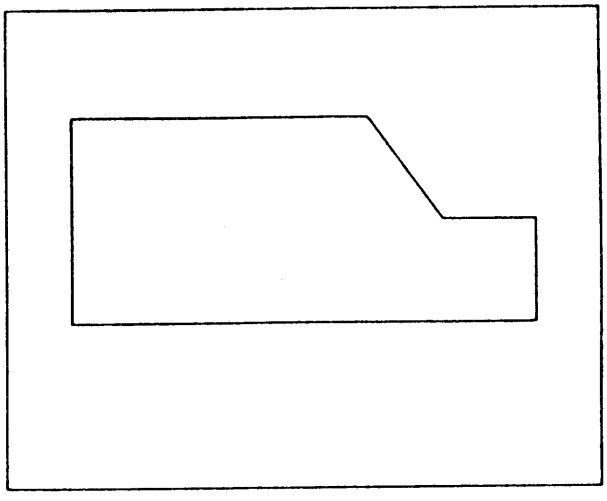
EXAMPLE: Setting the Visibility Attribute to OFF

```
! SEGREF 1              ; Execute (draw) Segment 1.
! SETATR 2 VIS OFF      ; Set visibility OFF for Segment 2, the wheel.
! VAL8 0                ; Clear screen.
! FLOOD
! VAL8 63
! SEGREF 1              ; Execute (draw) Segment 1; nested Segment 2 is
                          not visible.
```

SEG 1: EVERYTHING VISIBLE



SEG 1: NESTED SEG 2 NOT VISIBLE



SETCUR SETCUR

SYNTAX

ASCII	SETCUR flag
FORTRAN Call	CALL SETCUR (FLAG)
Binary	[199] [flag] (2 bytes)
	199 decimal = 307 octal = C7 hex

FUNCTION

The SETCUR command determines whether or not the cursor of the currently selected alphanumeric window or VT100 window will be displayed.

ASCII PARAMETER

flag — Eight-bit parameter which specifies whether the alphanumeric or VT100 window cursor is displayed:

flag = 1 or ON displays the cursor;
flag = 0 or OFF disables the cursor.

FORTRAN PARAMETER

INTEGER*2 FLAG

EXAMPLE

```
! SETCUR ON              ; Display cursor.
```

SETGL SETGL

ASCII SYNTAX

```
SETGL  global, value
```

FORTRAN SUBROUTINE CALLS

```
CALL SETGL  (GLOBAL,XAPSZ,YAPSZ,PID,SID,IGNORE,PICBUF)
```

Note: The following are provided for convenience.

```
CALL SETGL0 (XAPSZ,YAPSZ)      (Set value for pick aperture)
CALL SETGL1 (PID)              (Set value for pick ID register)
CALL SETGL2 (SID)              (Set value for segment ID register)
CALL SETGL3 (IGNORE)           (Set number of pick hits to ignore)
CALL SETGL4 (PICBUF)           (Set information to store in pick buffer)
```

HOST BINARY COMMAND STREAMS

```
[70] [global = 0] [highxap][lowxap] [highyap][lowyap]   (6 bytes)
[70] [global = 1] [pid3][pid2][pid1][pid0]              (6 bytes)
[70] [global = 2] [sid3][sid2][sid1][sid0]              (6 bytes)
[70] [global = 3] [highignore][lowignore]               (4 bytes)
[70] [global = 4] [picbuf]                              (3 bytes)
```

70 decimal = 106 octal = 46 hex

FUNCTION

The SETGL command sets the values of the following globally defined display list parameters:

- the size of the pick aperture
- the current pick ID register
- the current segment ID register
- the number of picks to ignore during picking, and
- the type of information to store in the pick buffer during picking.

The global parameter specifies which global parameter to set.

The value parameter varies depending on which parameter is being set.

SETGL SETGL

ASCII PARAMETERS

global — The global parameter to set, as follows.

0 = PICKAP	Size of pick aperture.
1 = PIDREG	Pick ID number.
2 = SEGREG	Segment ID number.
3 = IGNORE	Pick hits to ignore during picking.
4 = PICKBUF	Information to store in the pick buffer during picking.

value — The new value for the specified parameter.

global = 0 — Two 16-bit parameters which represent half of the size of the pick aperture in the x and y dimensions. The default pick aperture size is 20 x 20 (SETGL PICKAP 10 10).

global = 1 — The pick ID number (32 bits).

global = 2 — The segment ID number (32 bits).

global = 3 — The number of picks to be ignored during picking (32 bits).

global = 4 — The information type (8 bits), as follows.

0 = TREE	Store entire pick trees.
1 = SEGID	Store only segment IDs.
2 = PICKID	Store only pick IDs.
3 = SEGPID	Store segment ID/pick ID pairs. (Default)

FORTRAN PARAMETERS

INTEGER*2 GLOBAL, XAPSZ, YAPSZ, IGNORE, PICBUF

INTEGER*4 PID, SID

EXAMPLE

```
! SETGL PICKAP 10 20     ; Set the pick aperture size to 20 x 40.
! SETGL PICKBUF TREE     ; Specify to save entire trees in pick buffer
                          during picking.
```

SETSIZ SETSIZ

SYNTAX

ASCII	SETSIZ xscale, yscale
FORTRAN Call	CALL SETSIZ (XSCALE, YSCALE)
Binary	[198] [xscale] [yscale] (3 bytes)
	198 decimal = 306 octal = C6 hex

FUNCTION

The SETSIZ command sets the width and height of all text written into the alphanumeric windows.

The xscale parameter determines the width (scaling in the x dimension); the yscale parameter determines the height.

If a value of zero is given for either parameter, the current value is retained.

The default value is (2,2).

An xscale and yscale of (2,2) create a 20-pixel-high by 12-pixel-wide character.

For windows 1 through 7, this command will re-initialize the entire window, erasing the text buffer and placing the cursor in character position (0,0).

Note: The SETSIZ command has no effect within the VT100 window (window 8).

ASCII PARAMETERS

xscale	Eight-bit parameter which specifies the width of the text.
yscale	Eight-bit parameter which specifies the height of the text.

SETSIZ SETSIZ

FORTRAN PARAMETERS

INTEGER*2 XSCALE, YSCALE

EXAMPLE

```
! SETSIZ 3,3          ; Set the x and y text scaling to three times the
                       default size.
```

SHMODE SHMODE

SYNTAX

ASCII	SHMODE flag
FORTRAN Call	CALL SHMODE (FLAG)
Binary	[86] [flag] (2 bytes)
	86 decimal = 126 octal = 56 hex

FUNCTION

The SHMODE command specifies the mode of shading. Available choices are: smooth (flag = 1 or ON) and flat (flag = 0 or OFF). Flat shading specifies that no color interpolation is to be done when displaying a patch using the ZPATCH command. Smooth shading specifies that color values are to be bilinearly interpolated across the patch, as part of the Gouraud algorithm. There is no difference in performance between using one shading mode or the other.

ASCII PARAMETER

flag	1 or ON:	perform Gouraud shading. (Default)
	0 or OFF:	perform no interpolation (flat shading).

FORTRAN PARAMETER

INTEGER*2 FLAG

EXAMPLE

In this example, 24-bit true color output is specified for the ZPATCH command. In order for the example to work properly on an 8-bit system, you must have set up the look-up table values for dithered true color. See the ZPATCH command for further explanation.

If you want to try the following example with SHMODE OFF and then ON, enter the rest of the commands in a macro to save retyping.

```
! SHMODE OFF              ; Turn off shading.
! ZFUNCT GT               ; Set the ZFUNCT to GREATER THAN.
! ZCLEAR -32768           ; Clear depth-buffer memory to -32768.
! ZCLIP OFF               ; No clipping in z.

! ZPATCH 0 3 0,0,0 255,0,0 100,0,0 0,255,0 100,100,100 0,0,255

                          ; Send the first z-patch: a triangle shading
                            from red at (0,0,0) to green at (100,0,0) to
                            blue at (100,100,100).

! ZPATCH 0 3 0,100,0 255,255,0 80,100,0 255,255,0 150,0,100 100,100,0

                          ; Send the second z-patch: another triangle,
                            shading from deep yellow across the back
                            edge (0,100,0 to 80,100,0) to medium yellow
                            at the front corner (150,0,100). This triangle
                            intersects the first triangle at a z-depth
                            value of 50.
```

SPCHAR SPCHAR

SYNTAX

ASCII	SPCHAR char, flag, code
FORTRAN Call	CALL SPCHAR (ICHAR, IFLAG, ICODE)
Binary	[178] [char] [flag] [code] (4 bytes)
	178 decimal = 262 octal = B2 hex

FUNCTION

The SPCHAR command can be used to redefine or disable any of the special characters used by the Model One, thereby circumventing problems with certain host computers and operating systems. The parameter char specifies which of the special characters is to be defined or disabled.

The default values for special characters are listed below.

Char	ASCII Code	Hex equivalent	Purpose
0	CTRL D	04 or 84	Enter Graphics mode
1	CTRL P	10 or 90	Send break to host
2	CTRL V	16 or 96	WARMstart
3	@	40 or B0	Line kill
4	CTRL H	08 or 88	Backspace
5	CTRL F	06 or 86	ACK (Acknowledge)
6	CTRL U	15 or 95	NACK (Negative Acknowledge) (abort)
7	CTRL X	18 or 98	Invoke Debug mode
8	CTRL Q	13 or 93	Resume communications (XON)
9	CTRL S	11 or 91	Suspend communications (XOFF)
10	CTRL M	0D or 8D	Newline

Flag = 1 or ON indicates that the special character is to be redefined. The third parameter, code, specifies the new ASCII code of the special character. Flag = 0 or OFF indicates that the special character is to be disabled. The code parameter is ignored but must be present.

SPCHAR SPCHAR

FUNCTION, continued

Once you have made changes to the special characters, you may save the changes with the SAVCFG command. Then the new special characters will be initialized with every COLDstart command.

Note: Disabling the WARMstart character does not disable the WARMstart command.

ASCII PARAMETERS

char	Special character number; range is 0 to 10.
flag	Flag = 1 or ON, redefine special character; flag = 0 or OFF, disable special character.
code	8-bit parameter specifying the new ASCII code.

FORTRAN PARAMETERS

INTEGER*2	ICHAR, IFLAG, ICODE

EXAMPLE

```
! SPCHAR 0 1 #05        ; Change the Enter Graphics mode control code to
                          05 hex or 85 hex (CTRL E).
! SPCHAR 2 0 0          ; Disable the WARMstart character.
```

SYSCFG: Overview SYSCFG

SYNTAX

SYSCFG ALPHA port_mnemonic

SYSCFG EMULATE type

SYSCFG ERROR port_mnemonic

SYSCFG HOST port_mnemonic, ASCII or BINARY

SYSCFG SERIAL port_mnemonic [RTS on/off] [CTS on/off] [STOP 1/2] [BITS 7/8] [PARITY e/o/1/h/n] [BAUD rate] [CTRL on/off] [XIN on/off] [XOUT on/off] [QUEUE size]

SYSCFG SERIAL TABLET type

FUNCTION

The SYSCFG command has several different forms, which are each described in more detail on the following pages. The table below outlines the forms of the SYSCFG command.

Command	Function
SYSCFG ALPHA	Configures the Model One's alphanumeric port.
SYSCFG EMULATE	Configures the Model One for alphanumeric or VT100 emulation.
SYSCFG ERROR	Configures the Model One's error port.
SYSCFG HOST	Configures the host port to accept either 8-bit binary characters or ASCII hexadecimal characters from the host.
SYSCFG SERIAL	Configures the Model One's serial ports: keyboard, tablet, alphanumeric terminal, and host.
SYSCFG SERIAL TABLET	Sets the tablet port for the graphics input device that is being used.

Note: Each of the SYSCFG commands can be executed only from the local alphanumeric terminal, and cannot be included in a macro.

SYSCFG ALPHA SYSCFG ALPHA

ASCII SYNTAX

SYSCFG ALPHA port_mnemonic

FUNCTION

The SYSCFC ALPHA command specifies which port is to be used as the local alphanumeric port. This command can only be executed from the local alphanumeric terminal, and cannot be included in a macro.

ASCII PARAMETER

port_mnemonic Port mnemonic or port number. The ports are:

3	KEYBSIO	Keyboard port
2	TABLETSIO	Tablet or mouse port
1	HOSTIO	Host serial port
0	ALPHASIO	Alphanumeric terminal port

SYSCFG EMULATE SYSCFG EMULATE

SYNTAX

SYSCFG EMULATE type

FUNCTION

The SYSCFG EMULATE command configures the Model One to power up into the alphanumeric window, VT100 window, or no window.

This command can only be executed from the local alphanumeric terminal, and cannot be included in a macro.

PARAMETER

type — The type of emulation.

- ALPHA — Selects alpha window 0 for powerup and user display.
- OFF — Selects ALPHASIO for powerup and user display.
- VT100K — Selects the VT100 window for powerup and user display.

SYSCFG ERROR SYSCFG ERROR

ASCII SYNTAX

SYSCFG ERROR port_mnemonic

FUNCTION

The SYSCFG ERROR command specifies which port is to be used as the error port. This command can only be executed from the local alphanumeric terminal, and cannot be included in a macro.

ASCII PARAMETER

port_mnemonic The port mnemonic or port number. The ports are:

3	KEYBSIO	Keyboard port
2	TABLETSIO	Tablet or mouse port
1	HOSTIO	Host serial port
0	ALPHASIO	Alphanumeric terminal port

SYSCFG HOST SYSCFG HOST

ASCII SYNTAX

SYSCFG HOST port_mnemonic, ASCII or BINARY

FUNCTION

The SYSCFG HOST command configures the host port to accept either ASCII hexadecimal or 8-bit binary characters from the host. The command does not change the default host port.

Note: If you are installing a serial interface to the host, you should configure the HOSTIO port to match the characteristics of the terminal that was connected to the line previously.

ASCII PARAMETER

port_mnemonic The port mnemonic or port number. The ports are:

3	KEYBSIO	Keyboard port
2	TABLETSIO	Tablet or mouse port
1	HOSTIO	Host serial port
0	ALPHASIO	Alphanumeric terminal port

EXAMPLE

```
! SYSCFG HOST HOSTIO ASCII      ; Configure the host port to accept ASCII
                                  hexadecimal characters.
```

SYSCFG SERIAL SYSCFG SERIAL

ASCII SYNTAX

SYSCFG SERIAL port mnemonic [RTS on/off] [CTS on/off] [STOP 1/2] [BITS 7/8] [PARITY e/o/1/h/n] [BAUD rate] [CTRL on/off] [XIN on/off] [XOUT on/off] [QUEUE size]

FUNCTION

The SYSCFG SERIAL command configures the Model One's serial ports: keyboard, tablet, alphanumeric terminal, and host. This command can only be executed from the local alphanumeric terminal, and cannot be included in a macro.

Displaying the Current Configuration: To display the current configuration, use the DISCFG command. The following is an example of what you might see after entering the DISCFG command. In this example, the default configuration is displayed.

PORT	RTS	CTS	STOP BITS		XIN	XOUT	CTRL	PARITY	BAUD	QUEUE
KEYBSIO	OFF	OFF	2	7	ON	OFF	ON	NONE	300	0006
TABLETSIO	OFF	OFF	2	8	OFF	OFF	OFF	NONE	9600	0006
ALPHASIO	OFF	OFF	2	8	ON	OFF	ON	NONE	9600	0006
HOSTIO	OFF	OFF	1	8	OFF	ON	OFF	NONE	9600	000B

Changing the Configuration: To change the configuration of any serial port, use the SYSCFG SERIAL command, followed by the port to be configured, and the keywords and values for any parameter you want to change. Omit any parameter you want to leave unchanged.

Saving the New Configuration: After you have changed the configuration, you need to save the new configuration with the SAVCFG command. The SAVCFG command copies the new configuration into NVRAM (non-volatile RAM).

Restoring the Default Configuration: To reset the configuration of serial ports to the default values, use the DFTCFG command. The default values are stored in PROM (programmable read-only memory) are are not affected by the SYSCFG or SAVCFG commands.

SYSCFG SERIAL SYSCFG SERIAL

ASCII PARAMETERS

port_mnemonic — Supplies the mnemonic of the serial port to be configured. You can also use the port number in place of the mnemonic. The ports are:

3	KEYBSIO	Keyboard port
2	TABLETSIO	Tablet or mouse port
1	HOSTIO	Host serial port
0	ALPHASIO	Alphanumeric terminal port

RTS, CTS — Not currently applicable to the Model One/380.

STOP — Specifies whether 1 or 2 stop bits should be used.

BITS — Specifies whether 7 or 8 bits are sent per byte.

XIN — Indicates whether XON/XOFF is to be accepted at input.

ON specifies that output will be enabled or disabled according to the XON/XOFF signals received by the port.

OFF specifies that XON/XOFF signals should be ignored.

XIN does not apply to the TABLETSIO port.

XOUT — Indicates whether XON/XOFF should be sent when the port's queue is near full (ON) or simply not used (OFF).

XOUT does not apply to the TABLETSIO port.

CTRL — Indicates whether the Model One should accept control characters (including [CTRL-S] and [CTRL-Q]) from the port (ON) or ignore them (OFF).

PARITY — Specifies input/output parity. Parity may be even (E), odd (O), high (H), low (L), or not used (N).

BAUD — Specifies baud rate, which may be 75, 100, 134.5, 150, 300, 600, 1200, 1800, 2000, 2400, 4800, 19200, or 38400.

The baud rate for the host port cannot be set higher than 19200. If an illegal baud rate is specified, the command will generate an error and then terminate.

QUEUE — Specifies the queue size for the I/O queues. The size defines the new queue size to be 2 ** size (2 raised to the size power). The total of all queue sizes for the serial ports cannot be greater than 6K; verify this before doing a SAVCFG.

EXAMPLE

```
! SYSCFG SERIAL ALPHASIO BAUD 4800

                          ; Changes the baud rate for the alphanumeric
                            terminal port to 4800; leaves other parameters
                            unchanged.
Are you sure??            ; Prompt from Model One.
y                         ; User response to prompt confirming that the
                            configuration is as desired.

(Model One displays new configuration.  Graphics mode must be re-entered
(CTRL-D))

! SYSCFG SERIAL TABLETSIO PARITY 0 CTRL ON

                          ; Changes, for the tablet port, parity to odd and
                            indicates that control characters should be
                            accepted.
Are you sure??            ; Prompt from Model One.
y                         ; User response to prompt confirming that the
                            configuration is as desired.

(Model One displays new configuration.  Graphics mode must be re-entered
(CTRL-D))
```

SYSCFG SERIAL TABLET SYSCFG SERIAL TABLET

ASCII SYNTAX

SYSCFG SERIAL TABLET type

FUNCTION

The SYSCFG SERIAL TABLET command sets the tablet port for the XY digitizer that is being used.

You can only enter this command from the local terminal. You cannot include the command in a macro.

ASCII PARAMETER

type	GTCO	GTCO tablet.
	SUMMA	Summagraphics Bit-Pad.
	NGRID	Summagraphics Microgrid.
	RMOUSE	Mouse Systems mouse.

EXAMPLE

```
! SYSCFG SERIAL TABLET RMOUSE      ; Sets the tablet port for the Mouse
                                    Systems mouse.
```

SYSTAT SYSTAT

ASCII SYNTAX

```
SYSTAT  infotype   or
SYSTAT  infotype, segment
```

FORTRAN Calls

```
CALL SYST80 (FUNC,ARRSIZ,SID,FREBLK,NUM,SEGS,NUMBLK)    (general form)

CALL SYSTA0 (FREBLK)                 (Return number of free memory blocks
                                      available for segment definition.)
CALL SYSTA1 (ARRSIZ, NUM, SEGS)      (Return array of defined segments.)
CALL SYSTA2 (SID, NUMBLK)            (Return number of memory blocks used
                                      by specified segment.)
```

Binary

```
[228] [infotype = 0,1]          (2 bytes)
[228] [infotype = 2] [segment3][segment2][segment1][segment0]   (6 bytes)

228 decimal = 344 octal = E4 hex
```

FUNCTION

The SYSTAT command returns information on display list memory usage and availability. The value of the infotype parameter (0, 1, or 2) determines the type of information that the SYSTAT command returns.

If infotype = 0, the SYSTAT command returns a 32-bit word indicating the number of remaining free memory blocks available. These blocks are the size specified with the SEGINI command.

If infotype = 1, the SYSTAT command returns an array of defined segment numbers.

If infotype = 2, the SYSTAT command returns the number of memory blocks used by the specified segment.

SYSTAT SYSTAT

ASCII PARAMETERS

infotype The type of information to return.

0 = FREEMEM Return the number of free memory blocks available for segment definition.

1 = SEGS Return array of defined segment numbers. Format is (n,array), where n is the number of defined segments (16 bits), and array is an array of n segment IDs (32 bits). The returned array is unsorted.

2 = SIZE Return the number of memory blocks used by the specified segment (32 bits).

segment The number of the segment for which to return memory block information (32 bits). Use this parameter only when infotype = 2.

FORTRAN PARAMETERS

CALL SYST80 (FUNC, ARRSIZ, SID, FREBLK, NUM, SEGS, NUMBLK)

Input Parameters:	INTEGER*2	FUNC, ARRSIZ
	INTEGER*4	SID
Output Parameters:	INTEGER*2	NUM
	INTEGER*4	FREBLK, SEG(1), NUMBLK

FUNC The type of information to return.

ARRSIZ Size of the segments array in longwords.

SID The segment ID used for FUNC = 2.

FREBLK The number of free memory blocks available for segment definition.

NUM The number of defined segments.

SEGS Array of NUM segments IDs.

NUMBLK The number of blocks used in SID.

EXAMPLE

In this example, the blocksize is 256 and Segment 1 has been defined as specified in the SEGDEF command example.

```
! SYSTAT FREEMEM          ; Query system memory availability.
 0000004030               ; Model One returns: 4030 free memory blocks
                            available for segment definition.
! SYSTAT SEGS             ; Ask for the number of defined segments, and a
                            list of segment IDs.
  00002                   ; Model One returns: two segments have been
 0000000001                 defined, Segment 1 and Segment 2.
 0000000002
! SYSTAT SIZE 1           ; Ask for the number of memory blocks used by
                            Segment 1.
 0000000001               ; Model One returns: 1 memory block used by
                            Segment 1.
```

TEKEM TEKEM

SYNTAX

ASCII	TEKEM flag
FORTRAN Call	CALL TEKEM (FLAG)
Binary	[57] [flag] (2 bytes)
	57 decimal = 071 octal = 39 hex

FUNCTION

The TEKEM command invokes the Tektronix 4014 emulator. For complete details of the command, see the manual, Tektronix Emulator Technical Note.

ASCII PARAMETERS

flag See the Tektronix Emulator Technical Note for details.

FORTRAN PARAMETERS

INTEGER*2 FLAG

Note: The FORTRAN call, CALL TEKEM (FLAG), enters the Tektronix Emulator; it is then up to the user to ensure that the correct data is sent. After the emulator is exited, Model One FORTRAN library commands may again be used.

TEXT1 TEXT1

SYNTAX

ASCII	TEXT1 string
FORTRAN Call	CALL TEXT1 (STRLEN, STRING)
Binary	[144] [strlen] ([char1][char2]...[charn])
	144 decimal = 220 octal = 90 hex

FUNCTION

The TEXT1 command draws a horizontal text string into image memory with Font 1. If the text size is the default 16, each character uses 5 x 7 pixels. Descenders take an additional two pixels.

The parameter string specifies the text to be drawn. If the command is entered in ASCII mode from the local alphanumeric terminal or keyboard, then the string to be drawn is the set of ASCII characters which follow the TEXT1 command on the command line. If the TEXT1 command is sent from the host, then the length of the string must be specified. Strlen specifies the number of characters in the string, and is followed by strlen bytes containing the ASCII characters to be drawn.

The current point (CREG 0) specifies the starting point for the text string and remains unchanged by the command.

You can specify the size of the text and the baseline angle of the text using the TEXTC or TEXTN commands.

2-D transformations affect the rotation, scaling, and translation of text that is drawn with any of the text drawing commands.

ASCII PARAMETER

string The text to be drawn.

TEXT1 TEXT1

FORTRAN PARAMETERS

```
INTEGER*2     STRLEN, STRING(1)
```

STRLEN is an integer specifying the number of characters that are to be drawn. STRING is an integer array with two characters packed per 16-bit word, as in FORTRAN A2 format.

EXAMPLE

```
! MOVABS 0 0              ; Move current point to 0,0.
! TEXT1 Horizontal text ; Draw text string in default size 16 and angle 0.
! MOVABS 0 20             ; Move current point to 0,20.
! TEXTC 32 45             ; Set double-sized text at a 45 degree angle.
! TEXT1 Double-sized text at an angle
```

TEXT2 TEXT2

SYNTAX

ASCII	TEXT2 string
FORTRAN Call	CALL TEXT2 (STRLEN, STRING)
Binary	[145] [strlen] ([charl][char2]...[charn])
	145 decimal = 221 octal = 91 hex

FUNCTION

The TEXT2 command draws a horizontal text string into image memory with Font 2. Font 2 is a user defined character set which is downloaded using the TEXTDN command.

At power-on or COLDstart, each character in Font 2 defaults to the same character in Font 1. When Font 2 is downloaded, each character replaces the power-on default definition.

The TEXT2 command is issued in the same manner as the TEXT1 command. The current point (CREG 0) specifies the starting point for the text string and remains unchanged by the command. Strlen specifies the length of the string when text is not sent in ASCII mode.

ASCII PARAMETER

string The text to be drawn.

FORTRAN PARAMETERS

INTEGER*2 STRLEN, STRING(1)

STRLEN is an integer specifying the number of characters that are to be drawn. STRING is an integer array with two characters packed per 16-bit word, as in FORTRAN A2 format.