

QCKVU IMAGE SERVER

User's Manual

Artwork Conversion Software, Inc.

QCKVU IMAGE SERVER

Copyright

Copyright 1999-2002 by Artwork Conversion Software, Inc. All rights reserved. No part of this publication may be reproduced without the prior written consent of Artwork Conversion Software, Inc., 417 Ingalls St., Santa Cruz CA 95060

QCKVU IMAGE SERVER (QIS) is protected by both United States copyright law and international treaty provisions. Therefore this software must be treated just like a book, with one exception. We authorize you to make archival copies of the software for the purpose of backing up the software to protect your investment in it.

The phrase “just like a book” means that this software can be used by any number of people and may be freely moved from one computer location to another, so long as there is no possibility of it being used at one location while it is simultaneously being used at another. This copy of the software may not be used by two different people at the same time in different places without violating our copyright.

Warranty

We warrant the physical media and documentation to be free from defects for a period of 30 days from shipment. We further warrant that this product does substantially what the data sheet describes it will do. For a period of 30 days from shipment we will accept the return of the software and refund your purchase price if you find it does not perform as specified. You agree to examine and test the software within the 30 day limit. We will not accept returns after the 30 day limit.

In no event shall Artwork be held liable for any loss of profit, expense incurred or other commercial damage arising from your use of the software.

Governing Law

Artwork Conversion Software, Inc. is a California corporation and this warranty shall be interpreted and governed by the laws of California.

Attorneys Fees

In case of litigation arising from the licensing or use of this software the prevailing party shall be entitled to reasonable attorney fees and all costs of proceedings incurred in the litigation.

Trademarks

QCKVU IMAGE SERVER and QIS are trademarks of Artwork Conversion Software, Inc. DXF is a trademark of AutoDesk Co.. Other trademarks belong to their respective owners.

Acknowledgements

QCKVU IMAGE SERVER was written by Eric Chan. Documentation by Steve DiBartolomeo and Jeff Warrick.

Table of Contents

Chapter 1 - Introduction to QIS

Remote Viewing using QCKVU	1-1
Speed Issues	1-2
The Client Side	1-3

Chapter 2 - Installation Guide

Please see the special insert for Installation topics and page numbers.

Chapter 3 - Opening a GDSII File - Commands and Functions

Open_GDSII	3-1
Arguments	3-1
Example	3-1
Errors	3-1
Notes	3-1
Error and Warning Messages When Opening a GDSII File	3-2
Errors	3-2
Warnings	3-2
Set_Progress_Message	3-4
Arguments	3-4
Example	3-4
Errors	3-5
Notes	3-5

Chapter 4 - Structures - Commands and Functions

Open_Structure	4-1
Arguments	4-1
Example	4-1
Notes	4-1
Get_Structure_List	4-1
Arguments	4-1
Example	4-1
Errors	4-2
Notes	4-2
Get_Structure_Extents	4-2
Arguments	4-2
Example	4-2
Errors	4-2
Notes	4-2
Get_Structure_Children	4-2
Arguments	4-2
Example	4-3
Errors	4-3
Notes	4-3

Get_Structure_Root	4-3
Arguments	4-3
Example	4-3
Errors	4-3
Notes	4-3
Get_Structure_Tree	4-4
Use	4-4
Arguments	4-4
Example	4-4
Errors	4-5
Notes	4-5
Get_Structure_References	4-6
Use	4-6
Arguments	4-6
Example	4-6
Errors	4-6
Notes	4-6
Set_Structure_Outline	4-7
Arguments	4-7
Example	4-7
Errors	4-7
Notes	4-7
Set_Structure_Labels	4-7
Arguments	4-7
Example	4-8
Errors	4-8
Notes	4-8

Chapter 5 - Basic File Info - Commands and Functions

Get_DBU	5-1
Arguments	5-1
Example	5-1
Errors	5-1
Notes	5-1
Get_QIS_Report	5-1
Arguments	5-1
Example	5-1
Errors	5-2
Notes	5-2
Get_QIS_Version	5-2
Arguments	5-2
Example	5-2
Errors	5-2
Notes	5-2

Chapter 6 - Layers, Colors and Patterns - Commands and Functions

Get_Layer_List	.6-1
Arguments	.6-1
Example	.6-1
Errors	.6-1
Notes	.6-1
Get_Colfill_Pat	.6-1
Arguments	.6-1
Example	.6-1
Errors	.6-2
Notes	.6-2
Set_Layers_On	.6-2
Arguments	.6-2
Example	.6-2
Errors	.6-3
Notes	.6-3
Set_Layers_Off	.6-3
Arguments	.6-3
Example	.6-3
Errors	.6-4
Notes	.6-4
Set_Layers_Fill	.6-4
Arguments	.6-4
Example	.6-4
Errors	.6-5
Notes	.6-5
Set_Layers_Outline_Color	.6-6
Arguments	.6-6
Example	.6-6
Errors	.6-6
Notes	.6-7

Chapter 7 - Display Control - Commands and Functions

Get_Window	.7-1
Arguments	.7-1
Example	.7-1
Errors	.7-1
Notes	.7-1
Set_Window	.7-1
Arguments	.7-1
Example	.7-1
Errors	.7-1
Notes	.7-2
Set_Fill	.7-2
Arguments	.7-2
Example	.7-2
Errors	.7-2
Notes	.7-2

Set_Array_Mode	7-3
Arguments	7-3
Errors	7-3
Notes	7-3
Set_Outline	7-3
Arguments	7-3
Example	7-3
Errors	7-3
Notes	7-4
Set_Display_Filter_Size	7-4
Arguments	7-4
Example	7-4
Errors	7-4
Notes	7-4
Set_Geometry_Marker	7-5
Arguments	7-5
Example	7-5
Errors	7-5
Notes	7-5
Set_Reference_Marker	7-6
Arguments	7-6
Example	7-6
Errors	7-6
Notes	7-6
Set_Marker_Shape	7-6
Arguments	7-6
Example	7-6
Errors	7-7
Notes	7-7
Set_Text_Mode	7-7
Arguments	7-7
Example	7-7
Errors	7-7
Notes	7-7
Set_Background_Color	7-8
Arguments	7-8
Example	7-8
Errors	7-8
Notes	7-8
Set_Nesting_Level	7-8
Arguments	7-8
Example	7-8
Errors	7-8
Notes	7-9
Set_Structure_Outline	7-9
Arguments	7-9
Example	7-9
Errors	7-9
Notes	7-9

Set_Structure_Labels	7-10
Arguments	7-10
Example	7-10
Errors	7-10
Notes	7-10
Set_Scale_Bar	7-11
Arguments	7-11
Example	7-11
Errors	7-11
Notes	7-11
Set_Window	7-12
Arguments	7-12
Example	7-12
Errors	7-12
Notes	7-12

Chapter 8 - Getting an Image - Commands and Functions

Set_Image_Size	8-1
Arguments	8-1
Example	8-1
Errors	8-1
Notes	8-1
Set_Image_Format	8-1
Arguments	8-1
Example	8-2
Errors	8-2
Notes	8-2
Redraw	8-2
Arguments	8-2
Example	8-3
Errors or Picture not Drawn	8-3
Notes	8-3
Get_Image	8-3
Arguments	8-3
Example	8-3
Errors	8-4
Notes	8-4
Stop	8-4
Arguments	8-4
Example	8-4
Errors	8-4
Notes	8-4
Zoom_Home	8-4
Arguments	8-4
Example	8-4
Errors	8-4
Notes	8-4

Image_Ready	8-5
Arguments	8-5
Example	8-5
Errors	8-5
Notes	8-5

Chapter 9 - Getting Vectors - Commands and Functions

Set_Vector_Unit	9-1
Arguments	9-1
Example	9-1
Errors	9-1
Notes	9-1
Get_Vector_Path	9-2
Arguments	9-2
Example	9-2
Errors	9-2
Notes	9-2
Set_Reference_Vector_Format	9-2
Arguments	9-2
Example	9-2
Errors	9-3
Notes	9-3
Get_Vector	9-3
Primitives	9-3
Boundary	9-4
Boundary Syntax	9-4
Example	9-4
Path	9-4
Path Syntax	9-4
Example	9-5
Text	9-5
Text Syntax	9-5
Structure Reference	9-6
Structure Syntax	9-6
Example	9-6
Array Structure Reference	9-6
Array Structure Syntax	9-6
Example	9-6
Get_Display_Vector	9-7
Arguments	9-7
Example	9-7
Errors	9-7
Notes	9-7
Get_GDS_Vector	9-7
Arguments	9-7
Example	9-7
Errors	9-7
Notes	9-7

Get_Vertex_Info	9-7
Arguments	9-7
Example	9-7
Errors	9-8
Notes	9-8

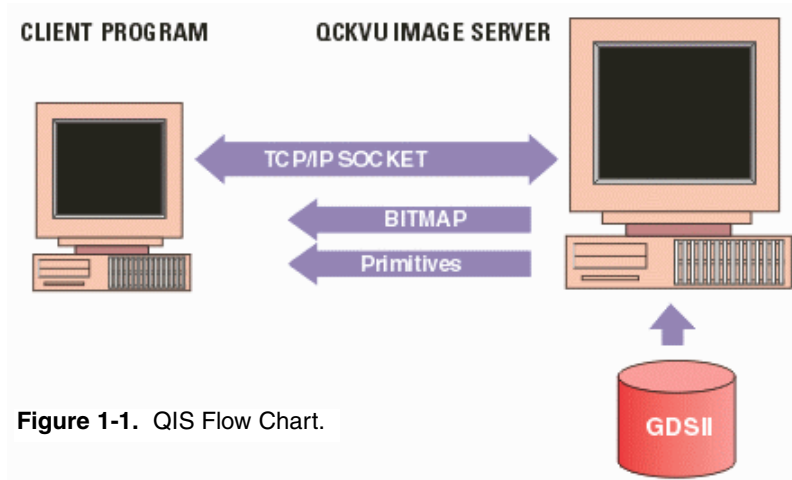
Appendix I - Technical Support

Address	A1-1
Email	A1-1
WEB Site	A1-1
FTP Site	A1-1
Dropping Files at the FTP Site	A1-2
Picking up Files from the FTP Site	A1-2
ZIP Compression	A1-2
ZIP Password Compression	A1-2
WEB Support	A1-3
What's the Current Version of the Software	A1-3
Downloading the Most Current Version	A1-3
Technical Bulletins and Solutions to Common Problems	A1-3

Chapter 1

Introduction to QIS

The Qckvu Image Server project is designed to convert Qckvu from a typical viewing application to a server where the image is not displayed on screen but sent to another application.



Commands that would normally be entered using a mouse or the keyboard are sent from the client to the qckvu server via a TCP/IP socket. The client program can request data such as a list of layers or structures and can at any time request an "image" - essentially a bitmap that represents what would normally have gone to the screen.

The client can also send "state" commands such as turning layers on and off, assigning fill patterns to layers and changing the display window size, coordinates and magnification.

This enables other applications which need to display and manipulate very large GDSII files to use the technology we have put into Qckvu; the TCP/IP approach means that only the API need be understood by the client program and that the server and the client program need not be written in the same language or run on the same operating system.

The Qckvu Image Server (QIS) will enable remote viewing of very large GDSII files; we believe that QIS will be useful for IC fabs and mask shops to allow clients to view and measure critical data without the need to transmit large data files.

Remote Viewing using Qckvu

The problem of "remote" viewing an extremely large data file is one that can now be addressed by taking advantage of the Internet together with improvements in computer hardware and software.

Consider the following situation. You are an IC designer, say based in the UK, and your designs are fabricated at a foundry located in Taiwan. You send your data to Taiwan (either by CD ROM, tape or via the Internet.) The foundry takes and modifies your GDSII data to enhance the yield for their particular process and may also include modifications needed for mask correction.

The foundry wants you to sign off on the mask set and you need to review the changes to be sure that critical parts of your design remain valid. A sticking point: the changes the fab has made to your design are considered proprietary and they are reluctant (or unwilling) to actually send back the entire GDSII file to you -- it's OK to look, but not to actually have the modified data.

Or the situation may be that the fab does not wish to make the needed changes but wants you to make them. They need to show you in detail the areas that are problematic and how best to modify these areas so that you meet their design rules.

How an Image Server Comes to the Rescue

One way to satisfy both parties is for the fab to install a GDSII Image server. This is a powerful GDSII viewer that runs on a large workstation; typically a Sun or HP server equipped with lots of RAM (from 2 to 12 GB), lots of hard disk space and multiple processors.

The image server has a couple of additional functions that a standard viewer does not.

First, it can receive commands via TCP/IP instead of just from the keyboard and mouse. This enables an external source to control the viewer.

Second, it can send the screen image as a bitmap back out the TCP/IP port. This enables the external program to take the bitmap and through it up on the screen.

Speed Issues

Can a system like this be fast enough? The speed is limited first by the speed of the viewer and the size of the file being viewed. One must use an image server that has been optimized for speed. Secondly, one must install the image server on fast, powerful hardware. This is generally not that difficult, since the foundry must have a powerful server since it routinely handles very large files and performs very compute intensive calculations on them such as fracturing and design rule checking.

By far the largest delay on the server side is loading the GDSII file for the first time. This can take several minutes for very large files. Once the file is loaded entirely into RAM, zooming and panning are extremely fast.

The other limitation is the speed of the connection. If the client is internal it is running on a 10Mb or 100Mb connection. If the client is far away the connection speed can be limited by any of the links between client and server.

However note that the size of the commands sent to the image server are only a few bytes. The size of the image returned by the server varies from about 20KB to about 300 KB depending on the screen size and the complexity of the image. Even when connecting through a 56K modem, it should only take a few seconds to transmit each screen. It's really no different than opening the home page of most WEB sites.

The Client Side

We've discussed the server -- what about the client side? The client can be written in any language and run on any platform since it communicates with the server via TCP/IP using a simple standard API. One can use Visual Basic, C++, Java or Tcl/Tk. The only requirement is that the language have support for TCP/IP (and most do) and the ability to process and display bitmaps.

The client can be simple or complex. Only about a few commands need to be implemented to make a minimal client: `open_gdsii`, `open_structure`, `zoom_window`, `get_image` and `exit`. The API is rich enough, however, to build a full featured client if desired.

Since the API is open and fully documented, a large company could decide to build their own client to their own specifications.

Installation Guide - UNIX

Artwork Conversion Software, Inc.

Santa Cruz, CA 95060

(831) 426-6163

info@artwork.com

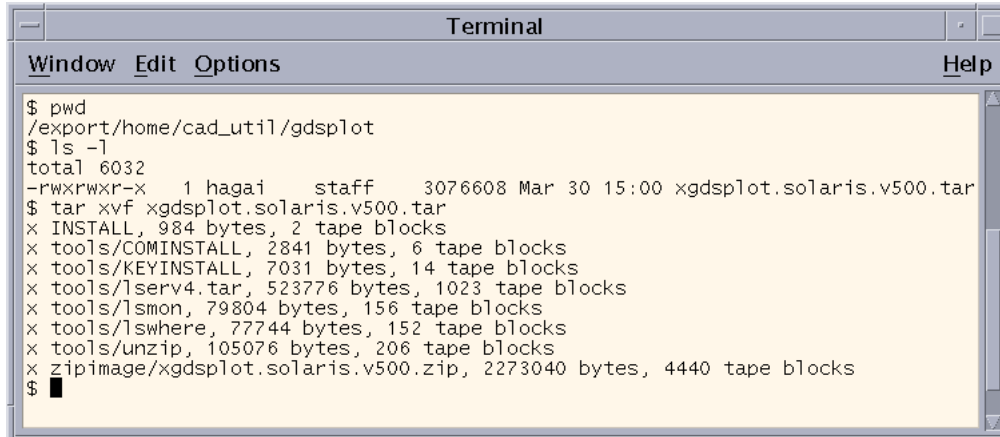
Installation Contents

Step 1: Untar the release into a directory	2-3
Step 2: Start the install script	2-3
Step 3: Enter the unzipping password	2-3
Updating?	2-4
New Installation	2-4
Network or Node Locked License	2-4
Network Licensing - Case 1	2-5
Network Licensing - Case 2	2-5
Installing the License Manager	2-6
Choose a Text Editor	2-7
Entering the Key Codes	2-8
Node Lock Installation	2-10

Unix Installation

Artwork's UNIX programs are shipped on CD or electronically as a **tar** file. To install the software you untar the release to disk and then run the INSTALL script that accompanies each program. We will illustrate the steps using the GDSPLLOT program on Solaris.

Step 1 - Untar the Release into a directory



```
Terminal
Window Edit Options Help
$ pwd
/export/home/cad_util/gdsplot
$ ls -l
total 6032
-rwxrwxr-x 1 hagai staff 3076608 Mar 30 15:00 xgdsplot.solaris.v500.tar
$ tar xvf xgdsplot.solaris.v500.tar
x INSTALL, 984 bytes, 2 tape blocks
x tools/COMINSTALL, 2841 bytes, 6 tape blocks
x tools/KEYINSTALL, 7031 bytes, 14 tape blocks
x tools/lserver4.tar, 523776 bytes, 1023 tape blocks
x tools/lsmom, 79804 bytes, 156 tape blocks
x tools/lswhere, 77744 bytes, 152 tape blocks
x tools/unzip, 105076 bytes, 206 tape blocks
x zipimage/xgdsplot.solaris.v500.zip, 2273040 bytes, 4440 tape blocks
$
```

Files extracted include:
Install scripts
licensing utilities
zip file
unzipping program.

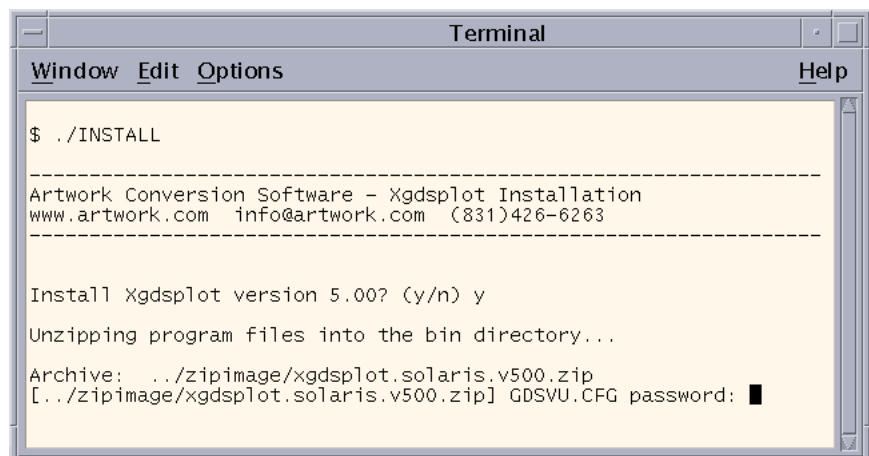
Step 2 - Start the INSTALL Script

Start the installation by executing the INSTALL script.

```
$ ./INSTALL
```

Step 3 - Enter your Unzipping Password

The program files are zipped using a password. You should have received the unzipping password in an email. Enter it now.



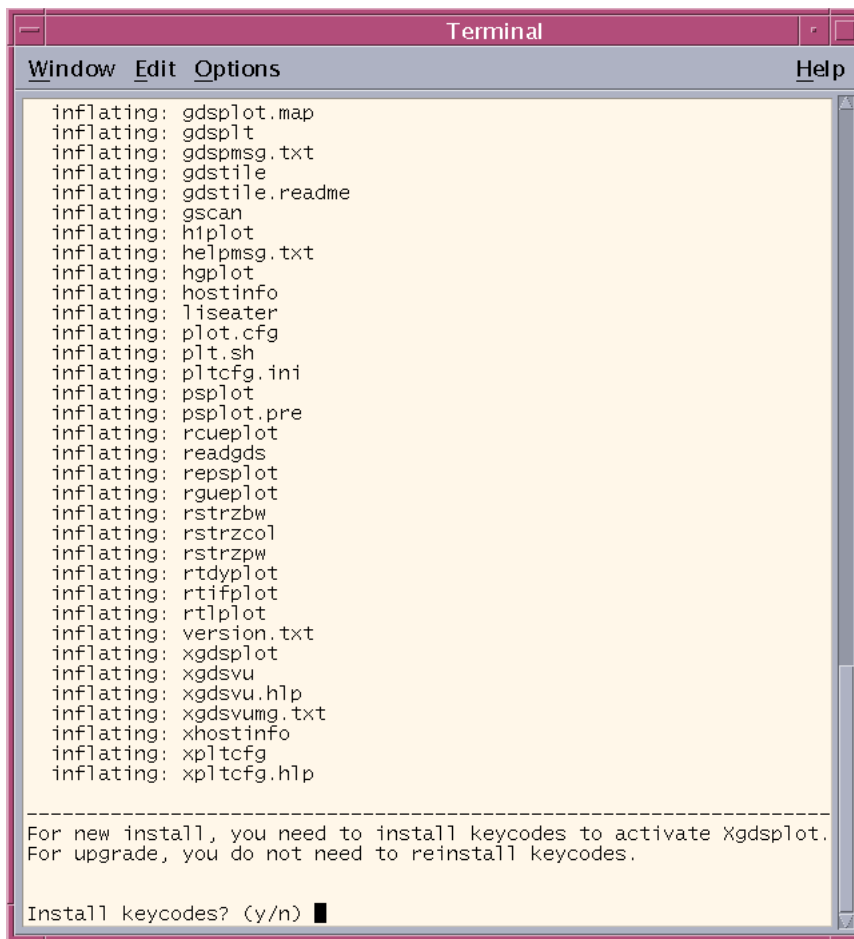
```
Terminal
Window Edit Options Help
$ ./INSTALL

-----
Artwork Conversion Software - Xgdsplot Installation
www.artwork.com info@artwork.com (831)426-6263
-----

Install Xgdsplot version 5.00? (y/n) y

Unzipping program files into the bin directory...

Archive: ../zipimage/xgdsplot.solaris.v500.zip
[../zipimage/xgdsplot.solaris.v500.zip] GDSVU.CFG password: █
```

A terminal window titled "Terminal" with a menu bar containing "Window", "Edit", "Options", and "Help". The main text area shows a list of files being inflated, each preceded by the word "inflating:". The files include: gdsplot.map, gdsplt, gdspsmsg.txt, gdstyle, gdstyle.readme, gscan, h1plot, helpmsg.txt, hgplot, hostinfo, liseater, plot.cfg, plt.sh, pltcfg.ini, psplot, psplot.pre, rcueplot, readgds, repsplot, rgueplot, rstrzbw, rstrzcol, rstrzpw, rtdyplot, rtifplot, rt1plot, version.txt, xgdsplot, xgdsvu, xgdsvu.hlp, xgdsvmg.txt, xhostinfo, xpltcfg, and xpltcfg.hlp. Below this list, a dashed line separates it from a message: "For new install, you need to install keycodes to activate Xgdsplot. For upgrade, you do not need to reinstall keycodes." At the bottom, a prompt asks "Install keycodes? (y/n) █".

```
inflating: gdsplot.map
inflating: gdsplt
inflating: gdspsmsg.txt
inflating: gdstyle
inflating: gdstyle.readme
inflating: gscan
inflating: h1plot
inflating: helpmsg.txt
inflating: hgplot
inflating: hostinfo
inflating: liseater
inflating: plot.cfg
inflating: plt.sh
inflating: pltcfg.ini
inflating: psplot
inflating: psplot.pre
inflating: rcueplot
inflating: readgds
inflating: repsplot
inflating: rgueplot
inflating: rstrzbw
inflating: rstrzcol
inflating: rstrzpw
inflating: rtdyplot
inflating: rtifplot
inflating: rt1plot
inflating: version.txt
inflating: xgdsplot
inflating: xgdsvu
inflating: xgdsvu.hlp
inflating: xgdsvmg.txt
inflating: xhostinfo
inflating: xpltcfg
inflating: xpltcfg.hlp

-----
For new install, you need to install keycodes to activate Xgdsplot.
For upgrade, you do not need to reinstall keycodes.

Install keycodes? (y/n) █
```

Assuming the password was entered correctly, the unzip program will inflate all of the files into the current directory.

Updating?

If you are upgrading software that has been previously installed (in this directory) then you are essentially complete. The previous key-code file will be present and should be still valid for the update.

New Installation?

If this is a new installation or if you are updating to a fresh directory then you need to install the keycodes.

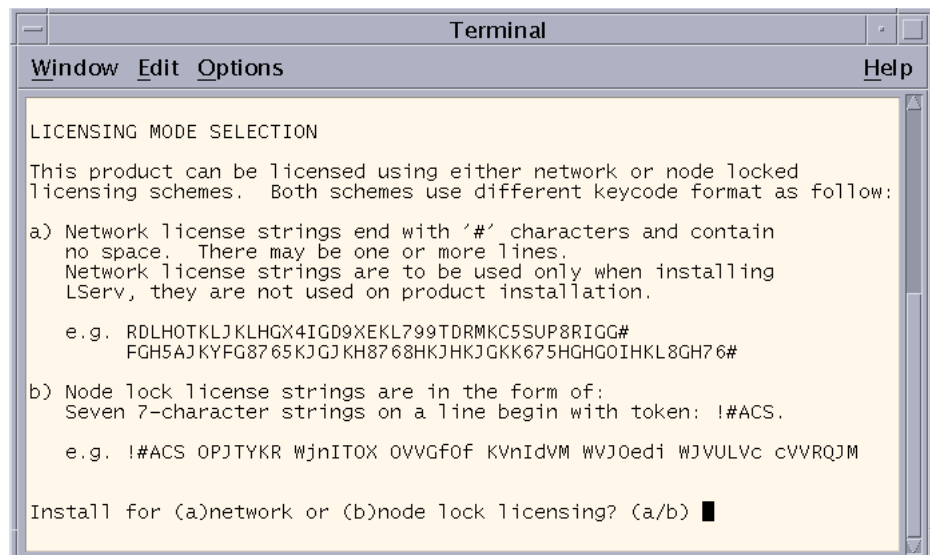
Network or Node Locked License?

There are two types of licensing.

Network - a license server program grants a license to anyone on the local network.

Node Locked - the application checks the local machine and compares the hostid of that machine against the hostid encrypted into the key strings.

You must answer this question correctly - otherwise the application will not run because your key codes will not properly authorize the correct license.

A terminal window titled "Terminal" with a menu bar containing "Window", "Edit", "Options", and "Help". The main text area displays "LICENSING MODE SELECTION" followed by an explanation: "This product can be licensed using either network or node locked licensing schemes. Both schemes use different keycode format as follow:". It then lists two options: (a) Network license strings end with '#' characters and contain no space. There may be one or more lines. Network license strings are to be used only when installing LServ, they are not used on product installation. An example is provided: "e.g. RDLHOTKLJ KLHGX4IGD9XEKL799TDRMKC5SUP8RIGG# FGH5AJ KYFG8765KJGJ KH8768HKJHKJGKK675HGHGOIHKL8GH76#". (b) Node lock license strings are in the form of: Seven 7-character strings on a line begin with token: !#ACS. An example is provided: "e.g. !#ACS OPJTYKR WjnITOX OVVGf0F KvnIdVM WVJOedi WJVULVc CVVRQJM". At the bottom, a prompt asks "Install for (a)network or (b)node lock licensing? (a/b) █".

```
LICENSING MODE SELECTION

This product can be licensed using either network or node locked
licensing schemes. Both schemes use different keycode format as follow:

a) Network license strings end with '#' characters and contain
no space. There may be one or more lines.
Network license strings are to be used only when installing
LServ, they are not used on product installation.

e.g. RDLHOTKLJ KLHGX4IGD9XEKL799TDRMKC5SUP8RIGG#
FGH5AJ KYFG8765KJGJ KH8768HKJHKJGKK675HGHGOIHKL8GH76#

b) Node lock license strings are in the form of:
Seven 7-character strings on a line begin with token: !#ACS.

e.g. !#ACS OPJTYKR WjnITOX OVVGf0F KvnIdVM WVJOedi WJVULVc CVVRQJM

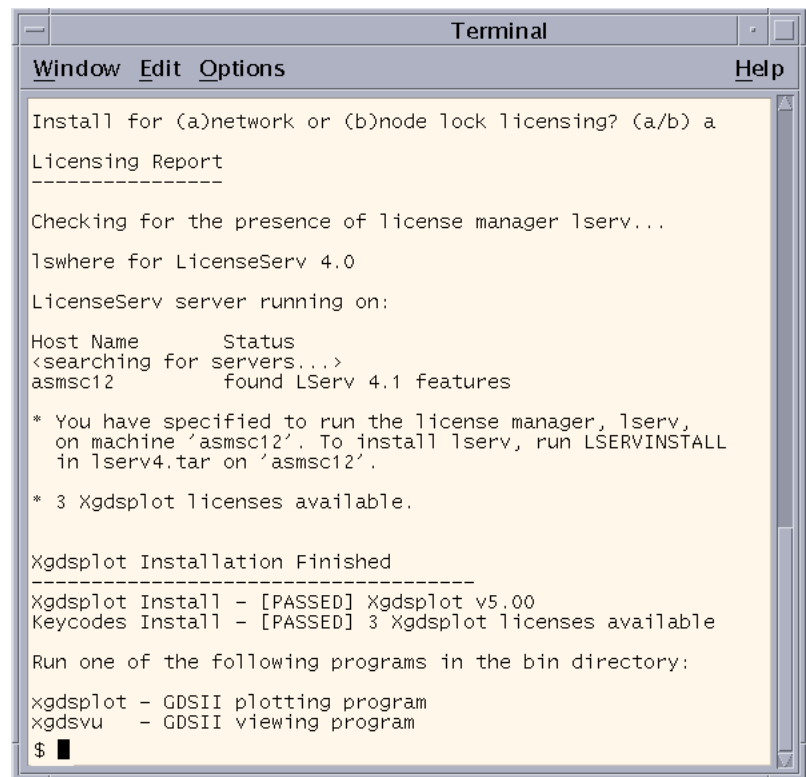
Install for (a)network or (b)node lock licensing? (a/b) █
```

Network Licensing Case I

You selected network licensing. Ideally the floating license manager has already been installed and is running correctly.

In that case the INSTALL script will attempt to find the license manager (i.e. which machine it is running on in your network) and then acquire a license for your program.

If it can do so you are essentially done.



```
Terminal
Window Edit Options Help

Install for (a)network or (b)node lock licensing? (a/b) a
Licensing Report
-----
Checking for the presence of license manager lserv...
lswhere for LicenseServ 4.0
LicenseServ server running on:
Host Name      Status
<searching for servers...>
asmsc12        found LServ 4.1 features

* You have specified to run the license manager, lserv,
  on machine 'asmsc12'. To install lserv, run LSERVINSTALL
  in lserv4.tar on 'asmsc12'.

* 3 Xgdsplot licenses available.

Xgdsplot Installation Finished
-----
Xgdsplot Install - [PASSED] Xgdsplot v5.00
Keycodes Install - [PASSED] 3 Xgdsplot licenses available

Run one of the following programs in the bin directory:
xgdsplot - GDSII plotting program
xgdsvu   - GDSII viewing program
$ █
```

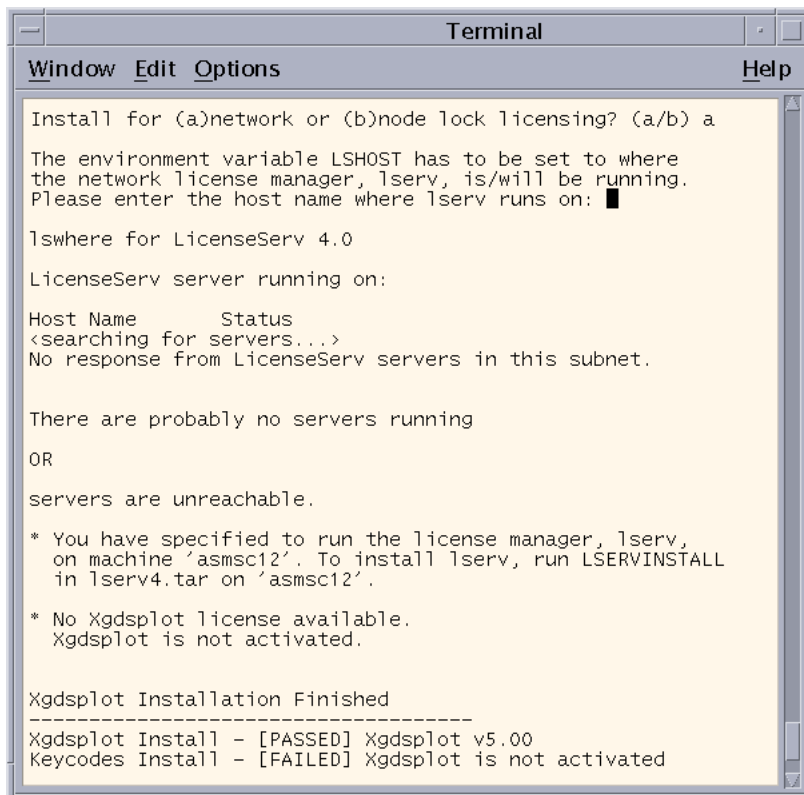
Network Licensing Case II

You selected network licensing. However the License Manager lserv has not yet been installed.

In that case the INSTALL script fails to find the license manager on the local subnet and instructs you to install it. To do so please see the section called Lserv Installation.

The application has been installed but you will not be able to run it at this point in time.

Once the license manager is running and you set the LSHOST variable on this machine to point to the machine running the license manager you will be OK.



```
Terminal
Window Edit Options Help

Install for (a)network or (b)node lock licensing? (a/b) a
The environment variable LSHOST has to be set to where
the network license manager, lserv, is/will be running.
Please enter the host name where lserv runs on: █

lswhere for LicenseServ 4.0
LicenseServ server running on:
Host Name      Status
<searching for servers...>
No response from LicenseServ servers in this subnet.

There are probably no servers running
OR
servers are unreachable.

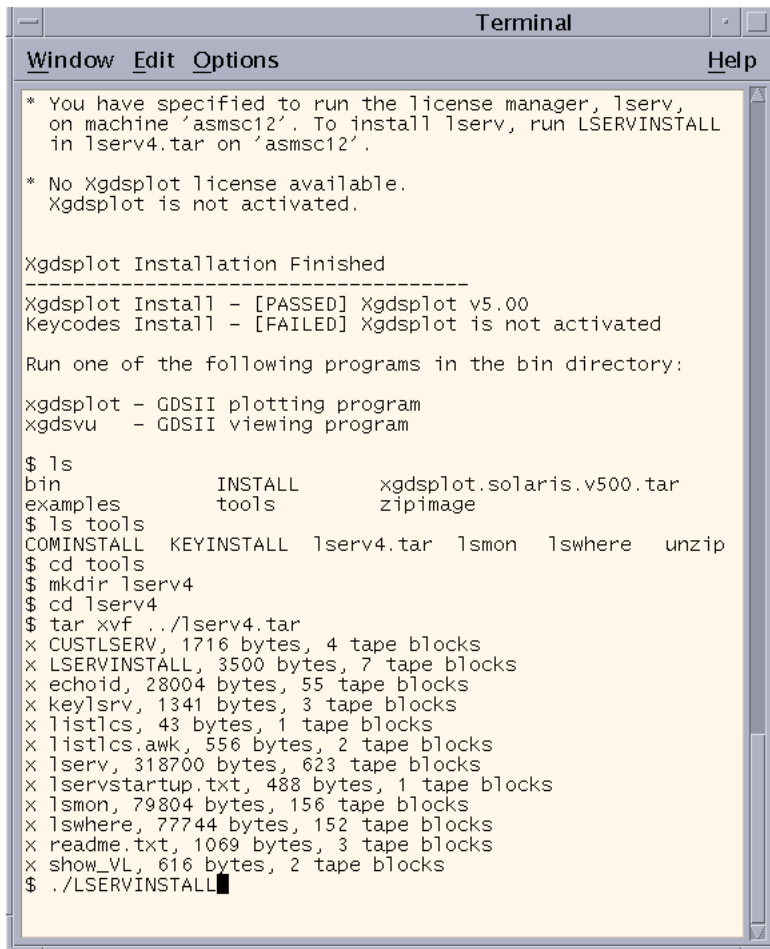
* You have specified to run the license manager, lserv,
  on machine 'asmsc12'. To install lserv, run LSERVINSTALL
  in lserv4.tar on 'asmsc12'.

* No Xgdsplot license available.
  Xgdsplot is not activated.

Xgdsplot Installation Finished
-----
Xgdsplot Install - [PASSED] Xgdsplot v5.00
Keycodes Install - [FAILED] Xgdsplot is not activated
```

Installing the License Manager

First, you need not install the license manager on the same machine that you are going to run the application. The license manager can be installed on any workstation on the local subnet as long as it is always on so that it can grant licenses to others at all times. This machine need not be a file server.

A terminal window titled "Terminal" with menu options "Window", "Edit", "Options", and "Help". The terminal displays the following text:

```
* You have specified to run the license manager, lserv,  
on machine 'asmsc12'. To install lserv, run LSERVINSTALL  
in lserv4.tar on 'asmsc12'.  
  
* No Xgdsplot license available.  
Xgdsplot is not activated.  
  
Xgdsplot Installation Finished  
-----  
Xgdsplot Install - [PASSED] Xgdsplot v5.00  
Keycodes Install - [FAILED] Xgdsplot is not activated  
  
Run one of the following programs in the bin directory:  
  
xgdsplot - GDSII plotting program  
xgdsvu   - GDSII viewing program  
  
$ ls  
bin          INSTALL      xgdsplot.solaris.v500.tar  
examples     tools          zipimage  
$ ls tools  
COMINSTALL  KEYINSTALL  lserv4.tar  lsmon  lswhere  unzip  
$ cd tools  
$ mkdir lserv4  
$ cd lserv4  
$ tar xvf ../lserv4.tar  
x CUSTLSERV, 1716 bytes, 4 tape blocks  
x LSERVINSTALL, 3500 bytes, 7 tape blocks  
x echoid, 28004 bytes, 55 tape blocks  
x keylsrv, 1341 bytes, 3 tape blocks  
x listlcs, 43 bytes, 1 tape blocks  
x listlcs.awk, 556 bytes, 2 tape blocks  
x lserv, 318700 bytes, 623 tape blocks  
x lservstartup.txt, 488 bytes, 1 tape blocks  
x lsmon, 79804 bytes, 156 tape blocks  
x lswhere, 77744 bytes, 152 tape blocks  
x readme.txt, 1069 bytes, 3 tape blocks  
x show_VL, 616 bytes, 2 tape blocks  
$ ./LSERVINSTALL
```

After you get the floating key codes, you may start with the license server installation. To start the installation type:

```
./LSERVINSTALL
```

```
Terminal
Window Edit Options Help

* No Xgdspplot license available.
  Xgdspplot is not activated.

Xgdspplot Installation Finished
-----
Xgdspplot Install - [PASSED] Xgdspplot v5.00
Keycodes Install - [FAILED] Xgdspplot is not activated

Run one of the following programs in the bin directory:

xgdspplot - GDSII plotting program
xgdsvu    - GDSII viewing program

$ ls
bin          INSTALL      xgdspplot.solaris.v500.tar
examples     tools          zipimage
$ ls tools
COMINSTALL  KEYINSTALL  lserv4.tar  lsmon      lswhere     unzip
$ cd tools
$ mkdir lserv4
$ cd lserv4
$ tar xvf ../lserv4.tar
x CUSTLSERV, 1716 bytes, 4 tape blocks
x LSERVINSTALL, 3500 bytes, 7 tape blocks
x echoid, 28004 bytes, 55 tape blocks
x keylsrv, 1341 bytes, 3 tape blocks
x listlcs, 43 bytes, 1 tape blocks
x listlcs.awk, 556 bytes, 2 tape blocks
x lserv, 318700 bytes, 623 tape blocks
x lservstartup.txt, 488 bytes, 1 tape blocks
x lsmon, 79804 bytes, 156 tape blocks
x lswhere, 77744 bytes, 152 tape blocks
x readme.txt, 1069 bytes, 3 tape blocks
x show_VL, 616 bytes, 2 tape blocks
$ ./LSERVINSTALL

Welcome to LServ 4.1 Installation (Oct 27, 1998)

Do you wish to check for an lserv running on this network <y or n>? n
```

If LSERV is not running on your network you may answer “no” to the question “do you wish to check for an lserv running on this network?”

Choose a Text Editor

You will need to identify the text editor you want to use. Enter the key codes to the text editor. A common editor is “vi”.

```
Terminal
Window Edit Options Help

Your editor is currently set to /usr/dt/bin/dtpad

Enter the name of another available editor on your system
or hit Return to continue

Editor Name: vi

Launching (vi) session to add license(s) to Licenserc file.

===== Hit Return to continue =====
```



```
Terminal
Window Edit Options Help

Customizing for /export/home/cad_util/gdsplot/tools/lserv4

To finish setting up the installed license
server to work with application programs,
the following conditions must be satisfied:

1. Environment variable LSHOST needs to be
   set on all client machines.

2. The license server needs to be setup to
   automatically start lserv at boot up time.

3. Lserv needs to be actively running all
   the time.

Refer to readme.txt for more information.

View readme.txt now <y or n>?y
```

At the end of the installation process, there are a few things that need to be done.

1. Define an environment variable called LSHOST as the name of the server.

2. Add a script to automatically start lserv upon booting the computer.

```
Terminal
Window Edit Options Help

View readme.txt now <y or n>?y

Environment Variable LSHOST
-----
Users should define an environment variable LSHOST having the
value of the server node name, e.g. asm001 (invoke uname -n)

Launching Lserv at Startup
-----
1. Create a S86lserv script file in /etc/rc3.d, with the text
   contained in the file lservstartup.txt, to automatically
   start up the license server at boot time.

2. The lserv server uses the network port number 5093. System
   Administrator should add the following line to the
   /etc/services file.

   LicenseServ    5093/udp

Trouble Shooting
-----
1. Having the license server program "lserv" be an active
   process. This can be discerned with the "ps" command

   ps -aef

   It is most quickly found with the following lengthy command

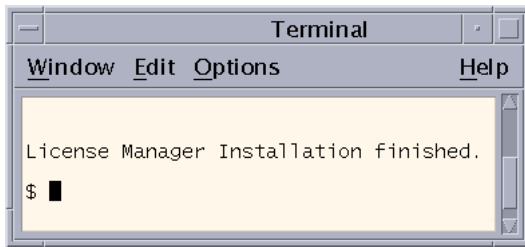
   ps -aef | grep lserv | grep -v grep

   and should result in output like the following:

   root  5073      1  0 12:12:54 ttty4      0:00 ./lserv -s ./Licenserc

2. The file Licenserc (keycode file) must reside together in the
   same directory where lserv resides.

===== Hit Return to continue =====
```



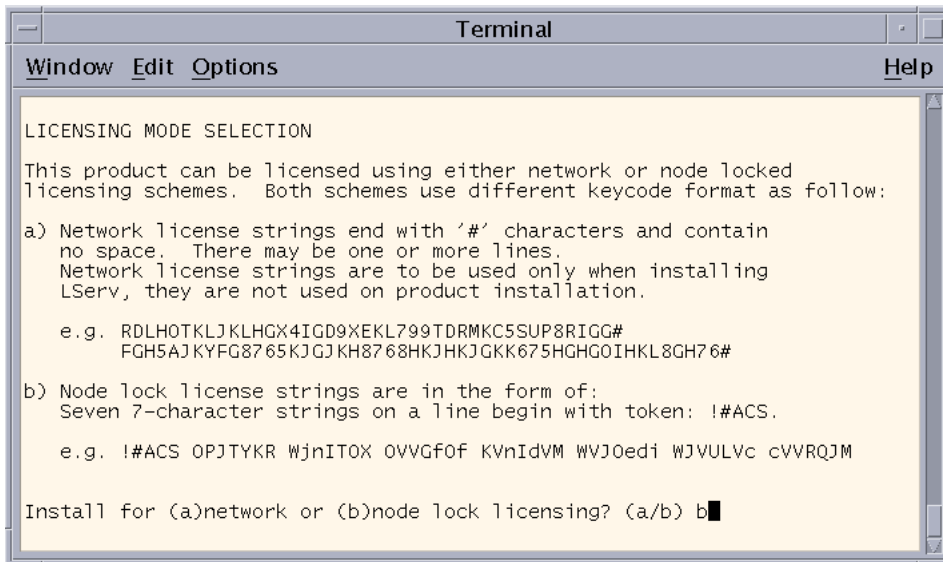
```
Terminal
Window Edit Options Help

License Manager Installation finished.
$ █
```

Now that the License Manager installation is complete you may start the program.

Node Lock Installation

In the case of node locked installation you must run the installation on the system to which the codes are licensed.



```
Terminal
Window Edit Options Help

LICENSING MODE SELECTION

This product can be licensed using either network or node locked
licensing schemes. Both schemes use different keycode format as follow:

a) Network license strings end with '#' characters and contain
no space. There may be one or more lines.
Network license strings are to be used only when installing
LServ, they are not used on product installation.

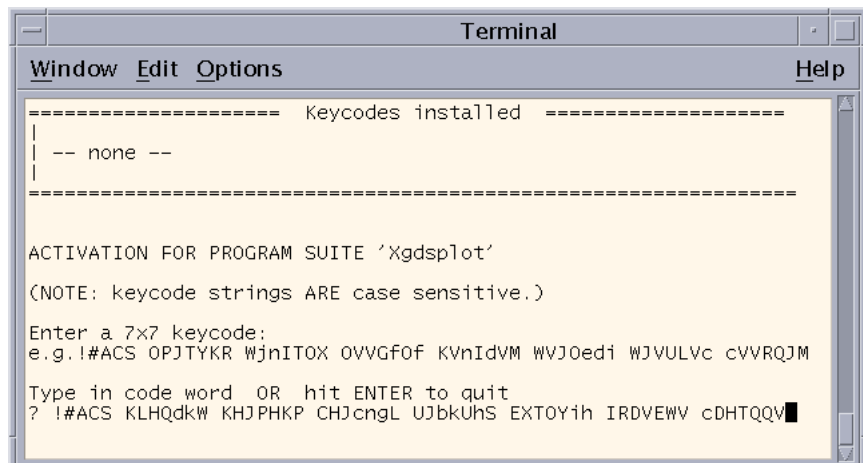
e.g. RDLH0TKLJKLHGx4IGD9XEKL799TDRMKC5SUP8RIGG#
FGH5AJKYFG8765KJGJKH8768HKJHKJGKK675HGHGOIHKL8GH76#

b) Node lock license strings are in the form of:
Seven 7-character strings on a line begin with token: !#ACS.

e.g. !#ACS OPJTYKR WjnITOX OVVGfOf KvIdVM WVJOedi WJVULVc cVVRQJM

Install for (a)network or (b)node lock licensing? (a/b) b █
```

Pick “b” to continue with node locked installation. At this point you need to copy and paste the key codes. Do not try to type it! (This is because of errors that can occur such as typing in the number 1 instead of the letter l). After copying the codes, hit enter to accept them and to continue with the installation.



```
Terminal
Window Edit Options Help

----- Keycodes installed -----
|
| -- none --
|
-----

ACTIVATION FOR PROGRAM SUITE 'XgdspIot'
(NOTE: keycode strings ARE case sensitive.)

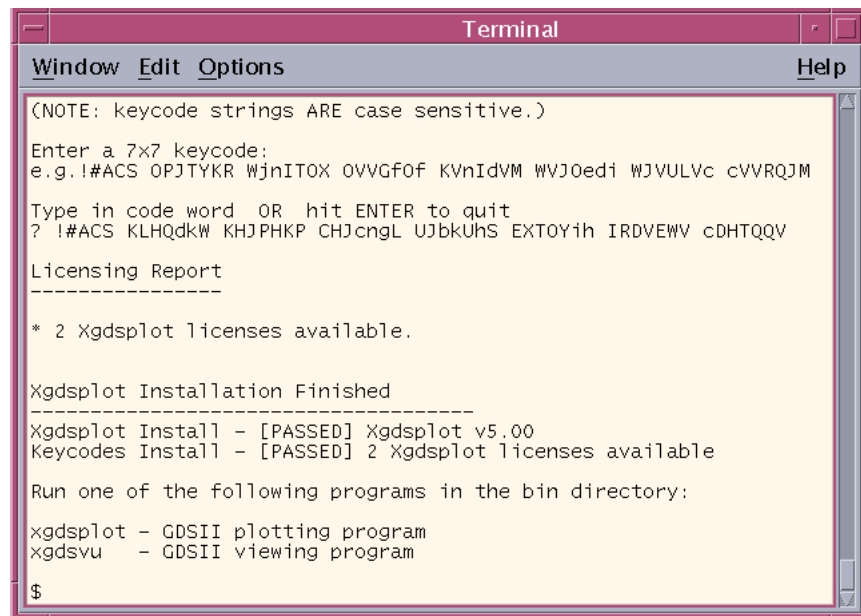
Enter a 7x7 keycode:
e.g. !#ACS OPJTYKR WjnITOX OVVGfOf KvIdVM WVJOedi WJVULVc cVVRQJM

Type in code word OR hit ENTER to quit
? !#ACS KLHQdkW KHJPHKP CHJcngL UJbkUhs EXT0Yih IRDVEWV cDHTQQV █
```


The installation script will evaluate the codes entered and give you a report of which programs got activated.

If all goes well, the installation will finish, and you will be ready to use the program.

If problems persist, try to reinstall the program and if the program still does not work consult the technical support appendix.

A screenshot of a terminal window titled "Terminal". The window has a menu bar with "Window", "Edit", "Options", and "Help". The terminal text shows the execution of an installation script. It starts with a note about case sensitivity for keycode strings. The user enters a 7x7 keycode: "e.g. !#ACS OPJTYKR WjnITOX OVVGfOf KvIdVM WVJOedi WJVULVc cVVRQJM". The script then prompts for a code word or to hit ENTER to quit. The user enters " ? !#ACS KLHQdkW KHJPHKP CHJcngL UJbkUoS EXT0Y1h IRDVEWV cDHTQQV". The script displays a "Licensing Report" showing that 2 Xgdsplot licenses are available. It then announces "Xgdsplot Installation Finished" and shows the results of the installation: "Xgdsplot Install - [PASSED] Xgdsplot v5.00" and "Keycodes Install - [PASSED] 2 Xgdsplot licenses available". Finally, it lists the programs to run: "xgdsplot - GDSII plotting program" and "xgdsvu - GDSII viewing program". The prompt "\$" is visible at the bottom.

```
(NOTE: keycode strings ARE case sensitive.)

Enter a 7x7 keycode:
e.g. !#ACS OPJTYKR WjnITOX OVVGfOf KvIdVM WVJOedi WJVULVc cVVRQJM

Type in code word OR hit ENTER to quit
? !#ACS KLHQdkW KHJPHKP CHJcngL UJbkUoS EXT0Y1h IRDVEWV cDHTQQV

Licensing Report
-----

* 2 Xgdsplot licenses available.

Xgdsplot Installation Finished
-----
Xgdsplot Install - [PASSED] Xgdsplot v5.00
Keycodes Install - [PASSED] 2 Xgdsplot licenses available

Run one of the following programs in the bin directory:

xgdsplot - GDSII plotting program
xgdsvu   - GDSII viewing program

$
```


Windows Installation

Hardware/OS Requirements2-2
Supporting DLL's and Drivers2-2
Installation Procedure2-3
Destination2-3
Components to Install2-3
Security2-3
Host ID2-3
Sentinel Parallel Port Key2-3
Floating Network License2-3
Key Strings2-4
Modifying the Key Strings2-4
The Hardware Key2-4
Damage to the Hardware Key2-4
Hardware Key Conflicts2-5
Parallel Port Switch Box2-5
Host ID Locking - Replacing the Hardware Key with your Network Card2-5
Getting your Host ID2-5
Common Problems and Solutions2-6

Artwork Conversion Software, Inc.

417 Ingalls St.
Santa Cruz, CA 95060
(831) 426-6163
support@artwork.com

Chapter 2

Windows Installation

Installation of Artwork's Windows based programs is the same for all applications. We use ***Install Shield*** to uncompress files from the CD or Internet download and copy them to the directory of your choice, install required DLL's, create the program group, update the registry and install the example files. You attach the "hardware key" to the parallel port and enter the "key strings" to complete the installation.

Hardware/OS Requirements

- Pentium Class or higher CPU.
- 64 Mbytes RAM recommended.
- 256 color display with at least 800 x 600 resolution. Our dialog boxes will not be fully visible on a 640 x 480 display.
- 5-10 Mbytes of disk space.
- Windows 95/98/NT4/2000.
- Mouse or other pointing device.
- Parallel Port for the Hardware Key or Network Card for MAC address locking.

Supporting DLLs and Drivers

- **Sentinel Driver for NT.** Since our programs cannot access the parallel port directly on NT a special driver provided by Rainbow (our key mfg) must be installed. This is automatically installed¹ if you have selected Hardware Key (Sentinel) Locking and you are running on NT.

¹: Some users have machines set up in a way that they do not have write privileges to certain directories including the Windows system directory. In that case it may not be possible for the Install Shield program to load drivers such as the Sentinel Driver. If this occurs you need to arrange with your system administrator to run the installation again with full privileges.

Installation Procedure

- 1 Insert the CD into your drive
- 2 Double click on the setup.exe (or on the self extracting .exe for downloads)
- 3 Follow the screen instructions and prompts. You can select the drive and directory where the programs will be installed. Example files are usually installed in a subdirectory called examples under the program directory. You can override the default selections. by pressing **Browse...** and choosing a new destination.



Figure 2-1. When the Destination Folder Menu comes up, the user can choose where the program will be installed.

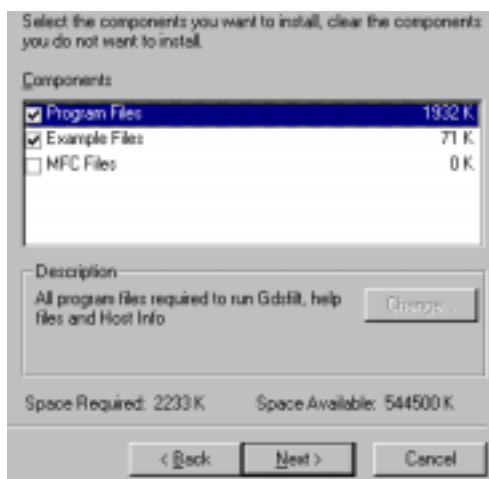


Figure 2-2. This screen shows an example of how the user can select which components to install.

Components

You can also select which components to install. Normally you do not need to install the examples during an upgrade.

Security

There are three different types of security:

Host ID - a unique serial number is derived from your network card's MAC address (note on NT machines with two network cards this can fail)

Sentinel Parallel Port Key - a Sentinel-C dongle is attached to one of the parallel ports. This provides a unique serial number for the application.

Floating Network License - a license server runs a program known as a license manager. The LM gets requests from the application over the network and returns permission to execute.

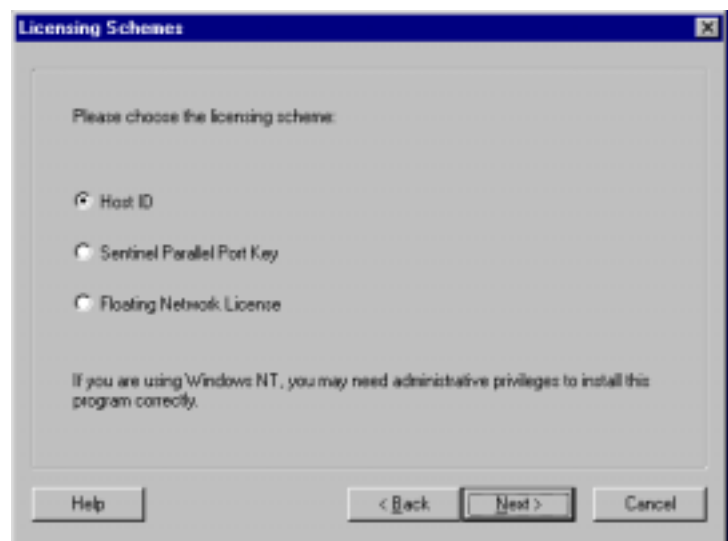


Figure 2-3. The Installation Licensing Schemes Menu.

When you purchase our software you select which type of licensing to use. The keycodes that are generated reflect the type of licensing. During the Install you must enter the type of security to be used.

Key Strings

If this is your first installation please type or paste them into the box provided. If you are updating or otherwise overwriting the execs you should not need to re-enter the key strings a second time as long as you install into the same directory as before.

```
EVdYjWJ OJDGfFR YDNXcJM UNXcUdV OXTUXMi MTNeRWX cDFRPKL
```

Our key strings consist of 7 groups of 7 characters. No numerals are used so it is impossible to confuse the letter “l” (larry) and the number “1” (one) or the number “0” (zero) and the letter “O” (Oscar). Please pay attention to upper and lower case since these do make a difference.

If possible obtain your keystings by email so that you can paste them into the box. Typing is prone to errors.

The key string you entered is stored in a file called **ACS.KEY**. The key file must be located in the same directory as the executables. Inside the file you will find an entry such as:

```
!#ACS EVdYjWJ OJDGfFR YDNXcJM UNXcUdV OXTUXMi MTNeRWX cDFRPKL
```

Modifying the Key Strings

If you need to modify a key string after the program is installed you can do so from the **Help** pulldown within your application as shown here:

Or you can use a text editor such as the notepad to directly edit the ACS.KEY file. Either approach will work.

The Hardware Key

Artwork’s programs can be secured using a hardware key that attaches to the parallel port. The key will work on LPT1, LPT2 or LPT3. If a printer was attached to the port re-attach it behind the key. The printer must be turned on for the key to work.

Damage to Hardware Key

The **Iomega parallel port ZIP drive** is known to damage the Sentinel-C key when attached behind it. If you must use the parallel Iomega Zip drive get a second parallel port and put the hardware key on a separate port.

The keys are fairly rugged; however we have found that moving them from machine to machine on a daily basis will eventually damage the key either through static electricity or by breaking a pin. Accidentally inserting a key into a serial port could also burn out the key since serial ports can have +12 and -12 volts on some pins.

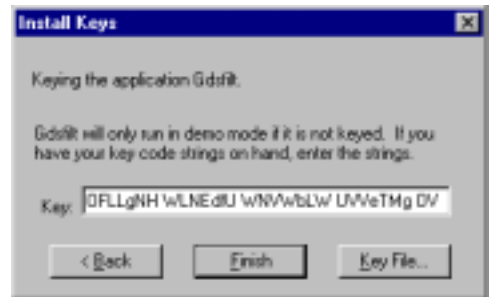


Figure 2-4. The Install Keys menu.

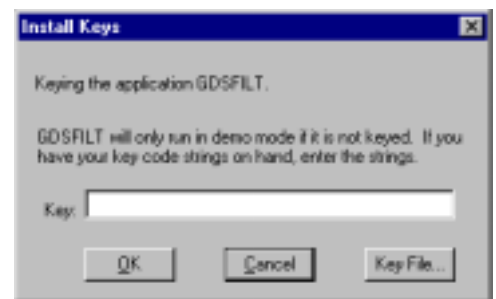


Figure 2-5. The Install Keys menu can also be accessed from the Help pulldown.

Hardware Key Conflicts

Our hardware key may or may not work in conjunction with other keys on the same port. Sometimes the order in which they are attached can make a difference. **If you encounter a conflict the best solution is to put our key on a second parallel port.** Generally speaking the Rainbow Sentinel hardware key does not work on the same parallel port together with other Rainbow Sentinel keys but does work on the same port with Rainbow Pro and SuperPro keys.

You can also avoid the hardware key altogether by using **hostid** locking to your network card's MAC address as described later in this chapter.

Parallel Switch Box?

A parallel switch box can also be used to select between two keys - however this may be of limited value if you need to toggle between two programs since our program does check the key during program operation and not just at start up.

Host ID Locking - Replace the Key with Your Network Card!

Artwork has developed a technique to lock its Windows based software to a network card installed in the computer instead of using a hardware key (dongle). From the network card's MAC address we derive a **hostid** which substitutes for the dongle's serial number.

Getting Your HostID

You can use the program called hostinfo.exe to get the hostid of your machine.. If it is able to read the hostid you will see a screen as follows:

You need to send us the Hostid address of your machine. You can cut and paste this into your email tool or into notepad or other text editor as needed.

In general, it will be able to get a reading if you have a network card installed and TCP/IP services enabled.

Hostid Locking does not always work even if you have a network card installed. It depends on Windows winsock.dll services and there are configurations where networking is configured in a fashion that prevents our program from obtaining a valid MAC address. For more details check our WEB site at

www.artwork.com/support/index.htm.

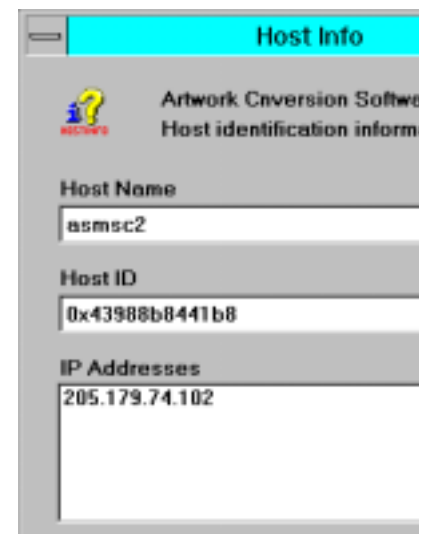


Figure 2-6. The Host Info Menu.

Common Problems and Solutions

Here are the most common installation problems reported by our users and the solution. Please go through this list prior to calling tech support.

Problem: Program runs in demo mode because it cannot find a valid key string.

Reason The key string file is missing. Look for the file called ACS.KEY . If not present you can run the key install from the Help pull down on your program.

Reason The key string file is not located in the same directory as the executables. Using the file manager make sure that the key string file ACS.KEY is located in the same directory as the executable. If you have more than one translator from Artwork insure that you have an ACS.KEY file in each executable directory and that each ACS.KEY file contains the correct key strings for that program.

Reason The keystring is typed in incorrectly. Fix it with a text editor or re-enter from the Help pull down. For example, we have noticed many users do not type in the blank space between each group of characters.

Reason The keystring and the program executed do not match. Find out why. You may have entered a key string for a different program. Remember that different Artwork Windows' programs should be installed in different directories.

Reason The keystring and the hardware key serial number do not match.

Problem: The hardware key is attached but cannot be read.

Reason Another hardware key is on the same parallel port and is interfering. Our hardware keys will work on LPT1:, LPT2: and LPT3:. Try moving one key to another port.

Reason The printer is attached behind the key and turned off thus shunting the key. Turn on the printer.

Reason certain laptops have non standard parallel ports that don't work. No easy solution here.

Reason The hardware key is not attached to the parallel port but to some other port.

Reason The hardware key has been damaged - usually either a pin is bent or cracked or static electricity has killed it. This is most likely if it is often moved from machine to machine.

Reason You are running on Windows NT and Install Shield could not install the Sentinel drivers because your system directory is read/execute only. Solution: Get your sysadmin to run the Install Shield program using his logon which should have full privileges.

z The hardware key is not attached at all.

Chapter 3

Opening a GDSII File - Commands and Functions

Open_GDSII

Arguments

String, the name of the stream file to open. May use double quotes if the file name has a space in it. The string is case sensitive in UNIX.

Example

Client	Server
-----	-----
Open_GDSII \n	
"demo5.gds" \n	
	Open_GDSII \n
Open_GDSII \n	
"c:\My Documents\demo5.gds" \n	
	Open_GDSII \n

Errors

Client	Server
-----	-----
Open_GDSII \n	
Set_Fill \n	
	QIS Error \n
	No argument provided \n

(If no argument is received)

Open_GDSII \n	
demo.6b.gds \n	
	QIS Error \n
	File does not exist \n

(If the selected file cannot be found. Make sure you have typed it correctly and specified the correct path.)

Open_GDSII \n	
demo5.txt \n	
	QIS Error \n
	File is not a GDSII file \n

(If the file specified is not a GDSII file.)

Open_GDSII \n	
demo5.gds	
	QIS runs out of memory during loading \n

Notes

QIS does not provide a directory service so the client needs some other method of knowing what files are available to QIS. A full path can be part of the file name. **Note:** This is only an issue if QIS is run on a different machine than the client.

A large file may take minutes to scan and load - QIS can provide progress information during this time.

Once the file is successfully scanned and loaded Qckvu will return an Open_GDSII.

Error and Warning Messages when Opening a GDSII File

Errors are fatal; nothing can be loaded. Warnings are not fatal. The file can be loaded but one should evaluate the warning to determine whether you wish to rely on the data.

Errors

More than 4GB of RAM needed to open the file

```
QIS_Error\n
xxxGB of memory is needed to view /fullpath/xxx.gds.\n
Maximum memory is 4GB.\n
```

If system does not have enough memory to load the GDSII file, the memory that would be needed is reported in GB or MB.

```
QIS_Error\n
x.xxGB of memory is needed to view /fullpath/xxx.gds.\n
The system does not have enough memory.\n
```

or

```
QIS_Error\n
x.xMB of memory is needed to view /fullpath/xxx.gds.\n
The system does not have enough memory.\n
```

If the requested GDSII file cannot be opened.

```
QIS_Error\n
The selected file 'xxx.gds' is invalid or empty.\n
```

If the file does not start with 0602, then it is not a valid GDSII file.

```
QIS_Error\n
The selected file 'xxx.gds' is not a GDSII file.\n
```

No structure definitions

```
QIS_Error\n
No structure defined in the GDSII file.\n
```

Memory related error during scanning.

```
QIS_Error\n
Memory error occurred during scanning.\n
```

Warnings

If a record cannot be read completely

```
QIS_Warning\n
Incomplete data at file position 123456, file might be truncated.\n
```

End of library record missing

```
QIS_Warning\n
Missing end of library.\n
```

QIS will treat the file as if the library was correctly closed. However, such a warning is an indication that the file might not be complete.

Structure is not closed with the "end_structure" record.

```
QIS_Warning\n
Missing end of structure for structure 'ABC'.\n
```

QIS will close the structure at the end of the library but this is a sign that the file may have been incorrectly created.

Structure definitions inside a structure.

GDSII does not allow a structure definition to be embedded within a structure definition. Should it detect a second begin_structure record before an end_structure record it will issue a warning:

```
QIS_Warning\n
Illegal structure definition at file positions:\n
ABC,24\n
DEF,789\n
JUNK,55987\n
```

These "extra" structure definitions are ignored; data following such definitions belong to the structure definition before it. Only the first 20000 illegal structure definitions are reported.

```
Structure Definition ABC
Path 1
Boundary 1
Structure Definition DEF
Path 2
End of Structure

DEF is ignored and path 2 belongs to ABC.
```

End structure record without a matching begin structure

```
QIS_Warning\n
Illegal end of structure at file positions:\n
248\n
77668\n
```

These unmatched end of structure records are ignored. Only the first 20000 illegal end of structures are reported.

Data records defined outside of the begin_structure--end structure bound.

Any such data is ignored and the following type of error message is returned.

```
QIS_Warning\n
Illegal data at file positions:\n
Boundary,123\n
Layer,177\n
Path,456\n
Width,480\n
Text,789\n
Structure Reference,4321\n
Structure Reference Name,4330\n
Array Reference,6677\n
Path,9900\n
Path,11000\n
Path,33333\n
```

Only the first 20000 illegal data/record are reported.

Structures referenced but not defined in the GDSII file

```
QIS_Warning\n
Undefined Structures\n
StructureABC,StructureXXX\n
```

Structures defines more than once with the same name

```
QIS Warning\n
Multiply_defined_structures\n
StructureABC, StructureXXX\n
```

Set_Progress_Message

Arguments

Boolean - On or Off

If On, then the server will return a string over and over again indicating progress (an estimate) in first scanning the file and then loading it. The client can use this information to update the user so that the user does not feel that the program is not responding. This is needed most when the files are very large -- say hundreds of MB and above.

Example

Client	Server
-----	-----
Set_Progress_Message \n	
On \n	
Open_GDSII \n	
"C:\demo5.gds"	
	Set_Progress_Message \n
	Scanning 10% \n
	Set_Progress_Message \n
	Scanning 25% \n
	Set_Progress_Message \n
	Scanning 40% \n
	Set_Progress_Message \n
	Scanning 60% \n
	Set_Progress_Message \n
	Set_Progress_Message \n
	Finished Scanning in XXX min XXX sec \n
	Scanning 80% \n
	Set_Progress_Message \n
	Loading 10% \n
	Set_Progress_Message \n
	Loading 20% \n
	Set_Progress_Message \n
	Loading 40% \n
	Set_Progress_Message \n
	Finished Loading in XXX min XXX sec \n

Errors

Client

Set_Progress_Message \n

Set_Fill \n

Server

QIS_Error \n

Missing argument - On/Off \n

(If no argument is received)

Set_Progress_Message \n

Of \n

QIS_Error \n

Invalid argument. Argument - On/Off \n

(If an invalid argument is received)

Notes

Scanning and loading percentage messages are sent every 0.5 seconds.

Chapter 4

Structures - Commands and Functions

Open_Structure

Arguments

String, the name of the structure to open. May use double quotes if the structure name has a space in it. The string is case sensitive.

Example

```
Client                               Server
-----                             -
Open_Structure \n
"TOPMOSTST" \n

Errors

Client                               Server
-----                             -
Open_Structure \n
Set_Fill \n

                                   QIS_Error \n
                                   Missing structure name \n
                                   open_structure \n
```

(When no argument is received)

```
Open_Structure \n
NONAME \n

                                   QIS_Error \n
                                   Structure 'NONAME' does not exist \n
                                   open_structure \n
                                   NONAME \n
```

(When the argument received is an invalid structure name)

Notes

Normally a client would get a list of structures using the `get_structure_list` command and possibly also get the top level structure. Then a user can be presented from a list of available structures.

Get_Structure_List

Arguments

none

Example

Assume that a file has 5 structures in it named Aa,Bb,Cc,Dd and Ee.

```
Client                               Server
-----                             -
Get_Structure_List \n

                                   Get_Structure_List \n
                                   Aa,Bb,Cc,Dd,Ee \n
```

Errors

No errors

Notes

Structures names are returned in the order in which they are defined in the GDSII file and are separated by commas.

A large GDSII file can easily have 10,000 structures each with a name of 32 characters so the client should be prepared to receive a large stream of data. We recommend allowing support for up to 64,000 structures.

Get_Structure_Extents

Arguments

name of structure

Example

Client	Server
-----	-----
Get_Structure_Extents \n	
"TOP"	
	Get_Structure_Extents \n
	50.002,-10.340,400.000,680.025 \n

Errors

Client	Server
-----	-----
Get_Structure_Extents \n	
Get_Structure_List \n	
	QIS_Error \n
	Missing structure name \n

(When no structure name is received)

Get_Structure_List \n	
NONAME \n	
	QIS_Error \n
	Structure 'NONAME' does not exist \n

(When an invalid structure name is received)

Notes

Returns the coordinates of the extents of the structure; either in DBU or in UU depending on which is in effect. LLx,LLy,URx,URy

Get_Structure_Children

Arguments

name of structure

Example

Client	Server
-----	-----
Get_Structure_Children \n	
"TOP"	
	Get_Structure_Children \n
	MULTIPLIER, ADDER, IOBUFFER \n
Get_Structure_Children \n	
"MULTIPLIER"	
	Get_Structure_Children \n
	AND, OR, FLIPFLOP \n

Errors

Client	Server
-----	-----
Get_Structure_Children \n	
Get_Structure_Root \n	
	QIS_Error \n
	Missing structure name \n
(When no structure name is received)	
Get_Structure_Children \n	
topmost \n	
	QIS_Error \n
	Structure 'topmost' does not exist \n

(When an invalid structure name is received)

Notes

Returns a list of any structures referenced by the named structure. Only structures immediately referenced are named -- not those more than 1 level down in the hierarchy.

Get_Structure_Root

Arguments

No arguments

Example

Client	Server
-----	-----
Get_Structure_Root \n	
	Get_Structure_Root \n
	TOPMOSTST, TOPMOST_REVA \n

Errors

No errors

Notes

GDSII files can have more than one "structure" root. The root or top level structure is the one that contains the most unique references to other structures.

However, a structure which is not referred to by any other structure is also considered at structure root.

Get_Structure_Tree

Use

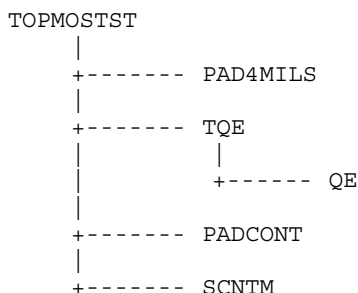
This command is used to get the tree - the list of structures that are referenced by your specified structure. You control how far down in the hierarchy QIS looks either by explicitly specifying a nesting level or if not specified, the current nesting level setting is used.

Arguments

structurename,nestinglevel \n
(nesting level is optional)

Example

From demo1.gds - This is the structure tree for the output that follows:



Client

```
Get_Structure_Tree \n
TOPMOSTST \n
```

Server

```

Vector_Data \n
S,PAD4MILS,TOPMOSTST,0.000,0.000,1.00000,0.00000,N,5,
0.000 0.000 99.000 0.000 99.000 99.000 0.000 99.000 0.000

0.000 \n

Vector_Data \n
S,TQE,TOPMOSTST,124.500,135.000,1.00000,90.00000,N,5,
132.000 127.500 132.000 150.000 117.000 150.000 117.000

127.500 132.000 127.500\n

Vector_Data\n
S,QE,TQE,124.500,135.000,1.00000,0.00000,N,5,
132.000 127.500 132.000 147.000 117.000 147.000 117.000

127.500 132.000 127.500\n

Vector_Data\n
S,PADCONT,TOPMOSTST,-1.500,79.500,1.00000,90.00000,N,5,
0.500 77.500 0.500 96.500 -18.500 96.500 -18.500 77.500

0.500 77.500\n

Vector_Data\n
S,SCNT,TOPMOSTST,124.500,97.500,1.00000,90.00000,N,5,
126.000 96.000 126.000 99.000 123.000 99.000 123.000 96.000

126.000 96.000\n

Get_Structure_Tree\n
```

If you send:

```
Get_Structure_Tree \n
TOPMOSTST,2 \n
```

2 means up to the second level of hierarchy,
the output would become this:
(without the QE reference because it's on the 3rd level)

```

Vector_Data\n
S,PAD4MILS, TOPMOSTST, 0.000, 0.000, 1.00000, 0.00000, N, 5,
0.000 0.000 99.000 0.000 99.000 99.000 0.000 99.000 0.000 0.000\n
Vector_Data\n
S,TQE, TOPMOSTST, 124.500, 135.000, 1.00000, 90.00000, N, 5,
132.000 127.500 132.000 150.000 117.000 150.000 117.000 127.500
132.000 127.500\n
Vector_Data\n
S,PADCONT, TOPMOSTST, -1.500, 79.500, 1.00000, 90.00000, N, 5,
0.500 77.500 0.500 96.500 -18.500 96.500 -18.500 77.500 0.500
77.500\n
Vector_Data\n
S,SCNT, TOPMOSTST, 124.500, 97.500, 1.00000, 90.00000, N, 5,
126.000 96.000 126.000 99.000 123.000 99.000 123.000 96.000
126.000 96.000\n
Get_Structure_Tree\n

```

(For detailed Structure Syntax, please see The Structure Reference Section page 8-6)

Errors

```

Client                               Server
-----                               -
Get_Structure_Tree \n
Get_Structure_List \n

QIS_Error \n
Missing structure name \n

```

(If no argument is received)

```

Get_Structure_Tree \n
NONAME \n

QIS_Error \n
Structure 'NONAME' does not exist \n

```

(If the argument is an invalid structure name)

```

Get_Structure_Tree \n
TOP,2 \n

QIS_Error
Invalid argument.  Argument format - string,All|number

```

(This is a possible error if the 2nd argument is provided but it's not a nesting value)

Notes

Use this command to get all structure references of a structure. It traverses down the hierarchy as specified by the current nesting level, or down to the specified level if provided. All references are output, the viewing window and array mode do not filter out the output.

The nesting level is either a level number of the keyword All if all levels are to be traversed. Level 0 and 1 would not generate any references because there are no references on those 2 levels.

Get_Structure_References

Use

This command is used to determine how many times and where a specified structure is inserted into the display window. You can get all of the instances or just one. You can also specify how far down the hierarchy tree to check.

Arguments

A structure name

Nesting level (All for all levels or a number for a specified level)

Example

```
Client                               Server
-----
Get_Structure_References \n
PAD4MILS,All \n

Vector_Data \n
S,PAD4MILS,TOPMOSTST,103.500,108.000,1.00000,90.00000,N,5,
105.000 108.000 105.000 127.500 90.000 127.500 90.000 108.000
105.000 108.000\n
Vector_Data \n
S,PAD4MILS,TOPMOSTST,157.500,108.000,1.00000,90.00000,N,5,
159.000 108.000 159.000 127.500 144.000 127.500 144.000
108.000 159.000 108.000\n
Get_Structure_References \n
```

(For detailed Structure Syntax, please see The Structure Reference Section page 8-6)

Errors

```
Client                               Server
-----
Get_Structure_References \n
Get_Structure_Tree \n

QIS_Error \n
Missing structure name \n

(If no argument is received)

Get_Structure_References \n
NONAME \n

QIS_Error \n
Structure 'NONAME' does not exist \n

(If the argument is an invalid structure name)

Get_Structure_Tree \n
TOP,2 \n

QIS_Error \n
Invalid argument.  Argument format - string,All|number \n
```

(This is a possible error if the 2nd argument is provided but it's not a nesting value)

Notes

Returns any SREF's of the structure the client has specified found within the "display" window.

Set_Structure_Outline

Arguments

Off / All / list of hierarchy levels - comma separated e.g. 1,2,3

Example

```
Client                               Server
-----                             -
Set_Structure_Outline \n
Off \n
```

or

```
Set_Structure_Outline \n
All \n
```

or

```
Set_Structure_Outline \n
2,3,4,5,6,7,8 \n
```

Errors

```
Client                               Server
-----                             -
Set_Structure_Outline \n
Get_Structure_Tree \n
```

```
Missing argument - Off/All/list of hierarchy level(s) \n
Set_Structure_Outline \n
```

(If no argument is received)

```
Set_Structure_Outline \n
A,2,B,C
```

```
Invalid argument.  Argument format - Off/All/list of hierarchy
level(s)
set_structure_outline
A,2,B,C
```

(If an invalid argument is received)

Notes

Use this command to control whether to draw an extent outline of the structure references on the specified levels. This command is controlled using hierarchy - to draw outline bounding box for references on the first level of the current viewing structure, pass in 1; for the second level, pass in 2. Multiple levels can be passed. e.g. 1,2,3,7,8. Related to this command is Set_Structure_Label which is used to label structures by name.

Set_Structure_Labels

Arguments

Off/All/list of hierarchy level(s) - comma separated e.g. 1,2,3

Example

Client	Server
-----	-----
Set_Structure_Labels \n	
2,3,4,5,6,7,8 \n	
Set_Structure_Labels \n	
Off \n	

Errors

Client	Server
-----	-----
Set_Structure_Labels \n	
Get_Structure_Tree \n	
	QIS_Error \n
	Missing argument - Off/All/list of hierarchy level(s) \n
	Set_Structure_Labels \n

(If no argument is received)

Set_Structure_Labels \n	
On \n	
	Invalid argument. Argument format - Off/All/list of
	hierarchy level(s) \n
	Set_Structure_Labels \n
	On \n

(If an invalid argument is received)

Notes

Use this command to label a structure insertion with its name. This is commonly used together with Set_Structure_Outline which outlines a structure insertion.

The text label height is automatically scaled to be about 9 screen pixels tall no matter what the zoom level is. The label is drawn starting at the structure insertion point.

Chapter 5

Basic File Info - Commands and Functions

Get_DBU

Arguments

None

Example

Client	Server
-----	-----
Get_DBU \n	
	Get_DBU \n
	0.001,1E-9 \n

Errors

No errors

Notes

This command tells QIS to return the database information stored in the header of the GDSII file. Two numbers are returned.

1st number is the user unit resolution. For example if you are working in um and you set a resolution of 0.001 um then the first number would be 0.001. If you are working in mils and you need 0.1 mil resolution the first number will be 0.1.

Second number: length of a database tick in meters. In the first case since you need 0.001 um resolution 1 database tick is 10E-9 meter. In the second case, the number would be 2.54E-5 m because you need 10 of these to make one mil.

Here are some examples:

User's Setup	DBU Info Returned
-----	-----
micron @ 0.001 res.	0.001,1E-9
micron @ 0.25 res.	0.25,2.5E-7
mm @ 0.001 res.	0.001,1E-6
mil @ 0.1 res.	0.1,2.54E-5

Get_QIS_Report

Arguments

No arguments

Example

Client	Server
-----	-----
Get_QIS_Report \n	

Client	Server
-----	-----
	Number of Structures: 47\n
	Structure References: 5427\n
	Array References: 510\n
	Boundaries: 5765 (1332,1260,2)\n
	Paths: 398 (145,87)\n
	Vertices: 52298\n
	Texts: 1371 (5702)\n
	Estimated Memory for Data: 0.6MB (668132 bytes)\n
	Total Memory for Data: 0.6MB (668132 bytes)\n
	Data Dropped: No\n
	Scan Time: 0 min 1 sec\n
	Load Time: 0 min 1 sec\n

Errors

No Errors

Notes

If the report is not available (e.g. before a GDSII file is opened), QIS sends out just the keyword
Get_QIS_Report\n

Get_QIS_Version

Arguments

No arguments

Example

Client	Server
-----	-----
Get_QIS_Version \n	
	Get_QIS_Version \n
	1.04 \n

Errors

Get_QIS_Version \n

Notes

If the version number is not available (some errors), QIS sends back just the keyword - Get_QIS_Version\n.

Chapter 6

Layers, Colors and Patterns - Commands and Functions

Get_Layer_List

Arguments

No arguments

Example

```
Client          Server
-----
Get_Layer_List \n

Get_Layer_List \n
1:0,2:20,3:12,5:1,6:3,9:0
```

Errors

No errors

Notes

This command returns a list of all layers/datatypes of the GDSII file.

Get_Colfill_Pat

Arguments

No arguments

Example

```
Client          Server
-----
Get_Colfill_Pat \n

63485
# Fill Patterns for GDSPLOT and GDSVU
# Each index is a new fill pattern and can consist of a
# 4x4,8x8,16x16 or 32x32 array. Valid entries for the array:
# R-red, G-green, B-Blue, C-cyan, Y-yellow, M-magenta
# K-black, .-white.
#
# Indices 0-31 are reserved for GDSPLOT/GDSVU.
# 16 special contact X, 17 contact +, 18 contact diamond.
# This default file prepared by Steve DiBartolomeo, 9-12-94
#0 WHITE
....
....
....
....
#1 BLACK
KKKK
KKKK
KKKK
KKKK
#2 RED
```


Client	Server
-----	-----
	RRRR
	RRRR
	RRRR
	RRRR
	#3 GREEN
	GGGG
	GGGG
	GGGG
	GGGG
	#4 YELLOW
	YYYY
	YYYY
	YYYY
	YYYY
	#5 BLUE
	BBBB
	BBBB
	BBBB
	BBBB
	#6 MAGENTA
	MMMM
	MMMM
	MMMM
	MMMM
	#7 CYAN
	CCCC
	CCCC
	CCCC
	CCCC
	(cont.....)

Errors

No errors

Notes

It returns the contents of colfill.pat QIS is using. Send this command with no argument and the client will get the same command with the number of bytes and the bytes to follow. e.g.

Set_Layers_On

Arguments

all/list of layer:datatype

Example

Client	Server
-----	-----
Set_Layers_On \n	
All \n	
or	
Set_Layers_On \n	
1,2,5,7 \n	

Client	Server
-----	-----
or	

```
Set_Layers_On \n
1:0,2:20,3:0,5:20 \n
```

Errors

Client	Server
-----	-----
Set_Layers_On \n	
Set_Fill \n	

```
QIS_Error \n
Missing argument - All/list of layer:datatype \n
Set_Layers_On \n
```

(If Set_Layers_On is received with no argument)

```
Set_Layers_On \n
a \n
```

```
QIS_Error \n
Invalid argument.   Argument format - All/list of
layer:datatype \n
Set_layers_on \n
a \n
```

(If the argument does not start with All or numbers which contain layer(s):datatype(s))

Notes

The argument should contain either all layers or all layer:datatype. Layers without datatype are ignored (All layers must be just the layer number or layer:datatype. If a datatype is found in any entry in the argument, all other entries without datatype are ignored. The entries with datatype are still used. This is more of a warning than an error.)

(The validity of the layer and datatype number is not checked and no error is generated if the layer or datatype does not exist.)

Set_Layers_Off

Arguments

all/list of layer:datatype

Example

Client	Server
-----	-----
Set_Layers_Off \n	
All \n	

Client	Server
-----	-----
or	

```
Set_Layers_Off \n
1,2,5,7 \n
```

Errors

Client	Server
-----	-----
Set_Layers_On \n	
Set_Fill \n	

```
QIS_Error \n
Missing argument - All/list of layer:datatype \n
Set_Layers_On \n
```

(If Set_Layers_On is received with no argument)

```
Set_Layers_On \n
a \n
```

```
QIS_Error \n
Invalid argument.   Argument format - All/list of
layer:datatype \n
Set_layers_on \n
a \n
```

(If the argument does not start with All or numbers which contain layer(s):datatype(s))

Notes

The argument should contain either all layers or all layer:datatype. Layers without datatype are ignored (All layers must be just the layer number or layer:datatype. If a datatype is found in any entry in the argument, all other entries without datatype are ignored. The entries with datatype are still used. This is more of a warning than an error.)

(The validity of the layer and datatype number is not checked and no error is generated if the layer or datatype does not exist.)

Set_Layers_Fill

Arguments

layer number - fill pattern e.g. 0-1023

Example

Client	Server
-----	-----
Set_Layers_Fill \n	
9-83 (layer 9 - 50% green) \n	

Client	Server
-----	-----

or

```
Set_Layers_Fill \n
9:1-83, (layer 9:datatype 1 - 50% green) \n
```

or

```
Set_Layers_Fill \n
9:1-83,10:20-84,18:0-85,22:0-86 \n
(for multiple layers:datatypes)
```

Errors

Client	Server
-----	-----

```
Set_Layers_Fill \n
Get_Layer_List \n
```

```
QIS_Error \n
Missing argument - list of layer:datatype-property \n
Set_Layers_Fill \n
```

(If no argument is received)

```
Set_Layers_Fill \n
A,B,C,D,1,2,3
```

```
QIS_Error \n
Invalid argument. Argument format - list of layer:datatype-
property \n
Set_Layers_Fill \n
A,B,C,D,1,2,3 \n
```

(If an invalid argument is received)

```
Set_Layers_Fill \n
9:1-83,2-4,6:3-55,4-44 \n
```

```
QIS_Error \n
Either all layers or all layer:datatype, layers without
datatype are ignored \n
Set_Layers_Fill \n
9:1-83,2-4,6:3-55,4-44 \n
```

(If the argument received contains both layers with datatypes and layers without
The argument MUST contain either ALL LAYERS or ALL LAYER:DATATYPES)

Notes

The syntax is layer-fill (9-83), or layer:datatype-fill (9:20-83). Layers should be specified with all just layers or all layers and datatypes, combinations are not supported (e.g. 9-83,10:20-84 is not supported).
If datatype is specified and the server is not in datatype mode, it will automatically switch to datatype mode; and vice versa.

Set_Layers_Fill (cont...)

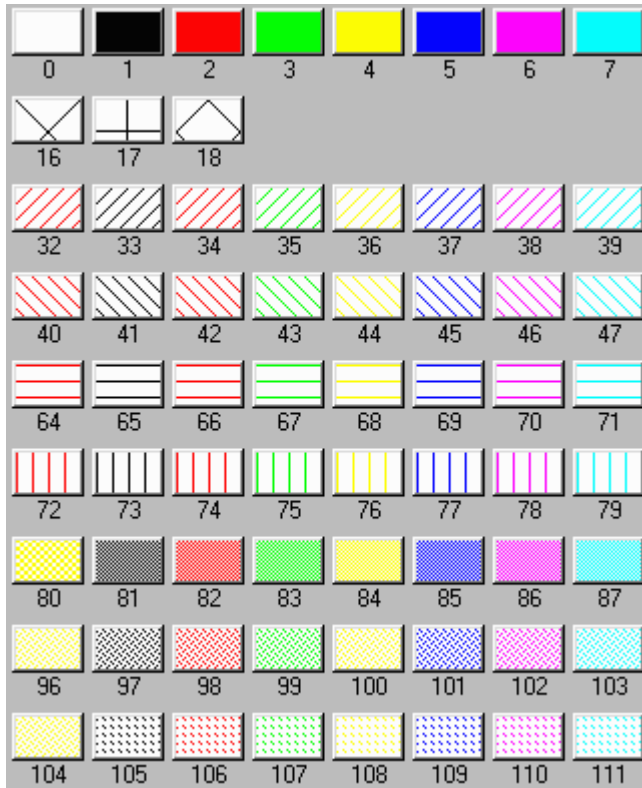


Figure 6-1. Just a small example of 1023 available fill patterns.

Set_Layers_Outline_Color

Arguments

layer number - outline color e.g. 0-7

Example

```
Client                               Server
-----
Set_Layers_Outline_Color \n
9-1 (layer 9 - black) \n
```

Errors

```
Client                               Server
-----
Set_Layers_Outline_Color \n
Set_Layers_Fill \n

QIS_Error \n
Missing argument - list of layer:datatype-property \n
Set_Layers_Outline_Color \n
```

(If no argument is received)

Client

Set_Layer_Outline_Color \n
B-2

Server

QIS_Error
Invalid argument. Argument format - list of layer:datatype-
property
Set_Layers_Outline_Color
B-2

(If an invalid argument is received)

Set_Layers_Fill \n
9:1-83,2-4,6:3-55,4-44 \n

QIS_Error \n
Either all layers or all layer:datatype, layers without
datatype are ignored \n
Set_Layers_Fill \n
9:1-83,2-4,6:3-55,4-44 \n

(If the argument received contains both layers with datatypes and layers without
The argument MUST contain either ALL LAYERS or ALL LAYER:DATATYPES)

Notes

Use this command to control the layer outline color. The outline colors are as follows:

0 = white
1 = black
2 = red
3 = green
4 = yellow
5 = dark blue
6 = purple
7 = light blue

Chapter 7

Display Control - Commands and Functions

Get_Window

Arguments

No arguments

Example

```
Client          Server
-----
Get_Window \n

Get_Window \n
500.000,500.000,3661.966,2460.000
```

Errors

No errors

Notes

This command asks the server what is the current zoom window coordinates in the GDSII file's database units or user units. The zoom window might be different than the coordinates sent by using the Set_Window because the server adjusts the coordinates received through the Set_Window command according to the aspect ratio of the image size in pixels. The client can either keep track of the aspect ratio and the zoom window coordinates or use this command to get the zoom window coordinates. The former should be more efficient than the later.

Set_Window

Arguments

4 numbers for the window coordinates

Example

```
Client          Server
-----
Set_Window \n
5000,5000,6000,6000 \n
```

Errors

```
Client          Server
-----
Set_Window \n
Set_Fill \n

QIS_Error \n
Missing argument - 4 numbers for the window coordinates \n
set_window \n
```

(If no argument is received)

Client	Server
-----	-----
Set_Window \n	
5000,5000 \n	
	QIS_Error \n
	Invalid Keyword \n
	5000,5000 \n

(If the argument received is an invalid pair of XY points)

Notes

The 4 numbers should be in DBU or UU, depending on the Set_Vector_Unit command. Also refer to the notes in Get_Window.

Set_Fill

Arguments

On or Off

Example

Client	Server
-----	-----
Set_Fill \n	
On \n	

Errors

Client	Server
-----	-----
Set_Fill \n	
Set_Window \n	
	QIS_Error \n
	Missing argument - On/Off \n
	Set_Fill \n

(When no argument is received)

Set_Fill \n	
of \n	
	QIS_Error \n
	Invalid Keyword \n
	of \n

(When the argument received is not On or Off)

Notes

Set_Fill On = Boundaries and Paths displayed with "FILL"

Set_Fill Off = Boundaries and Paths displayed with "OUTLINE"

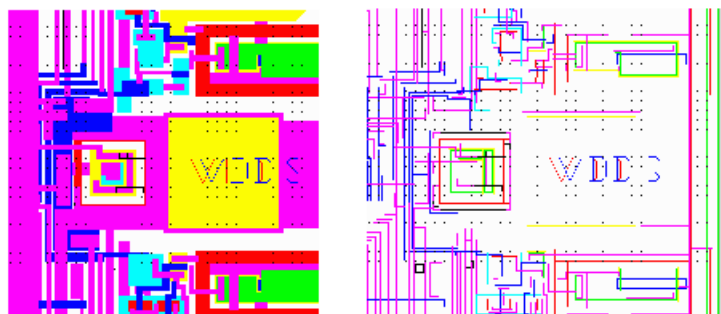


Figure 7-1. Filled and Outlined polygons (left to right).

Set_Array_Mode

Arguments

0, 1 or 2

Example

```
Client          Server
-----
Set_Array_Mode \n
1 \n
```

Errors

```
Client          Server
-----
Set_Array_Mode \n
Set_Window \n

QIS_Error \n
Missing argument - 0/1/2 \n
Set_Array_Mode \n
```

(When no argument is received)

```
Set_Array_Mode \n
3 \n

QIS_Error \n
Invalid argument. Argument - 0/1/2 \n
Set_Array_Mode \n
3 \n
```

(When the argument received is not 0,1, or 2)

Notes

Array Mode 0 = Outline Only
Array Mode 1 = Outer Row/Column
Array Mode 2 = Full Array

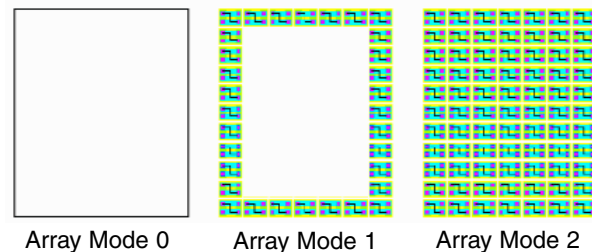


Figure 7-2. QIS various array modes.

Set_Outline

Arguments

On or Off

Example

```
Client          Server
-----
Set_Outline \n
On \n
```

Errors

```
Client          Server
-----
Set_Outline \n
Set_Window \n

QIS_Error \n
Missing argument - On/Off \n
Set_Outline \n
```

(When no argument is received)

```

Set_Outline \n
of \n

QIS_Error \n
Invalid argument.  Argument - On/Off \n
Set_Outline \n
of \n

```

(When the argument received is not On or Off)

Notes

Set_Outline On = Boundaries and Paths displayed with "OUTLINE"

Set_Outline Off = Boundaries and Paths displayed with "FILL"

(See Figure 7-1 for the difference between Fill and Outline)

Set_Display_Filter_Size

Arguments

Any variable greater than or equal to zero

Example

```

Client                      Server
-----
Set_Display_Filter_Size \n
5 \n

```

Errors

```

Client                      Server
-----
Set_Display_Filter_Size \n
Set_Window \n

QIS_Error \n
Missing argument - 1 number(s) \n
Set_Display_Filter_Size \n

```

(When no argument is received)

```

Set_Display_Filter_Size \n
B \n

QIS_Error \n
Invalid argument.  Argument
format - 1 number(s) \n

Set_Display_Filter_Size \n
B \n

```

(When the argument received is an invalid argument)

Notes

When viewing a stream file, often there are thousands of polygons that are so small that they really don't provide any information to the designer - yet they slow down the display. Qckvu's display filter allows the designer to tell Qckvu to disregard any entities smaller than a specified threshold. This speeds up the display. As you zoom in and these entities exceed your threshold, Qckvu then displays them.

0 - no elements are filtered.

1 - elements smaller than a single pixel are filtered out.

2 - elements smaller than two pixels are filtered out

X - elements smaller than X pixels are filtered out.

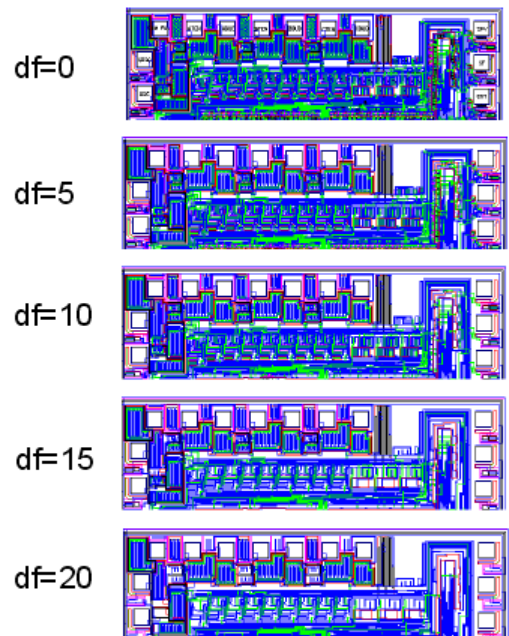


Figure 7-3. Various QIS Display Filter Options.

Set_Geometry_Marker

Arguments

On/Off/list of layer:datatypes

Example

```
Client          Server
-----
Set_Geometry_Marker \n
On \n

or

Set_Geometry_Marker \n
Off \n

or

Set_Geometry_Marker \n
On \n
1:2,2:4,3:8,4:5,5:1 \n
```

Errors

```
Client          Server
-----
Set_Geometry_Marker \n
Set_Fill \n

QIS_Error \n
Missing argument - On/Off/list of layer:datatype \n
Set_Geometry_Marker \n
```

(When no argument is received)

```
Set_Geometry_Marker \n
Of \n

QIS_Error \n
Invalid argument.   Argument format - On/Off/list of
layer:datatype\n
Set_Geometry_Marker \n
Of \n
```

(When the argument received is not On/Off/list of layer:datatype\n)

```
Set_Geometry_Marker \n
1,2,3:0 \n

QIS_Error \n
Either all layers or all layer:datatype, layers without datatype
are ignored\n
Set_Geometry_Marker \n
1,2,3:0 \n
```

(When the argument received mixes layers with layer:datatype syntax.\n)

Notes

This option allows the user to toggle Geometry Markers on/off or on/off on ONLY specific Layers:Datatypes. A combination of BOTH Layers and Layers:Datatypes or JUST LAYERS without datatypes is an invalid entry and will return an error message (shown above). The marker represents the first point in the start of the polygon.

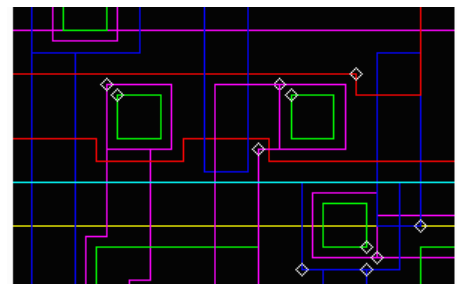


Figure 7-4. Geometry Markers turned On.

Set_Reference_Marker

Arguments

On or Off

Example

```
Client          Server
-----
Set_Reference_Marker \n
On \n
```

Errors

```
Client          Server
-----
Set_Reference_Marker \n
Set_Fill \n

QIS_Error \n
Invalid Keyword \n
Set_Reference_Marker \n
```

(When no argument is received)

```
Set_Reference_Marker \n
Of \n

QIS_Error \n
Invalid argument. Argument - On/Off\n
Set_Reference_Marker \n
Of \n
```

(When the argument received is not On/Off.)

Notes

When a structure or text string is inserted it has an insertion point. To view this point turn the Reference Markers ON. (See right.)

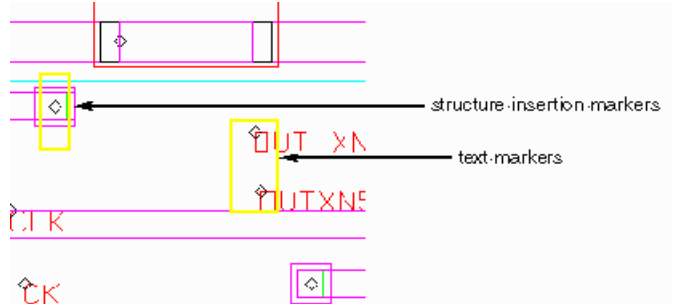


Figure 7-5. Structure and Text Markers turned ON
(Highlighted for clarity)

Set_Marker_Shape

Arguments

Diamond or X

Example

```
Client          Server
-----
Set_Marker_Shape \n
Diamond \n
```

Errors

```
Client          Server
-----
Set_Marker_Shape \n
Set_Fill \n

QIS_Error \n
Missing argument - Diamond/X \n
Set_Marker_Shape \n
```

(When no argument is received)

```
Set_Marker_Shape \n
- \n

QIS_Error \n
Invalid argument. Argument - Diamond/X\n
Set_Marker_Shape \n
- \n
```

(When the argument received is not Diamond or X.)

Notes

The size for X is 21 pixels by 21 pixels. Diamond is the default Marker Shape.

Set_Text_Mode

Arguments

On or Off

Example

```
Client          Server
-----
Set_Text_Mode \n
On \n
```

Errors

```
Client          Server
-----
Set_Text_Mode \n
Set_Fill \n

QIS_Error \n
Missing argument - On/Off \n
Set_Text_Mode \n
```

(When no argument is received)

```
Set_Text_Mode \n
Of \n

QIS_Error \n
Invalid argument. Argument - On/Off\n
Set_Text_Mode \n
Of \n
```

(When the argument received is not On or Off.)

Notes

On/Off to draw text.

Set_Background_Color

Arguments

Black or White

Example

```
Client                               Server
-----
Set_Background_Color \n
Black \n
```

Errors

```
Client                               Server
-----
Set_Background_Color \n
Set_Fill \n

QIS_Error \n
Missing argument - White/Black \n
Set_Background_Color \n
```

(When no argument is received)

```
Set_Background_Color \n
Blue \n

QIS_Error \n
Invalid argument.  Argument - White/Black\n
Set_Background_Color \n
Blue \n
```

(When the argument received is not Black or White)

Notes

This command allows the user to choose between black or white for the background color. Default = White.

Set_Nesting_Level

Arguments

All, or a nesting level number greater than or equal to 0.

Example

```
Client                               Server
-----
Set_Nesting_Level \n
2 \n
```

Errors

```
Client                               Server
-----
Set_Nesting_Level \n
Set_Fill \n

QIS_Error \n
Missing argument - All|1 number(s) \n
Set_Nesting_Level \n
```

(When no argument is received)

Notes

The argument All turns on All levels, 0 for level 0, 1 for up to 1 level, etc...

Added control to set nesting level to 0 which means no data are processed in the specified structure. In connection to this, one can specify to outline and/or label the specified structure on nesting level 0. The outline will be the extent box of the structure and the label will be centered within the extent box.

Set_Structure_Outline

Arguments

Off / All / list of hierarchy levels - comma separated e.g. 1,2,3

Example

```
Client          Server
-----
Set_Structure_Outline \n
Off \n
```

or

```
Set_Structure_Outline \n
All \n
```

or

```
Set_Structure_Outline \n
2,3,4,5,6,7,8 \n
```

Errors

```
Client          Server
-----
Set_Structure_Outline \n
Get_Structure_Tree \n
```

(If no argument is received)

```
Set_Structure_Outline \n
A,2,B,C
```

```
Missing argument - Off/All/list of hierarchy level(s) \n
Set_Structure_Outline \n
```

```
Invalid argument. Argument format - Off/All/list of hierarchy
level(s)
set_structure_outline
A,2,B,C
```

(If an invalid argument is received)

Notes

Use this command to control whether to draw an extent outline of the structure references on the specified levels. This command is controlled using hierarchy - to draw outline bounding box for references on the first level of the current viewing structure, pass in 1; for the second level, pass in 2. Multiple levels can be passed. e.g. 1,2,3,7,8. Related to this command is Set_Structure_Label which is used to label structures by name.

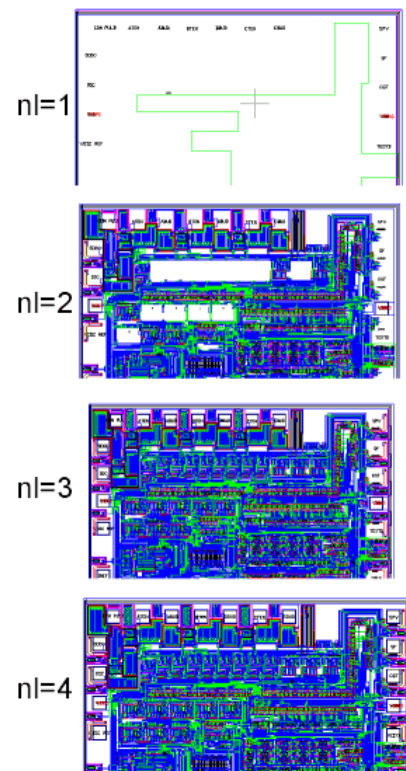


Figure 7-6. Various QIS Nesting Level Examples.

Set_Structure_Labels

Arguments

Off/All/list of hierarchy level(s) - comma separated e.g. 1,2,3

Example

```
Client                               Server
-----                             -
Set_Structure_Labels \n
2,3,4,5,6,7,8 \n
```

or

```
Set_Structure_Labels \n
Off \n
```

Errors

```
Client                               Server
-----                             -
Set_Structure_Labels \n
Get_Structure_Tree \n

QIS_Error \n
Missing argument - Off/All/list of hierarchy level(s) \n
Set_Structure_Labels \n
```

(If no argument is received)

```
Set_Structure_Labels \n
On \n
```

```
Invalid argument.  Argument format - Off/All/list of
hierarchy level(s) \n
Set_Structure_Labels \n
On \n
```

(If an invalid argument is received)

Notes

Use this command to label a structure insertion with its name. This is commonly used together with Set_Structure_Outline which outlines a structure insertion.

The text label height is automatically scaled to be about 9 screen pixels tall no matter what the zoom level is. The label is drawn starting at the structure insertion point.

Set_Scale_Bar

Arguments

UL, LL, UR, LR, off
(upper left, lower left, upper right, lower right)

Example

```
Client          Server
-----
Set_Scale_Bar \n
UR \n
```

Errors

```
Client          Server
-----
Set_Scale_Bar \n
Set_Fill \n

QIS_Error \n
Missing argument - UL/LL/LR/UR/On/Off \n
Set_Scale_Bar \n
```

(If no argument is received)

```
Set_Scale_Bar \n
UP \n

QIS_Error \n
Invalid argument.  Argument - UL/LL/LR/UR/On/Off \n
Set_Scale_Bar \n
UP \n
```

(If an invalid argument is received)

Notes

Control whether to draw a scale bar to show the distance in user units of approximately 100 screen pixels. The distance changes automatically as the zoom level changes. When zoomed in very tight, the scale bar will be shown in 1 database unit in user unit (0.001um) as long as the scale bar is fully within the width of the viewing area. If the distance of the scale bar cannot be determined (overflow or underflow), a "--" is shown with a fixed 80 pixel scale bar.

The scale bar can be placed in 1 of the 4 corners of the viewing area. Send 1 of these controls - UL,LL,UR,LR to turn on the scale bar and specify it's location. Send Off to turn off the scale bar. e.g.

This is what the scale bar looks like when turned ON.

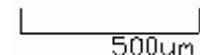


Figure 7-7. This is what the Scale Bar looks like when it's turned ON

Set_Window

Arguments

4 numbers for the window coordinates

Example

Client	Server
-----	-----
Set_Window \n	
5000,5000,6000,6000 \n	

Errors

Client	Server
-----	-----
Set_Window \n	
Set_Fill \n	
	QIS_Error \n
	Missing argument - 4 numbers for the window coordinates \n
	set_window \n

(If no argument is received)

Set_Window \n	
5000,5000 \n	
	QIS_Error \n
	Invalid Keyword \n
	5000,5000 \n

(If the argument received is an invalid pair of XY points)

Notes

The 4 numbers should be in DBU or UU, depending on the Set_Vector_Unit command. Also refer to the notes in Get_Window.

Chapter 8

Getting an Image - Command and Functions

Set_Image_Size

Arguments

X,Y

(where X = width (in pixels), and Y = the height (in pixels), of the desired image size.)

Example

```
Client          Server
-----
Set_Image_Size \n
512,512 \n
```

Errors

```
Client          Server
-----
Set_Image_Size \n
Set_Fill \n

QIS_Error \n
Missing argument - 2 number(s) \n
```

(If no argument is received)

```
Set_Image_Size \n
X,Y \n

QIS_Error \n
Invalid argument.  Argument format - 2 number(s) \n
Set_Image_Size \n
X,Y \n
```

(If an invalid argument is received)

Notes

Use this to specify the size of the images QIS generates.

If there are no errors, Set_Image_Size \n is returned by the server as an acknowledgement for a successful resize. Otherwise, the size is not changed.

Default image size is 800 by 600, however the client should be setting the image size accordingly depending on the client viewing area size.

Set_Image_Format

Arguments

GIF and Bitmap8C - QIS/Windows

GIF - QIS/UNIX/Linux - (which means this command does not allow the format to be changed, its always GIF)

Example

Use this command to get images in Windows GDI bitmap format.

```
Client          Server
-----
Set_Image_Format \n
Bitmap8C \n

8 bit compression in GDI BI_RLE8mode
```

If this is set, the image messages QIS sends out will be in this format:

```
Image_Ready\n      -- OR Get_Image
      12345\n      -- the number of bytes to follow
      BITMAPINFOHEADER  -- the first 40 bytes will be BITMAPINFOHEADER
      256 RGBQUAD Color Table  -- the next 1024 bytes will be an array of 256
                                RGBQUAD
      xxxxxxxx  -- then the image bytes in BI_RLE8 compression
                                format
```

Use the header, the color table and the image directly in the GDI BITMAPINFO structure then use SetDIBits to create a GDI bitmap. Refer to sample code for detail information.

Send Set_Image_Format\nGIF\n to toggle QIS back to output GIF images.

(For more information see Get_Image, page 8-3, and Image_Ready, page 8-5.)

Errors

```
Client          Server
-----
Set_Image_Format \n
Set_Fill \n

QIS Error \n
Missing argument \n
```

(If no argument is received)

```
Set_Image_Format \n
GI \n

QIS Error \n
Invalid argument. Argument - GIF / Bitmap8C
```

(If invalid argument is received)

Notes

Allows the user to get an image in Windows GDI bitmap format.

Redraw

Arguments

No argument

Example

```
Client      Server
-----
Redraw \n

Image_Ready \n
```

Errors or Picture not Drawn

```
Client      Server
-----
Redraw \n

Image_Ready \n
0 \n
```

Notes

Tells QIS to calculate the current image and send it to client.

Get_Image

Arguments

No arguments

Example

```
Client      Server
-----
Get_Image \n

Get_Image \n
12345 \n (total # of bytes to follow)
xxxxxxxxxxxxxxxxxxxx
```

Use the command below to get images in Windows GDI bitmap format.

```
Client      Server
-----
Set_Image_Format \n
Bitmap8C \n

8-bit compression in GDI BI_RLE8 mode
```

If this is set, the image messages QIS sends out will be in this format:

Image_Ready\n	-- OR Get_Image
12345\n	-- the number of bytes to follow
BITMAPINFOHEADER	-- the first 40 bytes will be BITMAPINFOHEADER
256 RGBQUAD Color Table	-- the next 1024 bytes will be an array of 256 RGBQUAD
xxxxxxx	-- then the image bytes in BI_RLE8 compression format

Use the header, the color table and the image directly in the GDI BITMAPINFO structure then use SetDIBits to create a GDI bitmap. Refer to sample code for detail information.

Send Set_Image_Format\nGIF\n to toggle QIS back to output GIF images.

(For more information see Set_Image_Format, page 8-1, and Image_Ready, page 8-5.)

Errors

Client	Server
-----	-----
Get_Image \n	
	Get_Image \n
	O \n

(If unable to get the image)

Notes

This command is used to get the partial image during the QIS drawing process. This is used to update the client's display during a long image draw.

Stop

Arguments

No arguments

Example

Client	Server
-----	-----
Stop \n	

Errors

No errors

Notes

Tells QIS to stop the drawing process.

Zoom_Home

Arguments

No arguments

Example

Client	Server
-----	-----
Zoom_Home \n	
	Image_Ready \n

Errors

No errors

Notes

Similar to redraw but zooms to extents; QIS may have cached the "home" view for faster response.

Image_Ready

Arguments

No arguments

Example

```
Client          Server
-----
Image_Ready \n

64 bytes \n

OR

Image_Ready \n
```

```
Server
-----
Image_Ready \n
12345 \n (total # of bytes to follow)
12.500,50.000,80.050,120.005\0\0....\0 # 0 padded to total

Gif Image \n
```

```
Image_Ready \n

Image_Ready \n
12345 \n (total # of bytes to follow)
12.500,50.000,80.050,120.005\0\0....\0 # 0 padded to total
64 bytes \n
BITMAPINFOHEADER # 40 bytes \n
256 RGBQUAD Color Table # 1024 bytes \n
Bitmap Image \n
```

Use the command below to get images in Windows GDI bitmap format.

```
Client          Server
-----
Set_Image_Format \n
Bitmap8C \n

8-bit compression in GDI BI_RLE8 mode
```

If this is set, the image messages QIS sends out will be in this format:

```
Image_Ready\n
12345\n
BITMAPINFOHEADER
256 RGBQUAD Color Table
xxxxxxx

-- OR Get_Image
-- the number of bytes to follow
-- the first 40 bytes will be BITMAPINFOHEADER
-- the next 1024 bytes will be an array of 256
  RGBQUAD
-- then the image bytes in BI_RLE8 compression
  format
```

Use the header, the color table and the image directly in the GDI BITMAPINFO structure then use SetDIBits to create a GDI bitmap. Refer to sample code for detail information.

Send Set_Image_Format\nGIF\n to toggle QIS back to output GIF images.

(For more information see Set_Image_Format, page 8-1, and Get_Image page 8-3.)

Errors

No errors

Notes

Server command indicating that the completed bitmap is available.

Chapter 9

Getting Vectors - Commands and Functions

Qckvu can return either a bitmap (image) or a stream of vectors representing all of the geometries that would be visible on the screen. The vector option is useful if the client wishes to do its own rendering (for example, overlaying the CAD data over scanned or simulated OPC data ...); the vectors are also useful if the client wishes to manipulate the boundary/path data or even convert the data into another format.

One item to remember is that when requesting vectors, you will only get vectors that would be visible on the screen. Hence it is important to make sure that the various settings for hierarchy display, layer display and filtering are properly set to insure that you get everything you want.

Set_Vector_Unit

Arguments

DBU or UU (Database Units or User Units)

Example

Client	Server
-----	-----
Set_Vector_Unit \n	
DBU \n	

Errors

Client	Server
-----	-----
Set_Vector_Unit \n	
Set_Fill \n	
	QIS_Error \n
	Missing argument - DBU/UU \n
	Set_Vector_Unit \n

(If Set_Vector_Unit is received with no argument)

Set_Vector_Unit \n	
ODF \n	
	Invalid argument. Argument - DBU/UU \n
	Set_Vector_Unit \n
	ODF \n

(If an argument other than DBU or UU is provided)

Notes

Controls how QIS outputs data. Data can either be returned in user units (with decimal point) or in DBU (integer).

Set_Get_Vector_Path

Arguments

Boundary or Path

Example

```
Client                               Server
-----                             -
Set_Get_Vector_Path \n
Path
```

Errors

```
Client                               Server
-----                             -
Set_Get_Vector_Path \n
Set_Fill \n

QIS Error \n
Missing argument - Path/Boundary \n
```

(If no argument is received)

```
Set_Get_Vector_Path \n
Pat \n

QIS Error \n
Invalid argument - Path/Boundary \n
```

(If an argument other than Path/Boundary is received)

Notes

Send the Set_Get_Vector_Path with the argument "Boundary" so Get_Vector/Get_Vertex_Info outputs boundar(ies) for paths. Paths with no width and paths with 1 vertex are still output as paths.

Send the Set_Get_Vector_Path with the argument "Path" if you want Get_Vector/Get_Vertex_Info to output paths as paths. This is the default.

Set_Reference_Vector_Format

Arguments

Use the keyword Long or Short as the argument to the command.
Default is long form.

Example

```
Client                               Server
-----                             -
Set_Reference_Vector_Format \n
Short \n

Set_Reference_Vector_Format \n
Short \n
S,PLUG2,160.500,127.500,1.00000,90.00000,N
A,PLUG2,160.500,127.500,1.00000,90.00000,N
```

Note: In short form, only the type (array or single structure reference), insertion point, scale, rotation and mirror are generated.

Client	Server
-----	-----
Set_Reference_Vector_Format \n	
	Set_Reference_Vector_Format \n
	S,PLUG2,TOPMOSTST,160.500,127.500,1.00000,90.00000,N,5,
	162.000 127.500
	162.000 147.000 147.000 147.000 147.000 127.500
	162.000 127.500
	A,PLUG2,TOPMOSTST,160.500,127.500,1.00000,90.00000,N,8,6,5,
	162.000 127.500
	162.000 147.000 147.000 147.000 147.000 127.500
	162.000 127.500

Note: In long form, it has the additional parent structure, the bounding box and for array references, the rows and columns.

Errors

Client	Server
-----	-----
Set_Reference_Vector_Format \n	
Set_Fill \n	
	QIS Error \n
	Missing Argument - Long/Short \n

(If no argument is received)

Set_Reference_Vector_Format \n	
Lon \n	
	QIS Error \n
	Invalid Argument - Long/Short \n

(If invalid argument is received)

Notes

Use this command to control whether to get all structure reference vectors in long form or short form. (Note, only for structure references and array structure references)

Get_Vector

The Get_Vector command tells QIS to return all "visible" data in the current window and to return it as a stream of primitives. This means that the client will either render this data itself or will use the data for some other purpose than to display it.

The basic primitives are:

Primitives

- boundaries
- paths
- text elements
- structure

Boundary

The boundary is a polygon. It must have a minimum of three vertices and can have up to 8192 vertices. It should not self-intersect. In addition to returning the coordinates of the boundary QIS also returns:

- layer
- datatype
- structure

Boundary Syntax

B,structure name,layer:datatype,number of vertices, x1 y1 x2 y2 ... xn yn

B	- indicates boundary data follows.
structure name	- the name of the structure containing this boundary. a string typically 32 characters long but could be longer for non standard GDSII. We have even seen structure names with spaces.
layer:datatype	- the layer and datatype of this boundary. integer 0-1024
number of vertices	- the number of vertices to follow. Note: GDSII always sends one extra vertex (i.e. the first and last are always on the same point)
(x1 y1)	- coordinate of the vertices. These are in user units or data base units, (depending on which on is chosen is Set_Vector_Unit). Therefore, if the GDSII file is in units of microns you will be getting microns back and will include decimal values.

Example

Client	Server:
-----	-----
Get_Vector \n	Vector_Data \n
	B,TOP,5:0,4,0 0 10 0 10 10 0 10 \n

Path

A path is a series of connected segments of equal width. While a path may have a width=0 it most likely will have a non-zero width. A path can have up to 8192 vertices. The path should not self intersect although it is possible that you might get one that does. The end of the path can be of three types:

- type 0 - [F]flush at vertex
- type 1 - [R]half round extension
- type 2 - [H]half width extension square

A path belongs to a layer and has a datatype (default datatype=0). It will also be tagged with the structure from which it originated.

Path Syntax

P,structure,layer:datatype,width,type,number_of_vertices, x1 y1 x2 y2 ... xn yn

P	- indicates path data follows.
structure name	- the name of the structure containing this path. a string typically 32 characters long but could be longer for non standard GDSII. We have even seen structure names with spaces.
layer:datatype	- the layer and datatype of this path. integer 0-1024

width	- width of the path in user units.
type	- end type F=flush R=1/2 round extension H= 1/2 square extension
number of vertices	- the number of vertices to follow. You can expect an absolute maximum of 8192 vertices.
x1 y1	- coordinate of the vertices. These are in user units - e.g. if the GDSII file is in units of microns you will be getting microns back and will include decimal values.

Example

Client	Server
-----	-----
get_vector \n	
	get_vector \n
	P, TOP, 5:0, 2.5, F, 6, 0 0 10 0 10 10 20 10 20 20 30 0 \n

Note: The path width returned by QIS is after all transformations. If the path was 2 um wide in its structure but the structure is inserted at 10X, then the returned width (and coordinates of course) would be 10X.

Text

Text is a string of text along with geometric information including:

- font
- reference location (any of 9)
- reference coordinate
- rotation
- mirror
- magnitude
- structure containing the text

The proper rendering of text depends on two external items -- a font file which defines the actual strokes used to produce each character and the "height" of the font -- neither which is part of the GDSII stream data.

The magnitude is only a relative value - A magnitude of 1 may produce text that is 0.1 or 100 units high depending on how the font is designed.

Text Syntax

T, structure, layer:texttype, x, y, font, scale, rotation, reflection(X or N), hor_just, vert_just, 4, x1 y1 x2 y2 x3 y3 x4 y4, "text string"

T	- indicates text data to follow
structure	- name of structure this text belongs to
layer:texttype	- layer number (0-1024) and texttype (0-1024) but typically=0
x, y	- text insertion coordinates
font	- one of 4 (0,1,2,3)
scale	- insertion magnitude
rotation	- rotation about insertion point (ccw from X axis)
reflection	- N=no reflection X=around X axis

horiz_just	- 0, 1 or 2	Text justification specifies where the insertion point is within the 9 points in the text extent box	0,0----	1,0----	2,0
vert_just	- 0, 1 or 2				
			0,1	1,1	1,2
			0,2----	1,2----	2,2

```

4                -   number of vertices to follow (which is 4)

(x1,y1)          bounding box of the text. this is used to reflect
(x2,y2)          the transformations that have been applied.
(x3,y3)
(x4,y4)

"text_string"    -   the actual string of text delimited by quote marks.

```

Structure Reference

Strictly speaking, a structure reference is not a primitive at all - it is merely a symbol but outputting might be useful for some applications. If you don't need this info you can throw away the data.

Structure Syntax

```
S,structure,parent,x,y,scale,rotation,reflection(X or N),5,x1 y1 ... x5 y5
```

```

S                -   indicates structure data to follow
structure        -   name of the structure (string)
parent          -   name of this structure's parent (one level up in hierarchy)
x,y             -   insertion coordinates
scale           -   insertion scale
rotation        -   insertion rotation (CCW from X axis)
reflection      -   N=none, X = around X axis
5              -   number of vertices to follow for the extents box
  x1 y1         -   vertices of the extent box. Note that there are 5 sets and
  x2 y2         that  x1 y1 == x5 y5
  x3 y3
  x4 y4
  x5 y5

```

Example

```
S,TESTCHIP,TOP,2000,4500,1,0,N,5,200 450 800 450 800 1250 200 1250 200 450
```

Array Structure Reference

In addition to structure reference, there is array structure reference. It starts with "A" for array, and it has the additional rows and columns after the reflection

Array Structure Syntax

```
A,structure,parent,x,y,scale,rotation,reflection(X or N),8,6,5,x1 y1 ... x5 y5
```

```

A                -   indicates array data to follow
structure        -   name of the structure (string)
parent          -   name of this structure's parent (one level up in hierarchy)
x,y             -   insertion coordinates
scale           -   insertion scale
rotation        -   insertion rotation (CCW from X axis)
reflection      -   N=none, X = around X axis
8              -   rows
6              -   columns
5              -   number of vertices to follow for the extents box
  x1 y1         -   vertices of the extent box. Note that there are 5 sets and
  x2 y2         that  x1 y1 == x5 y5
  x3 y3
  x4 y4
  x5 y5

```

Example

```
A,PLUG2,TOPMOSTST,160.500,127.500,1.00000,90.00000,N,8,6,5,162 127 162 147 147 147 147 127
162 127
```

Get_Display_Vector

Arguments

No arguments

Example

```
Client          Server
-----
Get_Display_Vector /n

B,TOP,5:0,4, 0 0 10 0 10 10 0 10 /n
```

Errors

No errors

Notes

This command works just like Get_Vector for this version. Data is filtered by the display filter. In future versions, this command will output data with less information than Get_Vector.

Get_GDS_Vector

Arguments

No arguments

Example

```
Client          Server
-----
Get_GDS_Vector \n
```

Errors

No errors

Notes

This command outputs all data regardless of the display filter value and blank line type as long as the extents of the data overlap with the current viewing area, the layer of the data (boundary, path and text) is turned on and the structure references is within the specified nesting level.

Get_Vertex_Info

Arguments

No arguments

Example

```
Client          Server
-----
Get_Vertex_Info \n

Get_Vertex_Info \n
246.050,300.650 \n
```


Errors

No errors

Notes

This command works very similar to `Get_Vector`. It outputs all primitives with vertex near the specified point in the `Get_Vector` format. A `Get_Vector_End` denotes the end of all the near by primitives.

The searching radius is around 4 screen pixels from the specified point. The program searches for boundary and path vertices, text and structure insertion points and if the structure outline bounding box is drawn or array reference mode is 0, then the 4 corners of structure references outline bounding boxes are also considered.

This command outputs data regardless of the display filter. Filter behavior is the same as `Get_GDS_Vector`.

Appendix I

Technical Support

If you have problems running our software please contact us so that we are aware of the types of problems you have encountered and can correct them. We also encourage suggestions about new features.

Artwork Conversion Software, Inc.

417 Ingalls St.

Santa Cruz, CA 95060-3500

Tel (831) 426-6163

Fax (831) 426-2824

Email

You can contact us at **support@artwork.com**

This email address is distributed to several users. You can attach small files to your email. If your attachments are large, we prefer that you ftp them to us. If you do attach files please use MIME attachment so that we can automatically detach them.

WEB

We maintain an information area at **http://www.artwork.com**. In addition to the basic datasheet, there is a page that reports the upgrade revision status, and a method of picking up the latest release for registered users. We suggest that you visit the WEB site regularly. A new page for each product: Frequently Asked Questions (FAQ) will be in place.

FTP Site

We prefer that you send us problem files or examples via ftp. The procedure is as follows:

- 1.) Gather the relevant files together (mask files, dxf, gerber, aperture list etc...)
- 2.) Create a readme.txt file identifying yourself, your phone, fax and describing the problem. Files sent to us without a readme.txt associated with them are deleted!
- 3.) Compress the files into a single zip file using PKzip, Winzip or equivalent. If you are worried about privacy or security you can use a password and send us the password by email, voice or fax. If your coming from a UNIX system you can also use gzip or tar and compress.
- 4.) Never send us a self extracting archive ending in .exe. We delete these files immediately - too much risk of a virus.
- 5.) ftp the file to us at ftp.artwork.com - follow the directions on the next page.

FTP Directions

Login = anonymous

Password = email address

Change directory to pub/put_in_here

Change to binary mode and put the file in this directory

We also place updates and newly released software on our ftp site. It is accessible both via the WEB page and directly. Files for download are available from the get_from_here directory.

Sample ftp dialog - dropping off a file

ftp: ftp.artwork.com	<i>IP = 128.242.120.83</i>
login: anonymous	<i>only anonymous login is allowed</i>
password: your_email_address	
> cd pub/put_in_here	<i>We've set up a special directory to put stuff to us.</i>
> binary	<i>set ftp's transfer protocol to binary</i>
> put filename.zip	<i>put the desired file zip file.</i>
> bye	<i>sign off.</i>

Picking up files from the ftp site

ftp: ftp.artwork.com	<i>artwork.com's IP address is 128.242.120.83</i>
login: anonymous	<i>only anonymous login is allowed</i>
password: your_email_address	
> cd pub/get_from_here/mtools	<i>under get_from_here are various directories of interest to us.</i>
> binary	<i>set ftp's transfer protocol to binary</i>
get mtools.sunos.v209.zip	<i>all files are zipped using a password you will need to contact artwork by mail to obtain the unzipping password.</i>
>> bye	<i>sign off.</i>

WEB Support

Updated Software can be easily downloaded

Manuals in PDF format can be downloaded and printed

The web address for this program is <http://www.artwork.com/gdsii/gdsplot/index.htm>

What's the Current Version of the Software?

Go the home page and click on the category for your software. You will find an index page for your program with an entry called Revision History. Click on Revision history and you will find the version number, date and a list of bug fixes or enhancements.

Download the most Recent Version

If you decide that you want to update you can download the latest version with a click.

Installs are password protected. You need to email us (support@artwork.com) to get the installation password. At that time your support status will be checked and only customers under support are given the password.

Technical Bulletins and Solutions to Common Problems

There is a support page (www.artwork.com/support/index.htm) where we've tried to post detailed bulletins on the most commonly asked questions and problems. These solutions are normally done in HTML and soon will be available in PDF.

ZIP Compression

Most of the UNIX files we make available on our ftp site are compressed using the ZIP protocol. UNIX users can download by anonymous ftp the ZIP programs. Don't attempt to download UNIX files onto a PC and unzip them there - the filenames will generally be modified and the install will no longer work.

<code>/pub/get_from_here/util/sunos_zip.tar</code>	for SunOS 4.1.x
<code>/pub/get_from_here/util/solaris_zip.tar</code>	for Solaris 2.x
<code>/pub/get_from_here/util/hp700_zip.tar</code>	for HP-UX 9.0x

You can install these in your `usr/local/bin` directory so that they are easily accessible.

Our zip files are password encrypted - to unzip them you must provide the correct password. The password changes regularly. Please call or email us to obtain the password.

Many public domain versions of unzip such as `gzip` do not support passwords. If yours is such we recommend that you download the ones we have provided.

For program updates you must be under support to obtain such a password.

