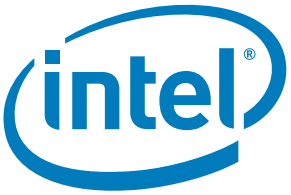


Intel® Embedded Graphics Drivers and Video BIOS v9.0

User's Guide

June 2008

Document Number: 274041-014US

INFORMATION IN THIS DOCUMENT IS PROVIDED IN CONNECTION WITH INTEL® PRODUCTS. NO LICENSE, EXPRESS OR IMPLIED, BY ESTOPPEL OR OTHERWISE, TO ANY INTELLECTUAL PROPERTY RIGHTS IS GRANTED BY THIS DOCUMENT. EXCEPT AS PROVIDED IN INTEL'S TERMS AND CONDITIONS OF SALE FOR SUCH PRODUCTS, INTEL ASSUMES NO LIABILITY WHATSOEVER, AND INTEL DISCLAIMS ANY EXPRESS OR IMPLIED WARRANTY, RELATING TO SALE AND/OR USE OF INTEL PRODUCTS INCLUDING LIABILITY OR WARRANTIES RELATING TO FITNESS FOR A PARTICULAR PURPOSE, MERCHANTABILITY, OR INFRINGEMENT OF ANY PATENT, COPYRIGHT OR OTHER INTELLECTUAL PROPERTY RIGHT. Intel products are not intended for use in medical, life saving, life sustaining, critical control or safety systems, or in nuclear facility applications.

Intel may make changes to specifications and product descriptions at any time, without notice. Designers must not rely on the absence or characteristics of any features or instructions marked "reserved" or "undefined." Intel reserves these for future definition and shall have no responsibility whatsoever for conflicts or incompatibilities arising from future changes to them. The information here is subject to change without notice. Do not finalize a design with this information.

The products described in this document may contain design defects or errors known as errata which may cause the product to deviate from published specifications. Current characterized errata are available on request.

Contact your local Intel sales office or your distributor to obtain the latest specifications and before placing your product order.

Copies of documents which have an ordering number and are referenced in this document, or other Intel literature, may be obtained from:

Intel Corporation

P.O. Box 5937

Denver, CO 80217-9808

or call in North America 1-800-548-4725, Europe 44-0-1793-431-155, France 44-0-1793-421-777, Germany 44-0-1793-421-333, other Countries 708-296-9333 or by visiting Intel's Web Site.

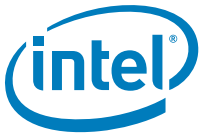
Any software source code reprinted in this document is furnished under a software license and may only be used or copied in accordance with the terms of that license.

Intel processor numbers are not a measure of performance. Processor numbers differentiate features within each processor family, not across different processor families. See http://www.intel.com/products/processor_number for details.

BunnyPeople, Celeron, Celeron Inside, Centrino, Centrino logo, Core Inside, FlashFile, i960, InstantIP, Intel, Intel logo, Intel386, Intel486, Intel740, IntelDX2, IntelDX4, IntelSX2, Intel Core, Intel Inside, Intel Inside logo, Intel. Leap ahead., Intel. Leap ahead. logo, Intel NetBurst, Intel NetMerge, Intel NetStructure, Intel SingleDriver, Intel SpeedStep, Intel StrataFlash, Intel Viiv, Intel vPro, Intel XScale, Itanium, Itanium Inside, MCS, MMX, Oplus, OverDrive, PDCharm, Pentium, Pentium Inside, skoool, Sound Mark, The Journey Inside, VTune, Xeon, and Xeon Inside are trademarks of Intel Corporation in the U.S. and other countries.

*Other names and brands may be claimed as the property of others.

Copyright © 2008, Intel Corporation. All rights reserved.



Contents

1.0 Introduction 11
1.1 Purpose 12
1.2 Intended Audience 12
1.3 Related Documents 12
1.4 Conventions 13
1.5 New Features for Version 9.0 14
1.6 Acronyms and Terminology 15
1.7 Downloading the IEGD and Video BIOS 17

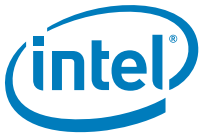
2.0 Architectural Overview 19
2.1 Introduction 19
2.1.1 Display Options 21
2.1.1.1 Types of Displays 21
2.1.1.2 Display Configuration 21
2.2 Features 22
2.2.1 Chipsets Supported 22
2.2.2 OS and API Support 22
2.2.3 EDID-Less Configuration 23
2.2.3.1 EDID-Less Panel Type Detection 23
2.2.4 sDVO Devices 23
2.2.5 Rotation 24

3.0 Platform Configuration Using CED 25
3.1 Before You Begin 26
3.2 Creating a Configuration in CED – Summary Steps 26
3.3 Starting the CED 28
3.4 Creating a New Configuration 29
3.4.1 Setting Color Correction 31
3.4.1.1 Overlay Color Correction 31
3.4.1.2 Color Correction Attributes (Framebuffer) 32
3.4.2 Changing Windows CE OS Options 34
3.4.3 Configuring Ports 36
3.4.3.1 Changing Port Attribute Settings 38
3.4.3.2 Changing I2C Settings 40
3.4.3.3 Changing Flat Panel Settings 41
3.4.4 Configuring Fastboot 43
3.4.4.1 Configuring Splash Video 45
3.4.4.2 How to Select the Video_Offset 46
3.4.5 Configuring the Video BIOS and EFI 46
3.5 Creating a New Package 50
3.5.1 Entering Linux OS Options 52
3.5.2 Entering Windows OS Options 53
3.5.3 Generating a VBIOS Package 55
3.5.4 Entering EFI Options 55
3.6 Creating a New Customized DTD 58
3.6.1 DTD Example Specifications 60
3.7 Generating an Installation 61
3.8 Configuring The System BIOS For Use With The IEGD 62
3.9 System BIOS Settings 62
3.9.1 GMCH PCI Device Enabling 62
3.9.2 Graphics Mode Select (GMS) 63
3.9.3 AGP (Accelerated Graphics Port) Aperture Size 63
3.10 VBIOS and Driver Configuration 63

3.11 Configuration Options 67
3.12 Display Detection and Initialization 74
3.12.1 Display Detect Operation 74
3.12.2 Detectable Displays 76
3.13 Advanced EDID Configuration 76
3.13.1 Sample Advanced EDID Configurations 77
3.13.2 User-Specified DTDs 78
3.14 Using an External PCI Graphics Adaptor as the Primary Device 81
3.15 Enhanced Clone Mode Support 84
3.15.1 Extended Clone Mode CED Configuration 85
3.15.2 Sample Clone Mode Configurations 86
3.16 Scaling and Centering Configurations 87
3.16.1 Upscaling for the Chrontel CH7308 LVDS Transmitters 87
3.16.2 Internal LVDS Scaling with EDID Panels 88
3.16.3 Centering Primary Display with Scaling Encoders 88
3.16.4 Enabling Render Scaling on Port Encoders without Hardware Scaling 89
3.16.5 Alignment in Clone Mode 89

4.0 Generating the VBIOS 91
4.1 Overview 91
4.2 System Requirements 91
4.3 Configuring and Building the VBIOS with CED 92
4.3.1 Selecting the Build Folder 92
4.3.2 Configuring the Video BIOS 93
4.3.2.1 COMMON_TO_PORT 93
4.3.2.2 post_display_msg 94
4.3.2.3 OEM Vendor Strings 94
4.3.2.4 Default Mode Settings 94
4.3.2.5 Default Refresh Settings 94
4.3.2.6 default_vga_height 94
4.3.3 Building the VBIOS 95
4.4 VBIOS, Driver Compatibility, and Data Dependencies 98
4.5 Supported VGA Display Modes for the IEGD Video BIOS 98
4.6 Supported VESA Display Modes for the IEGD Video BIOS 100

5.0 Configuring and Installing Microsoft Windows Drivers 103
5.1 Editing the Microsoft Windows INF File 103
5.1.1 Universal INF Configuration 103
5.1.2 INF File Backward Compatibility 104
5.1.2.1 INF File Backward Compatibility with IEGD Version 4.0 104
5.1.3 Dual Panel Configuration 104
5.1.4 Chipset Dual Display Example 104
5.1.5 Creating Registry Settings for Graphics Driver INF File 105
5.1.6 Dynamic Port Driver Configuration 106
5.1.6.1 iegd.PortDrvs_xxx 107
5.1.6.2 SourceDisksFiles 107
5.1.6.3 PortDrivers Registry Key 108
5.1.7 Creating an .sld file for Microsoft Windows XP Embedded Systems 108
5.1.8 Changing Default Display Mode 109
5.2 Installing the IEGD on Microsoft Windows 109
5.3 Uninstalling the Current Version of the Driver 111
5.4 Run-Time Operation 112
5.5 Viewing and Changing the Driver Configuration from Microsoft Windows 112



6.0 Configuring and Building the IEGD for Microsoft Windows CE Systems 119
6.1 Microsoft Windows CE Installation 119
6.1.1 Prerequisites 120
6.1.2 Integrating IEGD with Microsoft Windows CE Platform Builder 120
6.1.2.1 Catalog Feature File 121
6.2 Microsoft Windows CE Configuration 122
6.2.1 Basic Driver Configuration 122
6.2.1.1 Graphics Memory Configuration 123
6.2.1.2 Defining Graphics Memory Size 124
6.2.1.3 Framebuffer and Video Surface Size 124
6.2.1.4 Video Surface Allocation Rule 125
6.2.1.5 System to Video Stretch Blit 125
6.2.1.6 iegd.reg File Backward Compatibility 126
6.2.1.7 Blend Filtering 126
6.2.2 Configuration Sets 126
6.2.3 General Configuration 127
6.2.3.1 PortOrder Information 128
6.2.3.2 Vertical Extended Mode 129
6.2.4 Per Port Platform Customization 130
6.2.4.1 Per Port Customization – General Port Configuration 130
6.2.4.2 Per Port Customization – Custom DTD Timings 131
6.2.4.3 Per Port Customization — Custom Flat Panel Controls 132
6.2.4.4 Per Port Customization — Attribute Initialization 132
6.2.5 Miscellaneous Configuration Options 132
6.2.5.1 Text Anti-Aliasing 133
6.2.6 D3D Mobile Support 133
6.2.7 Sample iegd.reg File 134

7.0 Installing and Configuring Linux OS Drivers 147
7.1 Overview 147
7.2 Prerequisites 148
7.2.1 Supported Hardware 149
7.3 Installation 149
7.3.1 Installing Fedora Core 5 or Fedora 7 149
7.3.2 Installing SUSE 10 151
7.3.3 Installing Novell Linux POS 9 SP3 153
7.3.4 Installing Other Kernels 155
7.4 IKM Patch Instructions 156
7.4.1 Finding and Installing the Kernel Source (Headers) 156
7.4.2 Installing IKM with Fedora 156
7.4.3 Installing IKM with SUSE 158
7.4.4 Using the IEGD Kernel Module 159
7.5 Uninstalling the IKM 159
7.6 GART Patch Instructions (for Fedora Core 5 Only) 160
7.7 Editing the Linux OS Configuration File 161
7.7.1 Configuration Overview 161
7.7.2 Linux OS Configuration Using CED 161
7.7.3 Editing the Linux OS Configuration File Directly 162
7.7.4 The Linux OS Configuration File 162
7.7.4.1 Device Section 165
7.7.4.2 Screen Section 166
7.7.4.3 Monitor Section 166
7.7.4.4 ServerLayout Section 166
7.7.4.5 ServerFlags Section 167
7.7.5 XFree86 and Xorg Configuration Options 167
7.7.6 Sample Dual Independent Head (DIH) Configuration 171

7.7.7 Video Memory Management 173
7.7.8 Configuring Accelerated Video Decode for IEGD driver and US15W 173
7.7.8.1 Hardware Video Acceleration 173
7.7.8.2 IEGD Driver 173
7.7.8.3 VA Library (version 0.29) 174
7.7.8.4 IEGD Video Acceleration Driver 174
7.7.8.5 Helix Framework Installation 174
7.7.8.6 Intel® Media Codec Installation 175
7.7.8.7 Playing Video 175
7.7.8.8 Troubleshooting 175
7.7.9 Graphics Port Initialization 176
7.7.10 OpenGL Support 176
7.7.10.1 OpenGL Installation 177
7.7.10.2 OpenGL Use Considerations 178
7.7.11 Sample Advanced EDID Configurations for Linux OS 178
7.7.12 AGP GART Errors 179
7.8 Using the IEGD Runtime Configuration GUI on Linux Systems 179
7.8.0.1 iegdgui Setup 179
7.8.0.2 Using the iegdgui Runtime Configuration Utility 180
7.9 Enabling Damn Small Linux 184
7.9.1 Damn Small Linux OS Introduction 185
7.9.1.1 Extending Damn Small Linux OS 185
7.9.2 XFree86 Versus TinyX 186
7.9.3 Running IEGD on Damn Small Linux OS 186
7.9.4 RAM Size Constraint 187
7.9.5 Shrinking and Modifying the Extension 187

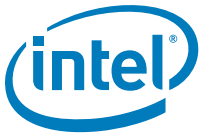
A Example INF File 189

B Port Driver Attributes 197

B.1 Standard Port Driver Attributes 197
B.1.1 Internal LVDS Port Driver Attributes (Mobile chipsets only) 199
B.1.2 CRT (Analog) Port Driver Attributes 199
B.1.3 HDMI Port Driver Attributes 199
7.9.5.1 Audio 199
B.1.3.1 SDVO-HDMI (CH7315) 200
B.1.3.2 Internal HDMI 200
B.1.3.3 HDCP 200
B.1.4 Internal TV Out Port Driver Attributes (Mobile chipsets only) 200
B.1.5 Chrontel CH7307 Port Driver Attributes 201
B.1.6 Chrontel CH7308 Port Driver Attributes 202
B.1.7 Chrontel CH7315/CH7317/CH7319/CH7320 Port Driver Attributes 202
B.1.8 Silicon Image SiI 1362/SiI 1364 Port Driver DVI Attributes 203
B.2 Default Search Order 203
B.3 Default GPIO Pin Pair Assignments 204
B.4 Default I2C Device Address Byte Assignment 204

C Intel® 5F Extended Interface Functions 205

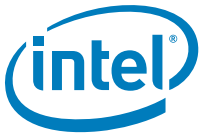
C.1 BIOS Extended Interface Functions 206
C.1.1 5F01h – Get Video BIOS Information 206
C.1.2 5F05h – Refresh Rate 206
C.1.2.1 5F05h, 00h – Set Refresh Rate 206
C.1.2.2 5F05h, 01h – Get Refresh Rate 207
C.1.3 5F10h – Get Display Memory Information 208
C.1.4 5F1Ch – BIOS Pipe Access 208
C.1.4.1 5F1Ch, 00h – Set BIOS Pipe Access 208
C.1.4.2 5F1Ch, 01h – Get BIOS Pipe Access 208



C.1.5 5F29h – Get Mode Information 209
C.1.6 5F61h – Local Flat Panel Support Function 209
C.1.6.1 5F61h, 05h – Get Configuration ID 209
C.1.7 5F68h – System BIOS Callback 210
C.2 Hooks for the System BIOS 210
C.2.1 5F31h – POST Completion Notification Hook 210
C.2.2 5F33h – Hook After Mode Set 211
C.2.3 5F35h – Boot Display Device Hook 211
C.2.4 5F36h – Boot TV Format Hook 212
C.2.5 5F38h – Hook Before Set Mode 212
C.2.6 5F40h – Config ID Hook 213
D Intel® OpenGL APIs 215

Figures

1 Intel Embedded Graphics Suite....19
2 Graphics Driver Architecture....20
3 Firmware Architecture....20
4 Sample CED Configuration Start Page....27
5 Chipset Configuration Page....29
6 Overlay Color Correction Page....32
7 Overlay Color Correction Page....33
8 Chipset Configuration Page....34
9 Port Configuration Page....36
10 Attribute Settings Page....39
11 sDVO Settings Page....40
12 Panel Settings Page....41
13 Fastboot Configuration Page....43
14 Splash Video with 8 MBytes of Stolen Memory Example....45
15 Video BIOS Configuration Page....47
16 IEGD Package Editor Page....50
17 Linux Options Page....52
18 Windows Options Page....54
19 VBIOS Generation Page....55
20 EFI Generation Page....56
21 EFI Generation Page....57
22 IEGD DTD Page....58
23 IEGD DTD Page....64
24 LVDS Configuration Page....65
25 IEGD Configuration Editor Page....66
26 IEGD DTD Editor....79
27 External PCI Graphics Card as Primary Driver and IEGD as Secondary Driver....82
28 IEGD as Primary Driver and External PCI Graphics Card as Secondary Driver....83
29 IEGD as Primary Driver With Two Displays and External PCI Driving a Tertiary Display....84
30 Video BIOS Directory Structure....93
31 Example Runtime Configuration GUI — Driver Info Tab....113
32 Example Runtime Configuration GUI — Display Config Tab....114
33 Example Runtime Configuration GUI — Display Attributes Tab....115
34 Example Runtime Configuration GUI — Color Correction Tab....116
35 Sample FILES Block from `platform.bib` File....121
36 Typical Memory Map Using Static Memory Model....123
37 Sample XF86Config File....163
38 Sample DIH Configuration....171
39 Example Linux Runtime Configuration GUI — Display Config Tab....181
40 Example Linux Runtime Configuration GUI — Display Attributes Tab....182
41 Example Linux Runtime Configuration GUI — Color Correction Tab (Framebuffer)....183
42 Example Linux Runtime Configuration GUI — Color Correction Tab (Overlay)....184
43 Damn Small Linux Partition....185



Tables

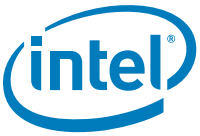
1 Intel® Embedded Graphics Drivers and Video BIOS 9.0 New Features 14
2 Acronyms and Terminology 15
3 Types of Displays 21
4 Display Configuration Definitions 21
5 Supported Display Configurations 21
6 Chipsets Supported by the Intel Embedded Graphics Suite 22
7 SDVO Devices Supported 23
8 Chipset Configuration Page Settings 30
9 Overlay Color Correction Values (applies to ALL color) 31
10 Framebuffer Color Correction Values (applies to R, G, B color) 32
11 Windows CE OS Settings 35
12 Port Configuration Settings 37
13 I2C Settings 40
14 Panel Settings Options 42
15 Fastboot Options 44
16 Video BIOS Settings Options 48
17 IEGD Package Editor Setting Options 51
18 Linux OS Settings Options 53
19 Windows OS Setting Options 54
20 IEGD DTD Setting Options 59
21 Timing Specification Example Values 60
22 GMCH Device 2, Function 1 BIOS Setting 62
23 GMS Settings 63
24 Parameter Configuration Format 67
25 Detectable Displays 76
26 Sample Advanced EDID Configurations 77
27 DTD Parameter Descriptions 80
28 Supported VGA Video Display Modes 99
29 VESA Modes Supported by Video BIOS 100
30 Example of Chipset Dual Display Parameter Setting 104
31 Framebuffer Color Correction Values (applies to R, G, B color) 116
32 Overlay Color Correction Values (applies to ALL color) 116
33 [HKLM\DRIVERS\Display\Intel] Registry Keys 122
34 [HKLM\Drivers\Display\Intel\<platform>\<config id>\]Registry Keys 127
35 PortOrder Information 128
36 X.org and X-version Conversion Reference 147
37 Supported Driver Options 167
38 Sample Advanced EDID Configurations for Linux OS 178
39 Standard Port Driver Attributes 197
40 Internal LVDS Port Driver Attributes 199
41 CRT (Analog) Port Driver Attributes 199
42 HDMI Port Driver Attributes 200
43 Internal TV Out Port Driver Attributes 200
44 Chrontel CH7307 Port Driver Attributes 201
45 Chrontel CH7308 Port Driver Attributes 202
46 Chrontel CH7315/CH7317/CH7319/CH7320 Port Driver Attributes 202
47 Silicon Image SiI 1362/SiI 1364 Port Driver DVI Attributes 203
48 Default Search Order 203
49 Default GPIO Pin Pair Assignments 204
50 Default I^2C Device Address Byte Assignment 204
51 Summary of Intel 5F Extended Interface Functions 205
52 Supported Intel® OpenGL APIs 215
53 Non-Supported Intel® OpenGL APIs 216

Revision History

This document may have been updated since the release shown below. See http://www.intel.com/design/intarch/manuals/274041.htm for the most recent version.

Date	Revision	Description
June 2008	014	Updated for use with Version 9.0 of the product.
October 2007	013	Updated for use with Version 8.0 of the product.
June 2007	012	Updated for use with Version 7.0 of the product.
December 2006	011	Updated for use with Version 6.1 of the product.
September 2006	010	Updated for use with Version 6.0 of the product, including support for the Intel® Q965 and Damn Small Linux*.
June 2006	009	Updated for use with Version 5.1 of the product, including support for the Texas Instruments TFP410* DVO encoder, Microsoft Windows Embedded for Point of Service (WEPOS)* operating system, and SuSE 10.
February 2006	008	Updated for use with Version 5.0 of the product, including support for the Intel® 852GM, Intel® 945G, and Intel® 945GM chipsets, the Silicon Image SiI 1362* and SiI 1364* sDVO transmitters, and External PCI as a Primary graphics adaptor.
October 2005	007	Updated for use with Version 4.1 of the product.
June 2005	006	Updated for use with Version 4.0 of the product, including support for the Intel® 915GV and Intel® 915GM chipsets, the Chrontel CH7307* and Chrontel CH7308* sDVO transmitters, and Advanced EDID Configuration.
May 2005	005	Updated for use with Version 3.4 of the product, including use of the enhanced Video BIOS, Windows* installer/uninstaller, runtime configuration GUIs, and display discovery feature.
July 2004	004	Updated for use with Version 3.2 of the product, including use of the dynamic port driver feature.
May 2004	003	Updated for usage with version 3.1 of the product, including details on PCF format and usage, Universal INF format, and updates to the User Build System.
February 2004	002	Updated chipset support to reflect current Embedded IA32 roadmap.
February 2004	001	Initial Release

§ §



1.0 Introduction

The Intel® Embedded Graphics Drivers (IEGD) comprise a suite of multi-platform graphics drivers designed to meet the requirements of embedded applications. Featuring Intel® Dynamic Display Configuration Technology (DDCT), the drivers run on the following Embedded Intel® Architecture (eIA) chipsets:

- Intel® GM45Express chipset
- Intel® System Controller Hub US15W chipset
- Intel® Q35 Express chipset
- Mobile Intel® GME965 Express chipset
- Intel® Q965 Express chipset
- Mobile Intel® 945GSE Express chipset
- Mobile Intel® 945GME Express chipset
- Intel® 945G Express chipset
- Intel® 915GV Express chipset
- Mobile Intel® 915GME Express chipset
- Mobile Intel® 910GMLE Express chipset

Note: If you need support for a chipset that is not listed above but is in the same family as those listed, please contact your Intel representative.

The IEGD supports five types of display devices:

- Analog CRT
- LVDS flat panels
- TMDS DVI displays
- HDMI
- TV Output

The IEGD is designed to work with fixed-function systems, such as Point-of-Sale (POS) devices, ATM machines, gaming devices, In-vehicle Information/Entertainment systems, etc. It can be configured to work with various hardware and software systems and supports both Microsoft Windows* and Linux* operating systems, including embedded versions of these operating systems.

The Intel Embedded Graphics Suite consists of both the IEGD and a Video BIOS (VBIOS) component. These two components are configurable and work together to provide a wide range of features. This document provides information on configuring and using both the IEGD and the VBIOS.

The IEGD provides the following features:

- Enhanced VBIOS support
- Dynamic Port Drivers
- Support for Dual Independent Head (DIH) displays
- Support of a Universal INF file
- EDID and EDID-less display support
- Display discovery and initialization
- Direct 3D* support
- Installer/Uninstaller GUI for Microsoft Windows OS
- Runtime configuration GUI for Microsoft Windows OS and Linux OS

1.1 Purpose

This manual provides information on both firmware and software, providing hardware design considerations, installation requirements, and static configuration options.

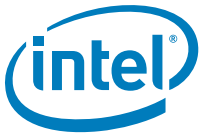
1.2 Intended Audience

This document is targeted at all platform and system developers who need to interface with the graphics subsystem. This includes, but is not limited to: platform designers, system BIOS developers, system integrators, original equipment manufacturers, system control application developers, as well as end users.

1.3 Related Documents

The following documents provide additional information on the hardware supported by the IEGD.

- *Intel® Embedded Graphics Drivers for Embedded Intel® Architecture-based Chipsets Product Brief*
  (Document Number: 315587)
- *Intel® 35 Express Chipset Family Datasheet*
  (Document Number: 31696602)
- *Mobile Intel® 965 Express Chipset Family Datasheet*
  (Document Number: 316273)
- *Intel® 965 Express Chipset Family Datasheet*
  (Document Number: 313053)
- *Mobile Intel® 915PM/GM/GMS and 910GML Express Chipset Datasheet*
  (Document Number: 305264)
- *Intel® 915G/915GV/915P Express Chipset Datasheet*
  (Document Number: 304467)
- *Intel® I/O Controller Hub 6 (ICH6) Family Datasheet*
  (Document Number: 301473)
- *Integrated Dual Independent Display on Intel® Digital Security Surveillance Multifunction Platforms Application Brief*
- *Display Panel Debugging with the Intel Graphics Memory Controller Hub*
  (Document Number: 305964)



- *VESA BIOS Extensions/Display Data Channel Standard,* available at the following Web address:
  http://www.vesa.org/public/VBE/VBEDDC11.PDF

  This document provides information on the 4F VBE functions, which are supported by the Intel embedded Video BIOS.
- *VESA BIOS Extension (VBE) Core Functions Standard Version 3.0,* available at the following Web address:
  http://www.vesa.org/public/VBE/vbe3.pdf

  Contains information on the VESA BIOS Extension (VBE) specification for standard software access to graphics display controllers that support resolutions, color depths, and framebuffer organizations beyond the VGA hardware standard.

1.4 Conventions

The following conventions are used throughout this document.

Boldface	Represents text that you type and text that appears on a screen.
Italics	Introduces new terms and titles of documents.
`Courier New`	Identifies the names of files, executable program names, and text that appears in a file.
Angle Brackets (<>)	Encloses variable values in syntax or value ranges that you must replace with actual values.
Vertical Bar (|)	Used to separate choices (for example, TRUE | FALSE)

1.5 New Features for Version 9.0

The table below presents new IEGD features and capabilities.

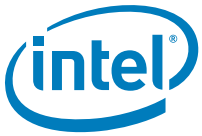
Table 1. Intel® Embedded Graphics Drivers and Video BIOS 9.0 New Features

New Features
Intel® System Controller Hub US15W Support • 2D (Xserver, Display, Overlay, Video Gamma Correction, Display Resolution/timings, Dynamic mode change, Render scaling, Twin, Clone, Extended, Dual Independent Display, Vertical Extended for CE, panning, centering, Rotation and Inverted Display) • GUI support • US15W supports all IEGD 9.0 supported OSes with the following condition: — US15W supports XP, XPe, Windows CE 6.0, Wind River Linux, Red Flag MIDINUX 2.0 and Red Hat Embedded. — US15W does NOT support Windows CE 5.0. — US15W does NOT support DSL. • Windows CE 2D — Scaling, Rendered Scaling, Panning, Centering. • Wind River Linux support (X.org version 7.3, Red Hat Embedded: Xorg 7.2, Red Flag: Xorg 7.2). • 3D (OpenGL 1.5) support • 3D (Direct3D)Compliant with DirectX 9.0c. • IEGD Configuration Editor (CED) support. • EFI# Support, including VESA mode support. — EFI supports only graphics mode but no text mode. — US15W requires fast boot in EFI + Linux. • Quick Boot + Sprite C Overlay handoff. • VBIOS Support • Hardware Accelerated Decode support (does not apply to Windows CE) • Second Overlay
sDVO Port driver support for HDMI with HDCP on CH7315
Intel® GM45 Express Chipset Support • 2D (Xserver, Display, Overlay, Video Gamma Correction, Display Resolution/timings, Dynamic mode change, Render scaling, Twin, Clone, Extended, Dual Independent Display, Vertical Extended for CE, panning, centering, Rotation and Inverted Display). • CED support. • HDMI support: — Internal HDMI (iHDMI) with HDCP (including Azalia audio) for GM45 only. — External HDMI with HDCP for sDVO enabled encoder supported on all platforms.
Intel® Mobile 945GSE Chipset Support
• Linux Power Management (ACPI Support of all C states, S States, Backlight modulation, Clock gating, interaction with system BIOS using ACPI)

This release also contains resolutions for errata. For details on errata, including status information, refer to the specification update located at the following Web address:

http://www.intel.com/design/intarch/specupdt/309380.htm

A manual process is available for obtaining the IEGD and the VBIOS. See "Downloading the IEGD and Video BIOS".



1.6 Acronyms and Terminology

The table below lists the acronyms and terminology used throughout this document.

Table 2. Acronyms and Terminology (Sheet 1 of 3)

Term	Description
ADD Card	APG Digital Display. An adapter card that can be inserted into the PCIe x16 port of Intel chipset family-based systems. ADD cards allow configurations for TV-out, LVDS, and TMDS output (i.e., televisions, digital displays, and flat panel displays).
AIM	Add In Module.
API	Application Programming Interface.
BDA	BIOS Data Area. A storage area that contains information about the current state of a display, including mode number, number of columns, cursor position, etc.
BIOS	Basic Input/Output System. The IEGD interacts with two BIOS systems: system BIOS and Video BIOS (VBIOS). VBIOS is a component of the system BIOS.
CED	Configuration EDitor. Graphical pre-installation utility allows easy creation of consolidated driver installation packages for Windows, Windows CE, and Linux operating systems, and VBIOS across numerous platforms and display combinations.
Clone Display Configuration	A type of display configuration that drives two display devices, each displaying the same content, but can have different resolutions and (independent) timings. Compare Twin Display Configuration and DIH Display Configuration.
DDCT	Intel® Dynamic Display Configuration Technology
DirectDraw*	A component of the DirectX* Graphics API in Microsoft Windows OS.
DIH Display Configuration	Dual Independent Head. A type of display configuration that supports two displays with different content on each display device. The IEGD supports Extended mode for Microsoft Windows systems and Xinerama for Linux systems.
DTD	Detailed Timing Descriptor. A set of timing values used for EDID-less devices.
DVI	Digital Video Interface.
EBDA	Extended BIOS Data Area. An interface that allows the system BIOS and Option ROMs to request access to additional memory.
EDID	Extended Display Identification Data. A VESA standard that allows the display device to send identification and capabilities information to the IEGD. IEGD reads all EDID data, including resolution and timing data, from the display, thus negating the need for configuring DTD data for the device.
EDID-less	A display that does not have the capability to send identification and timing information to the driver and requires DTD information to be defined in the driver.
eIA	Embedded Intel® Architecture.
EMI	Electromagnetic Interference.
Extended Clone Mode	A feature that allows you to have different sized displays in Clone mode.
Framebuffer	A region of physical memory used to store and render graphics to a display.

Table 2. Acronyms and Terminology (Continued) (Sheet 2 of 3)

Term	Description
GEN3	Napa Graphics Core in 910/915 family chipset.
GEN3.5	Napa+ Graphics Core in 945 family chipset.
GEN4	Graphics Core in 965 family chipset.
GDI	Graphics Device Interface. A low-level API used with Microsoft Windows operating systems.
GMA	Intel Graphics Media Accelerator. Refers to both the graphic hardware in Intel chipsets as well as the desktop/mobile driver. The GMA driver is not intended for use in embedded applications.
GMCH	Graphics and Memory Controller Hub.
GMS	Graphics Mode Select (stolen memory).
HAL	Hardware Abstraction Layer. An API that allows access to the Intel® chipsets.
HDMI	High-Definition Multimedia Interface, an uncompressed, all-digital audio/ video interface.
IAL	Interface Abstraction Layer. An API that allows access to graphics interfaces including the GDI, DirectDraw*, XFree86*.
IEGD	Intel® Embedded Graphics Drivers
IEGS	Intel® Embedded Graphics Suite. Runtime graphics driver plus a VBIOS component.
INF file	A standard Microsoft Windows text file, referred to as an information file, used by Microsoft Windows OS to provide information to the driver. The default .inf file for the IEGD is `iegd.inf`. You can create customized parameters using the CED utility.
LVDS	Low Voltage Differential Signaling. Used with flat panel displays, such as a laptop computer display.
Linux/XFree86	Open Source for X Window* System used on Linux systems.
NTSC	National Television Standards Committee. A TV standard used in North and Central America and in Japan.
OAL	Operating System Abstraction Layer. An API that provides access to operating systems, including Microsoft Windows and Linux.
Option ROM	Code which is integrated with the system BIOS and resides on a flash chip on the motherboard. The Intel Embedded Video BIOS is an example of an option ROM.
PAL	Phase Alternating Lines. A TV standard used in Europe, South America, Africa, and Australia.
PCI	Peripheral Component Interface.
Port Driver	A driver used with the sDVO interfaces of the Graphics and Memory Controller Hub (GMCH).
POST	Power On Self Test.
Reserved Memory	A region of physical memory in a Windows CE* system set aside for BIOS, VBIOS, and Graphics Driver operations. Reserved memory can be configured to be used by the operating system and other applications when not in use by the BIOS.
sDVO	Serial Digital Video Output.
Single Display Configuration	A type of display configuration that supports one and only one display device.
Stolen Memory	A region of physical memory (RAM) set aside by the system BIOS for input and output operations. The amount of stolen memory is configurable. Stolen memory is not accessible to the operating system or applications.
System BIOS	The standard BIOS used for basic input and output operations on PCs.

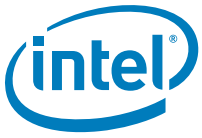


Table 2. Acronyms and Terminology (Continued) (Sheet 3 of 3)

Term	Description
TMDS	Transitioned Minimized Differential Signaling. Used with DVI displays, such as plasma TVs.
TOM	Top Of Memory.
TSR	Terminate and Stay Resident. A program that is loaded and executes in RAM, but when it terminates, the program stays resident in memory and can be executed again immediately without being reloaded into memory.
Twin Display Configuration	A type of display configuration that supports two display devices each of which has the same content, resolution, and timings. Compare Clone Display Configuration.
UBS	User Build System. A process for building a VBIOS.
VBIOS	Video Basic Input Output System. A component of system BIOS that drives graphics input and output.
VESA	Video Electronics Standards Organization.
VGA	Video Graphics Array. A graphics display standard developed by IBM* that uses analog signals rather than digital signals.

1.7 Downloading the IEGD and Video BIOS

The IEGD and the Video BIOS (VBIOS) are available via the Download Drivers link (under Related Information) at http://www.intel.com/go/iegd or directly via the Intel Embedded Graphics Drivers Downloads page, where the following is available:

- Intel Embedded Graphics Driver Configuration Editor (CED) release
  - — includes the IEGD drivers for VBIOS, Linux OS, and all Windows operating systems, plus an online help system

Note: The Embedded Video BIOS version 9.0 is recommended for use with each of the graphics drivers in most cases. Click the following link to see the FAQ page for details on the differences of these versions.

http://www.intel.com/design/intarch/swsup/graphics_faq.htm

After you have downloaded, installed, and run CED, you can configure and customize the drivers and VBIOS following the procedures in this document. Once they have been configured, you can integrate the VBIOS with the system BIOS ROM and install the IEGD on your operating system.

This page is intentionally left blank.

2.0 Architectural Overview

2.1 Introduction

The Intel Embedded Graphics Suite (IEGS) is composed of a runtime graphics driver and a Video BIOS (VBIOS) firmware component. (See the illustrations below.) Both the driver and VBIOS control the GMCH to perform display and render operations. The VBIOS is predominantly leveraged by System BIOS during system boot but is also used at runtime by the driver to handle full-screen text mode on Microsoft Windows* operating systems.

Figure 1. Intel Embedded Graphics Suite

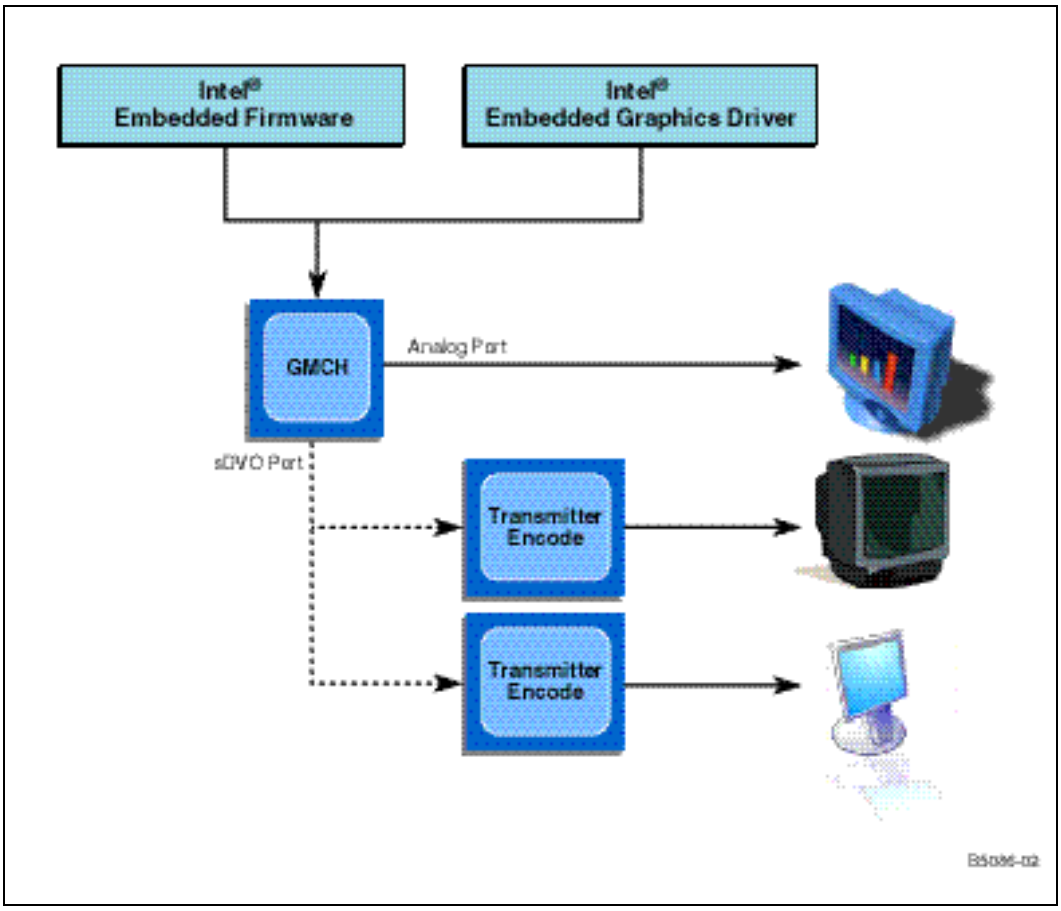


Figure 2. Graphics Driver Architecture

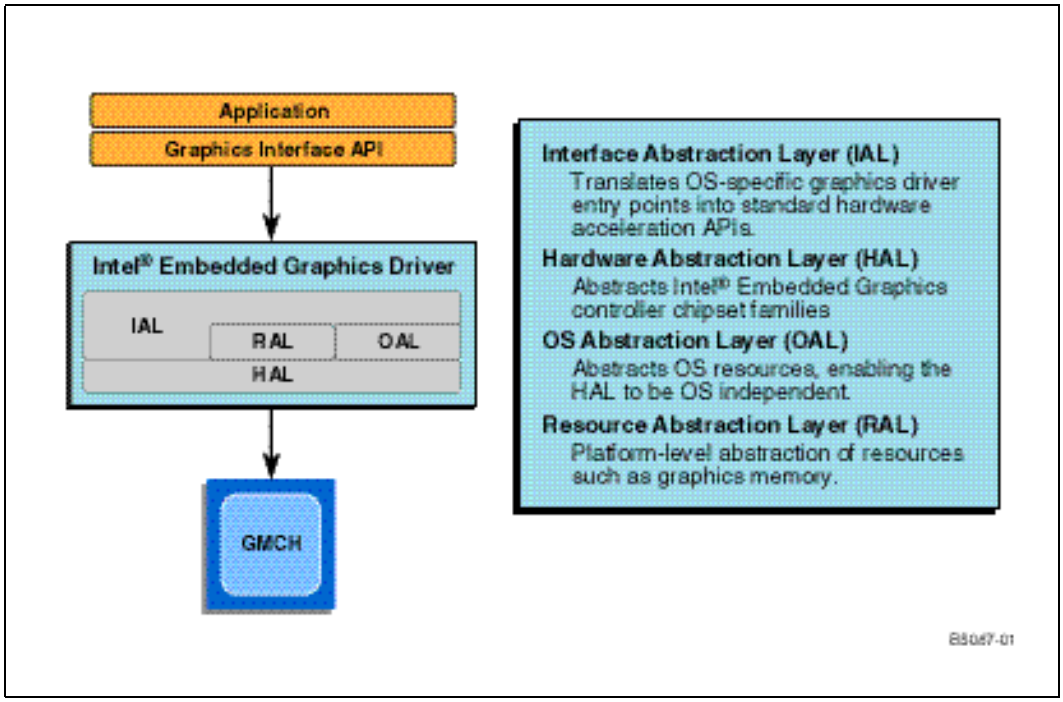
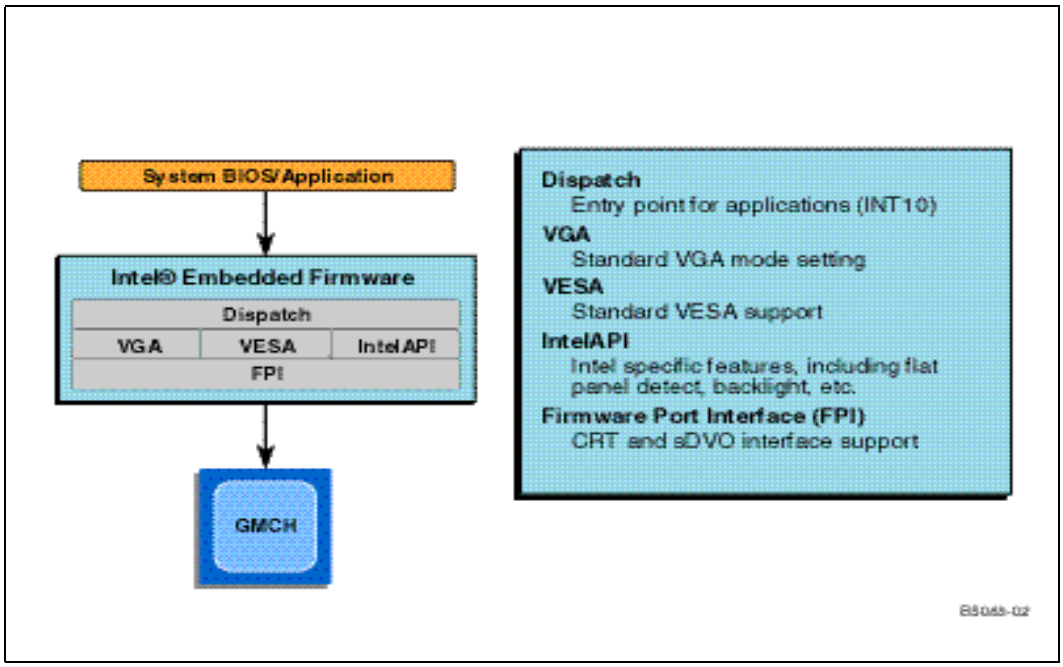
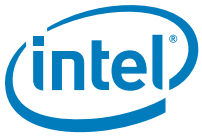


Figure 3. Firmware Architecture





2.1.1 Display Options

The following section describes the types of displays and configurations supported by the Intel Embedded Graphics Driver.

2.1.1.1 Types of Displays

The table below lists the types of displays supported by the IEGD.

Table 3. Types of Displays

Display	Description
CRT	Analog CRT, natively supported with RGB signaling.
Flat Panel	TMDS and LVDS compliant flat panels are supported with the use of an external transmitter via an sDVO port. Integrated LVDS flat panels are also natively supported on the Mobile Intel® 910GMLE, Mobile Intel® 915GME Express, Mobile Intel® 945GME,Mobile Intel® 945GSE Express, and Mobile Intel® GME965 chipsets.
TV	TV-out is supported via an external encoder sDVO port. ***Note:*** TV-Out is enabled via supported internal capability, or external transmitters sDVO (where available).

2.1.1.2 Display Configuration

IEGD supports driving two displays simultaneously. Several configurations are supported, dependent on operating system and chipset. The various display configuration are described in the table below.

Table 4. Display Configuration Definitions

Display Configuration Mode	Description
Single	Normal desktop configuration, single monitor
Twin	Two displays, same content, single resolution, same timings
Clone	Two displays, same content, different resolutions, independent timings
Extended	Two displays, continuous content
DIH	Dual Independent Head. Two displays, different content, independent resolutions

The table below summarizes which display configurations are supported by Intel chipsets.

Table 5. Supported Display Configurations

Chipset	Operating System		
	Windows XP*	Windows CE*	Linux
Intel® Q35 Intel® GME965, Intel® Q965, Intel® 945GM, Intel® 945G, Intel® 915GME, Intel® 910GMLE	Single, Twin, Clone, Extended	Single, Twin, Clone	Single, Twin, Clone, DIH
Intel® 915GV	Single, Twin, Clone	Single, Twin, Clone	Single, Twin, Clone

Twin and Clone modes are supported by IEGD through custom APIs. In contrast, Extended and DIH are supported natively by both Microsoft Windows and Linux operating systems (XFree86 and X.org*).

2.2 Features

The following sections describe major features supported by IEGD.

2.2.1 Chipsets Supported

The table below lists IEGD-supported chipsets.

Table 6. Chipsets Supported by the Intel Embedded Graphics Suite

Chipset	IEGD VBIOS Support	IEGD Support
Intel® GM45	Yes	Yes
Intel® US15W	Yes	Yes
Intel® Q35	Yes	Yes
Intel® GME965, Intel® Q965	Yes	Yes
Intel® 945G Intel 945GME Intel 945GSE	Yes Yes Yes	Yes Yes Yes
Intel® 915GV Intel® 915GME	Yes Yes	Yes Yes
Intel® 910GMLE	Yes	Yes

All supported chipsets provide support for a single analog output for CRTs. In addition, digital monitors, flat panels and TVs are supported through the GMCH sDVO interface.

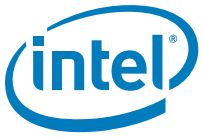
2.2.2 OS and API Support

The IEGD and Video BIOS support the following operating systems and APIs. For OpenGL APIs, see Appendix D, "Intel® OpenGL APIs".

- Linux XFree86 and X.org
- Damn Small Linux
- Microsoft Windows XP, Windows XP Embedded*
  - DirectX* 8.1 (DirectDraw* and Direct3D*)
  - DirectX 9 (DirectDraw and Direct3D)
- Microsoft Windows CE 5.0 and 6.0

Note: The following features are NOT supported in IEGD 9.0:

- Microsoft Vista 2D + 3D
- Vista DirectX 9.0L, DirectX 10.0 (Combine with MS Vista 2D + 3D)



2.2.3 EDID-Less Configuration

EDID-less support is the ability to run a display panel that does not have display timing information within the panel. Therefore, the user has to provide the display timing information to the graphics drivers. For the IEGD, this must be done through:

- CED
- Configuration file for the graphics drivers.

This document describes only the necessary edits to the configuration files that are required to implement the graphics driver and VBIOS, and not specific settings for EDID-less panel configuration. Please refer to the manufacturer's specifications for the DTD settings to use for your EDID-less panels.

2.2.3.1 EDID-Less Panel Type Detection

The Intel Embedded Graphics Suite supports EDID-less displays that do not export timing modes. This is accomplished by allowing configuration of a Detailed Timing Descriptor (DTD), and associating that DTD with a specific display port. The IEGS provides further flexibility in allowing numerous DTDs to be defined and having the selection of the DTD be configurable though selection of Configuration IDs. The selection of the Configuration ID can be done from the System BIOS, as long as it supports the Intel 5F40h function and passes the appropriate Configuration ID to the VBIOS. The VBIOS in turn notifies the Graphics Driver of which Configuration ID is active. This is not required however, but the VBIOS and/or Graphics Driver require the Configuration ID to be set prior to installation.

2.2.4 sDVO Devices

The IEGD supports many third-party digital transmitters connected to the sDVO ports of the GMCH. The driver code that supports each of these devices is abstracted and is a separate driver called a port driver. Port drivers can be dynamically loaded at the time IEGD is initialized, and IEGD can be configured to allow any number of these port drivers to be loaded. By default, all the port drivers for the devices listed in the following table as Included in Release Package will be loaded by default if the corresponding transmitter is detected. If a port driver is not specified in the configuration before installation, that device will not be detected, and the port driver will not be loaded. The configuration can be modified before installation to prevent certain port drivers from being loaded or to include additional port drivers to load.

Table 7. SDVO Devices Supported (Sheet 1 of 2)

Device	VBIOS/EFI Video Driver Support	Graphics Driver Support
Internal LVDS	√	√
Internal TV Out		√
Chrontel CH7021*	√	√
Chrontel CH7305*	√	√
Chrontel CH7301*	√	√
Chrontel CH7307*	√	√
Chrontel CH7308*	√	√
Chrontel CH7317*	√	√

Table 7. SDVO Devices Supported (Continued) (Sheet 2 of 2)

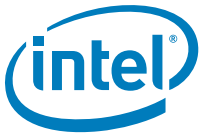
Device	VBIOS/EFI Video Driver Support	Graphics Driver Support
Chrontel CH7315/CH7319/CH7320*	√	√
Silicon Image SiI 1362*	√	√
Silicon Image SiI 1364*	√	√

2.2.5 Rotation

Rotation is the ability to rotate the display for the Intel Embedded Graphics Driver. Rotation support includes 0°, 90°, 180°, 270°. Rotation is supported only on the following chipsets using Windows XP and Linux operating systems:

- Intel® Q35 Express chipset
- Mobile Intel® GME965 Express chipset
- Intel® Q965 Express chipset
- Mobile Intel® 945GSE Express chipset
- Mobile Intel® 945GME Express chipset
- Mobile Intel® 945GM Express chipset
- Intel® 945G Express chipset
- Intel® 915GV Express chipset
- Mobile Intel® 915GME Express chipset
- Mobile Intel® 910GMLE Express chipset

Note: Rotation is not supported with the VBIOS.



3.0 Platform Configuration Using CED

The Intel® IEGD Configuration Editor (CED) is a Graphical User Interface (GUI) that allows you to create configurations, package the configurations, and create installations that can be loaded directly on a specific OS or Video BIOS platform. Configurations are associated with a specific chipset and can be created for any one of the following supported chipsets:

- Intel® GM45 Express chipset
- Intel® System Controller Hub US15W chipset
- Intel® Q35 Express chipset
- Mobile Intel® GME965 Express chipset
- Intel® Q965 Express chipset
- Mobile Intel® 945GSE Express chipset
- Mobile Intel® 945GME Express chipset
- Intel® 945G Express chipset
- Intel® 915GV Express chipset
- Mobile Intel® 915GME Express chipset
- Mobile Intel® 910GMLE Express chipset

IEGD configurations can be created for the following supported operating systems and Video BIOS:

- Linux XFree86 and X.org
- Fedora Core 5
- Fedora 7
- SUSE 10 SP1
- Novell Linux POS 9 SP3
- Damn Small Linux
- Microsoft Windows XP, Windows WP Embedded*
  - — DirectX* 8.1 and 9.0 (DirectDraw* and Direct3D*)
- Microsoft Windows CE 5.0 and 6.0 (Note that Intel® System Controller Hub US15W chipset does NOT support Windows CE 5.0)

Note: The following features are NOT supported in IEGD 9.0:

- Microsoft Vista 2D + 3D (WDDM)
- Vista DirectX 9.0L, DirectX 10.0 (Combine with MS Vista 2D + 3D)

The CED GUI is designed for ease of use and configuration of the IEGD. Online help is available for each configuration page and each data field is validated. If you enter an incorrect value, the CED displays an error message at the top of the page and displays the valid range of values for the field. You will not be able to finish a configuration until all fields contain valid values.

The following sections show how to create a configuration for any of the supported chipsets, operating systems, and the IEGD Video BIOS.

- "Starting the CED" on page 28
- "Creating a New Configuration" on page 29
- "Creating a New Package" on page 50
- "Creating a New Customized DTD" on page 58
- "Generating an Installation" on page 61

3.1 Before You Begin

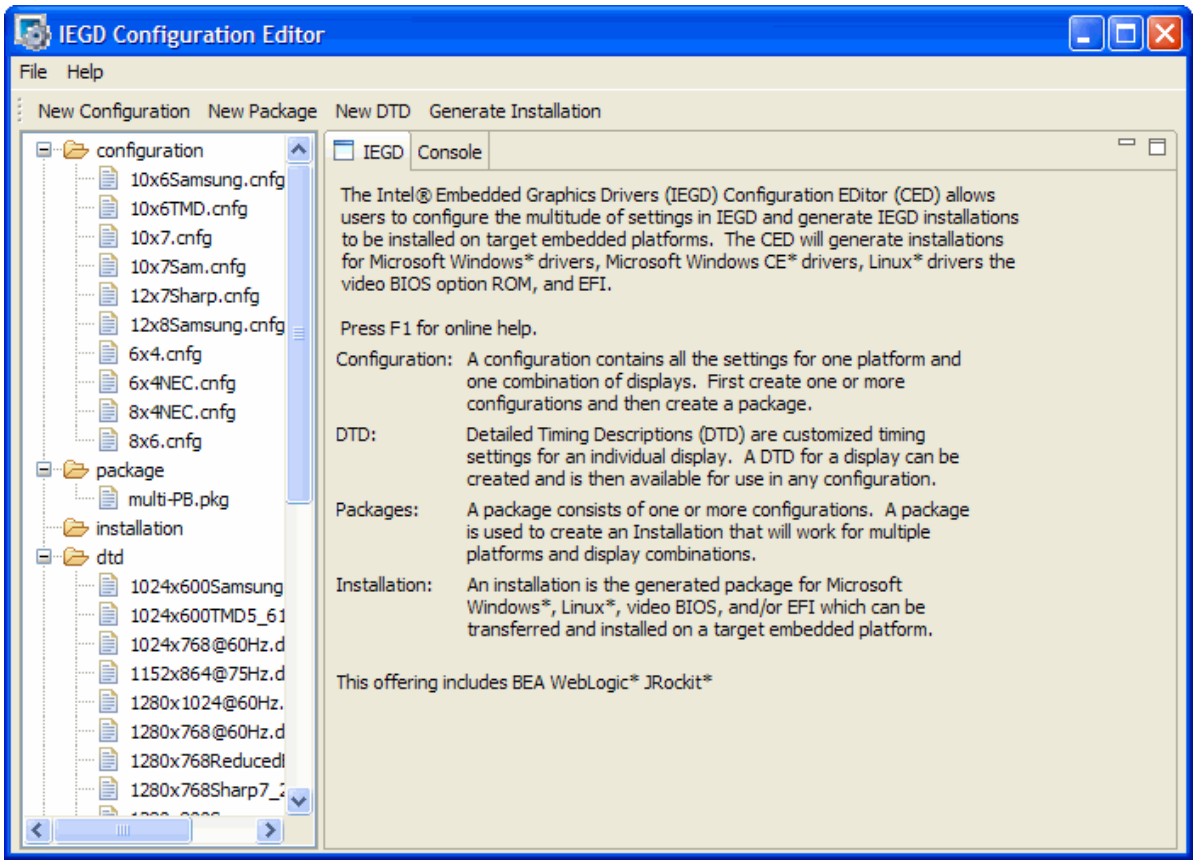
To configure the IEGD software using CED, you will need some information on the panel you are using. This information is usually found in the product specifications. In some cases the terminology used in the CED may not match the labels used in your panel's product specification. Refer to Table 21, "Timing Specification Example Values" on page 60 for hints on which specs correspond to CED DTD fields. After you obtain the correct specification values, you may need to derive other values for the DTD fields.

3.2 Creating a Configuration in CED – Summary Steps

The following steps present a sample CED configuration.

1. If you have custom panels and timings you may want to create your own Detailed Timings Descriptor (DTD); otherwise you can use the standard DTDs provided by CED. If needed, select **New DTD**.
   - Choose the DTD Type that most closely aligns with your display parameters, enter parameters, and then click **Finish**. For details, see "Creating a New Customized DTD" on page 58.
2. Select **New Configuration**.
   - Enter a name for the configuration, select the mode, chipset, ports, port drivers, DTDs, etc., for the configuration and then click **Finish**. For details, see "Creating a New Configuration" on page 29.
3. Select **New Package**.
   - Enter a name for the package, select the configurations for your package, the platforms for the installation, and then click **Finish**. For details, see "Creating a New Package" on page 50.
4. Select the created package and then select **Generate Installation**.
   The generated files are placed in the installation folder. The zip files (for Linux, Windows CE, and Windows operating systems) contain the generated XFree86, `iegd.reg`, or INF file. For details, see "Generating an Installation" on page 61.

Figure 4. Sample CED Configuration Start Page



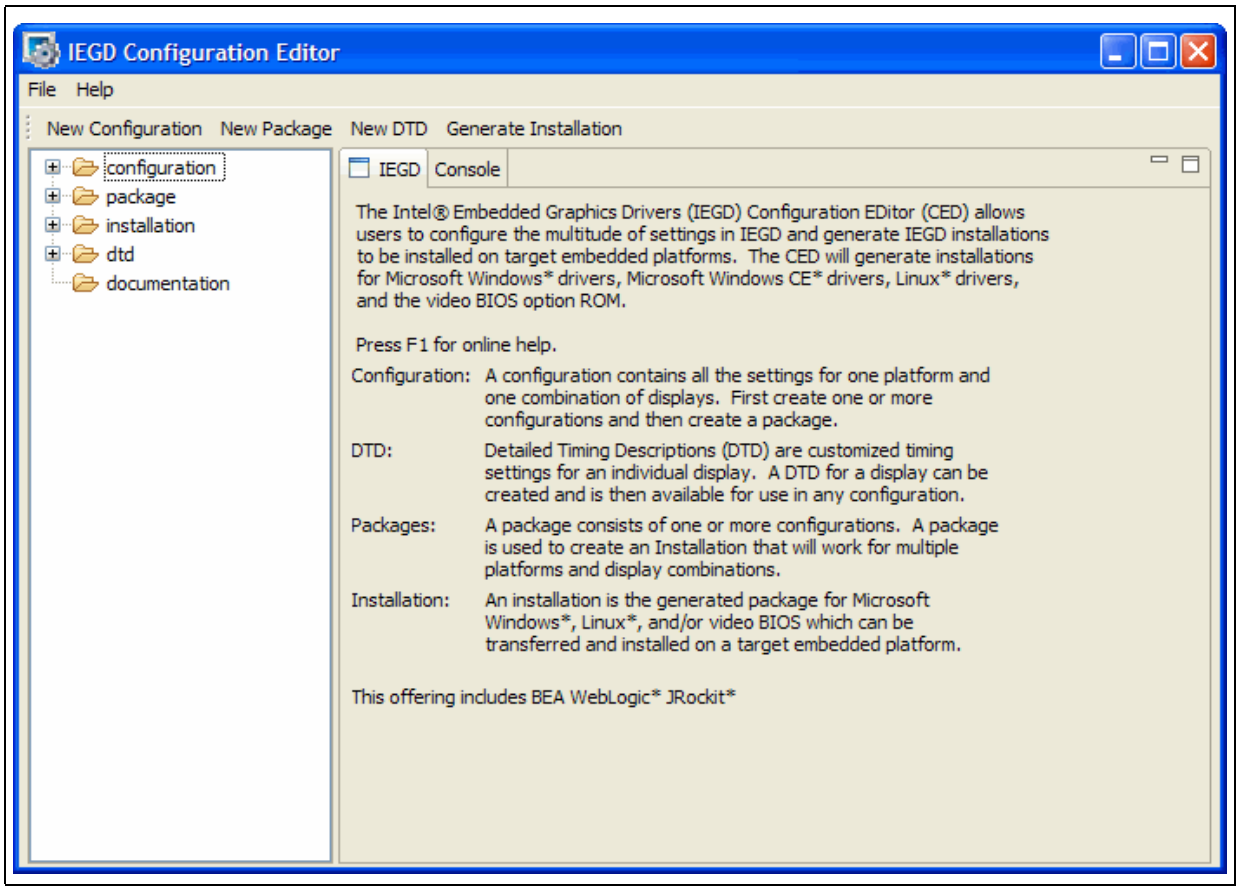
From this window, you can create configurations, package the configurations, and create installations from the packages that can be installed directly on a platform. The main window also provides a Console tab that displays information when you build a package or an installation.

The following sections show how to create a configuration for any of the supported chipsets, operating systems, and the IEGD Video BIOS.

- "Creating a New Configuration"
  - "Configuring Ports"
    - "Changing Port Attribute Settings"
    - "Changing I2C Settings"
    - "Changing Flat Panel Settings"
  - "Configuring Fastboot"
  - "Configuring the Video BIOS and EFI"
- "Creating a New Package"
  - "Entering Linux OS Options"
  - "Entering Windows OS Options"
- "Creating a New Customized DTD"
- "Generating an Installation"

3.3 Starting the CED

To start the IEGD CED, open the folder where you installed the CED and click the `iegd-ced.exe` icon. The IEGD CED splash window appears for a few moments followed by the IEGD Configuration Editor main window.



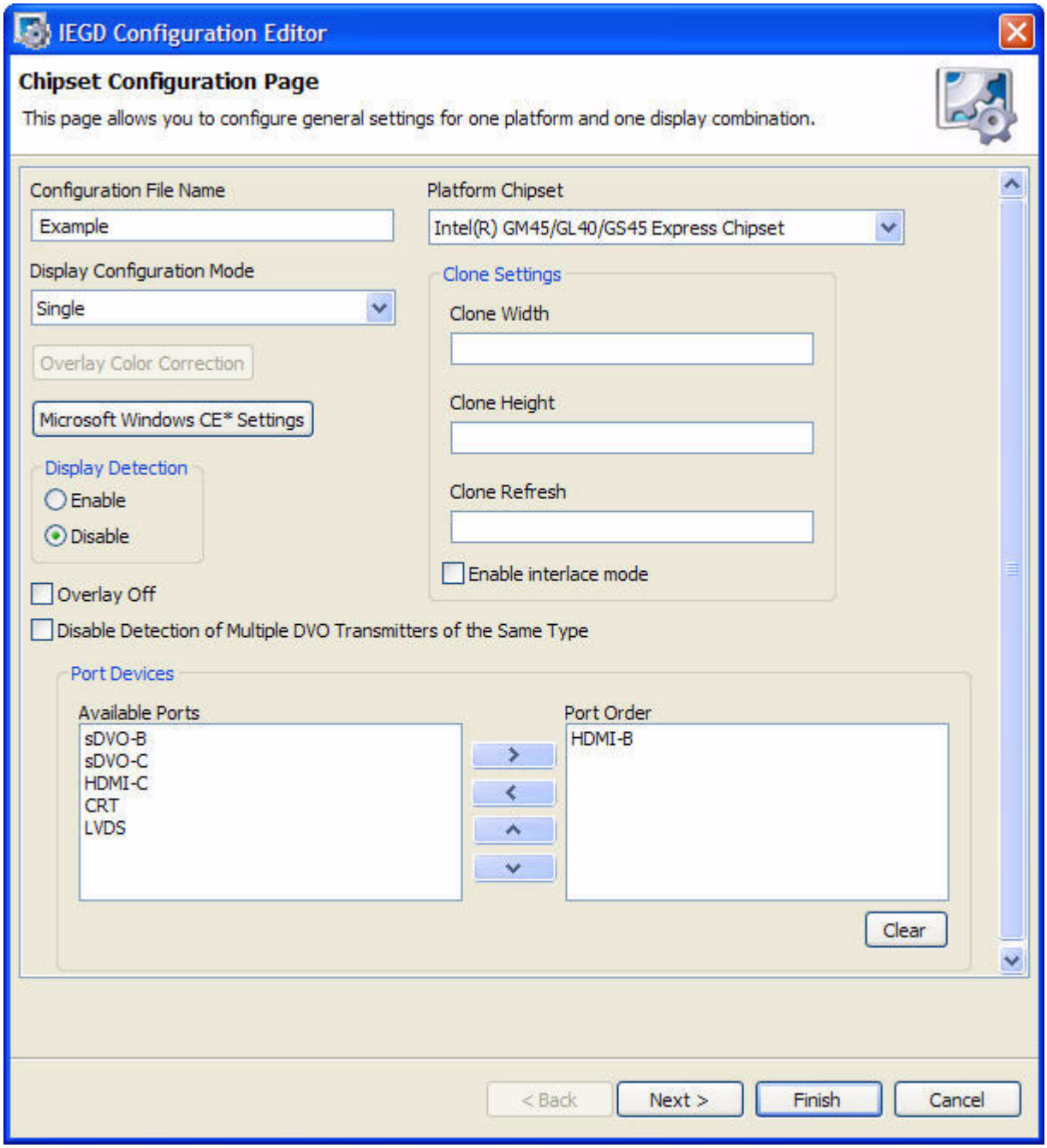
From this window, you can create configurations, package the configurations, and create installations from the packages that can be installed directly on a platform. The main window also provides a Console tab that displays information when you build a package or an installation.

The following sections show how to create a configuration for any of the supported chipsets, operating systems, and the IEGD Video BIOS.

3.4 Creating a New Configuration

To create a new configuration, click the **New Configuration** selection located on the top of the IEGD CED main window. The Chipset Configuration Page appears.

Figure 5. Chipset Configuration Page



The Chipset Configuration Page allows you to specify settings that apply to all OS, VBIOS, and EFI platforms.

The table below describes each setting on the Chipset Configuration page.

Table 8. Chipset Configuration Page Settings (Sheet 1 of 2)

Setting	Description
Configuration Name	Provide a name for the configuration you are creating. This name is required and is used when you create packages. The name can consist of any alphanumeric characters and any special characters and must be between 1 and 50 characters. You must enter a configuration before you can enter any other information on this page.
Platform Chipset	Select the target chipset for this configuration from the drop-down list.
Display Configuration Mode	Select the type of display configuration from the drop-down list. You can select any one of the following display configurations: • Single — Single display configuration • Twin — Two displays where both displays have the same resolution, refresh rate, and content • Clone — Two displays where both displays have the same content but can have different resolutions and timings. • DIH — Dual Independent Head. This is a configuration where both displays can have different resolutions, different refresh rates, and different content. ***Note:*** On Microsoft Windows* DIH configurations, the display DOES NOT automatically come up in extended display mode. You must go into the Display properties on the Control Panel and manually set the display to DIH mode.
Overlay Color Correction	Overlay Color Correction allows the Overlay plane to have color-correction settings that are different from the main frame buffer color-correction settings. See "Overlay Color Correction" on page 31. ***Note:*** Overlay color correction is not supported on the Intel® GM45 chipset.
Microsoft Windows CE Settings	If you are creating a package for a Microsoft Windows* CE platform, click the WinCE Settings button for additional settings that may be required for your configuration. Please see "Changing Windows CE OS Options" on page 34 for descriptions of these settings.
Display Detection	Display Detection allows you to specify if the driver should detect displays on the system. The default is Disabled. For more information on Display Detection, refer to "Display Detection and Initialization" on page 74.
Port Devices (Available Ports, Port Order)	The Port Devices section lists the ports available based on the chipset selected. The Available Ports box lists the ports that are available. You can move these port devices to the Port Order box to determine the search order for detecting attached displays. To move a port device to the Port Order box, either double-click the port device or click the port device to highlight it, and then click the right arrow button to move it from the Available Ports to the Port Order box. The Port Order section allows you to determine the search order for detecting attached displays for the Display Detection feature. When Display Detection is enabled, the Port Order determines which display is primary and which display is secondary. You can choose default ordering by not moving any of the Available Ports to the Port Order box and leaving the Port Order box empty. Default ordering is chipset-specific. See Table 48, "Default Search Order" on page 203 for more information on default port ordering based on chipset. When you move one or more ports to the Port Order box, you can configure each port by clicking **Next**. For each port listed in the Port Order box, you can click **Next** to configure each port. See "Configuring Ports" on page 36 for information on configuring ports.

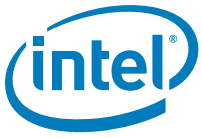


Table 8. Chipset Configuration Page Settings (Sheet 2 of 2)

Setting	Description
Clone Settings Clone Width Clone Height Clone Refresh Enable interlace mode	If you are creating a clone display configuration, you can specify the width, height, and refresh rate for the clone display in this section. For more information about clone display configurations, refer to "Enhanced Clone Mode Support" on page 84.
Disable off screen BMP support (No DFB)	This field enables the IEGD to pass the DIB call back to the OS. This is required in certain circumstances to improve performance.
Overlay Off	This field allows you disable Overlay support, which is enabled by default. ***Note:*** This field is only for Microsoft Windows* and Microsoft Windows CE operating systems only. The Linux* OS configuration for the XF86Conf provides a standard option that performs the same function.

See also the following topics:

- "Setting Color Correction"
- "Changing Windows CE OS Options"
- "Configuring Ports"
- "Configuring the Video BIOS and EFI"
- "Creating a New Package"

3.4.1 Setting Color Correction

Color Correction is available for both overlays and framebuffers, and is accessed under the **New Configuration** link at the top of the main CED window. For both overlay and framebuffer color correction, user-assigned values must be between 0.6 to 6. By default, gamma is 1.0 (no correction).

Note: Overlay color correction is not supported on the Intel® GM45 chipset.

3.4.1.1 Overlay Color Correction

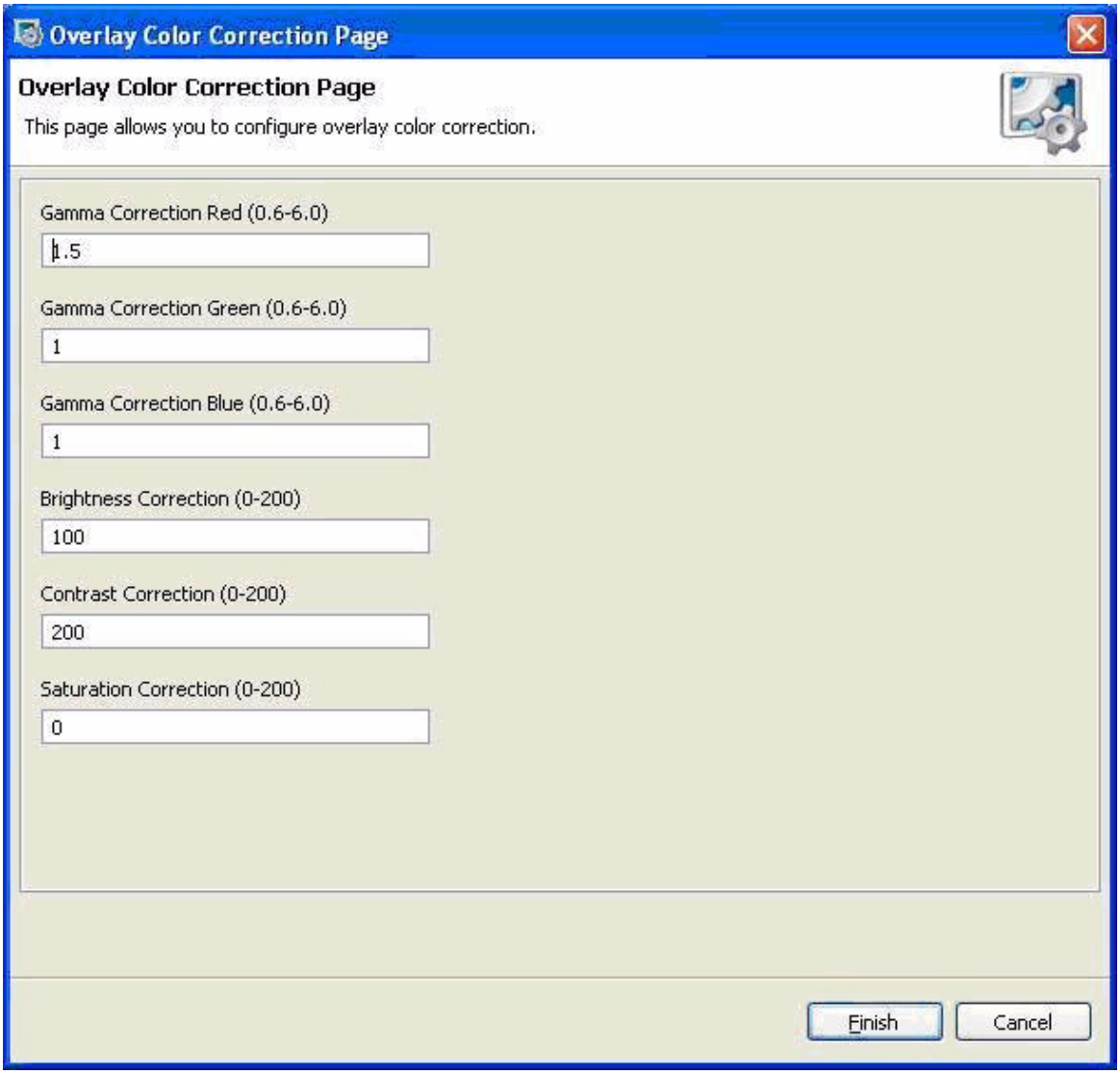
Overlay Color Correction allows the Overlay plane to have color-correction settings that are different from the main framebuffer color-correction settings. This feature allows you to color-correct for red, green, and blue, plus it enables you to adjust brightness, contrast, and saturation.

Table 9. Overlay Color Correction Values (applies to ALL color)

Gamma:	0.6 to 6.0 (default value is 1)
Brightness:	0 to 200 (default value is 100)
Contrast:	0 to 200 (default value is 100)
Saturation:	0 to 200 (default value is 100)

To assign overlay color correction, click the **Overlay Color Correction** button on the Chipset Configuration Page. The Overlay Color Correction Page appears, as shown in the figure below.

Figure 6. Overlay Color Correction Page



Add your desired values to the correction fields and then click **Finish**.

3.4.1.2 Color Correction Attributes (Framebuffer)

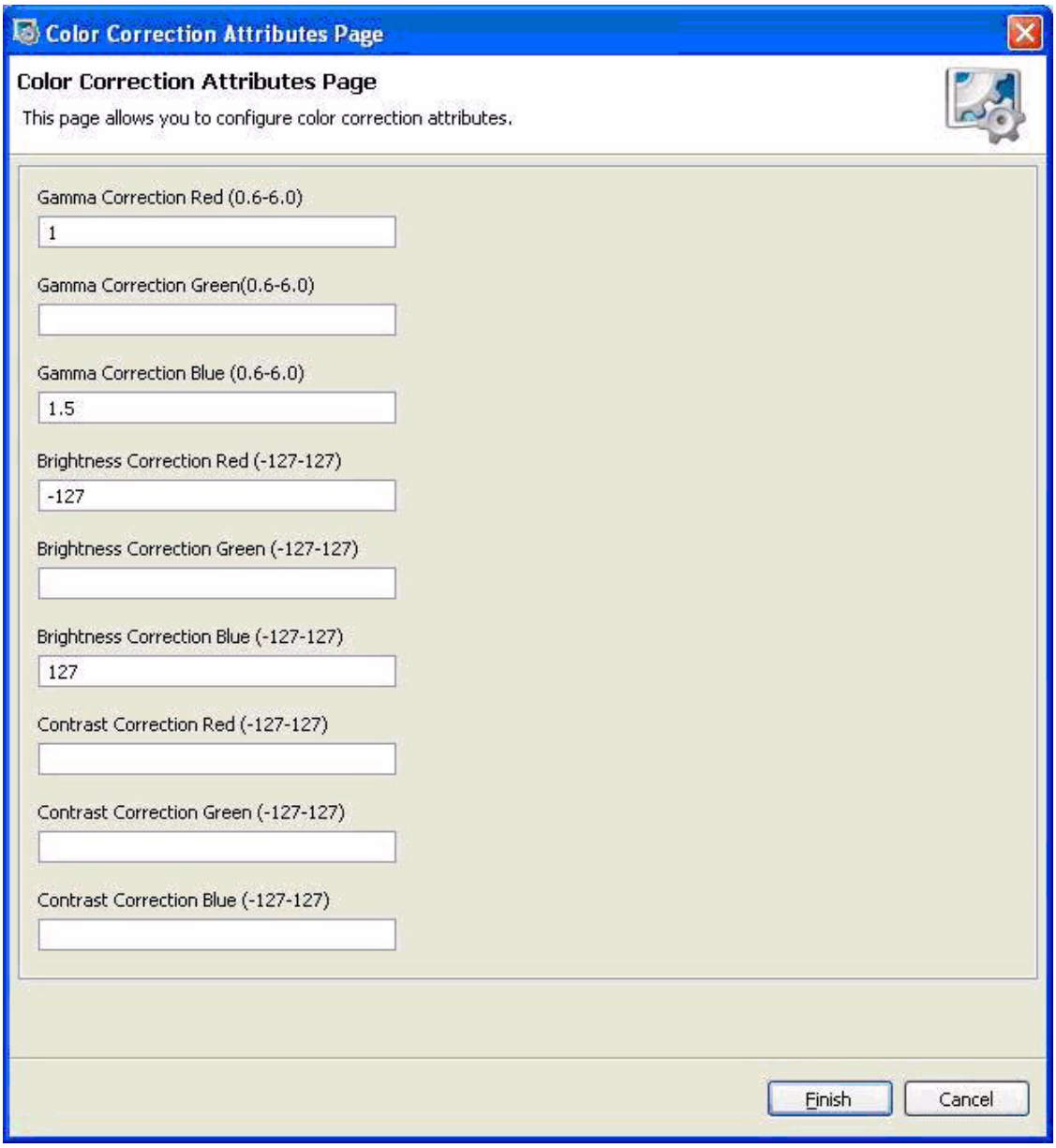
Color Correction Attributes allow you to adjust the main Framebuffer color attributes. This feature allows you to color-correct for red, green, and blue, and enables you to adjust brightness and contrast.

Table 10. Framebuffer Color Correction Values (applies to R, G, B color)

Gamma:	0.6 to 6.0 (default value is 1)
Brightness:	-127 to 127 (default value is 0)
Contrast:	-127 to 127 (default value is 0)

To assign overlay color correction, click the **Color Correction Attributes** button on the Chipset Configuration Page. The Overlay Color Correction Page appears, as shown in the figure below.

Figure 7. Overlay Color Correction Page

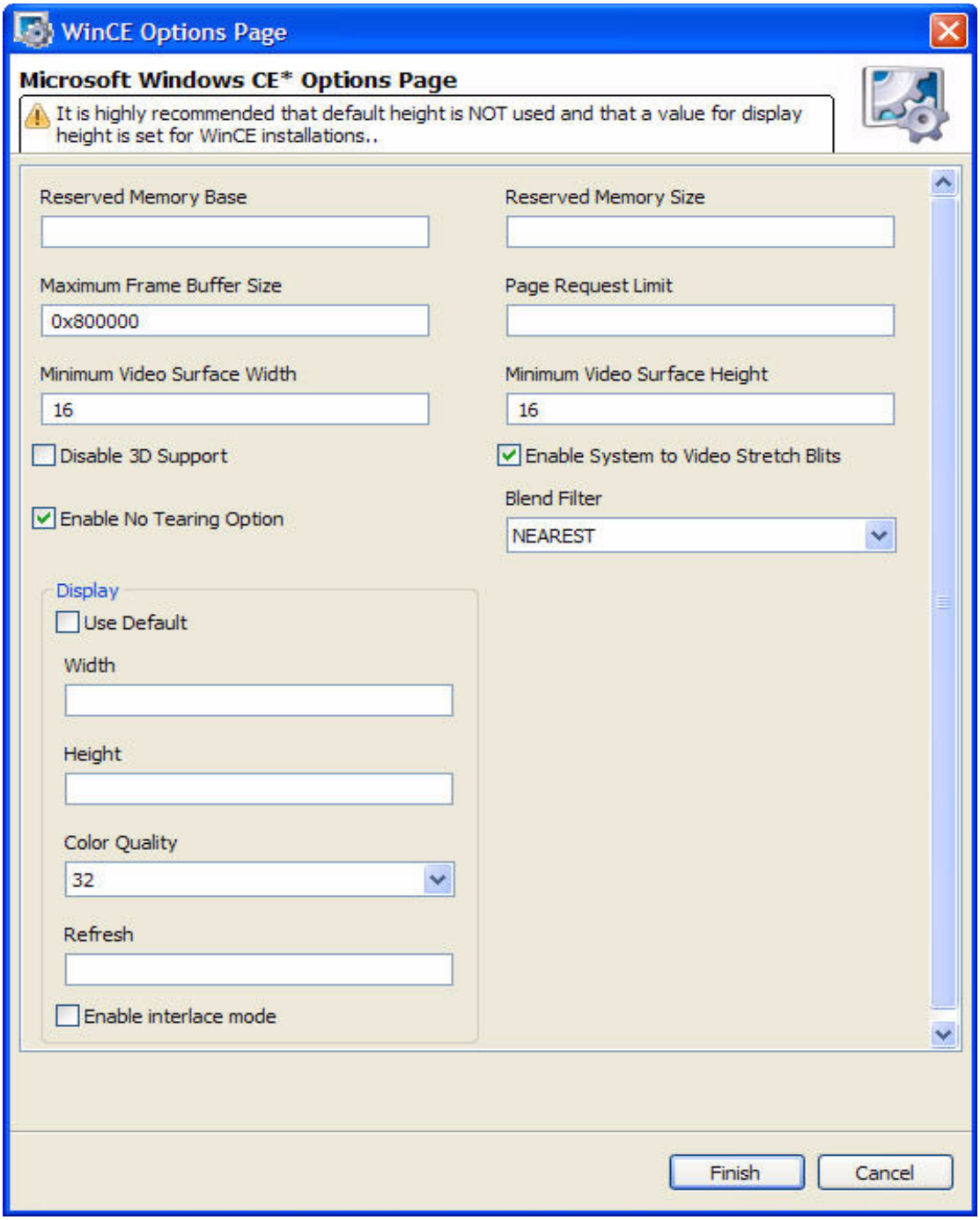


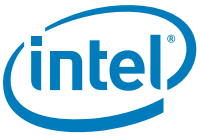
Add your desired values to the correction fields and then click **Finish**.

3.4.2 Changing Windows CE OS Options

The WinCE Options Page allows you to enter Windows CE OS-specific options into the configuration. When you click the **WinCE Settings** button from the IEGD Package Page, the following page appears.

Figure 8. Chipset Configuration Page





The table below describes each field on this page.

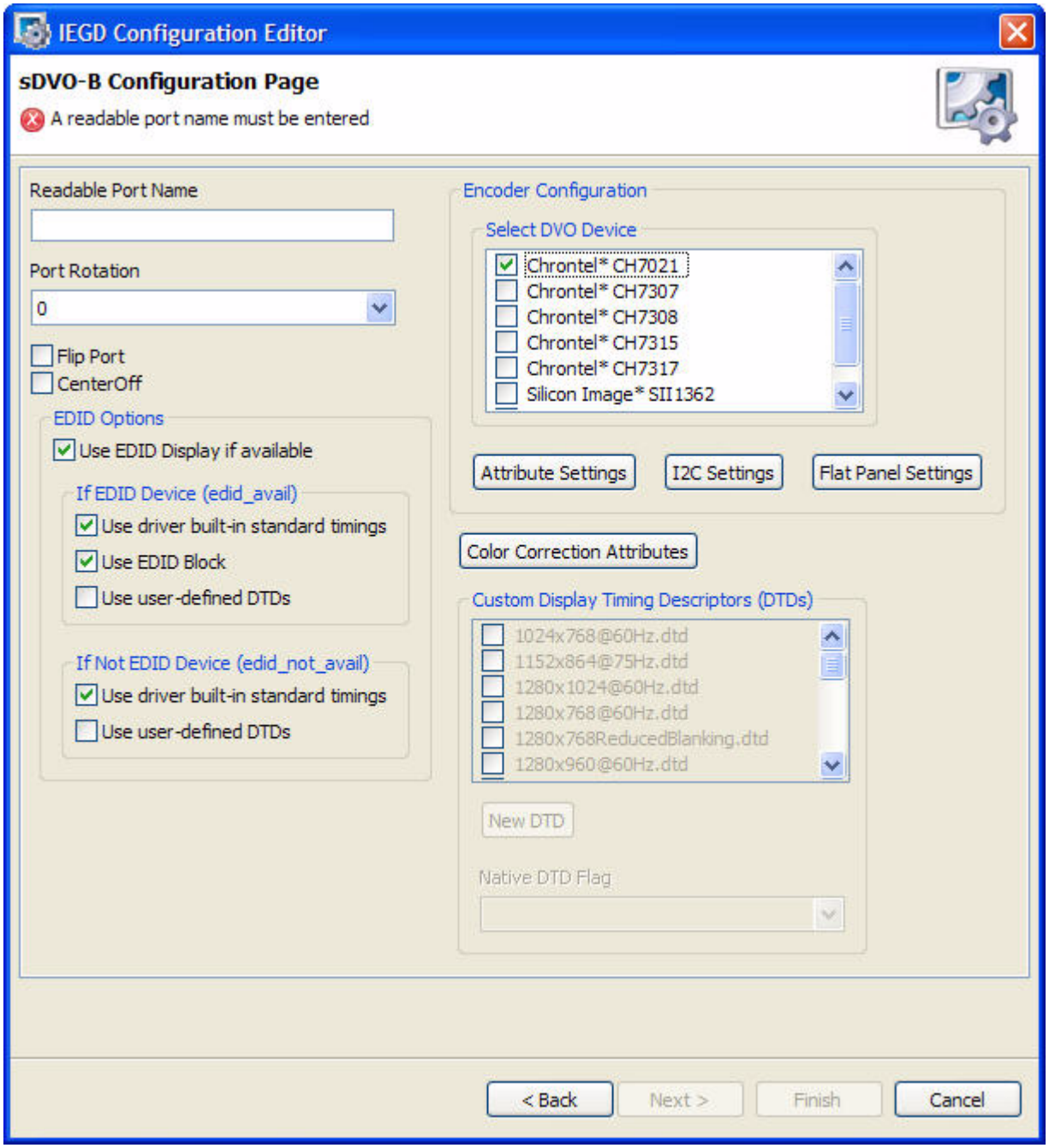
Table 11. Windows CE OS Settings

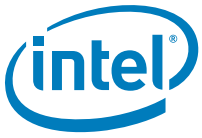
Windows CE OS Option	Description
Reserved Memory Base Reserved Memory Size	These two fields let you specify the amount and the starting point of statically reserved video memory. Video memory can be statically reserved or dynamically allocated on demand. If both Reserved Memory Base and Reserved Memory Size are non-zero, video memory allocation utilizes the static model. Base plus the Size must extend to TOM (Top Of Memory) and not conflict with other reserved memory arenas in the `config.bib` file. The default for both Reserved Memory Base and Reserved Memory Size is zero, indicating a dynamic allocation model. Default behavior disables static memory model.
Maximum Frame Buffer Size	The maximum size of the expected frame buffer. By providing this hint, the display driver can more efficiently organize GART memory, leading to a smaller video memory consumption. This value must be greater than or equal to the expected size of the frame buffer. Units represent the number of bytes and are specified in hexadecimal. Specifying zero causes the default frame buffer reservation sizing. The default is 0x300000
Page Request Limit	The Page Request Limit controls the maximum allocations of offscreen video surfaces, buffers, etc. This value represents the number of pages (4K) allocated and is independent of dynamic or static memory configuration. The maximum is 128MB (0x8000)
Minimum Video Surface Width Minimum Video Surface Height	In pixels, the minimum width and height of surfaces in order to be acceptable for allocation in video memory. Due to hardware restrictions that optimize memory access, it is advisable to reserve video memory for larger surfaces and allow GDI and DirectDraw* to allocate small surfaces from system memory. Default value for both width and height is 16.
Enable System to Video Stretch Blits	When checked, this enables system-to-video memory stretch blit operations to take advantage of hardware-accelerated filtering. Normally, it is more efficient to allow GDI to conduct system-to-video stretch blits, but the default filtering used by GDI is Nearest. See the Blend Filter below for hardware accelerated filtering options. The default is disabled.
Disable D3D	Specify whether to disable or enable D3D graphics. ***Note:*** For Windows CE 5.0 OS, this box must be checked as the IEGD drivers currently do not support D3D on Windows CE 5.0 systems.
Blend Filter	Provides selection of hardware-accelerated filtering methods for stretch blit operations. The choices are: • Nearest • Bilinear • Anisotropic Default: Nearest
Enable No Tearing Option	If enabled, all blit operations to the frame buffer are synchronized with video sync to eliminate any visible tearing on the display screen. Disabling this feature achieves a performance gain.
Display Use Default Width Height Color Quality Refresh Enable interlace mode	The Display section allows you select the default resolution, color depth, and refresh rate for the configuration. If you do not select a default display mode, the configuration uses the default display mode for the operating system it is installed on.

3.4.3 Configuring Ports

You can configure each port listed in the Port Order box of the Chipset Configuration Page by clicking **Next**. When you do, a port Configuration Page appears similar to the one shown here.

Figure 9. Port Configuration Page





The Port Configuration Page allows you to specify whether to use EDID timings or customized DTD timings for the display connected to this specific port. From this page, you can also specify Attribute Settings, I2C Settings, and Flat Panel Settings and create a new DTD that can be used with any configuration.

The table below describes each field on this page.

Table 12. Port Configuration Settings (Sheet 1 of 2)

Port Configuration Field	Description
Readable Port Name	Enter a name for the port. This is a required field and the name must be between 1 and 50 characters and may contain spaces.
Port Rotation	This list allows you select a rotation for the display connected to this port. You can choose between 0, 90, 180, and 270 degrees. The default is 0.
Flip Port	Check this box if you want the display connected to this port to be inverted horizontally. The default is not to invert horizontally.
EDID Options	This section allows you to set EDID options for the display. The IEGD supports three different types of EDID display modes: • Built-in display modes: These modes are hard-coded in the IEGD. These modes can be filtered based on the EDID block. • EDID Block: These are Detailed Timing Descriptors read from an EDID display. An EDID display can contain DTD as well as other information about the display. • User-specified DTDs. If you want to use the display's EDID information if it is available, click the **Use EDID Display if Available** check box. If the display attached to this port contains EDID information, you can choose one or more of the following options from the If EDID Device section to determine which set of timings to use for the display connected to the port: • Use driver built in standard timings — If this box is checked, the standard timings built into the IEGD are used. • Use EDID block — If this box is checked, the EDID block is used. • Use user defined DTDs — If this box is checked, a user-defined DTD is used. You can select which DTD to use by checking the appropriate box in the Custom Display Timings Descriptors (DTDs) section. If no DTDs are defined, you can click **New DTD** and create a custom DTD. For information on creating custom DTD, refer to Table 19 on page 54. If you select both **Use driver built in standard timings** and **Use EDID block**, the IEGD uses its built-in display timings and the timings provided by the display. If the display attached to this port does not contain EDID information, you can choose one or both of the following options from the If Not EDID Device section: • Use driver built in standard timings — If this box is checked, the standard timings are used. • Use user defined DTDs — If this box is checked, a user defined DTD is used. You can select which DTD to use by checking the appropriate box in the Custom Display Timings Descriptors (DTDs) section. If no DTDs are defined, you can click **New DTD** and create a custom DTD. For information on creating custom DTD, refer to Table 19 on page 54. See "Sample Advanced EDID Configurations" on page 77 for example configurations.

Table 12. Port Configuration Settings (Sheet 2 of 2)

Port Configuration Field	Description
Encoder Configuration	This section lets you to specify the type of encoder connected to an sDVO port and encoder Attributes, I2C settings, and Flat Panel settings for the port. The **Select sDVO Device** drop-down list contains the list of all supported sDVO devices. Select the device that will be connected to this port. To change the device's attributes, click the **Attribute Settings** button. Refer to "Changing Port Attribute Settings" for information on device attributes. To change the device's I2C settings, click the **I2C Settings** button. See "Changing I2C Settings" on page 40 for information on I2C settings. To change the device's flat panel settings, click the **Flat Panel Settings** button. See "Changing Flat Panel Settings" on page 41 for information for changing flat panel settings.
Color Correction Attributes	Color Correction Attributes allow you to adjust the main Frame Buffer color attributes. See "Color Correction Attributes (Framebuffer)" on page 32.
Native DTD	The Native DTD list lets you choose whether to use a display's built in timings.

See also the following topics:

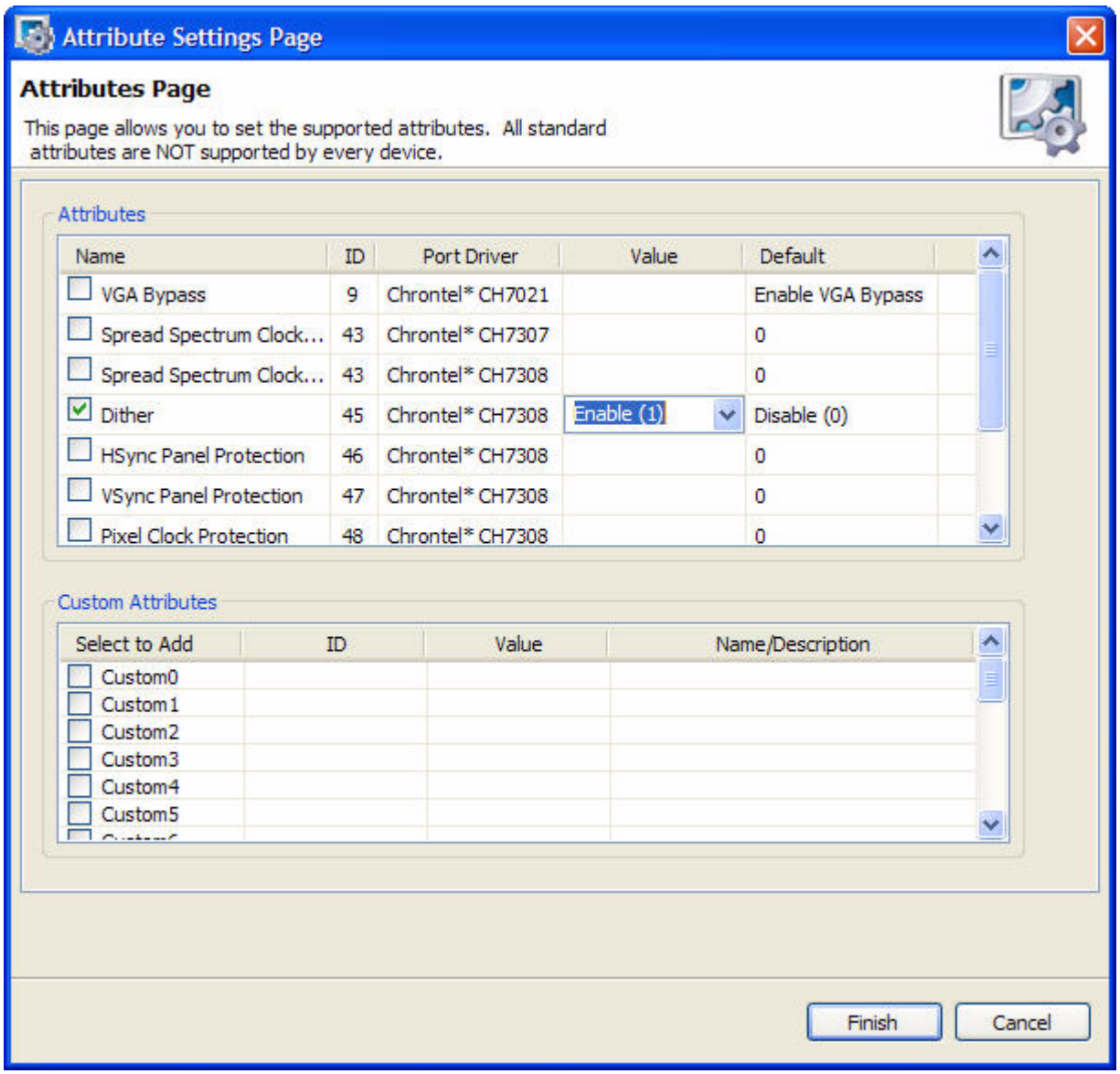
- "Changing Port Attribute Settings"
- "Changing I2C Settings"
- "Changing Flat Panel Settings"
- "Color Correction Attributes (Framebuffer)"
- "Creating a New Customized DTD"

3.4.3.1 Changing Port Attribute Settings

When you click the **Attributes Settings** button from the Encoder Configuration section of the Port Configuration Page, the CED displays a page of attributes for the selected encoder device. The actual page that appears depends upon the encoder device selected and only the attributes that apply to the selected encoder appear. For a full description of all attributes for all supported encoders, refer to "Port Driver Attributes" on page 197.

The figure below shows a sample Attributes Settings Page for the Chrontel CH7021, CH7307, and CH7308 encoder.

Figure 10. Attribute Settings Page



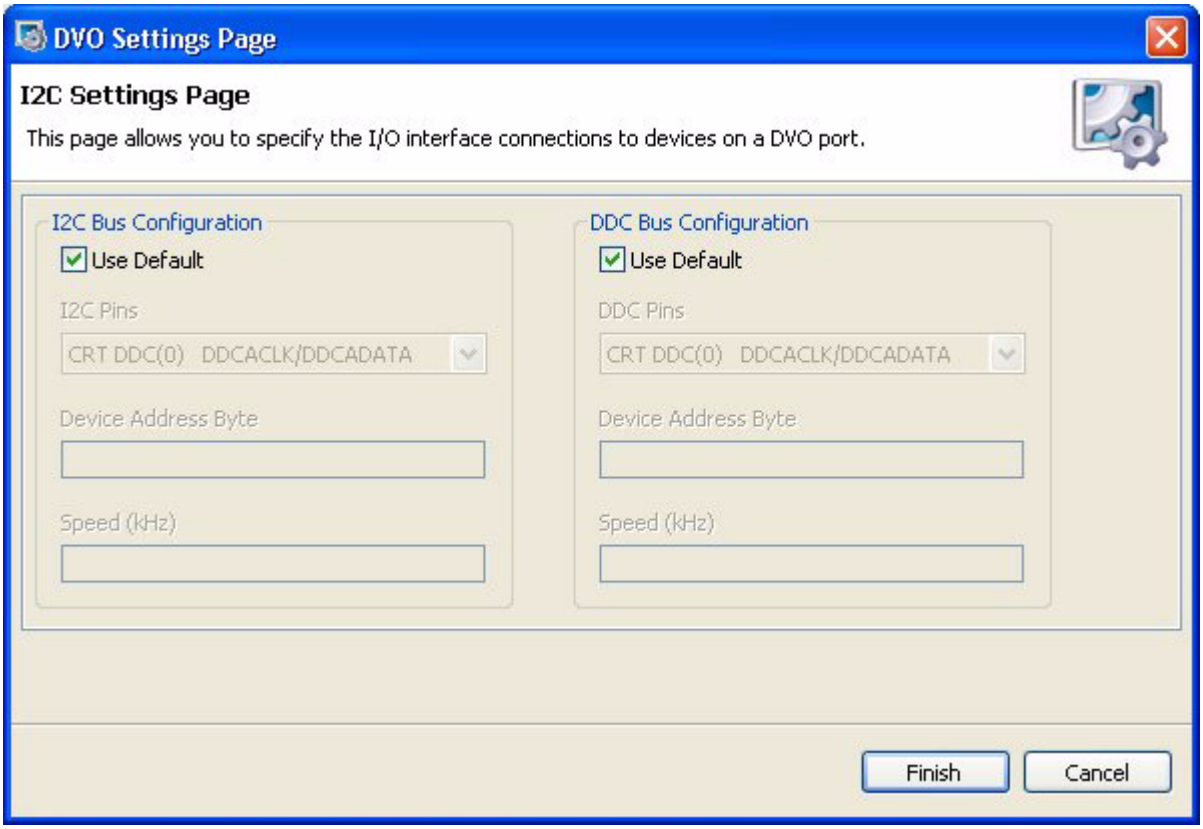
When the Attributes Settings Page first appears, it shows the **Use Default** box checked for all attributes.

To change a default value, clear the **Use Default** check box and enter a new value. For a description of all attributes for all supported encoders, see "Port Driver Attributes" on page 197.

3.4.3.2 Changing I2C Settings

The I2C Settings Page allows you to specify the I/O interface connections to devices on an sDVO port. When you click **I2C Settings** from the Port Configuration Page, the following screen appears.

Figure 11. sDVO Settings Page



To change the default settings for the **I2C Bus Configuration** or the **DDC Bus Configuration**, clear the **Use Default** box and enter new values. The following table describes each field on this page.

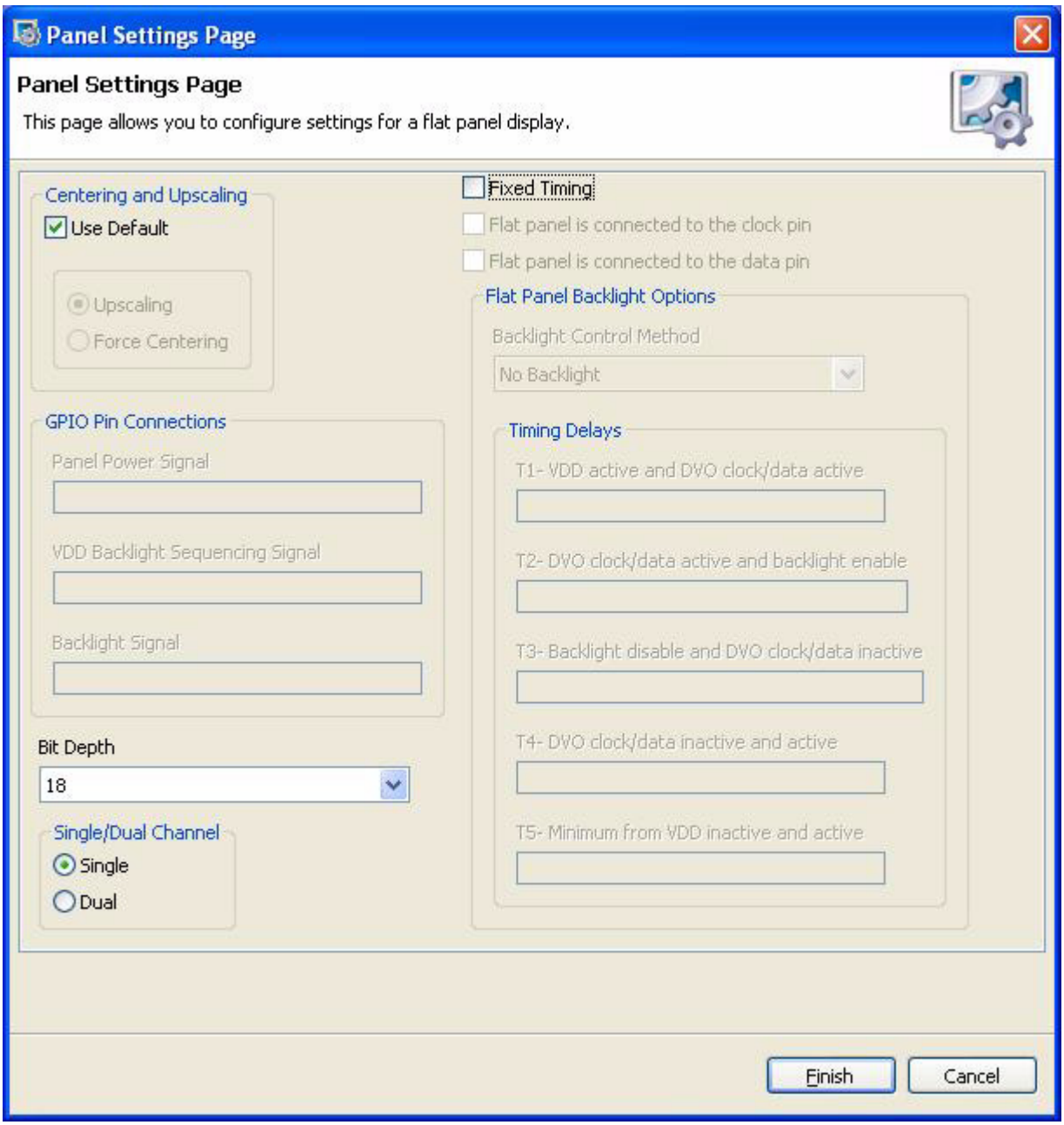
Table 13. I2C Settings

I2C/DDC Bus Setting	Description
I2C pins/DDC pins	This drop-down list allows you to select the pins that are used for I/O to the display connected to the encoder. This is the GPIO pin pair used on the I2C and DDC buses to read and write to device registers. You can choose between CRT DDC or sDVO I2C. The default is CRT DDC.
Device Address Byte	You can enter a device address byte for the device that this port is connected to in these boxes. The I2C device address is for reading and writing device registers. The device address byte must be in 8-bit format with the 7-bit slave address assigned to its bits 7:1 and bit 0 set to 0. The DDC Device Address Byte is the I2C device address for reading EDID data from the display through the DDC bus.
Speed (KHz)	Speed of I2C bus for the device and for the EDID device. The range for these two fields is 10-400 KHz.

3.4.3.3 Changing Flat Panel Settings

The Panel Settings Page allows you to specify settings for a flat panel display connected to this sDVO port. When you click **Flat Panel Settings** from the Port Configuration Page, the following screen appears.

Figure 12. Panel Settings Page



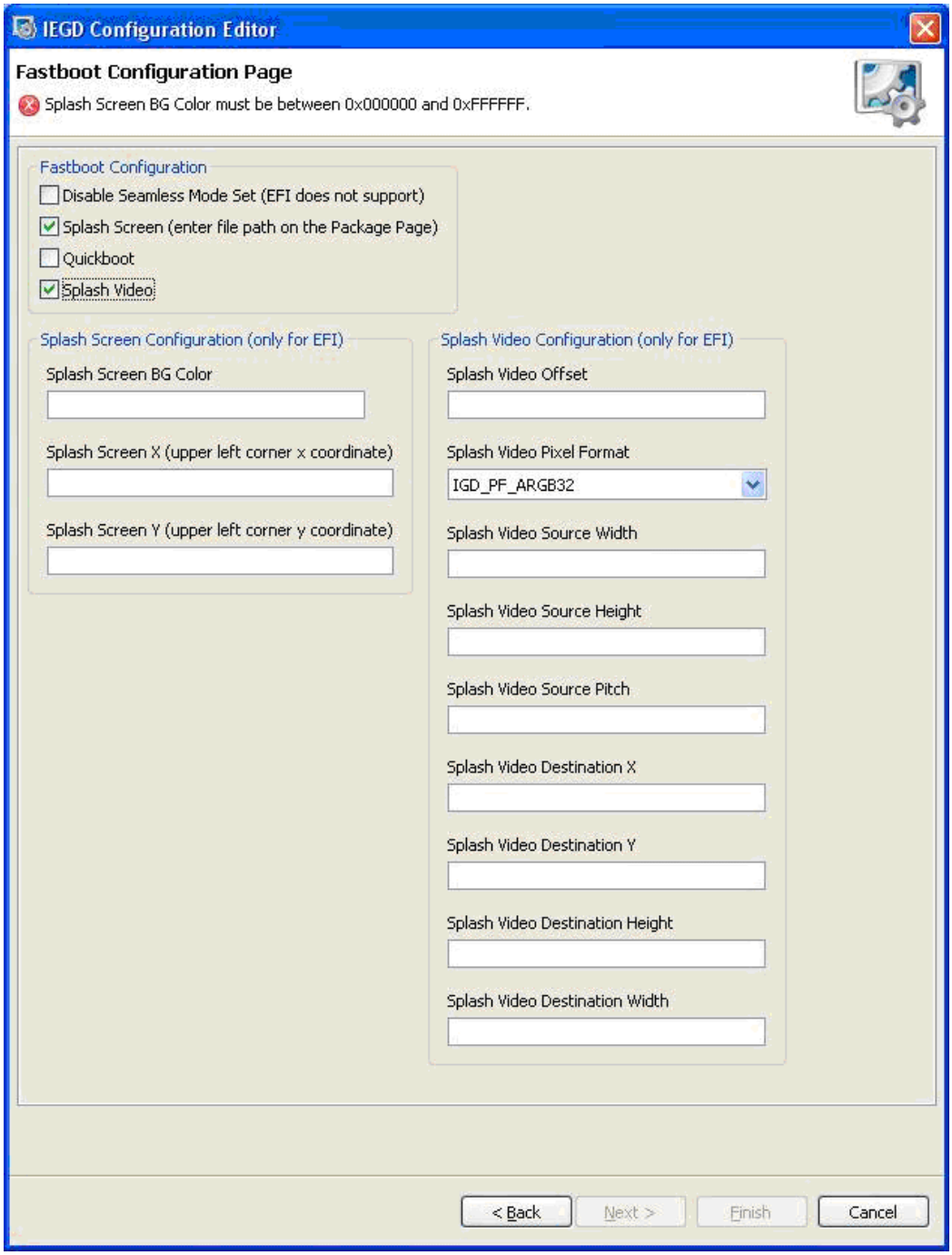
The table below describes each section of this page.

Table 14. Panel Settings Options

Flat Panel Settings	Description
Fixed Timing	This section indicates whether the attached display is a fixed timing display.
GPIO Pin Connections	If you select **Port Driver, GMCH**, or **ICH** from the Flat Panel Backlight Options list, you can specify the following GPIO pin connections. • Panel Power Signal — GPIO connection for panel power. • VDD backlight sequence signal — GPIO connection for backlight power on/off sequencing signal. • Backlight signal — GPIO connection to enable backlight signal.
Bit Depth	This list lets you select a color depth for the panel. You can choose either 18 or 24 bit color depth. The default is 18.
Single/Dual Channel	This check box determines the chip channel mode. Unchecked, the default, is for single channel mode. Checked is for Dual mode. Single mode is recommended for TV displays. For flat panels, refer to the panel's specification.
Flat Panel Backlight Options	This section provides options for controlling the backlight of the flat panel display and specifying timing delays. • The Backlight Control Methods list lets you choose among **No Backlight**, **Port Driver**, **GMCH**, or **ICH** to control the backlight. If choose Port Driver, GMCH, or ICH, you can specify the timing delays in the Timing Delays section and the GPIO pin connections in the GPIO Pin Connections section. The default is No Backlight.
Timing Delays	This section lets you specify timing delays for the backlight signals as follows: • VDD active and clock/data active: 1-512, increment by 1. Default is 255. • sDVO active and backlight enable: 2-256, increment by 2. Default is 127. • Backlight disable and sDVO clock/data inactive: 2-256, increment by 2. Default is 127. • Clock/data active and inactive: 1-512, increment by 1. Default is 255. • Minimum from VDD inactive and active: 1-1600, increment by 50. Default is 800.

3.4.4 Configuring Fastboot

Figure 13. Fastboot Configuration Page



The table below describes each section of this page.

Table 15. Fastboot Options (Sheet 1 of 2)

Fastboot Settings	Description
Disable Seamless Mode Set	The Seamless mode set feature ensures that on a properly configured embedded device there is only 1 mode set between power on and a fully functional system. Under normal circumstances a PC will set the mode several times during initialization which causes screen flicker and latency that is undesirable for an embedded device. With seamless mode set, the firmware sets the mode and the driver adopts the existing mode without altering the hardware state. This feature can be combined with splash screen or splash video for optimal effect. EFI does not support this feature.
Splash Screen	The Splash screen feature provides a user-configurable splash screen image that is loaded to the framebuffer at the earliest possible time by the EFI graphics driver and remains in place until overwritten by the OS or driver. Additionally the IEGD graphics driver can be configured to suppress OS drawing to the on-screen framebuffer until notified by an application. Instead, drawing is redirected to an off-screen framebuffer. When notified by the application, the IEGD driver will flip the already prepared off-screen framebuffer to be on-screen and cease redirection of drawing. In this manner the configured splash screen will be displayed early during boot and remain in place until a time when the OS is fully loaded and the application interface has been prepared.
Quickboot	The quickboot feature optimizes the speed that IEGD loads at the expense of compatibility and ease of use. Quickboot disables non-critical features that affect the initialization time of the driver that are not needed for targeted embedded applications.
Splash Video	The Splash Video feature provides a mechanism to use a portion of the off-screen pre-allocated video memory (“Stolen Memory”) as a video image that is displayed on an overlay to the framebuffer. The intention is that a video capture device external to IEGD will be configured to transfer a video stream to the configured location in video memory using DMA. The splash video remains in place until the IEGD driver is notified by an external application to disable the overlay.
Splash Screen BG Color (EFI only)	Splash Screen BG Color must be between 0x000000 and 0xFFFFFF.
Splash Screen X (EFI only)	The X location, in pixels, where the Firmware Splash Screen will be placed. This number is a signed number in 2's complement. Positive numbers are offset from the left of the screen. Negative numbers are offset from the right of the screen.
Splash Screen Y (EFI only)	The Y location, in pixels, where the Firmware Splash Screen will be placed. This number is a signed number in 2's complement. Positive numbers are offset from the top of the screen. Negative numbers are offset from the bottom of the screen.
Splash Screen File (EFI only)	The full path to a 32-bit bitmap file to be used as the splash screen. ***Note:*** Enter file path on the Package Page. See Figure 16 on page 50.
Splash Video Offset (EFI only)	The offset, in bytes, from the base of video memory where the Splash Video will be placed. Care must be taken to ensure that this location is past the end of the on-screen framebuffer and that the full Splash Video image within the pre-allocated video memory.
Splash Video Pixel Format (EFI only)	The pixel format of the Splash Video image in memory. The available pixel formats are encoded values used within the IEGD driver.
Splash Video Source Width (EFI only)	The width, in pixels, of the Splash Video image in memory.
Splash Video Source Height (EFI only)	The height, in pixels, of the Splash Video image in memory.
Splash Video Source Pitch (EFI only)	The pitch, in bytes, of the Splash Video image in memory. Pitch must be >= bytes per pixel * source width.
Splash Video Destination X (EFI only)	The X location, in pixels, where the Splash Video will be placed. This number is a signed number in 2’s complement. Positive numbers are offset from the left of the screen. Negative numbers are offset from the right of the screen.

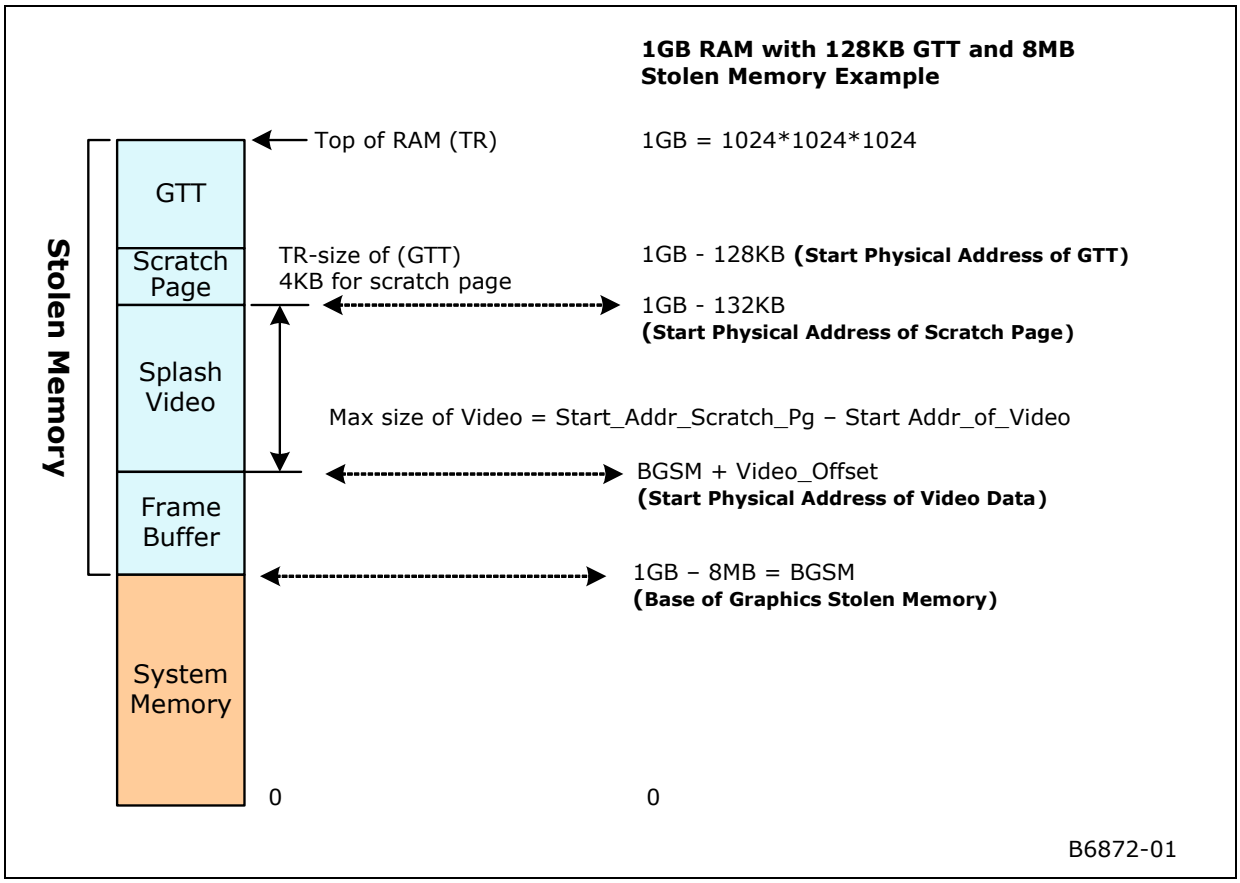
Table 15. Fastboot Options (Sheet 2 of 2)

Fastboot Settings	Description
Splash Video Destination Y (EFI only)	The Y location, in pixels, where the Splash Video will be placed. This number is a signed number in 2's complement. Positive numbers are offset from the top of the screen. Negative numbers are offset from the bottom of the screen.
Splash Video Destination Height (EFI only)	The height, in pixels, of the Splash Video window on the screen. This number must currently be the same as SrcHeight.
Splash Video Destination Width (EFI only)	The width of the screen. This number must currently be the same as SrcWidth.

3.4.4.1 Configuring Splash Video

The splash video feature can be used to display a video while the system is booting to the operating system. This section describes how to configure the options needed.

Figure 14. Splash Video with 8 MBytes of Stolen Memory Example



The Video DMA area is where the video will be streamed. It is part of the stolen memory of our Graphics Device.

The external PCI device that is connected to the camera needs to know the exact DDR RAM physical address to stream. Or dump the video data at that memory location.

To calculate the Start DDR RAM physical address:

Start_Phy_Ram_Addr = BGSM + Video_Offset

where **BGSM** = Base of Graphics Stolen Memory

and **Video_Offset** = Offset where the video data is present. This is what you enter into the CED tool.

There are two ways to calculate BGSM:

- The recommended method is to use the "setpci" command in Linux to find the BGSM from the PCI Config space.

  At the Linux command prompt, type the following:

  ```
  $setpci -s 0:2.0 0x5C.L
  ```

OR

- Find the amount of physical RAM populated in the system, for example, 1 Gbyte, and the stolen memory selected by the user in the system BIOS, for example, 8 Mbyte.

  BGSM = 1 Gbyte - 8 Mbytes = 0x4000 0000 - 0x80 0000 = 0x3F80 0000

3.4.4.2 How to Select the Video_Offset

Determine the size of the maximum resolution of the framebuffer.

Size = framebuffer_height * framebuffer_pitch

where **framebuffer_pitch** = framebuffer_width * Bytes_per_Pixel (page aligned)

For example, 1024x768 at 32-bit BPP:

Size = 768 * (1024 * 4) = 3145728 = 0x30 0000

For some usage models, the framebuffer pitch is set to 8192 bytes. In that case:

Size = 768 * (8192) = 6291456 = 0x60 0000

The Video_Offset can start from **0x30 0000** or **0x60 0000** (if the pitch is 8192). See the notes below on the recommended values for the Video Offset.

Max Size of Splash Video = Size of Stolen Memory - Max Frame buffer size – Size of GTT – Size of Scratch Page (4 KB)

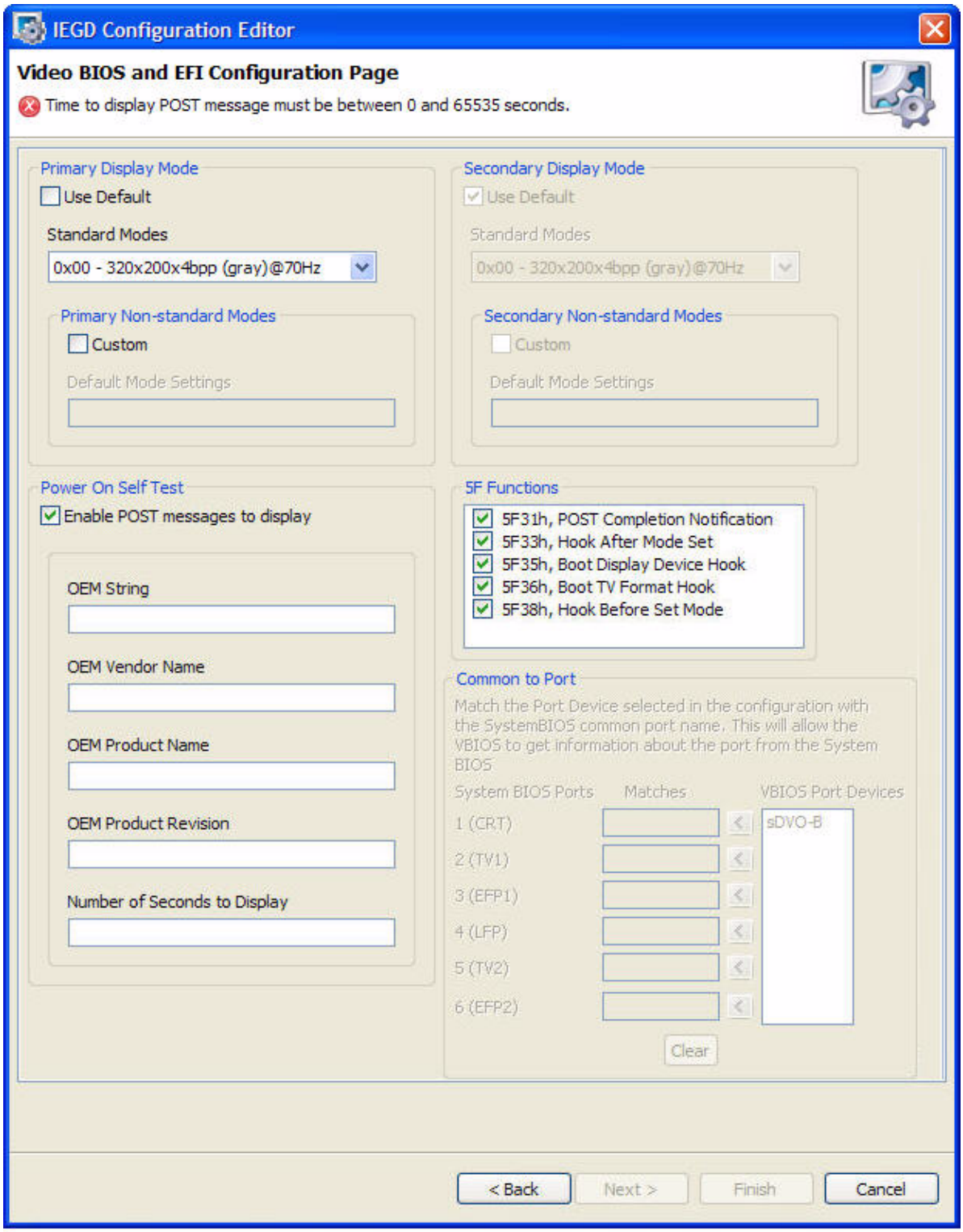
Notes:

1. For the Splash Video option the stolen memory MUST be a minimum of **8 Mbytes**. This is selected in the BIOS menu.
2. The recommended Video Offsets for the splash video are **0x600000** and **0x700000**.
3. If the Size of the Video frame is more than **1 Mbyte**, please choose **0x600000**.

3.4.5 Configuring the Video BIOS and EFI

The final page of the IEGD Configuration allows you to configure your video BIOS (if you are creating a configuration that includes the Video BIOS!) and EFI. You can configure the Video BIOS by clicking **Next** after you configure each port. When you do, the following Video BIOS and EFI Configuration Page appears.

Figure 15. Video BIOS Configuration Page



From this page, you can customize POST (Power On Self Test) messages and default display modes as well as matching port devices to System BIOS ports.

The table below describes each field on this page.

Table 16. Video BIOS Settings Options (Sheet 1 of 2)

Video BIOS Settings	Description
Primary Display Mode	This section allows you to specify a standard or a customized display mode for the primary display. You can select a standard mode from any of the standard modes listed in the drop-down list. If you want to use a customized mode for the primary display, check the **Custom** check box and enter the mode number in the box. For a complete list of customized VGA and VESA modes, refer to Table 28, "Supported VGA Video Display Modes" on page 99 and Table 29, "VESA Modes Supported by Video BIOS" on page 100.
Secondary Display Mode	This section allows you to specify a standard or a customized display mode for the secondary display. You can select a standard mode from any of the standard modes listed in the drop-down list. If you want to use a customized mode for the secondary display, check the **Custom** check box and enter the mode number in the box. For a complete list of customized VGA and VESA modes, refer to Table 28, "Supported VGA Video Display Modes" on page 99 and Table 29, "VESA Modes Supported by Video BIOS" on page 100.
5F Functions	These settings allow you to enable or disable the five System BIOS 15h interrupt hooks. (Please see "Intel® 5F Extended Interface Functions" on page 205 for more information on 5F functions.) All five functions are enabled by default.

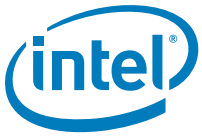


Table 16. Video BIOS Settings Options (Sheet 2 of 2)

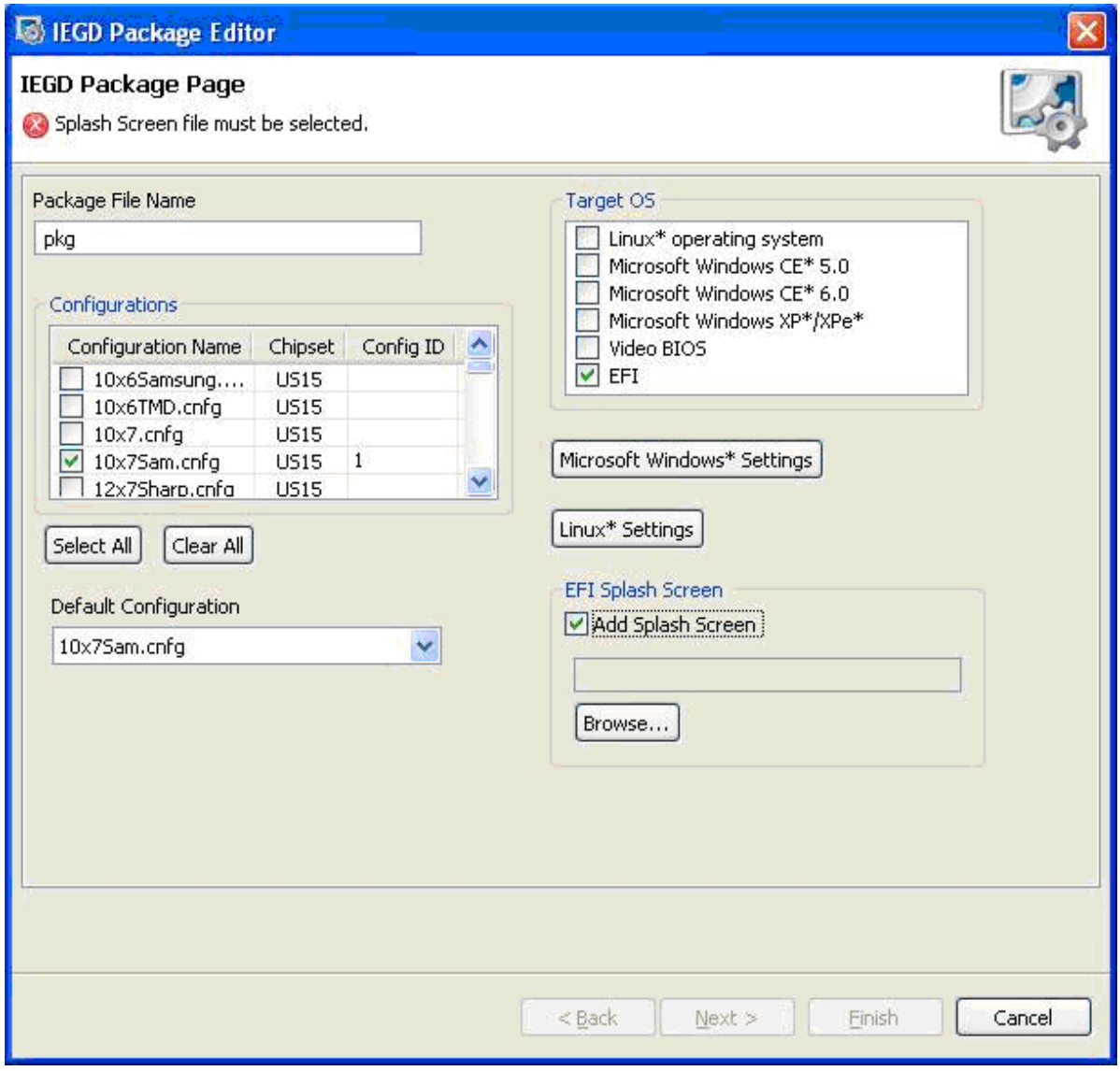
Video BIOS Settings	Description
Common to Port	The Common to Port section lets you match port devices with common System BIOS ports. This allows the Video BIOS to retrieve information about the port from the System BIOS. It allows you to associate standard display names used in most system BIOSs to specific ports that are recognized by IEGD (for example, LVDS, sDVO). The VBIOS makes this association when the VBIOS calls the System BIOS Intel® 5F interrupt functions. This setting consists of six numbers, where each number is associated with one of the System BIOS displays: 1 : CRT - Standard analog CRT 2 : TV1 - TV Output 1 3 : EFP1 - DVI Flat Panel 1 4 : LFP - Local Flat Panel (Internal LVDS display) 5 : TV2 - TV Output 2 6 : EFP2 - DVI Flat Panel 2 The values above are an example of the typical displays and corresponding order used by a system BIOS. However, this may vary depending on how your system BIOS has implemented the displays and the Intel 5F interrupt functions. The value in each position in the setting should be the associated port device. Using the typical settings above, if you want to associate CRT in the system BIOS with the internal CRT (port 1) and LFP in the system BIOS with internal LVDS (port 4) in the VBIOS, select **CRT** from the VBIOS Port Devices list and click the left arrow button next to the CRT row in the Matches column, and then select LFP from the VBIOS Port Devices list and click the left arrow button next to the LFP row in the Matches column. ***Notes:*** This feature must be compatible with the System BIOS. If the System BIOS does not properly implement the Intel 5F functions, then using the Common to Port feature could cause unpredictable results with the displays. If you are unsure, leave the Matches column blank for all ports to disable this feature. The Display Detect field on the Chipset Configuration page must be set to **Enable** in order for the Common to Port values to be used.
Enable POST messages to display	To enable Power On Self Test (POST) messages to display during the power on sequence, check this box. If left unchecked, the POST messages do not display.
OEM String	Enter a string of up to 100 characters. This string appears on the display when the Video BIOS starts up. The default is a blank string.
OEM Vendor Name	Enter a string of up to 80 characters that identifies the OEM Vendor. This string appears on the display when the Video BIOS starts up. The default is a blank string.
OEM Product Name	Enter a string of up to 80 characters that identifies the OEM Product Revision. This string appears on the display when the Video BIOS starts up. The default is a blank string.
OEM Product Revision	Enter a string of up to 80 characters that identifies the OEM Product Revision. This string appears on the display when the Video BIOS starts up. The default is a blank string.
Number of Seconds to Display	Enter the number of seconds to display the above information. The default is 1.

3.5 Creating a New Package

A package consists of one or more configurations and is used to create an installation that works for multiple operating systems and chipset platforms and displays.

To create a new package, click the **New Package** link at the top of the main CED window. The IEGD Package Page appears.

Figure 16. IEGD Package Editor Page



The table below describes each field on this page.

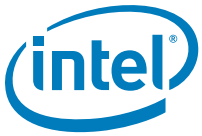


Table 17. IEGD Package Editor Setting Options

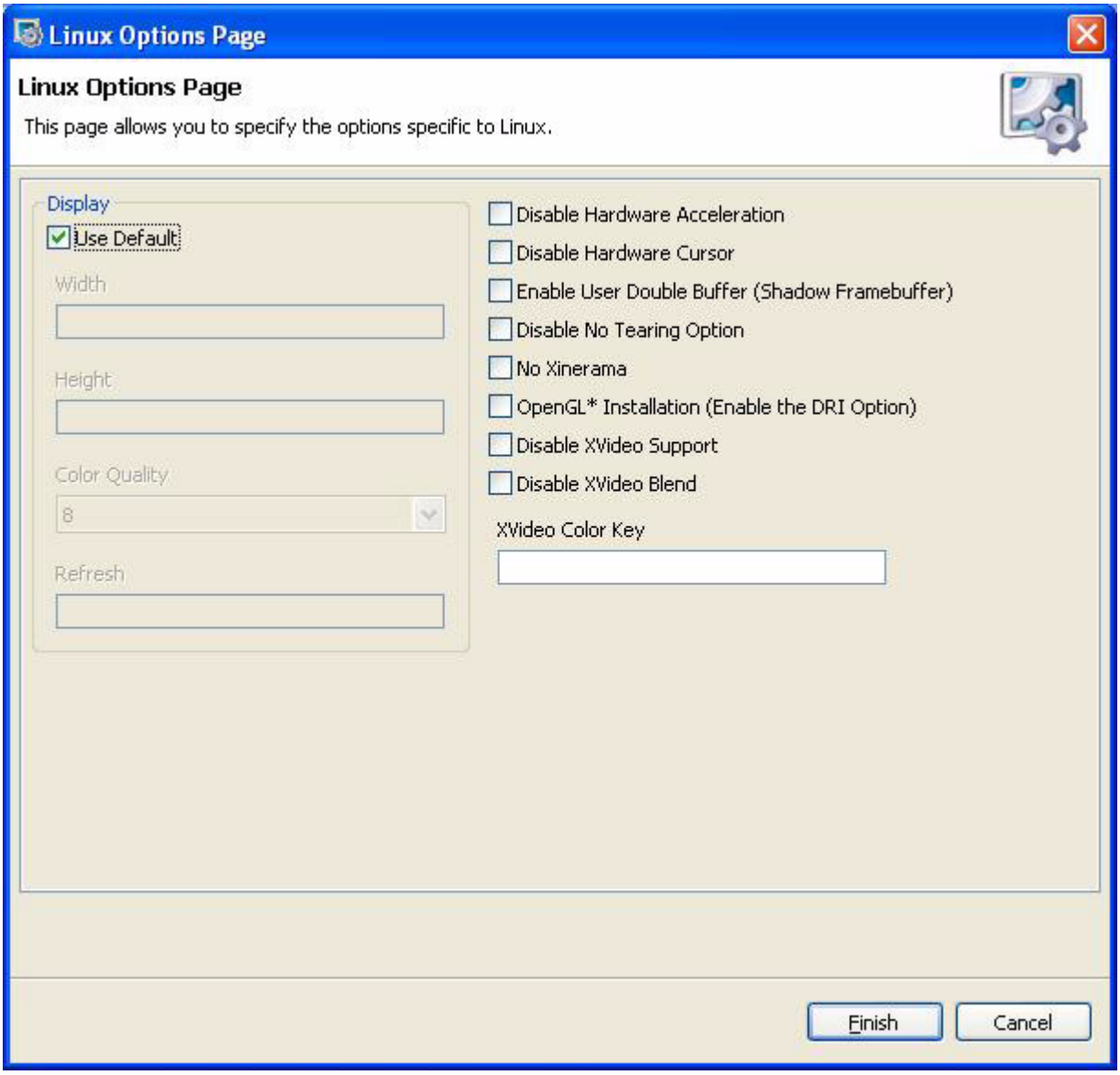
Package Option	Description
Package Name	Enter a name for the package. This is a required field and the name must be between 1 and 50 characters and may contain spaces.
Configurations	This blocks shows the configurations that are available to be packaged. Each package consists of one or more configurations, each of which is associated with a specific chipset. To select a configuration, click the check box next to the configuration name. You can select all available configurations by clicking the `Select All` button located below the Configurations block and clear all configurations by clicking the **Clear All** button. The Configuration Name column shows the name of each configuration and the Chipset column shows the chipset associated with each configuration. In the Config ID column, you must enter a configuration ID for each configuration. The configuration ID must be a number between 1 and 15. By default, the Package Editor automatically assigns the next available configuration ID when you select a configuration. You can change the default configuration ID by clicking in the edit box and entering a different value.
Default Configuration	The Default Configuration list box allows you to select a default configuration from the configurations you selected in the Configurations block. The default is **None**, which means that if you do not select a default, the lowest numbered configuration ID is used as the default.
Linux Settings	If you are creating a package for a Linux OS platform, click the **Linux Settings** button for additional settings that may be required for your configuration. Please see "Entering Linux OS Options" on page 52 for descriptions of these settings.
Microsoft Windows Settings	If you are creating a package for a Microsoft Windows* CE platform, click the **WinCE Settings** button for additional settings that may be required for your configuration. Please see "Changing Windows CE OS Options" on page 34 for descriptions of these settings.
Target OS	This block allows you to select one or more operating systems and Video BIOS for the package. For each target you select, the CED produces a configuration file for the selected OS or Video BIOS platform. Please see the following section for settings on the Target OS: • "Entering Linux OS Options" on page 52 • "Entering Windows OS Options" on page 53 • "Generating a VBIOS Package" on page 55 • "Entering EFI Options" on page 55

If you are not creating a VBIOS package, click **Finish**. When you click **Finish**, the CED creates a package that can be used for generating an installation.

3.5.1 Entering Linux OS Options

The Linux Options Page allows you to enter Linux OS-specific options into the configuration. When you click **Linux Settings** from the IEGD Package Page, the following page appears.

Figure 17. Linux Options Page



The table below describes each of these settings.

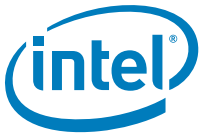


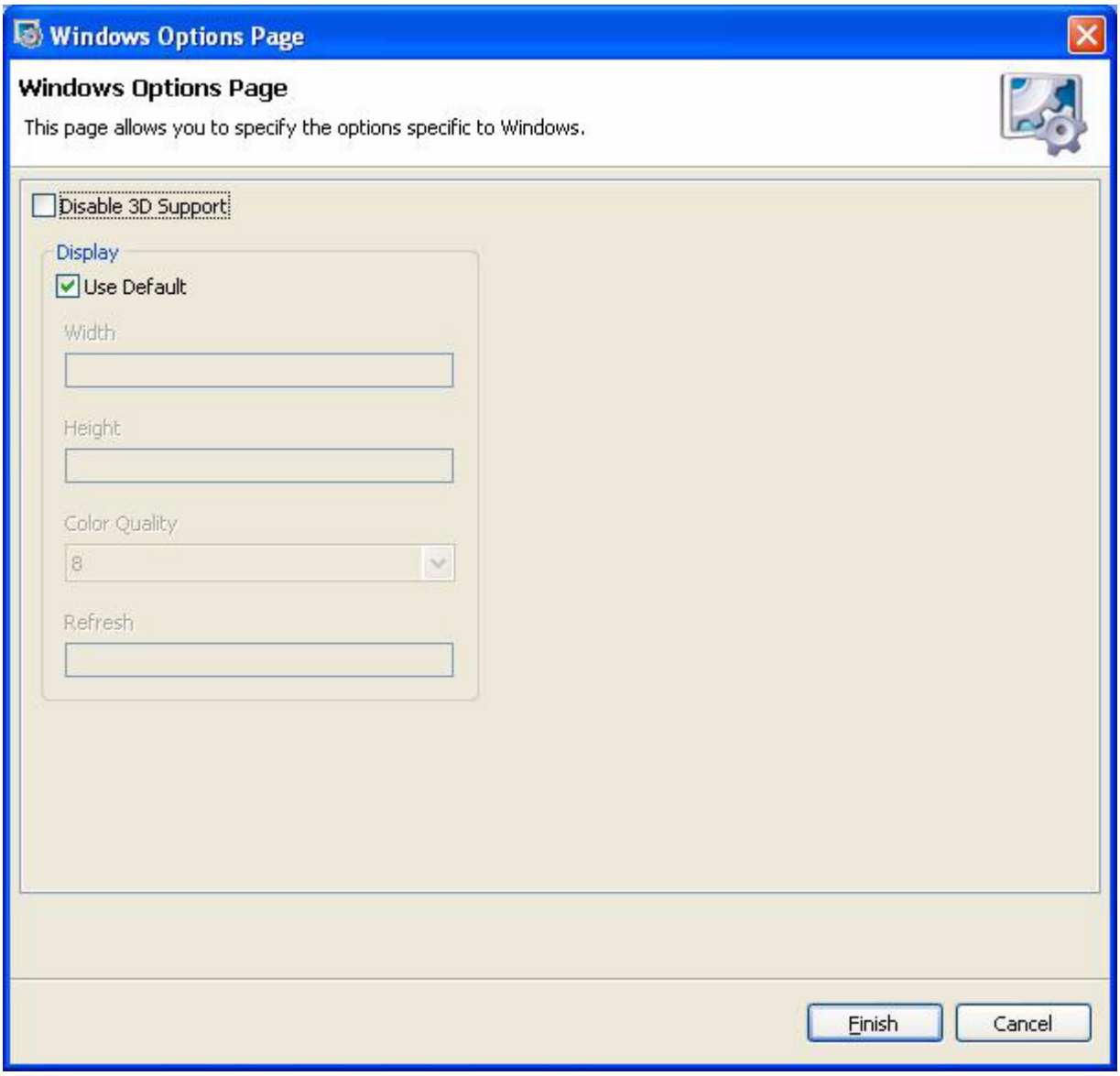
Table 18. Linux OS Settings Options

Linux OS Option	Description
Default Display Modes	The Default Display Modes section allows you select the default resolution, color depth, and refresh rate for the configuration. If you do not select a default display mode, the package uses the default display mode for the operating system it is installed on.
Disable Hardware Acceleration	Disable or enable hardware 2D acceleration. The default is to enable hardware acceleration, so to disable acceleration, click the check box.
Enable Hardware Cursor	Enable the use of the hardware cursor. By default, the hardware cursor is disabled.
Enable Use Double Buffer (Shadow Framebuffer)	Enable double buffering on the framebuffer. By default, is that double buffering is disabled. To enable it, click the check box.
Disable No Tearing Option	Disable No Tearing. By default, the No Tearing is enabled. Disabling this option results in a performance penalty as the driver is forced to synchronize page flips to the vertical blanking signal.
No Xinerama	Xinerama support. Xinerama is an extension to the X Window System which allows applications and window managers to use the two (or more) physical displays as one large virtual display. By default, Xinerama is enabled. To disable it, click the check box.
OpenGL* Installation (Enable the DRI Option)	OpenGL* (Enable the Direct Rendering Infrastructure (DRI) Option). DRI allows the client to directly write to DMA buffers that are used by the graphics hardware. To enable the DRI Option, click the check box. For a list of related application programming interfaces, see "Intel® OpenGL APIs" on page 215.
Disable XVideo Support	Disable XVideo support. In a dual independent head configuration, either the first display or the second display supports XVideo. Both displays can not support XVideo simultaneously. The default is XVideo support is enabled.
Disable XVideo Blend	Disable XVideo support using the 3D blend manager. This provides XVideo support in configurations that cannot be supported with overlay. For example, this is supported on both displays in a dual independent head setup. It is also supported when the display is rotated or flipped. Color key is only supported if ShadowFB is enabled and the VideoKey is defined. The default is XVideoBlend support is enabled.
XVideo Color Key	This sets the color key for XVideo and XVideoBlend. This value is either a 24-bit value or a 16-bit value, depending on the pixel depth of the screen. The color key is always enabled for XVideo, even when it is not defined. The color key is always disabled for XVideoBlend unless both this option is defined and the ShadowFB option is enabled. The default color key for XVideo is 0x0000ff00. For XVideo Blend, the color key is disabled by default.

3.5.2 Entering Windows OS Options

The Windows Options Page allows you to enter Windows OS-specific options into the configuration. When you click **Windows Settings** from the IEGD Package Page, the following page appears.

Figure 18. Windows Options Page



The table below describes each field on this page.

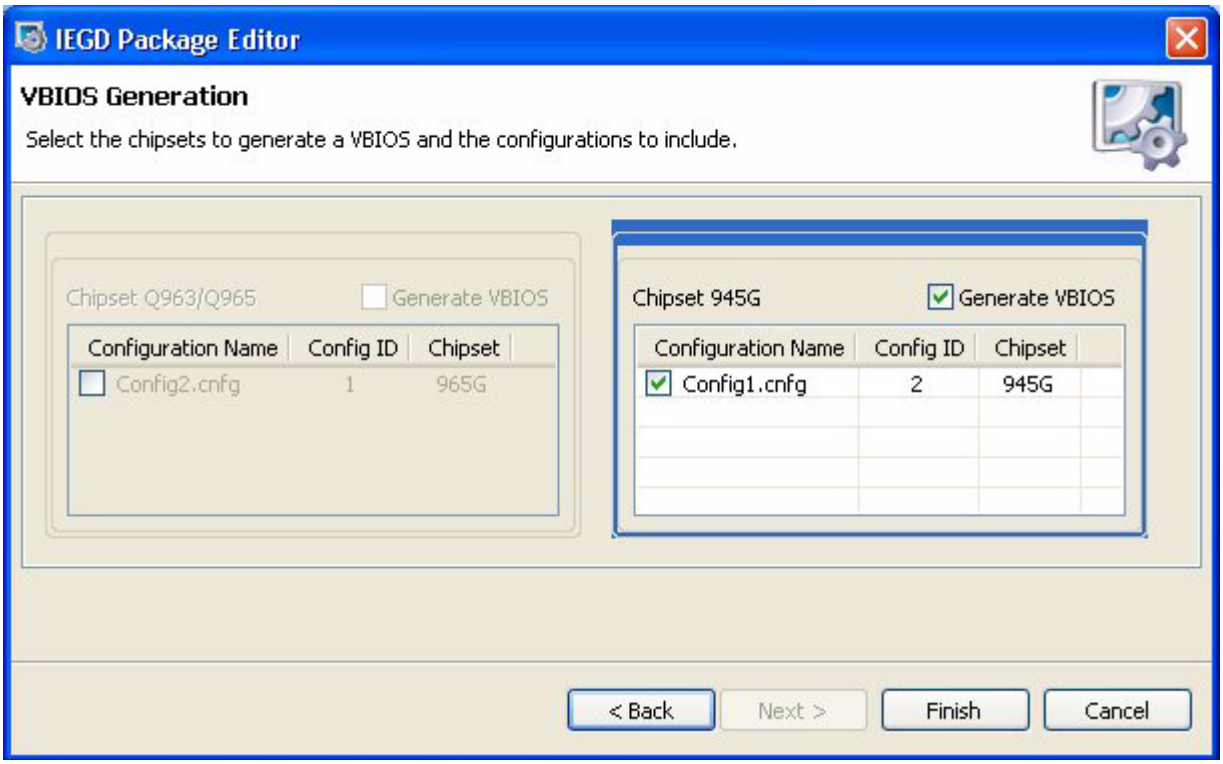
Table 19. Windows OS Setting Options

Windows OS Option	Description
Default Display Modes	The Default Display Modes section allows you select the default resolution, color depth, and refresh rate for the configuration. If you do not select a default display mode, the package uses the default display mode for the operating system it is installed on.
Disable 3D Support	Specify whether to enable D3D. The default is to enable 3D support (not checked).

3.5.3 Generating a VBIOS Package

If you are creating a package for a VBIOS installation, click **Next**. The CED displays the VBIOS Generation page.

Figure 19. VBIOS Generation Page

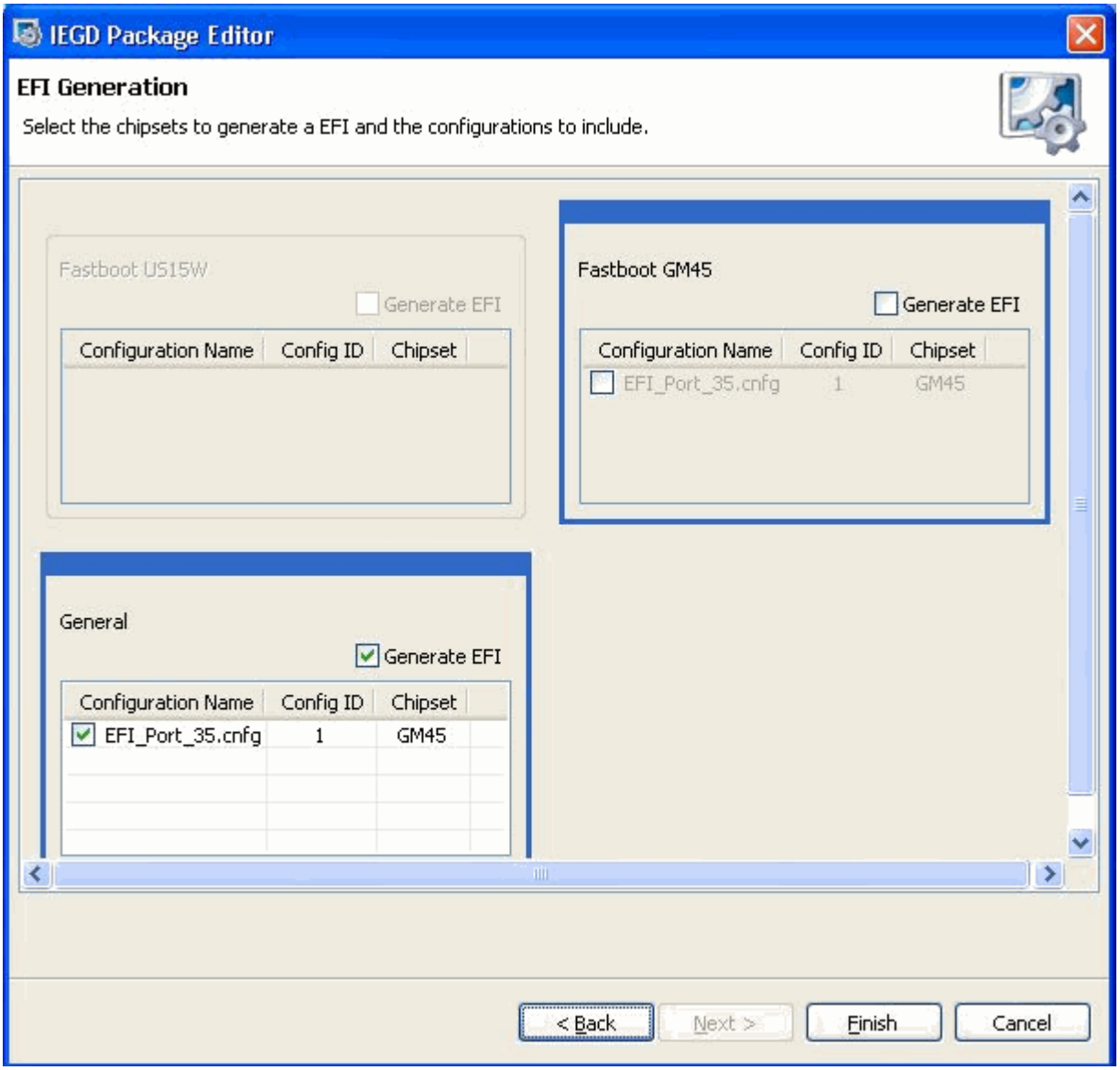


To generate a VBIOS, click the **Generate VBIOS** check box and select the configurations to include. After selecting the chipset and the configurations, click **Finish**. The CED generates a package that includes both the OROM and the TSR for the chipsets and the configurations you selected.

3.5.4 Entering EFI Options

If you are creating a package for a EFI installation, click **Next**. The CED displays the EFI Generation page.

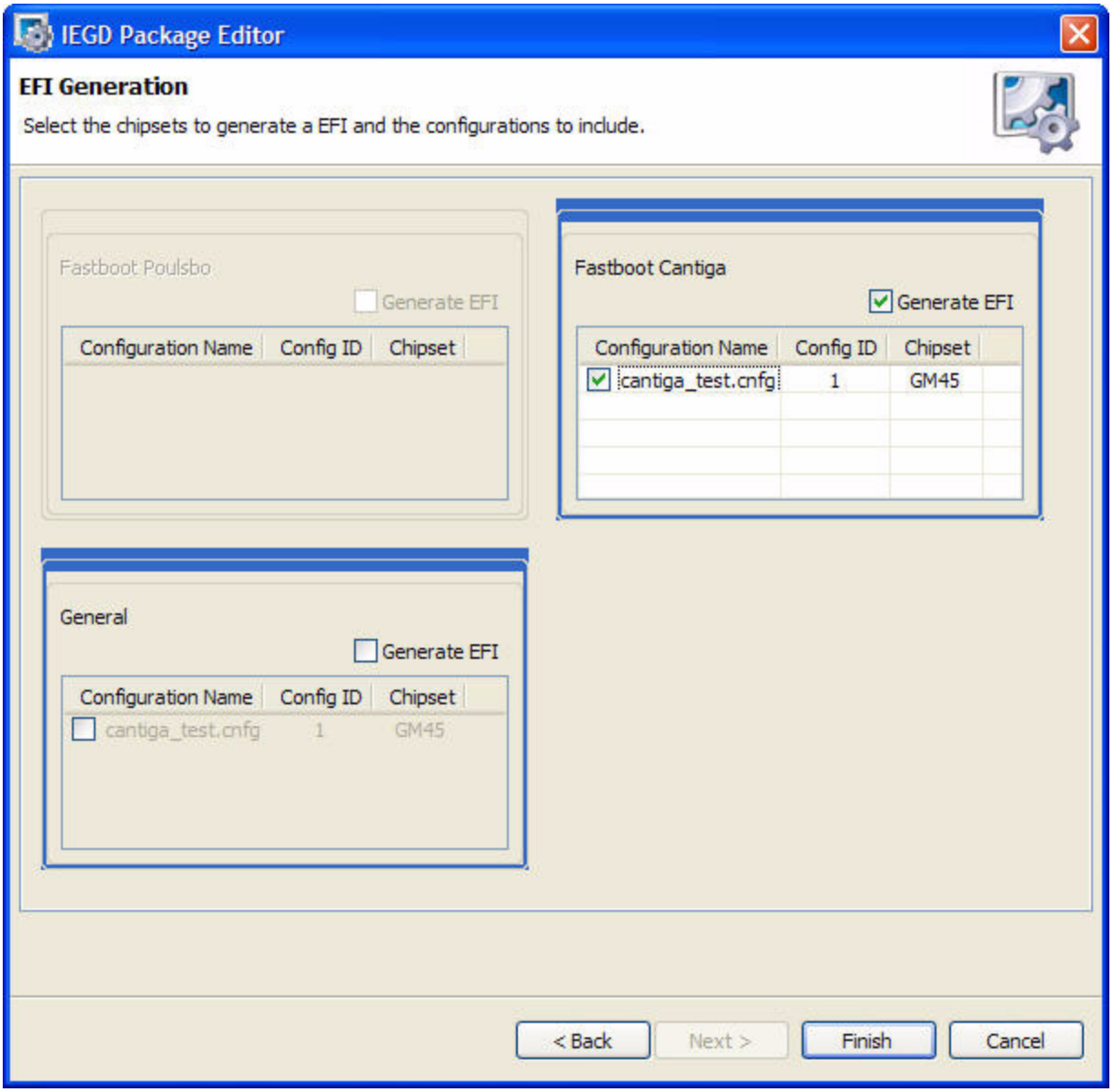
Figure 20. EFI Generation Page



To generate an EFI:

1. In the Fastboot and/or General modes sections, click the **Generate EFI** checkbox
2. Select the chipset and configuration(s) to include.
3. Click **Finish**.
   The CED generates a package that includes the EFI driver for the modes, chipsets and the configurations you selected.

Figure 21. EFI Generation Page



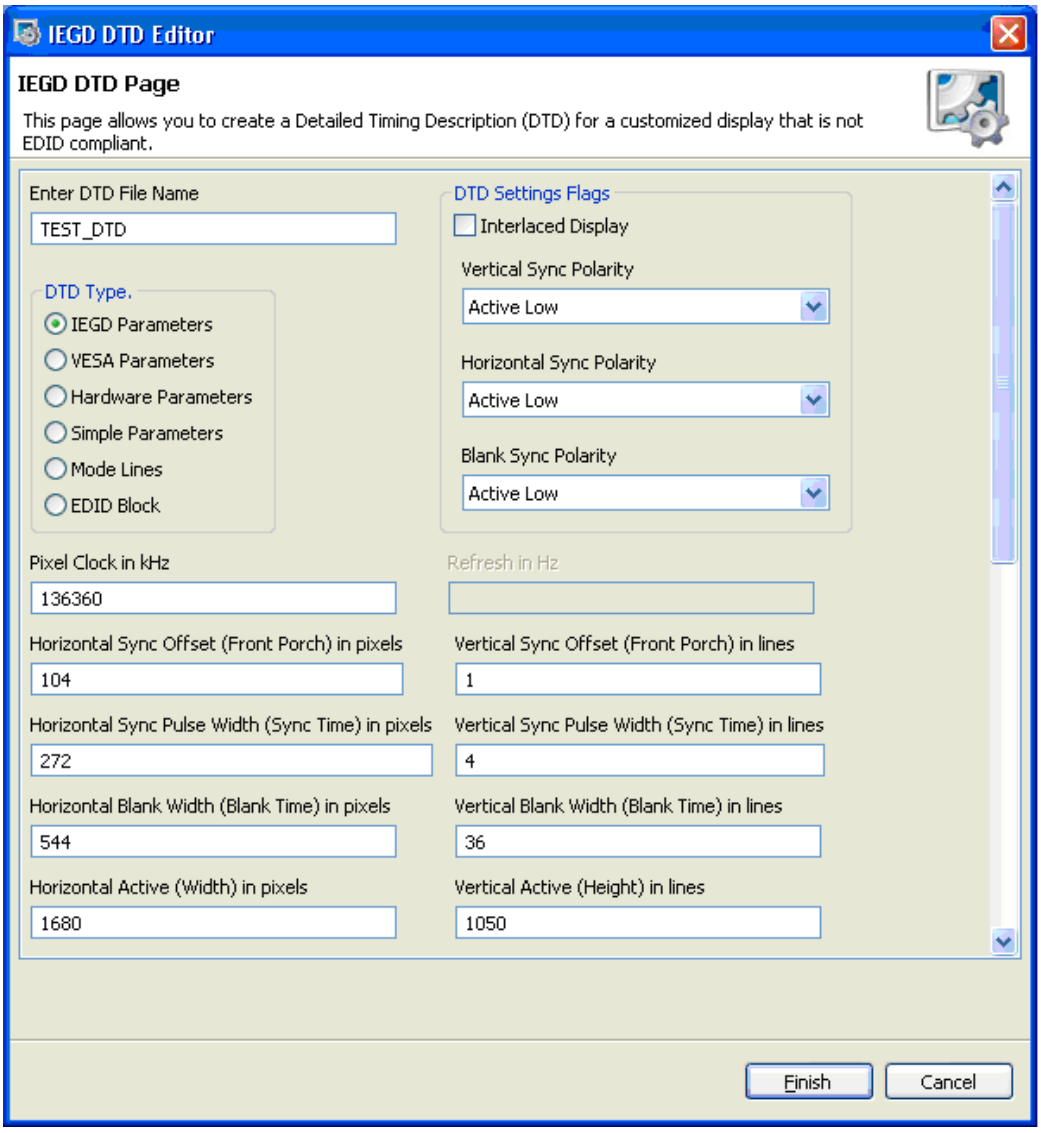
3.6 Creating a New Customized DTD

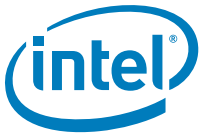
The CED allows you to create Dynamic Timings Definitions (DTD) for EDID-less displays or for displays for which you do not want to use the display's EDID settings.

You can create a new DTD by selecting the New DTD by clicking the **New DTD** link at the top of the main CED window, or you can create DTDs for each configured port when you create a new configuration. Please see "Creating a New Configuration" on page 29 for details on creating customized DTDs for specific configurations.

If you create a DTD from the **Create New DTD** link, the DTD is available for all configurations. When you select **New DTD** from the main CED window, the following IEGD DTD Page appears.

Figure 22. IEGD DTD Page





The table below describes each field on this page.

Table 20. IEGD DTD Setting Options

DTD Parameter	Description
Enter DTD File Name	Enter a name for this customized DTD. This is a required field and the name must be between 1 and 50 characters and may contain spaces and underscores.
DTD Type	Select the DTD Type that most closely aligns with your display parameters. Options are: • IEGD Parameters • VESA Parameters • Hardware Parameters • Simple Parameters • Mode Line • EDID Block
IEGD Parameters	The IEGD Parameters are the same as the current PCF/CED DTD parameters.
VESA Parameters	The VESA Parameters allow the user to create a DTD from a VESA monitor timing standard.
Hardware Parameters	The Hardware Parameters are the parameters that are used by IEGD.
Simple Parameters	The Simple Parameters (CVT Standard) is a process for computing standard timing specifications. The method for developing Reduced Blanking timings is not included.
Mode Line	The Mode Line is a video timing spec used by XFree86. The Xfree86 timing setting for Mode Line is "name" I A B C D E F G H. For example: "640x480@8bpp" 25.175 640 672 728 816 480 489 501 526.
EDID Block	The EDID Block is the detailed timing section (18 bytes) of the basic 128-byte EDID data structure. The detailed timing section starts at 36h of the 128-byte EDID data structure. Enter the EDID block 1 byte at a time. Example: a0 0f 20 00 31 58 1c 20 d2 1a 14 00 f6 b8 00 00 00 18
Horizontal Sync Offset	Specifies the amount of time after a line of the active video ends and the horizontal sync pulse starts (Horizontal Front Porch). Range 0-1023 [10 bits].
Horizontal Sync Pulse Width	Width of the Horizontal Sync Pulse (Sync Time) which synchronizes the display and returns the beam to the left side of the display. Range 0-1023 [10 bits].
Horizontal Sync Start	This parameter specifies the start of the horizontal active time. Range 0-40957.
Horizontal Sync End	This parameter specifies the end of the horizontal active time. Range 0-49148.
Horizontal Blank Width	This parameter indicates the amount of time it takes to move the beam from the right side of the display to the left side of the display (Blank Time). During this time, the beam is shut off, or blanked. Range 0-4095 [12 bits].
Horizontal Blank Start	This parameter specifies the start of one line of the video and margin period. Range 0-32766.
Horizontal Blank End	This parameter specifies the end of one line of the video and margin period. Range 0-65533.
Horizontal Active	Number of pixels displayed on a horizontal line (Width). Range 0-4095 [12 bits].
Pixel Clock	Pixel clock value in KHz. Range 0-0x7fffffff
Refresh	Also known as the Vertical Refresh, the rate the full display updates. Standard refresh rates are 50Hz, 60Hz, 75Hz, and 85Hz.
Vertical Sync Offset	Specifies the amount of time after last active line of video ends and vertical sync pulse starts (Vertical Front Porch). Range 0-4095 [12 bits].

Table 20. IEGD DTD Setting Options

DTD Parameter	Description
Vertical Sync Pulse Width	Specifies the Width of the Vertical Sync Pulse which synchronizes the display on the vertical axis and returns the beam to the top, left side of the display. Range 0-63 [6 bits].
Vertical Sync Start	This parameter specifies the start of the vertical sync. Range 0-4157.
Vertical Sync End	This parameter specifies the start of the vertical sync. Range 0-4220.
Vertical Blank Width	The amount of time for the complete vertical blanking operation to complete. It indicates the time it takes to move the beam from the bottom right to the top, left side of the display (Blank Time). During this time, the beam is shut off, or blanked. Range 0-4095 [12 bits].
Vertical Blank Start	This parameter specifies the start of display vertical blanking including margin period. Range 0-4094.
Vertical Blank End	This parameter specifies the end of vertical blanking. Range 0-8189.
Vertical Active	The number of active lines displayed (Height). Range 0-4095 [12 bits].
DTD Settings Flags	This section allows you to set flags for Interlace, Vertical Sync Polarity, Horizontal Sync Polarity, and Blank Sync Polarity. Each field in this section is described below. Interlaced Display: • Check for Interlaced • Unchecked for Non-interlaced Vertical Sync Polarity: • Active Low (Default) • Active High Horizontal Sync Polarity: • Active Low (Default) • Active High ***Note:*** These flags are IEGD-specific and do not correspond to VESA 3.0 flags.

3.6.1 DTD Example Specifications

The following table shows example product specifications that can be used in the timing fields.

Table 21. Timing Specification Example Values

Item		Symbol	Standard value			Unit
			Min.	Typ.	Max.	
Clock	Frequency	1/ts	29.91	33.231	36.55	MHz
	Period	ts	27.36	30.06	33.43	ns
	Hi-time	tsh	7	–	–	ns
	Low-time	tsl	7	–	–	ns
	DUTY ratio	th/tl	35	50	65	ns

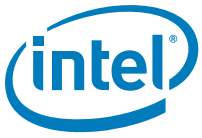


Table 21. Timing Specification Example Values (Continued)

Item		Symbol	Standard value			Unit
			Min.	Typ.	Max.	
Data	Setup time	tds	7	–	–	ns
	Hold time	tdh	4	–	–	ns
H sync.	Period	tlpl, tlpd	24.51	31.75	32.05	us
			880	1056	1088	clk
	Pulse width	tlw	3	128	200	clk
H display	Term	thd	800	800	800	clk
Enable	Setup time	tdrs	7	–	–	ns
	Hold time	tdrh	4	–	–	ns
V sync.	Period	tfpf, tfpd	520	525	680	Line
	Pulse width	tfw	1	2	3	Line
V display	Term	tvd	480	480	480	Line
	Start	tfd	10	33	40	Line
Phase difference	H sync. ~ enable	tdrds	50	216	260	clk
	H sync. ~ clock	tls	7	–	–	ns
	H sync. ~V sync.	tn	7	–	–	ns

3.7 Generating an Installation

After you have created a package, you can generate an installation for the package by following this procedure.

1. Select a package from the Package folder located on the left pane of the CED main window.
2. Click **Generate Installation**. While the installation is building, the CED displays a progress bar. When the installation is complete, the CED places the output in the Installation folder on the left pane of the CED window.

For each OS and VBIOS platform specified in the package, the CED generates a folder in the `...\workspace\installation` folder under the current folder. For example, if you select a package that contains configurations for all supported operating systems and the VBIOS, the CED generates the following folders, where X is the major version and x is the minor version:

```
...\workspace\installation\<package name_installation>\IEGD_X_x_Linux
...\workspace\installation\<package name_installation>\IEGD_X_x_WINDOWS
...\workspace\installation\<package name_installation>\IEGD_X_x_WINCE50
...\workspace\installation\<package name_installation>\IEGD_X_x_WINCE60
...\workspace\installation\<package name_installation>\IEGD_X_x_VBIOS
...\workspace\installation\<package name_installation>\IEGD_X_x_EFI
```

These folders contain all the subfolders required for the installation onto the target systems. To complete the installations on the target systems, refer to the following sections:

- "Installing and Configuring Linux OS Drivers" on page 147
- "Configuring and Installing Microsoft Windows Drivers" on page 103
- "Configuring and Building the IEGD for Microsoft Windows CE Systems" on page 119
- "Entering Linux OS Options" on page 52

3.8 Configuring The System BIOS For Use With The IEGD

Some aspects of configuring the Intel® Embedded Graphics Drivers are common across the Video BIOS (VBIOS), EFI, and the drivers for the supported operating systems. The following topics provide an overview for configuring both the VBIOS and the Intel Embedded Graphics Drivers and describe in detail the common components and tools. These topics also describe how to configure the system BIOS for the supported systems.

- "System BIOS Settings"
- "VBIOS and Driver Configuration"
- "Display Detection and Initialization"
- "Advanced EDID Configuration"
- "Using an External PCI Graphics Adaptor as the Primary Device"
- "Enhanced Clone Mode Support"

3.9 System BIOS Settings

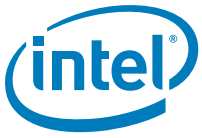
Before installing the Intel Embedded Graphics Drivers, you must first configure the system BIOS. The following sections describe the required settings. These descriptions are based on AMIBIOS8* from American Megatrends, Inc., which is the recommended system BIOS to use with the Intel Embedded Graphics Drivers. Settings may vary if a different system BIOS is used.

3.9.1 GMCH PCI Device Enabling

The PCI Device Enabling feature on the Graphics and Memory Controller Hub (GMCH) should be set as specified in the table below.

Table 22. GMCH Device 2, Function 1 BIOS Setting

OS	Chipset
	Intel® Q35, Intel® GME965, Intel® Q965, Intel® 945GM, Intel® 945G, Intel® 915GME, Intel® 915GV, Intel® 910GMLE
Microsoft Windows XP* and Microsoft Windows XPe*	Disabled
Microsoft Windows CE*	Disabled
Linux	Disabled



3.9.2 Graphics Mode Select (GMS)

The System BIOS typically allows a portion of physical memory to be dedicated to firmware and graphics driver use. This dedicated memory is known as stolen memory since it is not available to the operating system. The size of this memory is selectable and chipset-specific. Stolen memory is typically used by the firmware and graphics driver to locate the framebuffer, but can also be used as scratch and surface memory. Because it is programmatically set aside during boot by the System BIOS, access to it is direct and does not require OS memory allocation services. Firmware is fully responsible for stolen memory management.

Graphics Mode Select (GMS), or stolen memory, can be set to any of the sizes listed in the table below. Smaller sizes limit the framebuffer size during firmware boot. Larger sizes marginally increase surface allocation performance for the graphics driver.

Table 23. GMS Settings

Chipset	GMS Settings
Intel® Q35, Q965/ GME965	0, 1 Mbyte, 4 Mbytes, 8 Mbytes, 16 Mbytes, 32 Mbytes, 48 Mbytes, 64 Mbytes
Intel® 945G/ 945GME/945GSE	0, 1 Mbyte, 8 Mbytes
Intel® 915GV/ 915GME/910GMLE	0, 1 Mbyte, 8 Mbytes

3.9.3 AGP (Accelerated Graphics Port) Aperture Size

The AGP Aperture size controls the total amount of graphics memory that can be mapped in the AGP Aperture. This value can be set from 64 Mbytes up to 256 Mbytes, depending on the chipset. Refer to specific chipset details for information on the valid range.

3.10 VBIOS and Driver Configuration

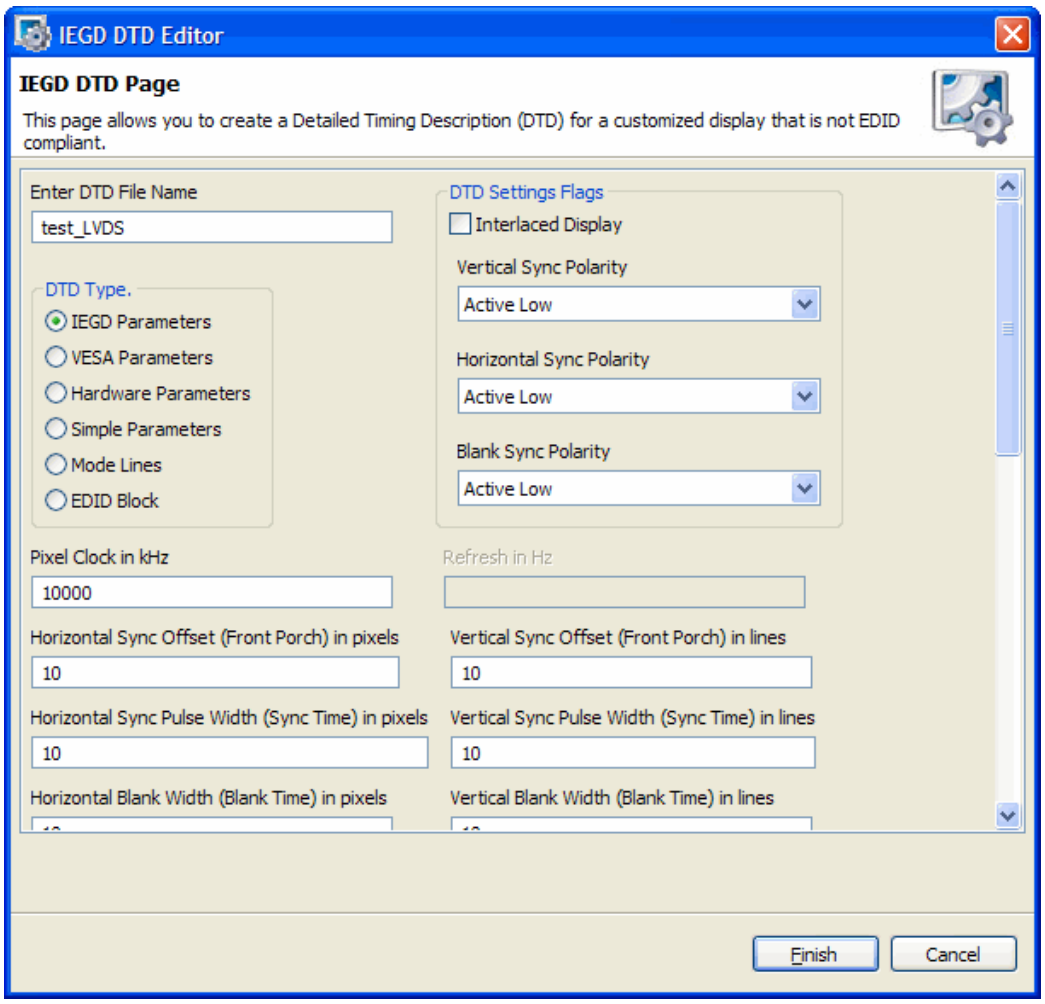
The Intel Embedded Graphics Suite allows user configuration of both the VBIOS and graphics driver as well as programming of Detailed Timing Descriptors (DTDs) for EDID-less panels for both the VBIOS and graphics driver. This is accomplished using CED, which offers several ways to input DTDs, each associated with a potential target panel and display mode for the system. CED generates DTD and configuration settings used by the IEGD VBIOS, Linux, and/or Windows drivers.

The following example is for a 945GME system setup with just an internal LVDS and sample timing parameters for illustration purposes only. You can use this example to set up DTD timings that are specific to your non-standard panels and then activate the panels using a custom mode.

To create a custom DTD setting:

1. From the CED main screen, select **New DTD**.
2. Enter a name for the DTD in the text box provided, for example, *test_LVDS*.
3. Using the data sheet from the panel being used, enter the DTD timings in the appropriate fields.
   The screen will be similar to the example below.

Figure 23. IEGD DTD Page



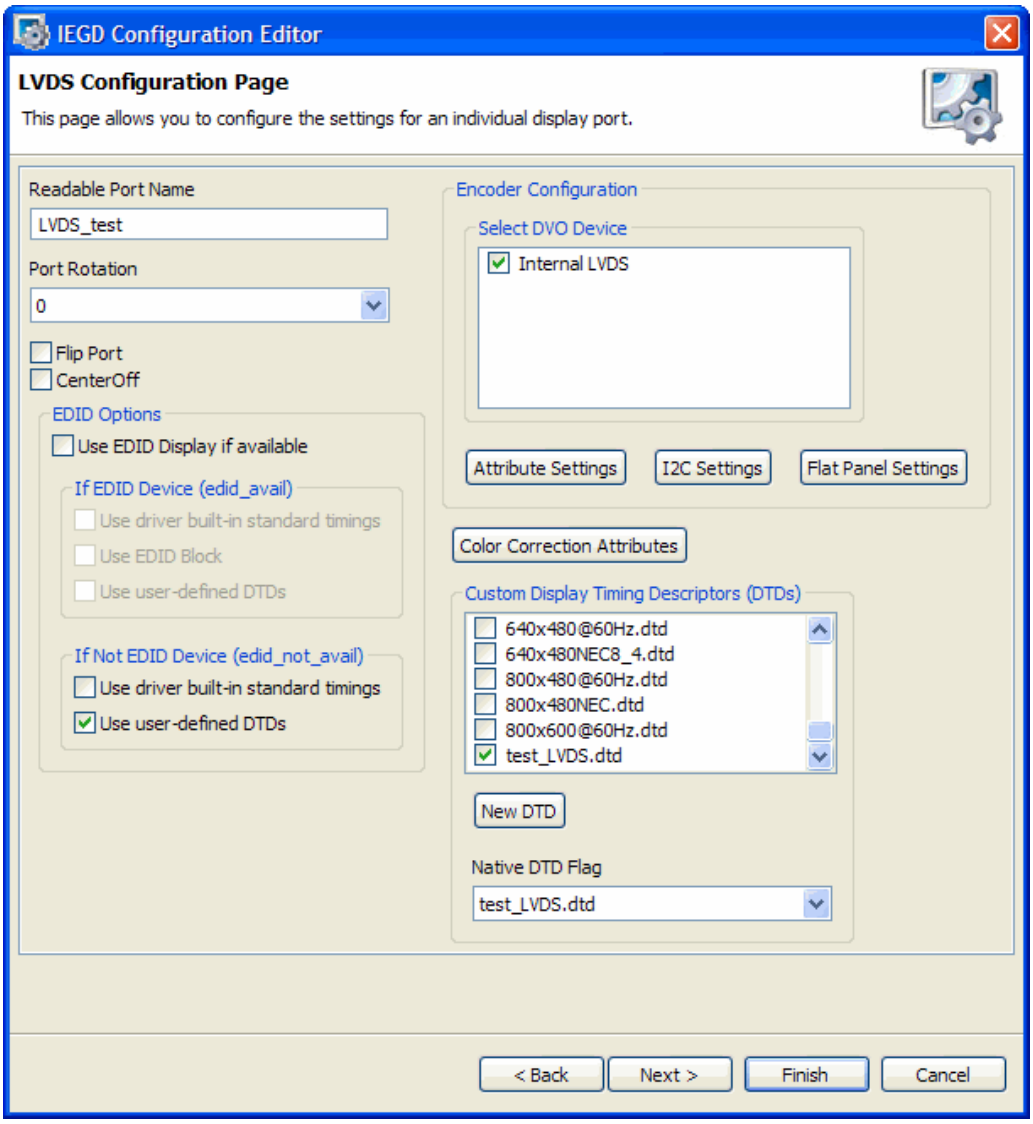
4. Click **Finish**.
   The custom DTD is complete.

To create a configuration and configure the LVDS options:

1. From the CED main screen, select **New Configuration**.
2. Enter a name for the configuration in the text box provided, for example, *LVDS_test*.
3. Select the platform chipset. This example uses the 945GME chipset.
4. In the list of available ports, select **LVDS** and then click **Next**.
5. On the LVDS Configuration Page, clear the checkboxes for **Use EDID Display if available** and **Use driver built-in standard timings**.
6. Select the checkbox for **Use user-defined DTDs**.
7. In the Encoder Configuration section, select **Internal LVDS**.
8. In the Custom Display Timing Descriptors (DTDs) list, select the DTD you created in the previous section, for example, *test_LVDS*.

The screen will be similar to the example below.

Figure 24. LVDS Configuration Page



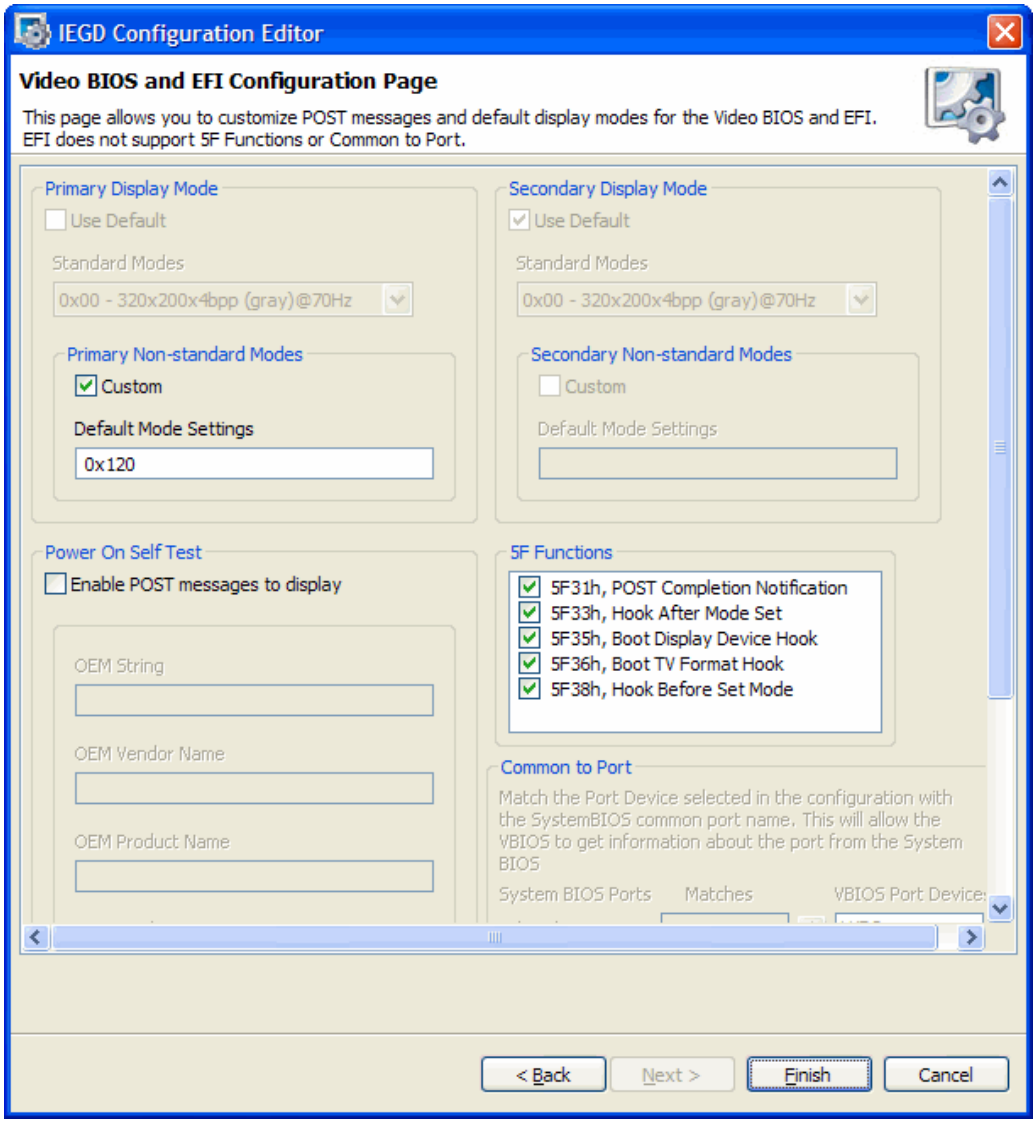
9. Click **Next**.
10. (Optional) Configure Fastboot options as described in "Configuring Fastboot" on page 43.
11. Click **Next**.

To set the custom mode:

1. From the IEGD Configuration Editor screen, in the Primary Display Mode section, clear the **Use Default** checkbox.
2. In the Primary Non-standard Modes section, select the checkbox for **Custom**.
3. In the Primary Non-standard Modes section, enter 0x120 in the Default Mode Settings text box. (See a description of the custom modes.)

The screen will be similar to the example below.

Figure 25. IEGD Configuration Editor Page



Custom Modes

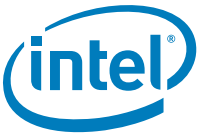
The custom modes begin with 0x120 (0x121 and 0x122 are the same modes in different pixel formats). If there was a second custom mode entered it would begin with 0x123 to 0x125.

From the above DTD 200x200 example, this is what the custom modes represent:

0x120 200x200@8bpp

0x121 200x200@16bpp

0x122 200x200@32bpp



And if the second custom mode was a 400x400 panel, its custom modes would be:

0x123 400x400@8bpp

0x124 400x400@16bpp

0x125 400x400@31bpp

3.11 Configuration Options

The table below describes available IEGD settings. The gray rows are block headings and the non-gray rows that follow each heading are settings within the block. Some of these block headings are contained within prior block headings.

Table 24. Parameter Configuration Format (Sheet 1 of 7)

Name	Range/Value	Description
configid	Integer (1-15)	Optional keyword used to specify which configuration is used. The config ID specified here must match one of the configuration IDs defined with CED. If this keyword is omitted, all configurations specified in the config file are used. Note that this keyword is not required for Linux OS and VBIOS configurations.
config	Integer (1-15)	More than one configuration is valid.
comment		A quoted string used to identify the origin of the .bin or .inf file.
name		A quoted string used to identify the configuration name. Name is a required field for VBIOS configuration.
general		Settings that are generic to the configuration.
displayconfig	1 – Single 2 – Clone 4 – Twin 8 – Extended Default: 8	Used to configure initial state of attached displays. 1 – Single. A single display. 2 – Clone. Primary and secondary displays enabled and configured with separate timing pipes. This allows different timings to be applied to each display. Resolutions can be different on both displays. 4 – Twin. Primary and secondary displays are enabled, but with only a single pipe. Both displays share the same resolutions and timings. 8 – Extended. Configures separate pipes to allow primary and secondary displays to have different resolutions and display different content. Upon first boot after the driver installation, this option will enable only the primary display, as the extended modes must be enabled in the operating system (i.e., **Extended Desktop** in the **Display Properties** sheet within Microsoft Windows).
displaydetect	0 - Disable 1 - Enable	Enable or disable Display Detection. Note that this parameter must be Enabled in order to use COMMON_TO_PORT values. Default is 0. Please see Section 3.12, "Display Detection and Initialization" on page 74 for detailed information on this parameter.

Table 24. Parameter Configuration Format (Sheet 2 of 7)

Name	Range/Value	Description
PortOrder	PortOrder must be specified as a quoted string containing five digits. The valid values are: 1 - Integrated TV Encoder (mobile chipsets only) 2 - sDVO B port 3 - sDVO C port 4 - Integrated LVDS port (mobile chipsets only) 5 - Analog CRT port 6 - Internal HDMI Default: 0 for all keys	Search order for detecting attached displays for the Display Detection feature. When Display Detection is enabled, the PortOrder determines which display is primary and which display is secondary. The port search order can be specified to ensure the port device (sDVO device) is found, based on the system integrator's routing choices. Default ordering is chosen by specifying zeros in the PortOrder keys. Default ordering is chipset specific; see Table 48, "Default Search Order" on page 203. Please see Section 3.12, "Display Detection and Initialization" on page 74 for more information on using `PortOrder` in combination with the Display Detect feature.
clonewidth cloneheight	Typical sizes: clonewidth – 800, cloneheight - 600 clonewidth – 1024, cloneheight - 768 clonewidth – 1280, cloneheight - 768 clonewidth – 1400, cloneheight – 1050	Width and height for a cloned display.
clonerefresh = 60	Typical refresh rates (expressed in Hz): 60 Hz, 75 Hz, 85 Hz	Refresh rate for a cloned display.
OverlayOff	0 - Overlay on (default) 1 - Overlay off	This parameter allows you disable Overlay support, which is enabled by default. **Note:** This parameter is only for Microsoft Windows* and Microsoft Windows CE. The Linux* OS configuration for the XF86Config provides a standard option that performs the same function.
No_DFB	0 - Off (Default) 1 - On	This parameter enables the IEGD to pass the DIB call back to the OS. This is required in certain circumstances to improve performance.
vbios		This block contains settings for the new Video BIOS. Note that you only need to specify the parameters you are actually using. You do not need to specify all the parameters in this block. If you omit any parameters, the vbios uses the default values.
COMMON_TO_PORT	6 digit value	Maps the ports from the system BIOS to a port number used by the graphics hardware. Please see Section 4.3.2, "Configuring the Video BIOS" on page 93 for more information on this parameter. Note that the `displaydetect` parameter must be set to Enabled in order for the COMMON_TO_PORT values to be used. The default is all zeroes: 000000

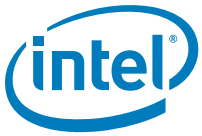


Table 24. Parameter Configuration Format (Sheet 3 of 7)

Name	Range/Value	Description
post_display_msg	0 - disable greater than 0 - enable and display POST message for the specified number of seconds	Enables or disables the POST (Power On Self Test) message. When you specify a value greater than 0, the message is displayed for the specified number of seconds. For example: post_display_msg = 5 This enables the POST message and displays it for approximately 5 seconds. The maximum value that can be entered here is 65535. The default is 1, enable and display the POST message for approximately 1 second.
oem_string	double-quoted string	This string appears on the display when the post_display_msg is enabled and the VBIOS starts up. The maximum string length is 100 characters. The default is " " (two double quotes with a single space in between).
oem_vendor	double-quoted string	This string appears on the display when the post_display_msg is enabled and the VBIOS starts up. The maximum string length is 80 characters. The default is " " (two double quotes with a single space in between).
oem_product_name	double-quoted string	This string appears on the display when the post_display_msg is enabled and the VBIOS starts up. The maximum string length is 80 characters. The default is " " (two double quotes with a single space in between).
oem_product_rev	double-quoted string	This string appears on the display when the post_display_msg is enabled and the VBIOS starts up. The maximum string length is 80 characters. The default is " " (two double quotes with a single space in between).

Table 24. Parameter Configuration Format (Sheet 4 of 7)

Name	Range/Value	Description
int15	5 digits	This parameter allows you to enable or disable the five System BIOS 15h interrupt hooks. The value must be 5 digits in length. Each digit is associated with one of the five System BIOS interrupt 15h hooks as shown below (left to right) 1 - 5F31h, POST Completion Notification Hook 2 - 5F33h, Hook After Mode Set 3 - 5F35h, Boot Display Device Hook 4 - 5F36h, Boot TV Format Hook 5 - 5F38h, Hook Before Set Mode (Please see Appendix C for more information on 5F functions.) The value of each digit must be a 0 or a 1 as follows: 0 - disable a System BIOS 15h hook 1 - enable a System BIOS 15h hook For example, `int15 = 11001` Enables 5F31h, 5F33h, and 5F38h hooks only. The 5F35h and 5F36h hooks are disabled. The default is 11111, enable all five hooks.
port	1 - Integrated TV Encoder (mobile chipsets only) 2 -sDVO B port 3 - sDVO C port 4 - Integrated LVDS port (mobile chipsets only) 5 - Analog CRT port	Used to define port specific settings.
name		A quoted string used to identify the port.
general		Settings that are generic to the port.
rotation	Windows* OS Range: 0x0 or 0 – 0 degrees 0x5A or 90 – 90 degrees 0xB4 or 180 – 180 degrees 0x10E or 270 – 270 degrees Linux OS Range: 0 – 0 degrees 90 – 90 degrees 180 – 180 degrees 270 – 270 degrees Default: 0	Rotation of the display.
flip	Windows OS: 0x0 or 0 – turn off horizontal flip 0x1 or 1 – turn on horizontal flip Default: 0 Linux OS Boolean: on - horizontal flip off - no horizontal flip Default: off	Flip of the display.

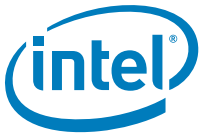


Table 24. Parameter Configuration Format (Sheet 5 of 7)

Name	Range/Value	Description
centeroff	Default: 0 – disabled, allow centering and add compatibility modes 1 – enabled, no centering, no added compatibility modes	When this option is enabled it DISABLES centering. Also, depending on the combination of "edid" + "user-dtd" + connected hardware, IEGD will add missing compatibility modes (6x4, 8x6, 10x7& 12x10) via centering. Use this option to disable this feature.
edid	0 – Do not read EDID from panel/CRT 1 – Attempt to extract EDID timing data from panel/CRT	If VBIOS/Driver reads EDID from panel/CRT.
edid_avail edid_not_avail	Range [16 bits] Valid values (specified in hex): bit 0 ---------- 0 - Do not use driver built-in standard timings 1 - Use driver built-in standard timings bit 1 (not applicable to `edid_not_avail`) ---------- 0 - Do not use EDID block 1 - Use EDID block and filter modes bit 2 ---------- 0 - Do not use user-defined DTDs 1 - Use user-defined DTDs bit3 - bit15 ---------- Reserved for future use.	These two parameters are used to control the available timings for any display. `edid_avail` is used when EDID values are read from the display. If an attempt to read EDID from the display fails or the `edid` parameter is set to 0, then the driver uses the `edid_not_avail` flags. The value for both parameters must be specified as a hex value. Defaults: edid_avail: 3 (hex). Bit 0 = 1, Bit 1 = 1, Bit 2 =0 (Use driver built-in standard timings and EDID block and filter modes.) edid_not_avail: 1 (hex). Bit 0 = 1, Bit 1 = 0, Bit 2 = 0. (Use driver-built-in standard timings.) Please see Section 3.13, "Advanced EDID Configuration" on page 76 for detailed information.
multidvo	0 – Do not attempt to detect a second decoder of same type 1 – After detect of a decoder, continue to attempt detection of same type of decoder until fail	If VBIOS/Driver detects a second decoder of same type. This value is hard-coded to "1" for Windows configuration and will ignore this setting.
dvo		sDVO device information.
i2cpin	<0-6>	The GPIO pin pair used on the I^2C bus to read and write to sDVO device registers.
ddcpin	<0-6>	The GPIO pin pair used as DDC bus to read panel EDID data.
i2cdab	<0x00-0xff>	I^2C device address for reading and writing device registers. The device address should be in 8-bit format with the 7-bit slave address assigned to its bits 7:1 and bit 0 set to 0.
ddcdab	<0x00-0xff>	I^2C device address for reading EDID data from display through the DDC bus.
i2cspeed	[10-400]. Units in KHz	Speed of I^2C bus for sDVO device.
ddcspeed	[10-400]. Units in KHz	Speed of I^2C bus for EDID device.
fpinfo		Panel-specific information.

Table 24. Parameter Configuration Format (Sheet 6 of 7)

Name	Range/Value	Description
bkltmethod	Range [0-3] 0 – no backlight 1 – Port Driver 2 – GMCH 3 – ICH ***Note:*** The only supported parameter for internal LVDS is 1 – Port Driver	Instructs which backlight method is required for the panel attached to the given port. If zero is supplied, or the key is not present, then no backlight control is provided.
bkltt1	Range [0 -0xfff]. Units of 0.1ms => the limit specified in your hardware specifications. For example, the maximum for the CH7307 is 409 ms.	(T1) Time delay between VDD active, and sDVO clock/data active. Zero indicates no delay required.
bkltt2		(T2) Time delay between sDVO clock/data active and Backlight enable.
bkltt3		(T3) Time delay between Backlight disable and sDVO clock/data inactive.
bkltt4		(T4) Time delay between sDVO clock/data inactive and VDD inactive.
bkltt5		(T5) Minimum delay between VDD inactive, and active.
gpiopinvee	Valid ICH GPIO pin, 0 indexed For example: gpiopinvdd = 3 gpiopinvee = 5 gpiopinenable = 1	GPIO connection for panel power.
gpiopinvdd		GPIO connection for backlight power on/off sequencing signal.
gpiopinbklt		GPIO to enable backlight signal.
UseGMCHClockPin	1 - Flat panel is connected to the clock pin 0 - Flat panel is not connected to the clock pin	This entry is needed when GMCH is selected as backlight control method.
UseGMCHDataPin	1 - Flat panel is connected to the data pin 0 - Flat panel is not connected to the data pin	This entry is needed when GMCH is selected as backlight control method.
dtd		Denotes a Detailed Timing Descriptor (DTD) block. Settings in this section, except for the `flags` parameter, correspond to the Detailed Timing Block described in the VESA standard "Extended Display Identification Data Standard", Version 3, November 13, 1997.
p_clock	Range [0-0x7fffffff]	Pixel clock value in KHz.
h_active	Range 0-4096 [12 bits]	Horizontal Active.
v_active	Range 0-4096 [12 bits]	Vertical Active.
h_sync	Range 0-1024 [10 bits]	Horizontal Sync Offset.
v_sync	Range 0-64 [6 bits]	Vertical Sync Offset.
h_syncp	Range 0-1024 [10 bits]	Horizontal Sync Pulse Offset.
v_syncp	Range 0-64 [6 bits]	Vertical Sync Pulse Width.
h_blank	Range 0-4096 [12 bits]	Horizontal Blanking.
v_blank	Range 0-4096 [12 bits]	Vertical Blanking.
h_border	Range 0-256 [8 bits]	Horizontal Border. Currently not supported.
v_border	Range 0-256 [8 bits]	Vertical Border. Currently not supported.
h_size	Range 0-4096 [12 bits]	Horizontal Size. Currently not supported.

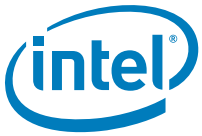


Table 24. Parameter Configuration Format (Sheet 7 of 7)

Name	Range/Value	Description
v_size	Range0-4096 [12 bits]	Vertical size. Currently not supported.
flags	Range [32 bits] Valid values: bit 31 --- 0 - Non-interlaced 1 - Interlaced bit 27 --- 0 - vertical sync polarity active low 1 - vertical sync polarity active high bit 26 --- 0 - horizontal sync polarity active low 1 - horizontal sync polarity active high bit 25 --- 0 - blank sync polarity active high 1 - blank sync polarity active low bit 17 --- 0 - Normal DTD 1 - Panel/display Native DTD All other bits --- Do not use any other bits; all other bits must be set to 0.	Interlace, Horizontal polarity, Vertical polarity, Sync Configuration, etc. Note that these flags are IEGD specific and do not correspond to VESA 3.0 flags. For example, to set Interlaced with Horizontal Sync Polarity high (bits 31 and 26), then the flags value = 0x84000000. (Binary = 10000100000000000000000000000000)
attr	0-0xFFFF	Attribute values that are specific to the sDVO device for the port. See Appendix B, "Port Driver Attributes" for specific attribute IDs and associated values.
id <Attribute ID>	0 - 2^{32}	id = <value>. Both the Attribute ID and its value should be specified in decimal. For example, to set brightness to 50, you specify id 0 = 50 See Appendix B, "Port Driver Attributes".

3.12 Display Detection and Initialization

The Display Detection and Initialization feature, when enabled, automatically detects displays and allocates ports without the need to change any configuration files. This feature is off by default and can be enabled either through CED or by directly editing the `iegd.inf` file for Microsoft Windows or the `XF86Config/xorg.conf` file for the Linux OS.

To enable the feature via CED, select the `DisplayDetect` option on the CED Chipset Configuration page. Please see Section 3.4, "Creating a New Configuration" on page 29.

Alternatively, you can enable the feature in Microsoft Windows by entering the following line in the **`[iegd_SoftwareDeviceSettings]`** section of the `iegd.inf` file:

HKR, All\<ConfigID>\General, DisplayDetect, %REG_DWORD%, 1

where <ConfigID> is the configuration ID (without the angle brackets).

To enable the feature in the Linux OS, enter the following line Option setting in the `XF86.conf or xorg.conf` file:

Option "Config/<ConfigID>/General/DisplayDetect" "1"

When the display detection feature is enabled, ports are allocated only when the display satisfies the following conditions:

1. The port is not in use (that is, it is not already allocated).
2. The display is detected by the port driver.

The first port that passes these conditions is allocated. If condition 2 fails for all ports, then the first port in the `PortOrder` setting that passes condition 1 is allocated. If the port is not detectable (specifically the internal LVDS or external LVDS using CH7308), the driver assumes the display is connected. Condition number 2. always passes for these displays.

When this feature is disabled, display allocation is done based on `PortOrder` and no display detection is performed.

See also:

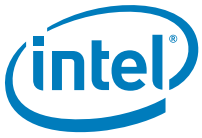
- "Display Detect Operation"
- "Detectable Displays"

3.12.1 Display Detect Operation

This section describes the logic of the Display Detection feature and provides several examples.

1. If Display Detect is disabled, the driver uses the first two ports identified in the PortOrder.
2. If Display Detect is enabled and you are using the 9.0 version of the VBIOS, the VBIOS performs the display detection. The driver then checks to see if the VBIOS returns the display allocations and if it does, the driver does not re-execute the display detection steps.

   If you are not using the 9.0 Legacy VBIOS, then the driver performs display discovery as described in the following steps.
3. The number of displays to be detected is based on the `DisplayConfig` settings in the configuration. If this is set to **Single**, then only one display is detected. If it is set to any other value, a maximum of two displays will be detected.



4. The IEGD goes through each port in the `PortOrder` settings and attempts to detect a display using the following algorithm:

   a. If a display is detected, it is based on the PortOrder sequence. Display allocation of the port is performed once the display has been detected. For example:

      PortOrder = "53240" (CRT, sDVO, LVDS)

      Displays Connected = CRT

      Primary display allocation: Searches for a display connected according to the PortOrder sequence. The first detected display is a CRT, so the Primary display is CRT.

      Secondary display allocation: Searches for a display connected according to the PortOrder sequence. The first detected and non-allocated display is sDVO, so the Secondary display is sDVO.

   b. If no display is detected on any of the ports, then the DisplayDetect option is turned off and ports are allocated in the order defined by PortOrder. For example:

      PortOrder = "32000"

      Displays Connected = None

      Primary display allocation: Searches for a connected display according to the PortOrder. Because no displays are detected, the Primary display is set to sDVO-C.

   c. The driver cannot detect the presence of a display connected to the Internal LVDS and external LVDS displays connected to some sDVO devices (for example, an LVDS connected to the CH7308). Consequently, the driver assumes that an LVDS display is connected if it is in the PortOrder. If you only want to use the internal LVDS when no CRT and devices are connected, then put LVDS in the PortOrder after them. For example:

      PortOrder = "53240" (CRT, LVDS)

      Display Connected = None

      Primary display allocation: Searches for a display connected according to PortOrder sequence. Since no display is connected and since LVDS is specified in the PortOrder, the driver assumes that an LVDS display is connected. Consequently, the Primary display is set to LVDS.

   d. Because the driver cannot detect the presence of a display connected to the Internal LVDS and certain external LVDS displays, it therefore always assumes that they are connected if they are listed in the PortOrder. Be careful not to set the PortOrder that prevents the driver from detecting a connected display. For example:

      PortOrder = "54320" (CRT, LVDS sDVO)

      Displays Connected = CRT

      Primary display allocation: Searches for a connected display according to the PortOrder. In this case, the Primary display is set to CRT.

      Secondary display allocation: Searches for a connected display according to the PortOrder. Even though sDVO is connected, the driver assumes that the internal LVDS is also connected. Consequently, the driver never detects the display connected to the sDVO port. To change this, move sDVO-C before LVDS in the PortOrder ("53420" rather than "54320").

e. If the port drivers are not loaded for any of the ports specified in the PortOrder, the driver enables port 5 (CRT) only. For example:
PortOrder = "32000" (sDVO)
PortDrivers = "" (None)
Primary display allocation: Searches for displays connected according to the PortOrder. Since no port drivers are available for the specified ports, CRT port 5 is enabled. Consequently, the Primary display is set to CRT.

3.12.2 Detectable Displays

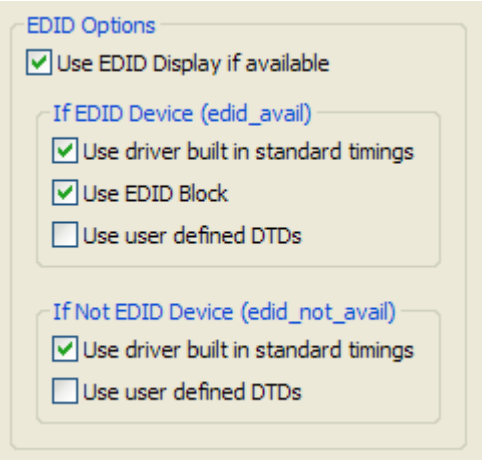
The table below provides a list of displays that are detectable by the IEGD.

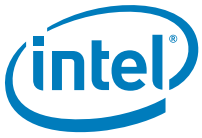
Table 25. Detectable Displays

Transmitter	Display Type	Detectable by IEGD?
GMCH Analog CRT	VGA	Yes
GMCH Integrated LVDS	LVDS	No (assumed attached)
GMCH Integrated TV Out	TV Out	N/A
CH7021	VGA Bypass	TBD
CH7307	DVI	Yes
CH7308	LVDS	No (assumed attached)
CH7315	HDMI/DVI	TBD
CH7317	VGA Bypass	TBD
CH7319	DVI	Yes
CH7320	DVI	Yes
SiI 1362	DVI	Yes
SiI 1364	DVI	Yes

3.13 Advanced EDID Configuration

Shown in the following EDID Options example, the If EDID Device (`edid_avail`) and If Not EDID Device (`edid_not_avail`) options in CED are found on the CRT, sDVO, LVDS, and TV Out configuration pages.





These options control the available timings for any display. The `edid_avail` parameter is used when EDID information is read from the display. If the driver is unable to read EDID information from the display or if the `edid` parameter is set to "`0`" (disable), then the settings of the `edid_not_avail` parameter are used.

The default behavior of `edid_avail` is to use the driver's built-in standard timings and EDID block and filter modes. The default for `edid_not_avail` is to use the driver's built-in standard timings. Please see Table 24 for more information on these parameters.

The IEGD supports three different types of EDID display modes:

1. **Built-in display modes**. These modes are hard-coded in the IEGD. These modes can be filtered based on the EDID block.
2. **EDID-DTDs**: These are Detailed Timing Descriptors read from the EDID block. EDID can have these DTDs along with other information about the display.
3. **User-specified DTDs** defined in CED. See Section 3.13.2.

The Advanced EDID Configuration supports different possible combinations of display modes when an EDID display is present along with user-specified DTDs.

3.13.1 Sample Advanced EDID Configurations

The table below presents various EDID configurations and the EDID settings in CED used for those configurations.

Table 26. Sample Advanced EDID Configurations (Sheet 1 of 2)

Configurations	CED Settings	Description
1. Use only filtered built-in and any EDID-DTDs when the display has EDID information 2. Use all built-in modes when the display doesn't contain EDID information	edid = 1 edid_avail = 3 edid_not_avail = 1	Default values.
1. Use only filtered built-in modes and EDID-DTDs when the display has EDID. 2. Use only user-DTDs otherwise.	edid = 1 edid_avail = 3 edid_not_avail = 4	This configuration allows the IEGD to use its built-in display modes and the modes provided by the display. If the IEGD is unable to read EDID information from the display, then the IEGD uses the user-DTDs defined in CED.

Table 26. Sample Advanced EDID Configurations (Sheet 2 of 2)

Configurations	CED Settings	Description
1. Use only user-DTDs regardless of connected display. (Typically used for a custom panel that only supports user-defined DTDs.) 2. Use limited set of timings when a panel EDID is present, but the IEGD cannot read the EDID information.	edid = 0 edid_avail = (any value) edid_not_avail = 4	Only user-DTDs defined in CED are used.
1. Use EDID-DTDs for an EDID display. 2. Use user-DTDs for a non-EDID display.	edid = 1 edid_avail = 2 edid_not_avail = 4	This configuration uses the EDID-DTDs when an EDID display is detected and EDID information is read from the display. If the driver detects a non-EDID display, then the IEGD uses user-DTDs defined in CED.
1. Use only EDID-DTDs and user-DTDs for an EDID display. 2. Use user-DTDs only for a non-EDID display.	edid = 1 edid_avail = 6 edid_not_avail = 4	This configuration uses both EDID-DTDs and user-DTDs when the IEGD detects an EDID display. If the driver detects a non-EDID display, then the IEGD uses user-DTDs defined in CED.

3.13.2 User-Specified DTDs

CED provides the ability to input DTD data directly. There are numerous sources of DTD data: VESA, panel manufacturers, etc. The figure below shows an example DTD Editor page, and the table below presents DTD parameter descriptions.

Figure 26. IEGD DTD Editor

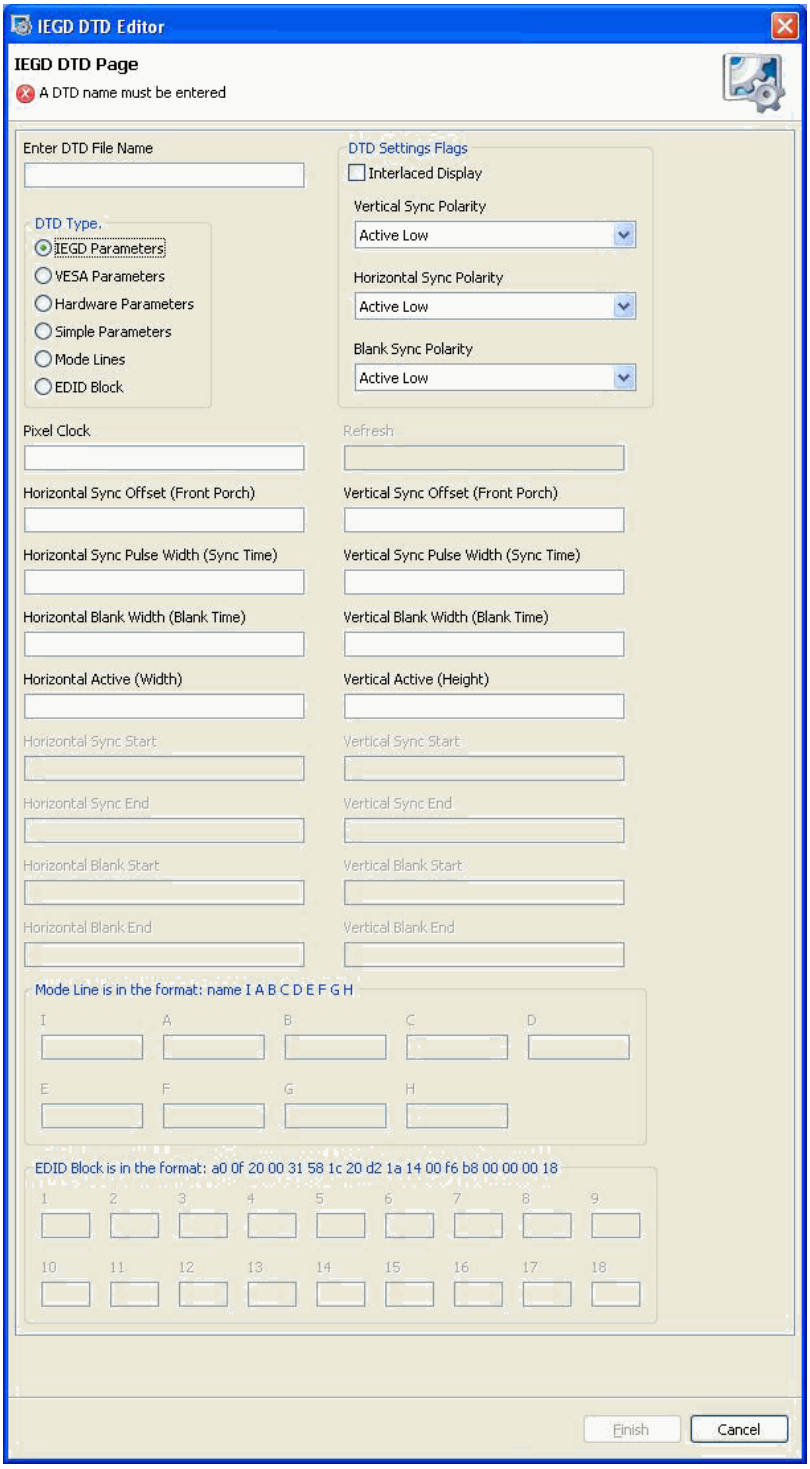


Table 27. DTD Parameter Descriptions

Parameter	Description
Enter DTD File Name	Enter a name for this customized DTD. This is a required field and the name must be between 1 and 50 characters and may contain spaces and underscores.
DTD Type	Select the DTD Type that most closely aligns with your display parameters. Options are: • IEGD Parameters • VESA Parameters • Hardware Parameters • Simple Parameters • Mode Line • EDID Block
IEGD Parameters	The IEGD Parameters are the same as the current PCF/CED DTD parameters.
VESA Parameters	The VESA Parameters allow the user to create a DTD from a VESA monitor timing standard.
Hardware Parameters	The Hardware Parameters are the parameters that are used by IEGD.
Simple Parameters	The Simple Parameters (CVT Standard) is a process for computing standard timing specifications. The method for developing Reduced Blanking timings is not included.
Mode Line	The Mode Line is a video timing spec used by XFree86. The Xfree86 timing setting for Mode Line is "name" I A B C D E F G H. For example: "640x480@8bpp" 25.175 640 672 728 816 480 489 501 526.
EDID Block	The EDID Block is the detailed timing section (18 bytes) of the basic 128-byte EDID data structure. The detailed timing section starts at 36h of the 128-byte EDID data structure. Enter the EDID block 1 byte at a time. Example: a0 0f 20 00 31 58 1c 20 d2 1a 14 00 f6 b8 00 00 00 18
Horizontal Sync Offset	Specifies the amount of time after a line of the active video ends and the horizontal sync pulse starts (Horizontal Front Porch). Range 0-1023 [10 bits].
Horizontal Sync Pulse Width	Width of the Horizontal Sync Pulse (Sync Time) which synchronizes the display and returns the beam to the left side of the display. Range 0-1023 [10 bits].
Horizontal Sync Start	This parameter specifies the start of the horizontal active time. Range 0-40957.
Horizontal Sync End	This parameter specifies the end of the horizontal active time. Range 0-49148.
Horizontal Blank Width	This parameter indicates the amount of time it takes to move the beam from the right side of the display to the left side of the display (Blank Time). During this time, the beam is shut off, or blanked. Range 0-4095 [12 bits].
Horizontal Blank Start	This parameter specifies the start of one line of the video and margin period. Range 0-32766.
Horizontal Blank End	This parameter specifies the end of one line of the video and margin period. Range 0-65533.
Horizontal Active	Number of pixels displayed on a horizontal line (Width). Range 0-4095 [12 bits].
Pixel Clock	Pixel clock value in KHz. Range 0-0x7fffffff
Refresh	Also known as the Vertical Refresh, the rate the full display updates. Standard refresh rates are 50Hz, 60Hz, 75Hz, and 85Hz.
Vertical Sync Offset	Specifies the amount of time after last active line of video ends and vertical sync pulse starts (Vertical Front Porch). Range 0-4095 [12 bits].

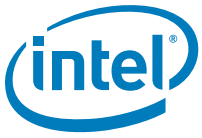


Table 27. DTD Parameter Descriptions (Continued)

Parameter	Description
Vertical Sync Pulse Width	Specifies the Width of the Vertical Sync Pulse which synchronizes the display on the vertical axis and returns the beam to the top, left side of the display. Range 0-63 [6 bits].
Vertical Sync Start	This parameter specifies the start of the vertical sync. Range 0-4157.
Vertical Sync End	This parameter specifies the end of the vertical sync. Range 0-4220.
Vertical Blank Width	The amount of time for the complete vertical blanking operation to complete. It indicates the time it takes to move the beam from the bottom right to the top, left side of the display (Blank Time). During this time, the beam is shut off, or blanked. Range 0-4095 [12 bits].
Vertical Blank Start	This parameter specifies the start of display vertical blanking including margin period. Range 0-4094.
Vertical Blank End	This parameter specifies the end of vertical blanking. Range 0-8189.
Vertical Active	The number of active lines displayed (Height). Range 0-4095 [12 bits].
DTD Settings Flags	This section allows you to set flags for Interlace, Vertical Sync Polarity, Horizontal Sync Polarity, and Blank Sync Polarity. Each field in this section is described below. Interlaced Display: Check for Interlaced Unchecked for Non-interlaced Vertical Sync Polarity: Active Low (Default) Active High Horizontal Sync Polarity: Active Low (Default) Active High ***Note:*** These flags are IEGD-specific and do not correspond to VESA 3.0 flags.

3.14 Using an External PCI Graphics Adaptor as the Primary Device

The IEGD can be configured to work with an external PCI graphics adaptor card as the primary graphics adaptor device with the Intel internal graphics device (GMCH) as the secondary graphics device. You can configure your system to boot with a PCI graphics adaptor in the System BIOS. When an external PCI graphics adaptor is designated as the primary graphics adaptor, the Intel GMCH becomes the secondary graphics device.

Note: The term *secondary* adaptor refers to the adaptor that is not the *boot-up*, or VGA-Compatible, adaptor. The secondary adaptor is not necessarily the secondary display as assigned by the OS.

You can configure an external PCI card to work with the IEGD as follows:

- The external PCI card as the primary graphics adaptor and the GMCH internal graphics device as the secondary.
- The external PCI card as the secondary graphics adaptor and the GMCH internal graphics device as the primary.

Note: This feature is not supported on Microsoft Windows CE systems.

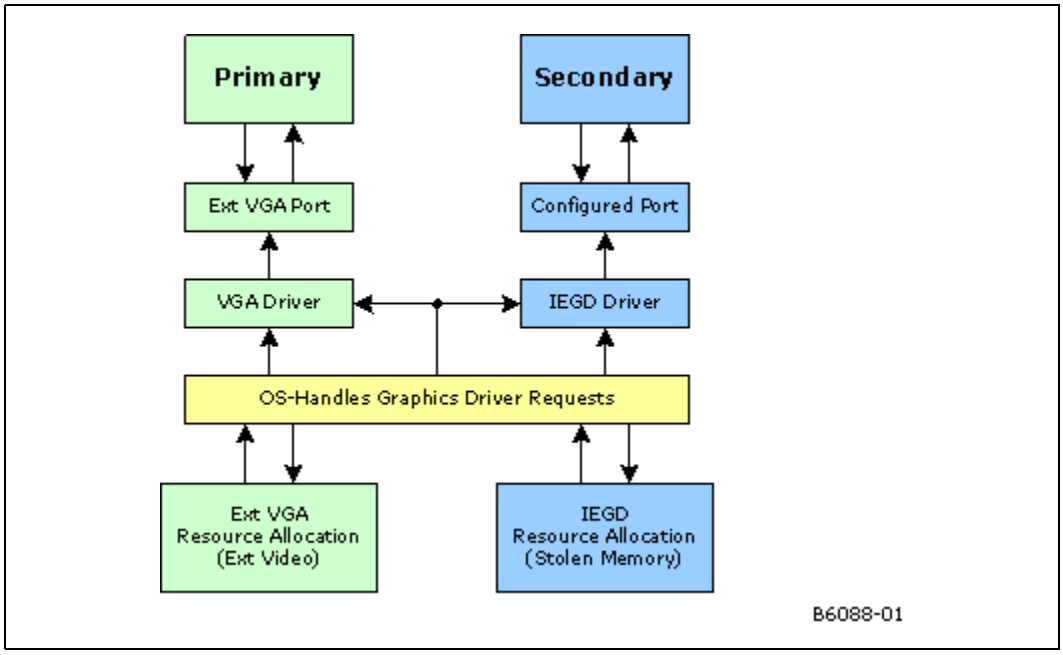
The IEGD allows you to specify which display is primary, secondary, and tertiary. It allows Twin and Clone configurations on the internal graphics device when the external PCI display is the primary graphics adaptor. It also allows Twin and Clone configurations on the internal graphics device when the external PCI device is the secondary graphics adaptor.

An external PCI graphics driver runs independently without sharing resources with the IEGD.

The following figures show several configurations when an external PCI adaptor is the primary graphics device and when it is the secondary graphics device.

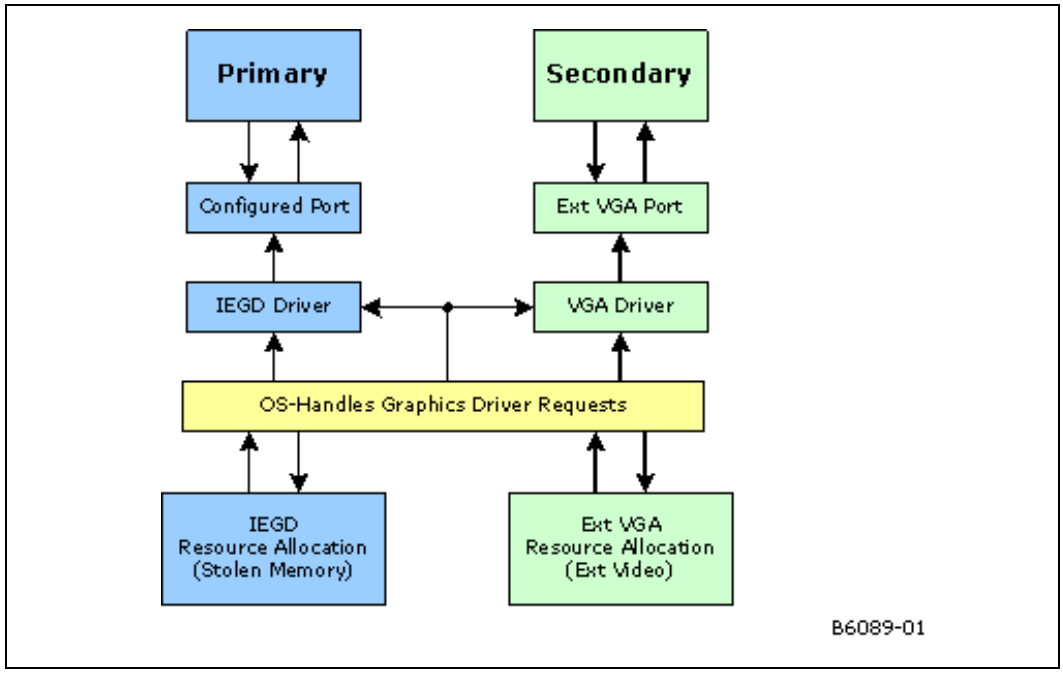
The figure below shows an External PCI card as the primary graphics adaptor card and the IEGD driver as the secondary. The drivers do not share hardware resources. The OS decides the framebuffer content and handles that by drawing to the respective driver independently.

Figure 27. External PCI Graphics Card as Primary Driver and IEGD as Secondary Driver



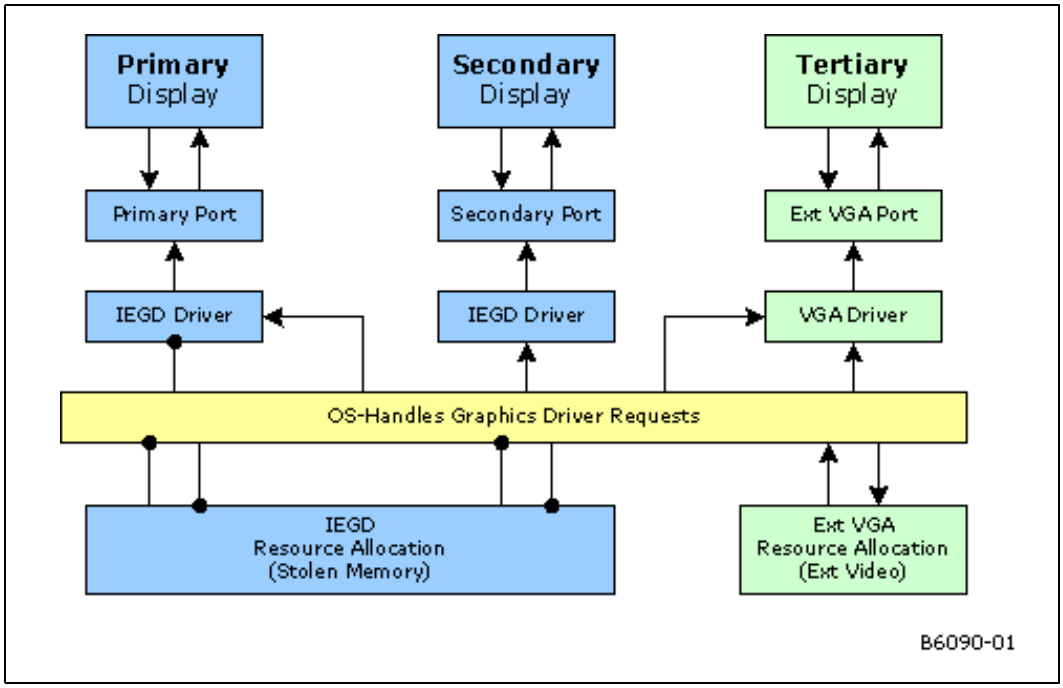
The figure below shows the interaction between the IEGD driver and the External VGA driver when the IEGD is booted as the primary driver. Again, the drivers do not share hardware resources. The OS decides the framebuffer content and handles it by drawing to the respective driver independently.

Figure 28. IEGD as Primary Driver and External PCI Graphics Card as Secondary Driver



The figure below shows a sample configuration where the internal graphics device is primary and configured to use two ports to drive two displays while an external PCI graphics adaptor is used to drive a tertiary display. Note that regardless of the number of ports being assigned to a driver, the external PCI graphics run independently without sharing resources with the IEGD driver.

Figure 29. IEGD as Primary Driver With Two Displays and External PCI Driving a Tertiary Display



3.15 Enhanced Clone Mode Support

The Enhanced Clone Mode feature allows you to specify a clone display size that is different from the primary display. It also allows you to change the clone display size at runtime using the IEGD Runtime GUI (see Section 5.5, "Viewing and Changing the Driver Configuration from Microsoft Windows" on page 112 or Section 7.8, "Using the IEGD Runtime Configuration GUI on Linux Systems" on page 179 for Linux systems).

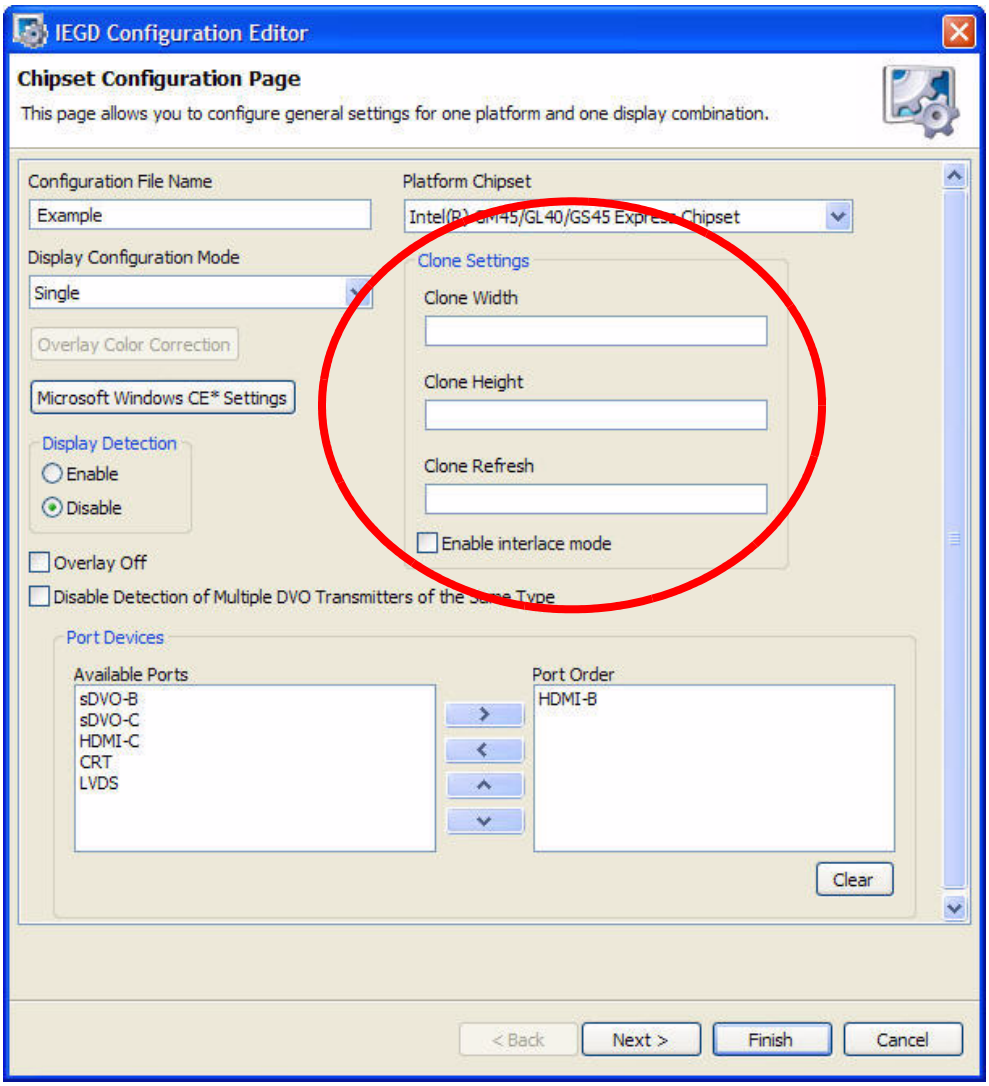
In Clone mode, the framebuffer is always allocated to match primary display size. On the clone display (secondary display) the image is centered if the display is bigger than the framebuffer. Centering is done only if the requested resolution and refresh rate is not available for the clone display.

Extended Clone mode is implemented through the use of four CED parameters:

- Clone Width — allows you to specify a width for the clone display
- Clone Height — allows you to specify a height for the clone display
- Clone Refresh — allows you to specify a refresh rate for the clone display
- Enable interlace mode — allows you to use interlace mode for the clone display

3.15.1 Extended Clone Mode CED Configuration

The following CED screenshot shows a sample Extended Clone mode setting configuration.



See also "Sample Clone Mode Configurations".

3.15.2 Sample Clone Mode Configurations

The following examples illustrate Clone Mode configurations for the following combinations:

- CRT + integrated LVDS (Example 1 on page 86)
- CRT + Fixed size DVI display (Example 2 on page 86)

Example 1. Mobile Intel® GM45 Express Chipset, Internal LVDS, CRT, and LVDS

This example shows how to set up a clone mode configuration consisting of an internal LVDS, CRT, and external LVDS device.

1. Choose a CRT that supports resolutions larger than 1024x768 and configure the following settings on the Chipset Configuration Page:
   - Platform Chipset = Mobile Intel® GM45 Express Chipset
   - Display Configuration Mode = Clone
   - Clone Width = 1024
   - Clone Height = 768
   - Clone Refresh = 60
   - Port Order = CRT, LVDS
2. Click **Next** on the Chipset Configuration Page and provide port names for the CRT, LVDS, and sDVO-B port devices.
3. On the LVDS Port Configuration Page, click the **Flat Panel Settings** button and set the Width and Height to **1024** and **768** respectively.
4. Package and generate an installation for the configuration and move the resulting `iegd.inf` file to the target machine. (Please see "Creating a New Package" on page 50 for specific instructions.)

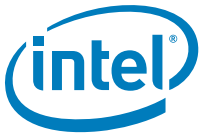
After you have moved the iegd.inf file to the target machine, do the following:

1. Set 800x600 on CRT and ensure that an 800x600 image appears at the top, left corner of the LVDS display or that the image has been scaled to match the panel size based on panel used.
2. Set 1280x1024 on CRT and check that the LVDS display is panning. Ensure that the clone mouse pointer is in sync with the primary display.

Example 2. Intel® 915GV, Chrontel 7307, CRT, and DVI

This sample shows how to set up a clone mode configuration consisting of a CRT and DVI display on a Chrontel* 7307 serial sDVO transmitter.

1. Choose a CRT that supports resolutions larger than 1024x768 and configure the following settings on the Chipset Configuration Page:
   - Platform Chipset = Intel® 915GV
   - Display Configuration Mode = Clone
   - Clone Width = 1024
   - Clone Height = 768
   - Clone Refresh = 60
   - Port Order = CRT, sDVO-B
2. Click **Next** on the Chipset Configuration Page and provide port names for the CRT and sDVO-B port devices.



3. On the sDVO-B Port Configuration Page, click the **Flat Panel Settings** button and set the Width and Height to **1024** and **768** respectively. Select **CH7307** from the Select sDVO Device list.
4. Package and generate an installation for the configuration and move the resulting `iegd.inf` file to the target machine. (Please see "Creating a New Package" on page 50 for specific instructions.)

After you have moved the `iegd.inf` file to the target machine, do the following:

1. Set 800x600 on CRT and ensure that an 800x600 image appears at the top, left corner of the DVI display or that the image has been scaled to match the panel size based on panel used.
2. Set 1280x1024 on CRT and check that the DVI display is in panning mode. Ensure that the clone mouse pointer is in sync with the primary display.

3.16 Scaling and Centering Configurations

This release supports the following scaling and centering configurations:

- Upscaling for the Chrontel CH7308 LVDS Transmitters
- Internal LVDS Scaling With EDID Panels
- Alignment in Clone mode
- sDVO as Primary
- Render Scaling modes to native panels connected to non-scaling port encoders

See the following topics for configuration details:

- "Upscaling for the Chrontel CH7308 LVDS Transmitters"
- "Internal LVDS Scaling with EDID Panels"
- "Centering Primary Display with Scaling Encoders"

3.16.1 Upscaling for the Chrontel CH7308 LVDS Transmitters

The IEGD can upscale lower-resolution modes (those smaller than the size of the respective panel) to the native size of the panel connected to a Chrontel CH7308* LVDS transmitter.

The IEGD uses a user-supplied DTD with the native flag set (also known as native DTD) as native timing for the panel connected to either a CH7308 transmitter.

If a native DTD is not supplied by the user, the IEGD takes the first available matching `FP info width` and `height` timings as native timing for the panel if standard timings were selected as part of `edid_avail` or `edid_not_avail` flags.

In order to support upscaling, the LVDS transmitters require the pipe to be set to native timing of the panel regardless of the user selected resolution. It also requires finding the native timing (also known as native DTD) of the panel based on user-supplied configuration information.

The CH7308 (sDVO) port drivers make the list of supported modes limited up to the size of panel. The port drivers also mark one of the timings as native DTD as follows (it goes to the next step only if native DTD isn't found in the current step).

1. It finds the timing with the user-defined DTD with the native DTD flag set. This becomes the native DTD for the panel.
2. If the panel is an EDID panel and user selected to use EDID DTDs, then the port driver marks the EDID DTD as native DTD.
3. If the user supplies a DTD without the native DTD flag set, then the port driver marks this one as the native DTD.
4. If none of the above steps works, the port driver finds the first matching timing for FP width, height and marks it as native DTD.

If none of the above steps work, then there is no native DTD and no upscaling is performed.

3.16.2 Internal LVDS Scaling with EDID Panels

The Internal LVDS connected to an EDID Panel supports scaling of modes other than native mode. To support this, the port driver exports information to the EDID parser that it can scale. The EDID parser does not remove other modes (that is, non-native modes) from the mode table. It only marks the native mode. When the IEGD queries the port driver on which modes are supported, the port driver then removes any modes that cannot be scaled (up or down depending on the port's hardware capability). When mode-setting occurs, the second display in Clone mode can indeed support non-native modes even though the panel had EDID. This occurs only if a native mode can be found the port driver can scale. Otherwise, the port driver ignores the scaling information and the IEGD proceeds normally.

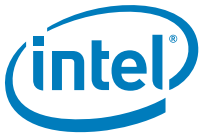
The driver also supports Internal LVDS Scaling on EDID-less panels. The steps that enable this are the same as those described for the scaling of Chrontel LVDS transmitters in Section 3.16.1.

3.16.3 Centering Primary Display with Scaling Encoders

In Clone mode, the IEGD expects the primary display to have a framebuffer size (OS Aware mode) that matches the display’s native size of panel timings. When a display is designated as the primary display in a Clone mode configuration, and the user wants the primary to be centered (as explained in Section 3.16.5), users may want this setup to align a primary display on a scaling encoder with a secondary one that can only center. This won’t work by default for certain port encoders such as the internal LVDS, which default to hardware scaling. But IEGD has a mechanism to override hardware scaling thus forcing centering.

When possible, the IEGD allows centering of 640x480, 800x600, and 1024x768 resolutions on the primary display. In some cases (depending on panels), the image may appear on the top-left. It may also produce unusable output on some displays (such as a TV). Thus this type of configuration is more appropriate for LVDS panels.

To disable hardware scaling (and force centering for primary display on above modes), users only need to set the “Panel_Fit” attribute (attribute “0x12”) to ‘0’ (zero).



3.16.4 Enabling Render Scaling on Port Encoders without Hardware Scaling

The IEGD Render Scaling feature allows the driver to support any one of the standard modes (640x480, 800x600, 1024x768 or 1280x1024) as a drawable framebuffer size output to a native panel and connected via a port encoder that does not hardware scale. To achieve this, the GPU engine repeats all rendering operations twice (from the original OS-targeted back buffer) to a separate front buffer, which is rendered via the 3D engine for scaling. This feature is enabled by turning on the "Panel-Fit" attribute (attribute 0x12) on a port driver that does not support that attribute. But this only happens if there is a native mode timing (see Section 3.16.1 for information about how native mode timing is determined).

Users should be aware that this feature can impact performance and produce scaled output which is inferior in quality to hardware encoder scaling.

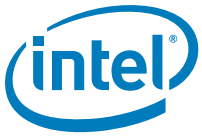
3.16.5 Alignment in Clone Mode

In Clone mode, both can be configured with separate timings and different resolutions. The content is the same on both displays. In the case where resolutions are different on the cloned displays, the display identified as primary drives the display mode and framebuffer size. There are three options for the cloned displays in this situation:

- *Panning*: If the clone display is smaller than the primary display, the displayed image can be off the screen with only the display showing a window into the overall image. Panning allows movement of the window, which is viewing the image based on the movements of the cursor.
- *Centering*: If the clone display is larger than the primary display mode, the display image can be centered in the clone display. Black borders are displayed around the image on the display.
- *Scaling:* There are two types of scaling in Clone mode, as described below.
  - *Hardware Encoder Scaling*: This feature adjusts the resolution of the image from the primary display to fit the resolution of the clone display. This permits scaling up to a larger display (upscaling), or scaling down to a smaller display (downscaling). It also allows the full image to be displayed within the full resolution of the clone display.

    Some systems may have cloned displays that cannot scale but have a primary display that can scale (such as an internal LVDS). In non-panning modes (i.e., centering/hardware scaling), this display combination will result in the primary display being scaled up (internal LVDS) but the clone display will be centered. Section 3.16.3 explains how to force the primary display to center — thus allowing both displays to center. Or, *Render Scaling* as follows explains how to make both displays scale up to full size.
  - *Render Scaling:* For clone display, a situation is possible where the primary display uses a hardware scaling port encoder and the secondary display uses a non-scaling port encoder. Assuming both displays are output via native panels, the resulting output should see the primary scaling of any smaller mode to full panel size. But the secondary display will center the smaller modes. The above explanation (see *Hardware Encoder Scaling*) explains how to align both displays to be centered. Using the Render Scaling feature, the opposite can be achieved. Ensure the non-scaling encoder is primary and enable Render Scaling on that port (see Section 3.16.4). This will make the GPU render-scale the smaller mode and achieve the full panel size. The clone display (now the scaling encoder) will, however, take the render-scaled image as its input (and output) to the clone display panel. This feature will be upgraded in the future so that the clone display can independently take in the original framebuffer image as its input.

This page is intentionally left blank.



4.0 Generating the VBIOS

4.1 Overview

The Intel Embedded Video BIOS incorporates many of the features and capabilities of the Intel® Embedded Graphics Drivers. The 9.0 version of the VBIOS includes the following new features:

- Intel® GM45 Express chipset
- Intel® System Controller Hub US15W chipset
- Support for Intel® Q35 Express chipset

4.2 System Requirements

The new Video BIOS can be built on a host Microsoft Windows* system and moved to the target system. The host system must have a 32-bit Microsoft Windows operating system installed with the capability to execute DOS commands from a command line window.

The target system must contain one of the following Intel chipsets:

- Intel® GM45 Express chipset
- Intel® System Controller Hub US15W chipset
- Intel® Q35 Express chipset
- Mobile Intel® GME965 Express chipset
- Intel® Q965 Express chipset
- Mobile Intel® 945GSE Express chipset
- Mobile Intel® 945GME Express chipset
- Intel® 945G Express chipset
- Intel® 915GV Express chipset
- Mobile Intel® 915GME Express chipset
- Mobile Intel® 910GMLE Express chipset

The target system must contain a minimum of 64 Mbytes of RAM.

See also these VBIOS configuration topics:

- "Configuring and Building the VBIOS with CED"
- "VBIOS, Driver Compatibility, and Data Dependencies"
- "Supported VGA Display Modes for the IEGD Video BIOS"

4.3 Configuring and Building the VBIOS with CED

The Intel® Embedded VBIOS is built with the Intel Configuration Editor (CED). The VBIOS will use the configuration that you specify in CED. The VBIOS is selected to be built when you specify the Video BIOS as a Target OS in your package configuration. After specifying the Video BIOS, follow all CED prompts, and be sure to select "Generate VBIOS" when available. The VBIOS will then be built when you select "Generate Installation" in CED.

Before building your VBIOS, you must set up your DOS environment with two items.

1. Install the Open Watcom* C/C++ compiler.
   The User Build System for the VBIOS relies on the Open Watcom C/C++ compiler to be able to build a 16-bit DOS binary required for the BIOS. The VBIOS has been tested with version 1.7a of the Open Watcom compiler. This can be downloaded from the following location:
   http://www.openwatcom.com
2. Set up directory paths.
   You must set up the PATH environment variable in DOS to be able to execute the Watcom compiler. If Watcom was installed with its default path, CED will by default be able to use it.

When you generate a VBIOS, the CED produces the following folders and files:

- Compiled_VBIOS folder
  - `iegdtsr.exe` (Terminate and Stay Resident executable)
  - `VGA.BIN` (Option ROM)
- `IEGD_X_x_VBIOS.zip` (this file is generated by the build system, where X is the major version and x is the minor version)

The `iegdtsr.exe` can be copied to any folder on the target machine. To run the TSR, boot the target machine with DOS, and then run the iegdtsr.exe from the DOS command line.

The `VGA.bin` file is the binary option ROM that can be merged with your system BIOS per the instructions provided by your system BIOS vendor.

The `IEGD_X_x_VBIOS.zip` file, where X is the major version and x is the minor version, contains default builds of the TSR executable and Option ROM for the various chipsets. The filenames are `iegdtsr-def.exe` and `vga-def.bin` and are located in the tsr or orom folder of the specific chipset folder (see the figure Figure 30).

See also the following topics:

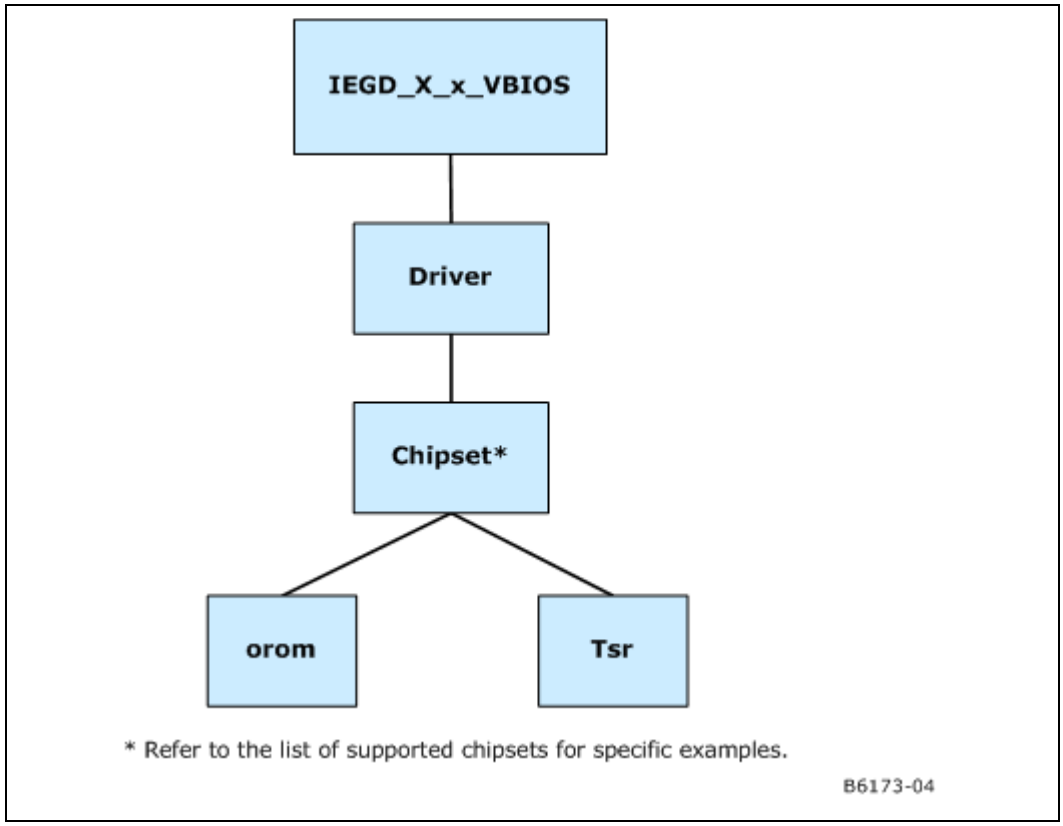
- "Selecting the Build Folder"
- "Configuring the Video BIOS"
- "Building the VBIOS"

4.3.1 Selecting the Build Folder

The 9.0 version of the VBIOS contains specific folders used for creating a VBIOS that is either an option ROM (OROM) that can be merged with the system BIOS, or an executable Terminate and Stay Resident (TSR) program for debugging purposes. There are also separate directories for the different chipsets that are supported. CED will build both the TSR and OROM.

Figure 30 shows the directory structure for the Video BIOS libraries contained within CED.

Figure 30. Video BIOS Directory Structure



4.3.2 Configuring the Video BIOS

CED is used to configure the VBIOS. The display settings will be used in the same manner as they are used for the driver.

4.3.2.1 COMMON_TO_PORT

This setting allows you to associate standard display names used in most system BIOSs to specific ports that are recognized by IEGD (e.g., LVDS, sDVO-B, sDVO-C). The VBIOS makes this association when the VBIOS calls the System BIOS Intel® 5F interrupt functions.

This setting is a six digit number, where each digit is associated with one of the system BIOS displays (from left to right):

1 : CRT - Standard analog CRT
2 : TV1 - TV Output 1
3 : EFP1 - DVI Flat Panel 1
4 : LFP - Local Flat Panel (Internal LVDS display)
5 : TV2 - TV Output 2
6 : EFP2 - DVI Flat Panel 2

The values above are an example of the typical displays and corresponding order used by a system BIOS. However, this may vary depending on how your system BIOS has implemented the displays and the Intel 5F interrupt functions.

The value in each position in the setting should be the associated port number. Using the typical settings above, if you want to associate CRT in the system BIOS with the internal CRT (port 5) and LFP in the system BIOS with internal LVDS (port 4) in the VBIOS, set COMMON_TO_PORT to be 500400.

Warning: This feature must be compatible with the system BIOS. If the system BIOS does not properly implement the Intel 5F functions, then using the `COMMON_TO_PORT` feature could cause unpredictable results with the displays. If you are unsure, set `COMMON_TO_PORT` to all zeros (000000) to disable this feature.

Note: The `displaydetect` parameter must be set to `Enabled` in order for the `COMMON_TO_PORT` values to be used.

4.3.2.2 post_display_msg

This setting is a binary setting that enables (1) or disables (0) post messages to the display.

4.3.2.3 OEM Vendor Strings

The following settings are string values that allow you to set the values that are returned from the Intel 4F interrupt functions.

```
oem_string
oem_vendor_name
oem_product_name
oem_product_rev
```

4.3.2.4 Default Mode Settings

These settings establish the default VGA or VESA mode to use for the primary (0) and secondary (1) displays. The values should be set to a valid standard VGA or VESA mode (in hexadecimal format, for example, 0x117). Note that a VGA mode can only be set on one display and a second display is disabled unless the `DisplayConfig` parameter is set to `twin` or `clone` mode.

```
default_mode_0
default_mode_1
```

4.3.2.5 Default Refresh Settings

These settings allow you to specify which refresh rate is used for certain VESA modes on the primary and secondary displays. For example, mode 0x117 specifies refresh rates of 60 Hz, 75 Hz, and 85 Hz. This setting allows use to specify which of those three rates to use (specified in decimal, e.g., default_refresh_0=60).

```
default_refresh_0
default_refresh_1
```

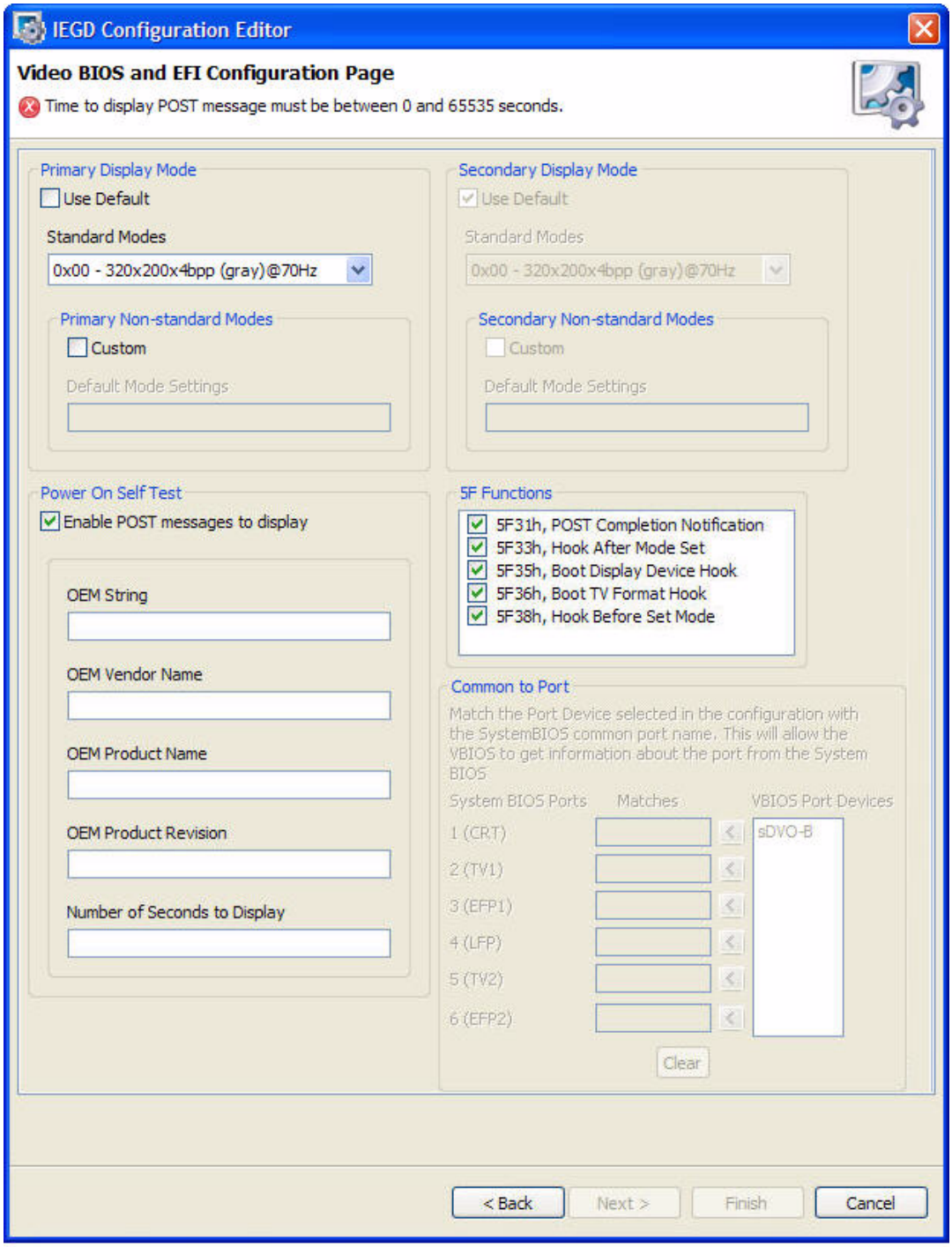
4.3.2.6 default_vga_height

This setting allows you to specify which resolution is used for certain VGA modes. Because only one VGA mode can be supported on both displays, this setting applies to the primary display mode (default_mode_0). For example, mode 3 specifies three possible resolutions: 640x200, 640x350, and 720x400. In this example, setting default_vga_height=350 indicates the resolution 640x350.

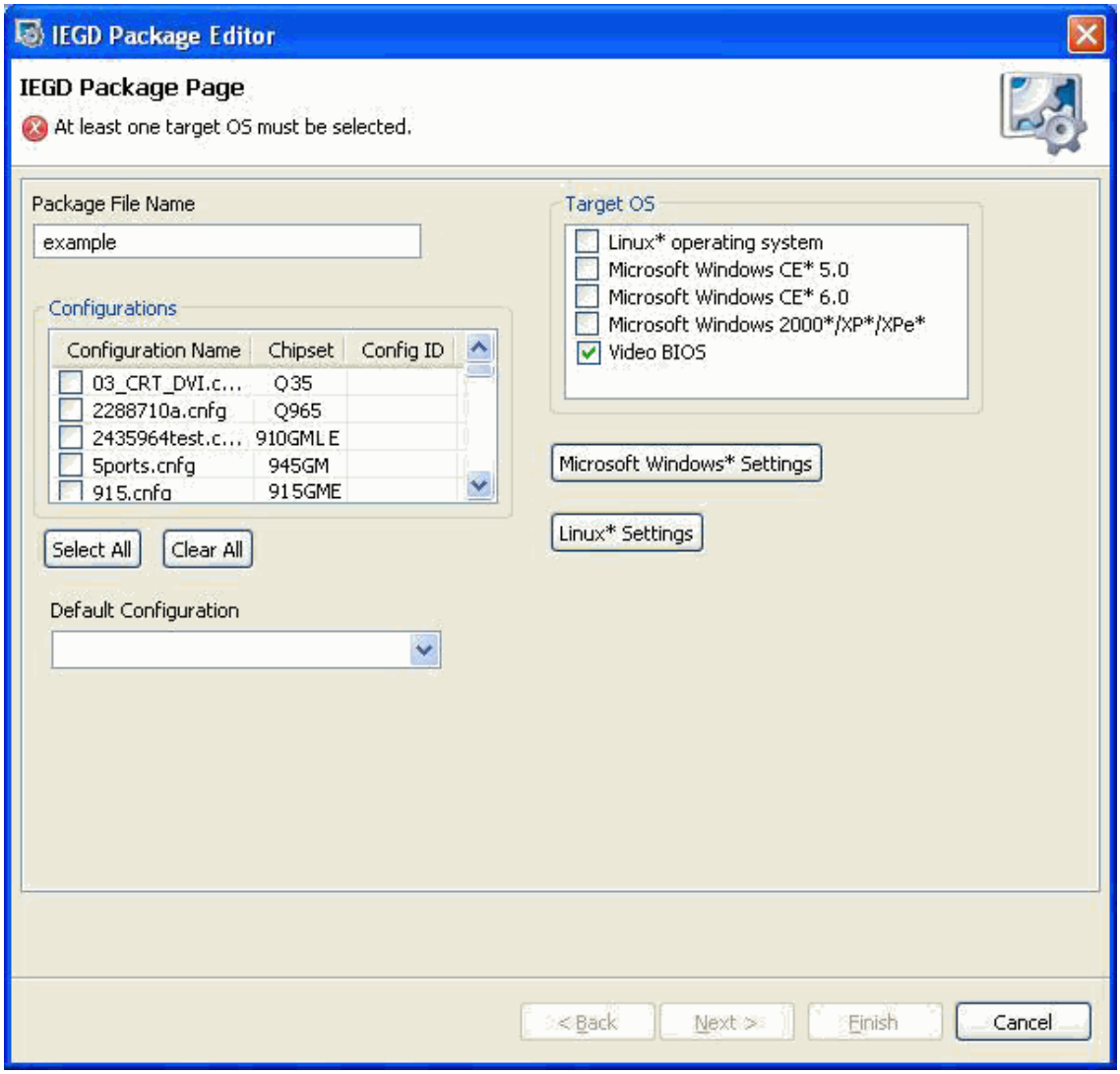
4.3.3 Building the VBIOS

CED is used to build the VBIOS. The following steps and screenshots outline a typical CED VBIOS build procedure.

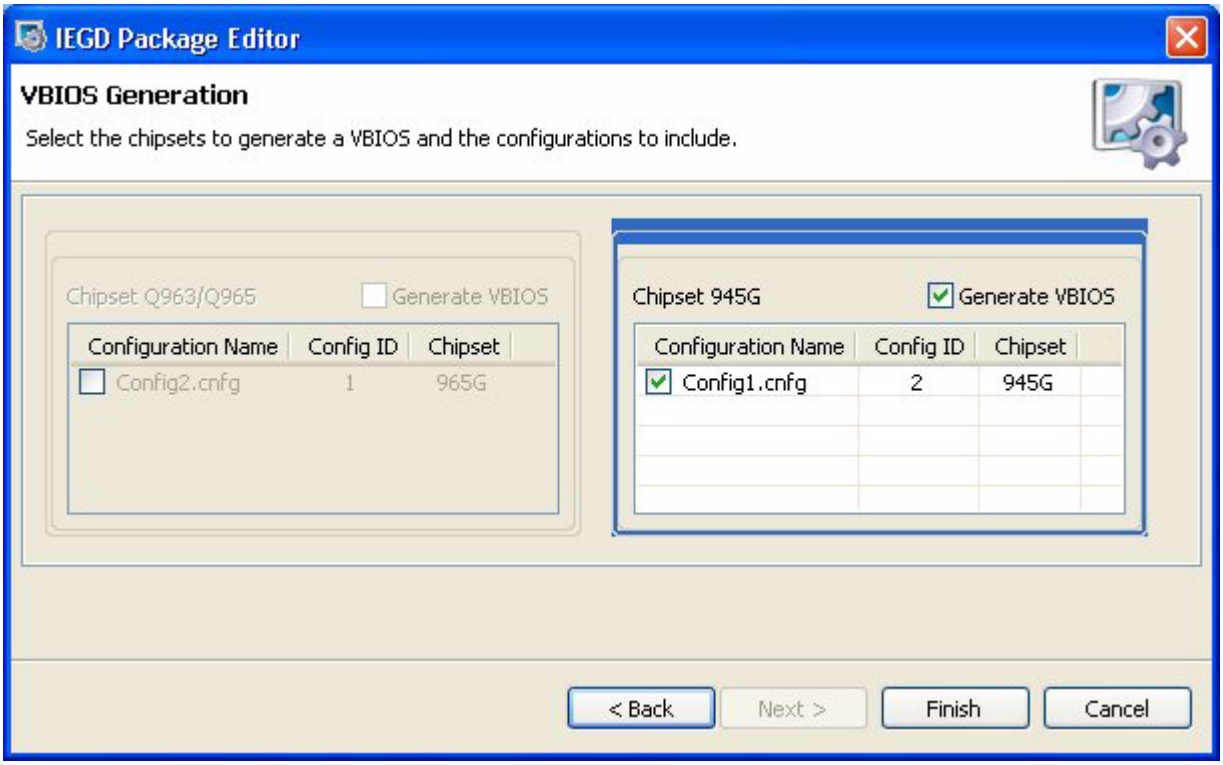
1. Define your configuration via CED, being sure to complete the Video BIOS Configuration Page.



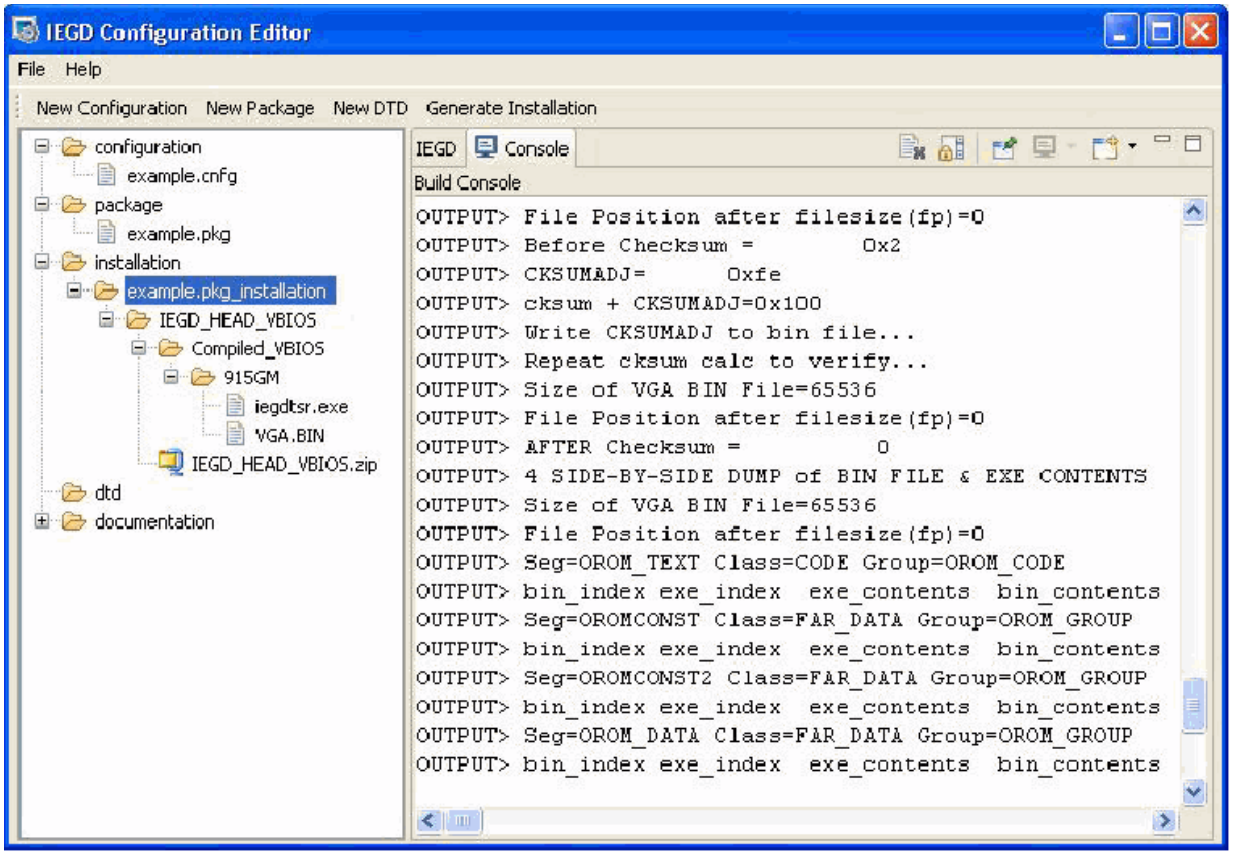
2. When you define the package you are building, be sure to select "Video BIOS" as "Target OS".



3. Generate the installation. The following may display:



4. Generated files should now be in your CED Installation folder.



4.4 VBIOS, Driver Compatibility, and Data Dependencies

The Intel Embedded Graphics Drivers do not depend on any data from the VBIOS, and will either use driver settings or select default values for the attached displays. This allows the driver to properly operate with incompatible BIOS or BIOS replacements.

The Intel Embedded Graphics Drivers will retrieve settings, such as panel ID and other display settings from the Embedded VBIOS. The Embedded VBIOS allows for configuration of display timings that can also be used for the Intel Embedded Graphics Drivers.

4.5 Supported VGA Display Modes for the IEGD Video BIOS

The VBIOS supports many VESA and standard VGA modes. See Table 28 and Table 29 for the VGA and VESA modes and vertical refresh rates that are supported by the VBIOS.

Note: Although IBM labeled certain EGA modes with a (*) suffix and the VGA modes with a (+) suffix (such as mode 3, 3* and 3+), the VGA modes are so common that this document does not use the (+) suffix to refer to them.

The actual availability of any particular mode depends on the capabilities of the display device, the amount of memory installed, and other system parameters.

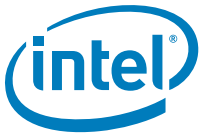


Table 28. Supported VGA Video Display Modes

Video Mode	Pixel Resolution	Color Depth (bpp)	Mode Type	Display Adapter	Font Size	Character Resolution	Dot Clock (MHz)	Horiz. Freq. (KHz)	Vert Freq (Hz)	Video Memory (KBytes)
00h	320 x 200	16 (gray) (4 bpp)	Text	CGA	8 x 8	40 x 25	25	31.5	70	256
	320 x 350	16 (gray) (4 bpp)		EGA	8 x 14	40 x 25	25	31.5	70	256
	360 x 400	16 (4 bpp)		VGA	9 x 16	40 x 25	28	31.5	70	256
01h	320 x 200	16 (4 bpp)	Text	CGA	8 x 8	40 x 25	25	31.5	70	256
	320 x 350	16 (4 bpp)		EGA	8 x 14	40 x 25	25	31.5	70	256
	360 x 400	16 (4 bpp)		VGA	9 x 16	40 x 25	28	31.5	70	256
02h	640 x 200	16 (gray) (4 bpp)	Text	CGA	8 x 8	80 x 25	25	31.5	70	256
	640 x 350	16 (gray) (4 bpp)		EGA	8 x 14	80 x 25	25	31.5	70	256
	720 x 400	16 (4 bpp)		VGA	9 x 16	80 x 25	28	31.5	70	256
03h	640 x 200	16 (4 bpp)	Text	CGA	8 x 8	80 x 25	25	31.5	70	256
	640 x 350	16 (4 bpp)		EGA	8 x 14	80 x 25	25	31.5	70	256
	720 x 400	16 (4 bpp)		VGA	9 x 16	80 x 25	28	31.5	70	256
04h	320 x 200	4	Graph	All	8 x 8	40 x 25	25	31.5	70	256
05h	320 x 200	4 (gray)	Graph	CGA	8 x 8	40 x 25	25	31.5	70	256
	320 x 200	4 (gray)		EGA	8 x 8	40 x 25	25	31.5	70	256
	320 x 200	4		VGA	8 x 8	40 x 25	25	31.5	70	256
06h	640 x 200	2	Graph	All	8 x 8	80 x 25	25	31.5	70	256
07h	720 x 350	Mono	Text	MDA	9 x 14	80 x 25	28	31.5	70	256
	720 x 350	Mono		EGA	9 x 14	80 x 25	28	31.5	70	256
	720 x 400	Mono		VGA	9 x 16	80 x 25	28	31.5	70	256
08h-0Ch	Reserved			-		-				
0Dh	320 x 200	16 (4 bpp)	Graph	E/VGA	8 x 8	40 x 25	25	31.5	70	256
0Eh	640 x 200	16 (4 bpp)	Graph	E/VGA	8 x 8	80 x 25	25	31.5	70	256
0Fh	640 x 350	Mono	Graph	E/VGA	8 x 14	80 x 25	25	31.5	70	256
10h	640 x 350	16 (4 bpp)	Graph	E/VGA	8 x 14	80 x 25	25	31.5	70	256
11h	640 x 480	2 (4 bpp)	Graph	VGA	8 x 16	80 x 30	25	31.5	60	256
12h	640 x 480	16 (4 bpp)	Graph	VGA	8 x 16	80 x 30	25	31.5	60	256
13h	320 x 200	256 (8 bpp)	Graph	VGA	8 x 8	40 x 25	25	31.5	70	256

4.6 Supported VESA Display Modes for the IEGD Video BIOS

The VBIOS supports many VESA and standard VGA modes. See Table 28 and Table 29 for the VGA and VESA modes and vertical refresh rates that are supported by the VBIOS.

The following table lists the supported VGA display modes. The actual availability of any particular mode depends on the capabilities of the display device, the amount of memory installed, and other system parameters.

Table 29. VESA Modes Supported by Video BIOS (Sheet 1 of 2)

Video Mode	Pixel Resolution	Colors (bpp)	Mode Type	Display Adapter	Vertical Frequency (Hz)	Video Memory (Mbytes)
101h	640 x 480	256 (8 bpp)	Graph	VGA	60	0.5
	640 x 480	256 (8 bpp)	Graph	VGA	75	0.5
	640 x 480	256 (8 bpp)	Graph	VGA	85	0.5
103h	800 x 600	256 (8 bpp)	Graph	SVGA	60	1
	800 x 600	256 (8 bpp)	Graph	SVGA	75	1
	800 x 600	256 (8 bpp)	Graph	SVGA	85	1
105h	1024 x 768	256 (8 bpp)	Graph	XVGA	60	1
	1024 x 768	256 (8 bpp)	Graph	XVGA	75	1
	1024 x 768	256 (8 bpp)	Graph	XVGA	85	1
107h	1280 x 1024	256 (8 bpp)	Graph	SXGA	60	2
	1280 x 1024	256 (8 bpp)	Graph	SXGA	75	2
	1280 x 1024	256 (8 bpp)	Graph	SXGA	85	2
111h	640 x 480	64K (16 bpp)	Graph	VGA	60	1
	640 x 480	64K (16 bpp)	Graph	VGA	75	1
	640 x 480	64K (16 bpp)	Graph	VGA	85	1
114h	800 x 600	64K (16 bpp)	Graph	SVGA	60	2
	800 x 600	64K (16 bpp)	Graph	SVGA	75	2
	800 x 600	64K (16 bpp)	Graph	SVGA	85	2

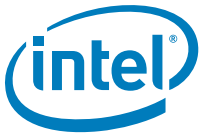
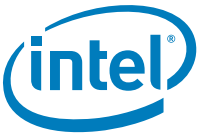


Table 29. VESA Modes Supported by Video BIOS (Continued) (Sheet 2 of 2)

Video Mode	Pixel Resolution	Colors (bpp)	Mode Type	Display Adapter	Vertical Frequency (Hz)	Video Memory (Mbytes)
117h	1024 x 768	64K (16 bpp)	Graph	XVGA	60	2
	1024 x 768	64K (16 bpp)	Graph	XVGA	75	2
	1024 x 768	64K (16 bpp)	Graph	XVGA	85	2
11Ah	1280 x 1024	64K (16 bpp)	Graph	SXGA	60	4
	1280 x 1024	64K (16 bpp)	Graph	SXGA	75	4
	1280 x 1024	64K (16 bpp)	Graph	SXGA	85	4
112	640 x 480	16M (32 bpp)	Graph	VGA	60	2
	640 x 480	16M (32 bpp)	Graph	VGA	75	2
	640 x 480	16M (32 bpp)	Graph	VGA	85	2
115	800 x 600	16M (32 bpp)	Graph	SVGA	60	4
	800 x 600	16M (32 bpp)	Graph	SVGA	75	4
	800 x 600	16M (32 bpp)	Graph	SVGA	85	4
118	1024 x 768	16M (32 bpp)	Graph	XVGA	60	4
	1024 x 768	16M (32 bpp)	Graph	XVGA	75	4
	1024 x 768	16M (32 bpp)	Graph	XVGA	85	4
11B	1280 x 1024	16M (32 bpp)	Graph	SXGA	60	8
	1280 x 1024	16M (32 bpp)	Graph	SXGA	75	8
	1280 x 1024	16M (32 bpp)	Graph	SXGA	85	8

This page is intentionally left blank.



5.0 Configuring and Installing Microsoft Windows Drivers

5.1 Editing the Microsoft Windows INF File

This section describes the driver-level information (`iegd.inf`) for the Microsoft Windows* operating system, which includes the following[1]:

- Microsoft Windows XP*
- Microsoft Windows XP Embedded*
- Microsoft Windows Embedded for Point Of Service* (WEPOS)

Note: Configuration and Installation information for the Microsoft Windows CE operating system is described in Chapter 6.0, "Configuring and Building the IEGD for Microsoft Windows CE Systems".

See also the following topics:

- "Universal INF Configuration"
- "INF File Backward Compatibility"
- "Dual Panel Configuration"
- "Chipset Dual Display Example"
- "Creating Registry Settings for Graphics Driver INF File"
- "Dynamic Port Driver Configuration"
- "Creating an .sld file for Microsoft Windows XP Embedded Systems"
- "Changing Default Display Mode"

5.1.1 Universal INF Configuration

Multiple display configurations can be specified in a single INF file. Each configuration is uniquely identified by the ConfigId parameter.

The driver reads the PanelId from the System BIOS during initialization and uses the configuration whose ConfigId matches the PanelId. If the System BIOS does not set a valid PanelId (for example, panelId = 0), the driver reads a configuration using ConfigId = 1. (A ConfigId value of 0 is invalid.)

You can override this behavior by specifying a ConfigId parameter as follows:

```
HKR,, ConfigId, %REG_DWORD%, %DEFAULT_CONFIG_ID%
```

In this case, the driver ignores the PanelId returned by the System BIOS. Instead, the IEGD uses the configuration information using the specified ConfigId.

1. These versions of the drivers are not WHQL (Windows Hardware Quality Labs) certified.

5.1.2 INF File Backward Compatibility

The current version of the IEGD uses the new INF file format. You cannot use the new INF file with pre-5.0 versions of the IEGD. However, you can still use pre-5.0 INF file formats with the current version of the IEGD.

5.1.2.1 INF File Backward Compatibility with IEGD Version 4.0

Version 4.0 of the IEGD provides backward compatibility with pre-4.0 versions of the INF file. This support is implemented through the `PcfVersion` key in the INF file, shown below:

```
HKR,, PcfVersion,     %REG_DWORD%, 0x0400
```

The IEGD uses this key to determine which version of the `.inf` file it is interpreting. When this key is present in the `.inf` file and its value is 0x0400, the driver reads it as a 4.0 `.inf` file. If this key is omitted from the `.inf` file or if its value is less than 0x0400, the driver reads the `.inf` file as a pre-4.0 file.

Note the following rules:

- If you use a pre-4.0 version of the `.inf` file with version 4.0 of the IEGD, the driver translates pre-4.0 configuration parameters to 4.0 parameters.
- You cannot use 4.0 parameters in a pre-4.0 `.inf` file. If you try, the driver ignores them.
- You cannot use pre-4.0 parameters in a 4.0 `.inf` file. If you try, the driver ignores them.

For example, the `usestdtimings` parameter is a pre-4.0 parameter. If it is specified in a 4.0 INF file, the driver ignores it. Similarly, if you attempt to add the `edid_avail` and `edid_non_avail` parameters to a pre-4.0 `.inf` file (that is, an `.inf` file where the `PcfVersion` key is not present), they are ignored by the driver.

The `PcfVersion` key is generated automatically by the CED utility and is placed in the `[iegd_SoftwareDeviceSettings]` section of the `.inf` file. The default `iegd.inf` file already contains the `PcfVersion` key. Please see Appendix A, "Example INF File" to view a sample `.inf` file.

5.1.3 Dual Panel Configuration

Below are the settings required to set the INF file to enable extended display configurations. Typically, these settings are output from the CED utility. However, the INF file may also be edited directly. See Table 30 for a description of these settings.

```
HKR, Config\%DEFAULT_CONFIG_ID%\General, DisplayConfig, %REG_DWORD%, 8
HKR, Config\%DEFAULT_CONFIG_ID%\General, PortOrder, %REG_SZ%, "5200"
```

5.1.4 Chipset Dual Display Example

The table below presents the dual display example for an Intel chipset.

Table 30. Example of Chipset Dual Display Parameter Setting (Sheet 1 of 2)

Dual Display Combination	Port Order
CRT + Internal LVDS	"54000"
CRT + sDVOB	"52000"
CRT + sDVOC	"53000"

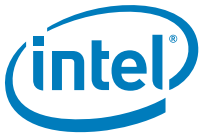


Table 30. Example of Chipset Dual Display Parameter Setting (Sheet 2 of 2)

Dual Display Combination	Port Order
Internal LVDS + CRT	"45000"
Internal LVDS + sDVOB	"42000"
Internal LVDS + sDVOC	"43000"
sDVOB + CRT	"25000"
sDVOB + Internal LVDS	"24000"
sDVOB + sDVOC	"23000"
sDVOC + CRT	"35000"
sDVOC + Internal LVDS	"34000"
sDVOC + sDVOB	"32000"

5.1.5 Creating Registry Settings for Graphics Driver INF File

The driver settings are configured using CED. It generates the following output, which is then inserted into the graphics driver INF file before driver installation. CED simply translates the configuration options to the INF file. See Table 24, "Parameter Configuration Format" on page 67 for details on the specific settings and values, which also apply to the settings and values of the INF file. The values of the INF file may also be directly modified. See the example below for syntax and usage. Also, see Appendix A, "Example INF File" for a complete sample INF file.

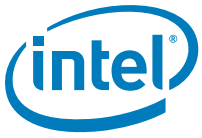
```
HKR,, PcfVersion,    %REG_DWORD%, 0x0700

HKR,, No_D3D, %REG_DWORD%, 0
HKR,, PortDrivers, %REG_SZ%, "lvds"

;-------------------------------------------------------------------------------
[iegd_SoftwareDeviceSettings_nap]
HKR,, InstalledDisplayDrivers, %REG_MULTI_SZ%, iegddis
HKR,, MultiFunctionSupported, %REG_MULTI_SZ%, 1
HKR,, VgaCompatible, %REG_DWORD%, 0
HKR,, PcfVersion,    %REG_DWORD%, 0x0700

HKR,, No_D3D, %REG_DWORD%, 0

HKR,, PortDrivers, %REG_SZ%, "sdvo lvds tv"

;-------------------------------------------------------------------------------
[iegd_SoftwareDeviceSettings_gn4]
HKR,, InstalledDisplayDrivers, %REG_MULTI_SZ%, iegddis
HKR,, MultiFunctionSupported, %REG_MULTI_SZ%, 1
HKR,, VgaCompatible, %REG_DWORD%, 0
HKR,, PcfVersion,    %REG_DWORD%, 0x0700

HKR,, No_D3D, %REG_DWORD%, 0

HKR,, PortDrivers, %REG_SZ%, "sdvo lvds"

;===============================================================================
[Strings]

;-----------------------------------------------------------------------
; Localizable Strings
;-----------------------------------------------------------------------
Intel="Intel Corporation"
DiskDesc="Embedded Installation"

i865="865 Embedded Graphics Controller"
```

```
i915GD0="915G/915GV/910GL Embedded Graphics Controller Function 0"
i915GD1="915G/915GV/910GL Embedded Graphics Controller Function 1"
i915AL0="915GM/915GMS/910GML Embedded Graphics Controller Function 0"
i915AL1="915GM/915GMS/910GML Embedded Graphics Controller Function 1"
i945LP0="945G Embedded Graphics Controller Function 0"
i945LP1="945G Embedded Graphics Controller Function 1"
i945CT0="945GM Embedded Graphics Controller Function 0"
i945CT1="945GM Embedded Graphics Controller Function 1"

i965BW0="965G Embedded Graphics Controller Function 0"
i965BW1="965G Embedded Graphics Controller Function 1"
iG9650="G965 Embedded Graphics Controller Function 0"
iG9651="G965 Embedded Graphics Controller Function 1"
iQ9650="Q963/Q965 Embedded Graphics Controller Function 0"
iQ9651="Q963/Q965 Embedded Graphics Controller Function 1"
i946GZ0="946GZ Embedded Graphics Controller Function 0"
i946GZ1="946GZ Embedded Graphics Controller Function 1"
i965GM0="GM965 Embedded Graphics Controller Function 0"
i965GM1="GM965 Embedded Graphics Controller Function 1"
;------------------------------------------------------------------------
; Non Localizable Strings
;------------------------------------------------------------------------
SERVICE_BOOT_START     = 0x0
SERVICE_SYSTEM_START   = 0x1
SERVICE_AUTO_START     = 0x2
SERVICE_DEMAND_START   = 0x3
SERVICE_DISABLED       = 0x4

SERVICE_KERNEL_DRIVER  = 0x1

SERVICE_ERROR_IGNORE   = 0x0; Continue on driver load fail
SERVICE_ERROR_NORMAL   = 0x1; Display warn, but continue
SERVICE_ERROR_SEVERE   = 0x2; Attempt LastKnownGood
SERVICE_ERROR_CRITICAL = 0x3; Attempt LastKnownGood, BugCheck

REG_EXPAND_SZ = 0x00020000
REG_MULTI_SZ  = 0x00010000
REG_DWORD     = 0x00010001
REG_SZ        = 0x00000000
```

5.1.6 Dynamic Port Driver Configuration

The IEGD supports many third-party digital transmitters connected to the sDVO ports of the GMCH though device drivers called port drivers. These port drivers are dynamically loaded at startup. The driver configuration can be modified to add or remove availability of specific port drivers.

This section describes the portions of the `iegd.inf` file that can be modified to either add or remove a port driver for the Microsoft Windows version of the Intel® Embedded Graphics Drivers.



5.1.6.1 iegd.PortDrvs_xxx

The first step in either adding or removing a port driver is to identify the family of the chipset you are using. 915 and 945 are Napa (nap), and 965 is Gen 4 (gn4). Next locate the appropriate [iegd.PortDrvs_xxx] section for your graphics family. Below are the default settings for the blocks of associated port drivers for a particular graphics chipset family:

```
[iegd.PortDrvs_nap]
sdvo.sys
lvds.sys
tv.sys

[iegd.PortDrvs_gn4]
sdvo.sys
lvds.sys
```

To remove one or more port drivers, delete the associated line from the `iegd.PortDrvs_xxx` block. To add a port driver, add the associated line into the appropriate `iegd.PortDrvs_xxx` block. For example, to add a new port driver for a device named "NewPD", add the following line to the `iegd.PortDrvs_alm` block:

```
NewPD.sys
```

5.1.6.2 SourceDisksFiles

The next step to either add or remove a port driver is to identify the specific port driver file names in the `SourceDisksFiles` blocks. The default settings are as follows:

```
[SourceDisksFiles]
iegdmini.sys = 1
iegddis.dll  = 1
iegd3dg3.dll= 1
iegd3dg4.dll= 1
lvds.sys     = 1
sdvo.sys     = 1
tv.sys       = 1
```

To remove a port driver, delete the associated line in the `[SourceDisksFiles]` block. To add a port driver, add the associated line to the block. For example, to add a port driver for a device whose driver is named `NewPD.sys`, add the following line:

```
NewPD.sys  = 1
```

5.1.6.3 PortDrivers Registry Key

The next step is to modify the registry key in the appropriate `[iegd_SoftwareDeviceSettings_xxx]` section that defines the list of available port drivers. Below are the default values of this registry key in the `iegd.inf` file:

For the `[iegd_SoftwareDeviceSettings_nap]` block:

```
HKR,, PortDrivers, %REG_SZ%, "sdvo lvds tv"
```

For the `[iegd_SoftwareDeviceSettings_gn4]` block:

```
HKR,, PortDrivers, %REG_SZ%, "sdvo lvds"
```

Remove or add port driver names as appropriate to the list of port drivers specified within the quoted string. For example, to add support for a new port driver named "NewPD", the registry key would be defined as follows:

```
HKR,, PortDrivers, %REG_SZ%, "lvds NewPD"
```

5.1.7 Creating an .sld file for Microsoft Windows XP Embedded Systems

Microsoft Windows XP Embedded* operating systems require the use of an `.sld` (system level definitions) file. The following steps detail how to create such a file for IEGD from your custom `iegd.inf` file that you created using CED.

1. Run Component Designer.
2. In the **File** menu, select **Import**.
3. In the **Choose File for Import** dialog, select **Setup Information files (*.inf)**. in the **File of type** drop-down list.
4. Select **iegd.inf** from installation directory.
5. In the **Inf Processing Options** dialog, select **Automatic** in the **Parsing Options** dialog and click **OK**.
6. Click **Start** in the **Import File** dialog box. Close the dialog on completion. There should not be any errors.
7. If there are no errors, **Save** the `.sld` file.
8. Run Component Database Manager and import the `.sld` file created above.

Note: Multiple versions will be created.

9. To move the binaries, copy the IEGD/Driver files into the root repository:
   `\Windows Embedded Data\Repository`
10. In Target Designer, all the IEGD drivers are found under `Hardware\Devices\Display Adapters` and can be selected by dragging and dropping into your build.

5.1.8 Changing Default Display Mode

When installing the Intel® Embedded Graphics Drivers, Microsoft Windows selects a default display mode for the initial startup of the system. This is a 640 x 480 resolution in four-bit color mode.

In some cases, particularly with EDID-less LVDS displays, the 640 x 480 resolution may not be supported, so the default mode selected by Microsoft Windows must be changed. Otherwise, the display may not work after installation.

This default mode can be changed by adding the following registry keys to the `[iegd_SoftwareDeviceSettings]` section of the `iegd.inf` file:

```
HKR,, DefaultSettings.XResolution, %REG_DWORD%, 1024

HKR,, DefaultSettings.YResolution, %REG_DWORD%, 768

HKR,, DefaultSettings.BitsPerPel,  %REG_DWORD%, 32

HKR,, DefaultSettings.VRefresh,    %REG_DWORD%, 60
```

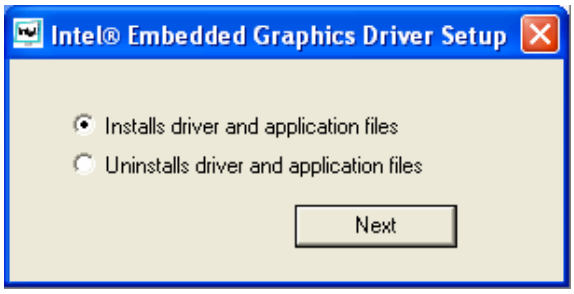
The example above makes the default resolution 1024 x 768, with a 32-bit color depth and a refresh rate of 60 MHz.

5.2 Installing the IEGD on Microsoft Windows

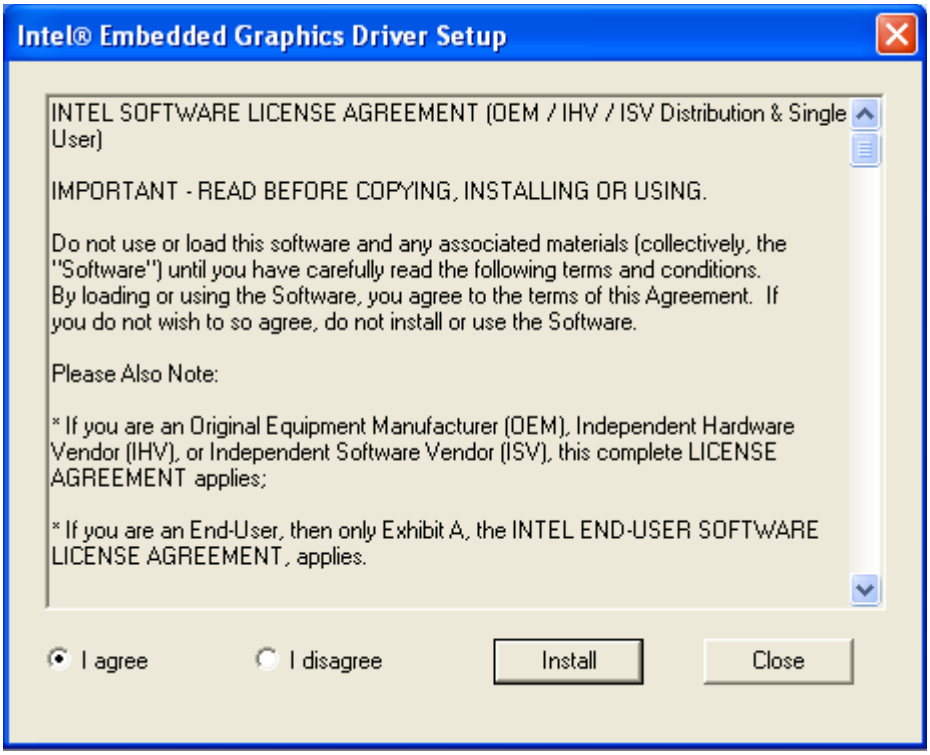
You can install and uninstall the IEGD on a Microsoft Windows system by using the `setup.exe` program located in the `Windows\Utilities` folder. The following procedure shows how to install the IEGD. Section 5.3, "Uninstalling the Current Version of the Driver" on page 111 provides instructions for uninstalling the current version of the IEGD.

Warning: If you have a previous version of the IEGD installed on your system, you must remove it using the Microsoft Windows **Add or Remove Programs** utility located in the **Control Panel**. Do not use the current version of the IEGD Install program to uninstall previous versions of the driver. If you do, unpredictable results may occur. You can use this program only to uninstall the driver from the current version. Each version of the driver has its own version of the installer/uninstaller utility.

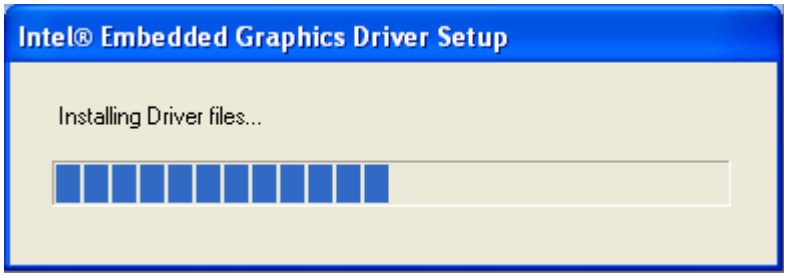
1. Double-click the `setup.exe` icon in the `Utilities` folder. The following dialog box appears.



2. To install the driver, make sure that **Installs driver and application files** is selected, and then click **Next**. The accept license screen appears.

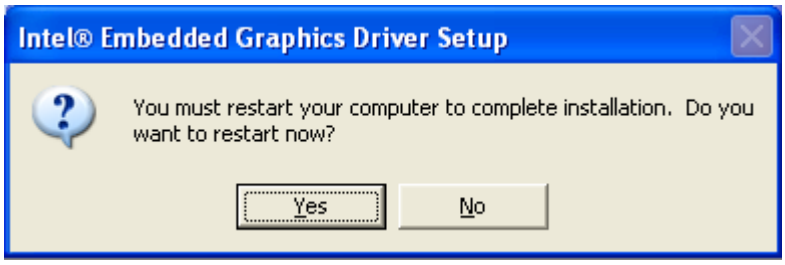


3. Click **I agree**, and then click **Install**. The installation begins and shows a progress bar as follows:



Note: If you get an "unsigned driver" warning, disregard and click **Continue** to allow the installation to continue.

4. After the driver and application files have been copied, the system must be restarted to complete the installation. If you want the installation program to restart your computer, click **Yes**.



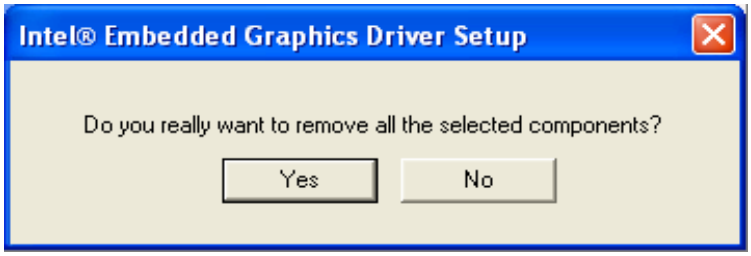
5.3 Uninstalling the Current Version of the Driver

You can use the `setup.exe` Microsoft Windows GUI program to remove the driver from your system. When you run the uninstaller program, it removes the following items from the system:

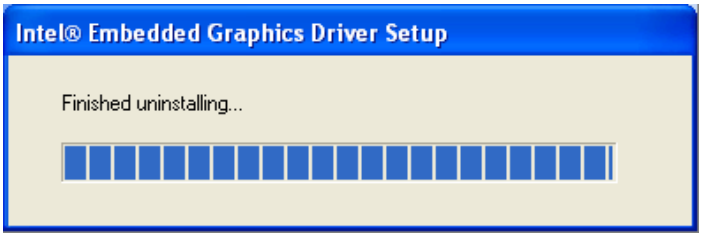
- The IEGD drivers
- The `.inf` and `.pnf` files from the `windows\system32\inf` folder.
- The `DisplayPage.dll` and `qt-mt332.dll` from the `windows\system32` folder
- Data registry items by running `regsvr32.exe` with the uninstall option.

Warning: You cannot use the **Add or Remove Programs** utility in the **Control Panel** to remove previous versions of the driver (each version of the driver has its own installer/uninstaller utility). To uninstall previous versions, click **Start ->Control Panel ->System ->Hardware ->Device Manager**, expand **Display adapters** then right-click the specific display adapter to be uninstalled, mouse-over and click **Uninstall**. Repeat if there is more than one previous version of the driver loaded.

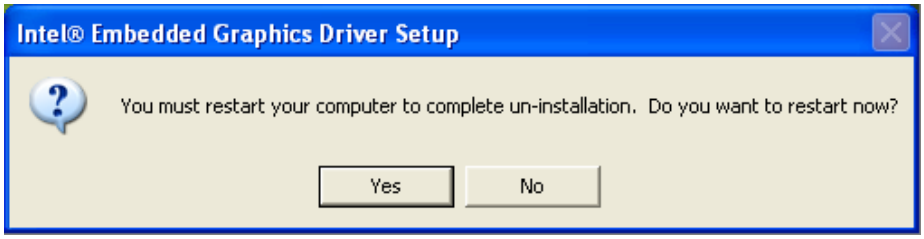
1. Click the `setup.exe` icon located in the `Utilities` subfolder of the `Windows` folder.
2. In the dialog box, select **Uninstalls driver and application files**, and then click **Next**. The following prompt appears:



3. Click **Yes** to remove the driver. A progress bar displays and when the driver has been removed, the following screen appears.



4. To complete the uninstallation, you must restart your system. If you want to restart your system now, click **Yes** in the following dialog box.



5.4 Run-Time Operation

Resolution, refresh rate, and color bit depth can be changed after installation and reboot via a Microsoft Windows display property sheet. On Microsoft Windows XP, extended desktop can be enabled and disabled, along with swapping primary and secondary displays. Other operations such as enabling and disabling ports (display output), rotation, port configuration, and attribute control are accessible via the standard display driver escape protocol.

5.5 Viewing and Changing the Driver Configuration from Microsoft Windows

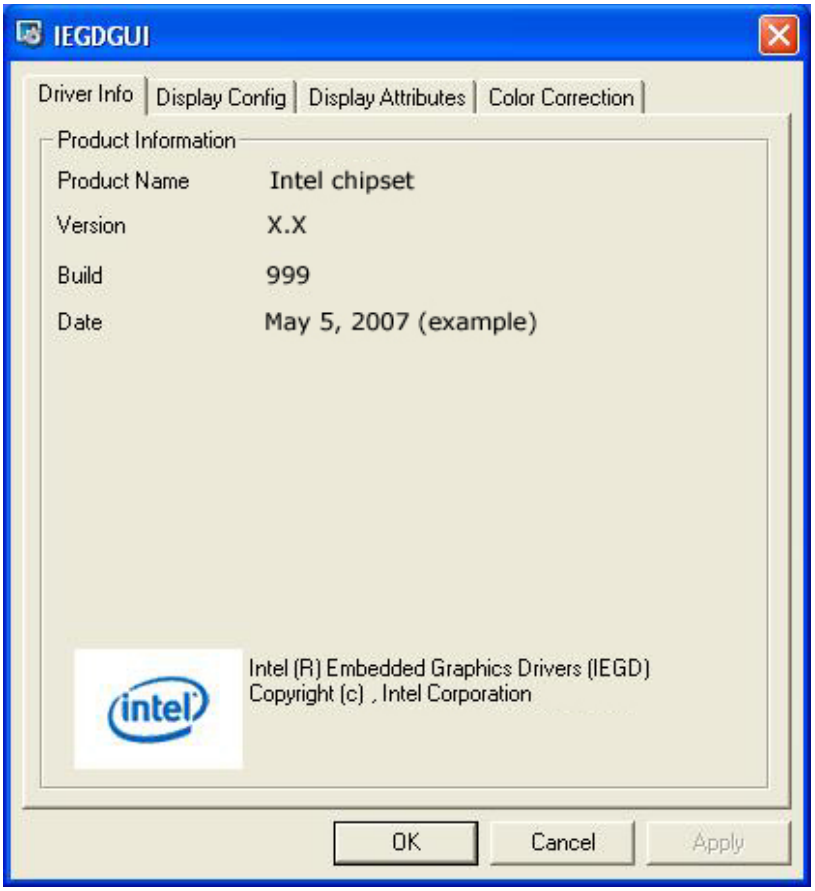
You can change certain configuration attributes of the IEGD using the `iegdgui.exe` program located in the `\Utilities` folder. On Microsoft Windows XP systems, you can access the IEGD Configuration tabs through the **Advanced Settings** tab of the **Display Properties** icon from the Windows dialog box. This program launches the IEGD Configuration GUI that consists of the following four tabs:

- **Driver Info** — Contains the driver information.
- **Display Config** — Contains current display information and allows configuration of display configurations, display resolutions and bit depth for primary and secondary displays, flip, rotation, and enabling/disabling for a given port.
- **Display Attributes** — Contains the supported Port Driver (PD) attributes and allows configuration of PD attributes.
- **Color Correction** — Contains color-correction information for the framebuffer and overlay. Using this tab, you can change the framebuffer and overlay color settings.

To view or change the driver settings using the GUI interface, follow this procedure.

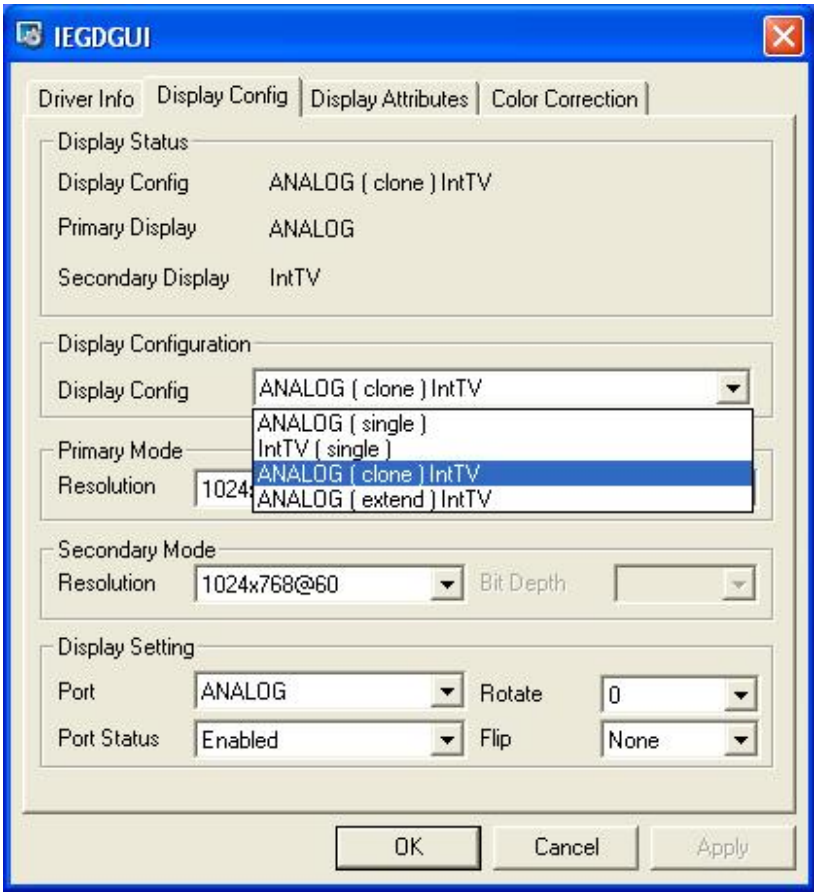
1. Double-click the `iegdgui.exe` icon in the `Utilities` folder. On Microsoft Windows XP systems, you can click the **Advanced Settings** tab in the **Display Properties** icon from the dialog box. The IEGD Configuration GUI Driver Info tab appears, showing information about the driver.

Figure 31. Example Runtime Configuration GUI — Driver Info Tab



2. Click the **Display Config** tab to show the current configuration.

Figure 32. Example Runtime Configuration GUI — Display Config Tab



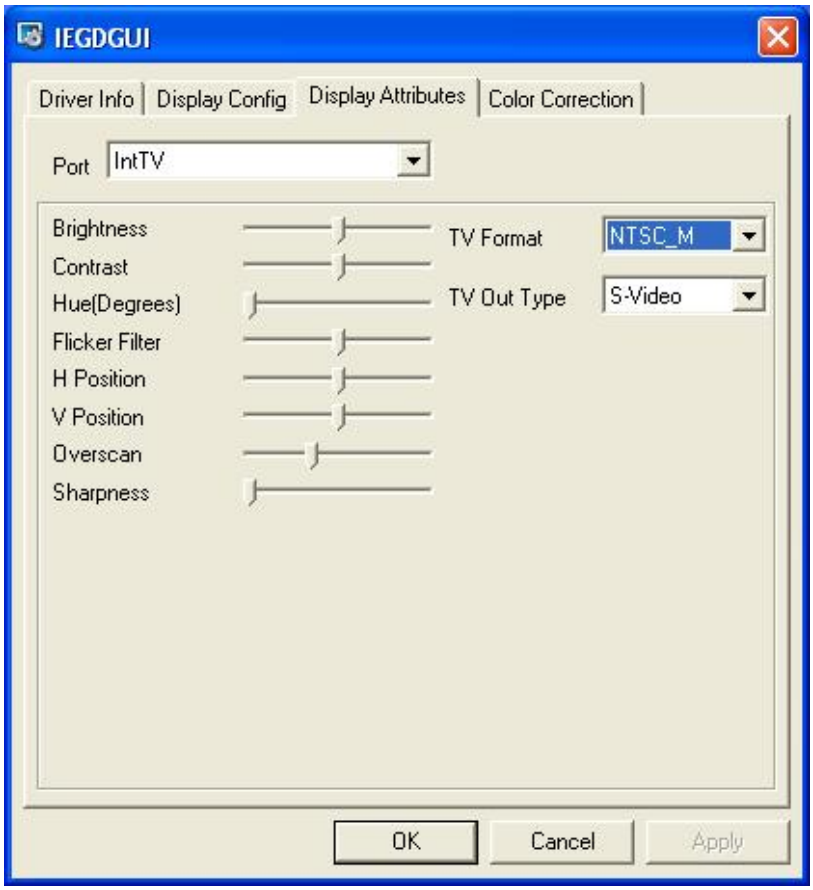
The **Display Status** section of the above dialog shows the current configuration for the **Primary** and **Secondary** displays.

3. In the **Display Configuration** section of the dialog, select the required display configuration in the **Display Config** drop-down list. This allows the user to choose between Single, Twin, Clone and Extended for all connected ports. A maximum of two ports per display configuration is currently allowed.
4. In the **Primary Mode** and **Secondary Mode** sections of the dialog, change resolution and bit depth of the primary and secondary displays via the **Resolution** and **Bit Depth** drop-down lists.
5. In the **Display Settings** section of the dialog, view and change the settings for a port, rotate and flip the display via the appropriate drop-down lists:
   - **Port**: Allows you to select the required port.
   - **Port Status**: Allows you to enable or disable the selected port.
   - **Rotate**: You can rotate the display 0, 90, 180, and 270 degrees.
   - **Flip**: Inverts the display horizontally.

Note: If you change any configuration settings in the **Display Config** dialog box, click **Apply** for the changes to take effect.

6. Click the **Display Attributes** tab to view and change the attributes for a port. The screen that appears depends upon the port drivers used.

Figure 33. Example Runtime Configuration GUI — Display Attributes Tab



The figure Figure 33 shows the attributes that can be changed for the selected port in the **Port** drop-down list. You can change the Port Driver by selecting the appropriate one for your device. The attributes that appear on this tab depend upon the selected port driver. Please see Appendix B, "Port Driver Attributes," for a complete list of port driver attributes.

7. Click the **Color Correction** tab to view and change color corrections. The figure Figure 34 shows a sample Color Correction tab screen.

   Color Correction is available for both overlays and framebuffers, and is accessed under the New Configuration link at the top of the main CED window.

Figure 34. Example Runtime Configuration GUI — Color Correction Tab

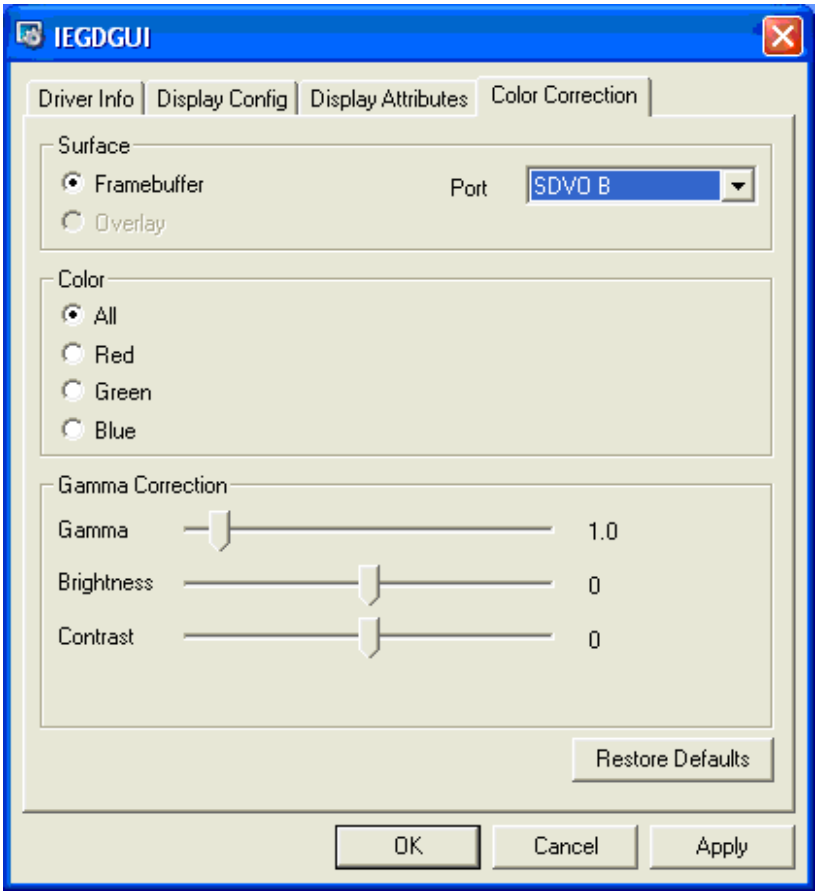
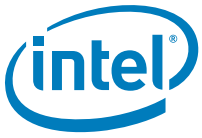


Table 31. Framebuffer Color Correction Values (applies to R, G, B color)

Gamma:	0.6 to 6.0 (default value is 1)
Brightness:	-127 to 127 (default value is 0)
Contrast:	-127 to 127 (default value is 0)

Table 32. Overlay Color Correction Values (applies to ALL color)

Gamma:	0.6 to 6.0 (default value is 1)
Brightness:	0 to 200 (default value is 100)
Contrast:	0 to 200 (default value is 100)
Saturation:	0 to 200 (default value is 100)



The following sub-steps present an example color-correction procedure:

a. Select **Framebuffer** in the **Surface** section and select the appropriate port for the color correction to be applied to or select **Overlay** in the Surface section for color correction to be applied to the overlay.

b. Select the required color to be corrected in the **Color** section.

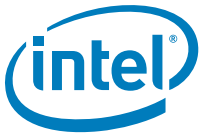
c. Select the required color attribute to be corrected in the **Gamma Correction** section.

d. Click **Restore Defaults** to restore the default values.

Note: If you make any changes to the color-correction settings, click **Apply** to have the changes take effect.

Note: The hardware does not support brightness, saturation, and contrast of the overlay and second overlay with RGB pixel format.

This page is intentionally left blank.



6.0 Configuring and Building the IEGD for Microsoft Windows CE Systems

The Microsoft Windows CE* drivers are configured and built from the options provided on the General Settings Page (see Creating a New Configuration) and the Microsoft Windows CE Options Page (see Creating a New Package). After you configure the IEGD for a Microsoft Windows CE system, package the IEGD and generate an installation. The CED produces an `iegd.reg` (see Sample iegd.reg File) file and a IEGD_X_x_WINCEXX.zip file (where _X is the major version, _x is the minor version, and XX is either 5_0 or 6_0) that you use to build an image for a Microsoft Windows CE system using the Microsoft Windows CE Platform Builder.

To build an IEGD image for a Microsoft Windows CE system, do the following:

1. Enter IEGD configuration settings using the CED. (Please see Creating a New Package and Creating a New Package.)
2. Package the configuration. (See Creating a New Package.)
3. Generate an installation using the `Generate Installation` option on the CED main window (see Generating an Installation). This produces an `iegd.reg` file and an `IEGD_5_0_WINCEXX.zip` file. The `iegd.reg` file contains registry entries and the `IEGD_X_x_WINCEXX.zip` file contains required driver files.
4. Integrate the `iegd.reg` file with the Microsoft Windows CE Platform Builder. Please see Integrating IEGD with Microsoft Windows CE Platform Builder.

See also the following topics:

- Microsoft Windows CE Installation
- Microsoft Windows CE Configuration

6.1 Microsoft Windows CE Installation

The following sections describe how to install the IEGD on the Microsoft Windows CE 5.0 and 6.0 operating systems.

6.1.1 Prerequisites

The development system should have the following software installed:

- Microsoft Windows XP Professional, SP2
- Platform Builder for Microsoft Windows CE 5.0 or 6.0 (with latest service packs)

The target system must contain one of the following Intel chipsets:

- Intel® GM45 Express chipset
- Intel® System Controller Hub US15W chipset
- Intel® Q35 Express chipset
- Mobile Intel® GME965 Express chipset
- Intel® Q965 Express chipset
  Mobile Intel® 945GSE Express chipset
- Mobile Intel® 945GME Express chipset
- Intel® 945G Express chipset
- Intel® 915GV Express chipset
- Mobile Intel® 915GME Express chipset
- Mobile Intel® 910GMLE Express chipset

The target system must contain a minimum of 64 Mbytes of RAM.

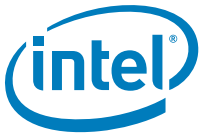
6.1.2 Integrating IEGD with Microsoft Windows CE Platform Builder

The integration/installation of the driver binaries depends upon the requirements of the target device; while `ddi_iegd.dll` is required, port drivers may be optionally included.

Note: Windows CE 6.0 does not employ the catalog file mentioned below. For Windows CE 5.0, to integrate the driver binaries into the Platform Builder, the catalog file included with the release should first be imported to the Platform Builder's catalog (for details about how this is done, see Catalog Feature File). After the catalog file is imported, find "Intel Embedded Graphics Driver" under Third Party -> Device Drivers -> Display and drag-and-drop it into the BSP.

In order for the Platform Builder to use IEGD, the `iegd.reg` file included with the release has to be properly included into the BSP. For Windows CE 5.0, this means adding the following lines into the `platform.reg` file. Note that you must specify the correct path to the `iegd.reg` file.

```
;*******************************************************
;* IEGD Section for WinCE 5.0 Platform Builder
;*******************************************************
IF BSP_DISPLAY_IEGD

; @CESYSGEN IF CE_MODULES_PCI

; GWES will be able to auto-detect the display adapter if a candidate value
; points to the adapter's PCI instance. Values from Candidate1 to Candidate32
; (decimal) are possible; GWES will examine them sequentially until it finds
; a match.
[HKEY_LOCAL_MACHINE\System\GDI\DisplayCandidates]
    "Candidate6"="Drivers\\Display\\Intel"
```



```
[$(PCI_BUS_ROOT)\Template\IEGD]
"DisplayDll"="ddi_iegd.dll"
"Class"=dword:03
"SubClass"=dword:00
"ProgIF"=dword:00

"VendorID"=multi_sz:"8086", "8086", "8086", "8086", "8086", "8086", "8086", "8086",
"8086", "8086", "8086", "8086", "8086", "8086", "8086", "8086", "8086", "8086",
"8086", "8086", "8086", "8086", "8086", "8086"
"DeviceID"=multi_sz:"3582", "2572", "2562", "357B", "3577", "1132", "7125", "7123",
"7121", "2582", "2782", "2592", "2792", "2772", "2776", "27A2", "27A6", "2982",
"2983", "29A2", "29A3", "2992", "2993", "2972", "2973"
; include the path to the iegd.reg file in the release package
#include <specify_path_here>\iegd.reg

; @CESYSGEN ENDIF CE_MODULES_PCI
ENDIF ; BSP_DISPLAY_IEGD
```

Finally, to include the actual driver binaries into the OS image, you must reference them in the BSP's BIB file by appending the path to ddi_iegd.dll and the port drivers into platform.bib, as shown below.

Figure 35. Sample FILES Block from `platform.bib` File

```
FILES
;       Name                            Path                                Memory Type
;       ------------------------        -------------------------           -----------
;       @CESYSGEN ENDIF CE_MODULES_DEVICE
;       @CESYSGEN IF CE_MODULES_DEVICE


        ddi_iegd.dll                    <specify_path_here>\ddi_iegd.dll    NK
```

6.1.2.1 Catalog Feature File

For Windows CE 5.0, IEGD's Catalog Feature File, `iegd.cec`, is provided in the release package. To import IEGD into the workspace's catalog, complete the following steps:

1. From the File menu, select **Manage Catalog Features**.
2. Choose **Import**.
3. In the **Import Catalog Features** dialog box, select the `.cec` file, and then click **Open**.
4. From the **View** menu, select **Catalog** to display the Catalog.

6.2 Microsoft Windows CE Configuration

The following sections describe how to configure the IEGD on the Microsoft Windows CE operating system. All the IEGD-specific registry keys are located within the path `[HKEY_LOCAL_MACHINE\DRIVERS\Display\Intel]`

All keys use one of the following syntax:

`"<keyname>"=dword:<value>,`

or

`"<keyname>" = <value>`

where <value> in the second case is a string in double quotes.

Note: Unless specified otherwise, the "value" field is in hex format.

The `iegd.reg` file contains display configuration registry entries for the IEGD. A sample `iegd.reg` file is provided along with the driver package. The content of this file may be included through the "#include" directive in `platform.reg` (see Integrating IEGD with Microsoft Windows CE Platform Builder), or it may be copied into the proper section in `platform.reg`.

6.2.1 Basic Driver Configuration

This section discusses basic driver configuration keys located in `[HKEY_LOCAL_MACHINE\Drivers\Display\Intel\ALL\1\General]`.

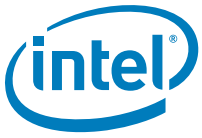
The table below lists the keys in the "`Intel`" folder.

Table 33. [HKLM\DRIVERS\Display\Intel] Registry Keys

Registry Entry	Description	Possible Ranges
PCFversion	Specifies the version of the current configuration file.	400 or 700
ConfigId	This selects the configuration set.	1, 2, 3, 4, or 5
PortDrivers	List of port drivers to be dynamically loaded when the system boots. The dll's must exist in the `c:\Windows` directory. sDVO transmitter port drivers to load when the system boots.	Space separated string enclosed in quotes, where each port driver name is listed in the string. The default string included with the release has all supported port drivers.

See also the following topics:

- Graphics Memory Configuration
- Defining Graphics Memory Size
- Framebuffer and Video Surface Size
- Video Surface Allocation Rule
- System to Video Stretch Blit
- iegd.reg File Backward Compatibility
- Blend Filtering



6.2.1.1 Graphics Memory Configuration

The Intel Embedded Graphics Suite (IEGS = VBIOS + Graphics driver) provides the ability to dedicate additional memory for graphics functions on the Microsoft Windows CE platform. This is known as *reserved memory*. The amount of reserved memory is selected by firmware. The reservation size is passed to the graphics driver through a scratch register available on the GMCH. Reserved memory is useful in minimizing the amount of memory stolen from the OS for memory-limited, embedded systems. For instance, if firmware utilizes a 640 x 480, 32-bit framebuffer, a total of 1.2 Mbytes is required. Stolen memory would need to be configured as 8 Mbytes or higher, since the next smaller option is only 1 Mbyte, too small for the 640 x 480, 32-bit framebuffer. In such a case, stolen memory can be programmed to 1 Mbyte. The additional memory required for the framebuffer can then be provided by reserved memory, allowing a minimum amount of memory to be removed from the OS.

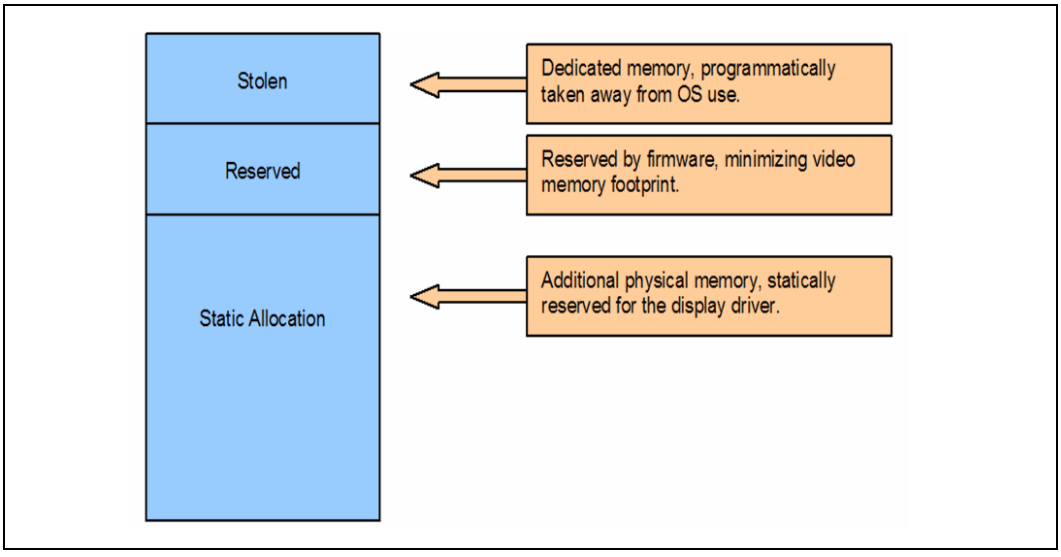
Note: Reserved memory is only available on the Microsoft Windows CE operating system, and must be accounted for in the `config.bib` memory layout file.

Additionally, the Microsoft Windows CE display driver can be configured for either static or dynamic allocation of video memory. The static model preallocates physical memory for the display driver and provides a more efficient surface allocation scheme. The dynamic model allocates surface memory on demand from the system and will incur a small performance hit. However, the dynamic model has the advantage of deallocation of video memory when not required, thus making it available to other applications.

The static memory model requires a base and size specification registered in the `project.reg` file. The base + size must reach to top of memory (TOM). Since this is not required to be specified in the config.bib memory map, care must be taken not to overlap any other memory arenas with the static allocation. See page 124 for further details on how to configure the static memory model.

The figure Typical Memory Map Using Static Memory Model shows a typical memory map, using a static memory model.

Figure 36. Typical Memory Map Using Static Memory Model



6.2.1.2 Defining Graphics Memory Size

The IEGD supports static as well as dynamic location of graphics memory. For static video memory, the two registry settings ReservedMemoryBase and ReservedMemorySize denote where and how big this area will be. This portion of memory will be taken care off by the built-in GART driver. The size should include the stolen memory (BIOS setting, if applicable).

If either of these registry settings are zero or not defined in the registry, then dynamic memory allocation is used for video memory management. The usage will be dependent on system resources. This is in line with the Dynamic Video Memory Technology.

As an example of static video memory, if the stolen memory is 8 Mbytes and you want a total of extra 56 Mbytes of graphics memory for a total of 64 Mbytes of graphics memory you would want to have these settings: (On a 128 Mbyte machine)

```
[HKEY_LOCAL_MACHINE\Drivers\Display\Intel\ALL\1\General]

      "ReservedMemoryBase"=dword:4000000

      "ReservedMemorySize"=dword:4000000
```

Which means that the managed graphics memory pool will begin at physical address 0x4000000 (64 Mbytes) and is 64 Mbytes of size.

Note: Always remember to include the amount of stolen memory in this number.

The lower 64 Mbytes in the above machine will be used for the NK.bin and RAM, so therefore you must change your `config.bib` accordingly. The configuration for the given machine is (debug build):

```
NK          80220000  01650000  RAMIMAGE
RAM         81700000  02790000  RAM
```

In this configuration, the NK.BIN image takes 22.3125 Mbytes and the rest of 41.6875 Mbytes is RAM. The release build should have much smaller NK area. For better control, users may turn AUTOSIZE=OFF to calculate exactly where the image and RAM starts and ends followed by the location of the video memory pool with no possibility of runtime deviation.

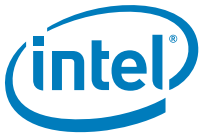
These two settings, together with the registry settings mentioned above, will determine the memory layout.

The example `config.bib` and the memory configuration settings in the `iegd.reg` are validated settings that can be used for static video memory of 16 Mbytes in size starting from the 48 Mbytes address, thus including the stolen memory.

6.2.1.3 Framebuffer and Video Surface Size

Two additional optional registry settings are available to limit the framebuffer size of the display driver and the total size of offscreen video surfaces.

The `MaxFbSize` registry entry will control the maximum size of the framebuffer only. Actual usage will depend on the mode being used.



The `PageReqLimit` registry entry will control the total size in pages (4 Kbytes) of all video surfaces, buffers allocated for any use. Both of these registry configurations apply to both the static as well as dynamic video memory management explained in the previous section. The default below indicates that a maximum of 2 Mbytes are used for the framebuffer and a maximum of 16 Mbytes are permitted for all offscreen videosurface allocations.

```
"MaxFbSize"=dword:200000
"PageReqLimit"=dword:1000
```

In the case of Microsoft Windows CE, because the OS does not allow for dynamically setting the framebuffer size, the `MaxFbSize` can be changed to match the mode setting being used in order to minimize on video memory wastage. The following are different suggested values for `MaxFbSize` for different display modes. These values have not been validated. Note that 640x480 is calculated as 640x512 and 800x600 is calculated as 800x768 for stride alignment purposes.

640x512X16 = A0000
640x512X24 = F0000
640x512X32 = 140000
800x768X16 = 12C000
800x768X24 = 1C2000
800x768X32 = 258000
1024x768X16 = 180000
1024x768X24 = 240000
1024x768X32 = 300000

6.2.1.4 Video Surface Allocation Rule

Another two optional registries entries determine a minimum width and height dimension that allows video surface allocations to succeed.

In the Microsoft Windows CE GDI, video surface allocations can happen with a REQUIRE_VIDEO_MEMORY or a PREFER_VIDEO_MEMORY flag. The following options force surface allocations with the PREFER_VIDEO_MEMORY flag to be allocated in system memory if the width and height are lower then what is stated in the following entries.

“MinVidSurfX”=dword:10
“MinVidSurfY”=dword:10

In this example, surfaces allocated with the PREFER_VIDEO_MEMORY where the width and height are both less than 16 pixels are forced to be in system memory.

6.2.1.5 System to Video Stretch Blit

System to Video Memory stretch blits are not natively supported on Intel GMCH devices. This feature allows you to enable a soft copy of system surfaces to video surfaces in order to conduct an accelerated stretch blit. The advantage of this is that the stretch blit then utilizes the blend engine and hardware filtering can be applied. The filtering options are listed in Blend Filtering.

A value of 1 for the “SysToVidStretch” enables system to video stretch blits, as described above, while a value of 0, disables this feature and forwards all system to video stretch blits to the emulator provided by the operating system.

```
[HKEY_LOCAL_MACHINE\Drivers\Display\Intel\ALL\1\General]
"SysToVidStretch"=dword:0
```

6.2.1.6 iegd.reg File Backward Compatibility

The Intel Embedded Graphics Driver expects a configuration file in the PCFVersion 700 format. However, the driver will maintain backward support with version 4.0. This support is implemented through the `PcfVersion` key as shown below:

```
[HKEY_LOCAL_MACHINE\Drivers\Display\Intel\ALL\1\General]
"PcfVersion"=dword:400
```

The IEGD uses this key to determine the format of the configuration file. When this key is present, IEGD parses the configuration file using the format specified by the key (400 or 700). If this key is not present, then IEGD assumes 4.0 format.

6.2.1.7 Blend Filtering

The blend filtering method can be selected via the `BlendFilter` registry setting. The filter method chosen will be used for all stretch blit operations, including video to video stretch blits, as well as system to video stretch blits. The default filtering method is `Nearest`.

Available Blend Filter options:

```
0 == Default == Nearest
1 == Binlinear
2 == Anisotropic
3 == 4x4
```

```
[HKEY_LOCAL_MACHINE\Drivers\Display\Intel\ALL\1\General]
"BlendFilter"=dword:2
;  Blend filter to use for all stretch blits
;  BlendFilter 0 == NEAREST
;  BlendFilter 1 == BILINEAR
;  BlendFilter 2 == ANISOTROPIC
;  BlendFIlter 3 == 4X4
```

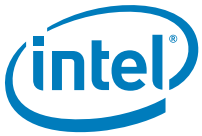
6.2.2 Configuration Sets

The Intel® Embedded Graphics Drivers allows multiple configuration sets for OEMs who wish to use the same iegd.reg file across different platforms. There can be up to 16 instances of configurations. The registry key described in the previous section, ConfigId, ensures the display driver selects the right instance. Each instance may contain multiple groups of per-config and per-config+per-port platform customizations.

The configuration sets are defined in the registry tree as

```
[HKEY_LOCAL_MACHINE\Drivers\Display\Intel\<platform>\<config id>],
```

Where <config id> is the configuration number. The "ConfigID" key described in the previous section selects the active configuration set.



6.2.3 General Configuration

Registry keys described in this section can be found in [HKEY_LOCAL_MACHINE\Drivers\Display\Intel\<platform>\<config id>\], where 'config id' is the configuration number, and where <platform> is one of the following: ALL, US15W, GM45, Q35, GME965, Q965, 945G, 945GME, 945GSE, 915GV, 915GME, and 910GMLE. The driver first attempts to find the configuration or platform on which it is booted, but if the configuration for that platform is not present, the driver uses the ALL platform setting.

Table 34. [HKLM\Drivers\Display\Intel\<platform>\<config id>\]Registry Keys (Sheet 1 of 2)

Registry Entry	Description	Possible Ranges
Width	Width of the display	Width and Height must be expressed as hexidecimal values. For example: 1024 x 768: 400 x 300 800 x 600: 320h x 258 640 x 480: 280 x IE0
Height	Height of the display	See above.
Depth	Color depth in bpp (bits per pixel)	Depth must be expressed as a hexidecimal number and must be one of the following values: 8bpp: 8 16bpp: 10 24bpp: 18 32bpp: 20 (Note that the Intel 915 chipsets do not support 24 bpp.)
Refresh	The refresh rate of the display.	Refresh rate must be in hex: 60 : 3c 70 : 46 75 : 4b 85 : 55 etc... This value can be any valid refresh rate as long as the display port supports it. A refresh of '0' takes the first refresh that matches `width`, `height` and `depth`.
NO_D3D	Specify whether to enable D3D. ***Note:*** For Windows CE 5.0, this registry entry must be set to 1 as IEGD currently does not support D3D Mobile on Windows CE 5.0 systems.	0 = Enable D3D 1 = Disable D3D Default is 0.
ReservedMemoryBase ReservedMemorySize	Video memory can be statically reserved or dynamically allocated on demand. If both *ReservedMemoryBase* and *ReservedMemorySize* are non-zero, then Video memory allocation utilizes the static model.	The ReservedMemoryBase plus the ReservedMemorySize must extend to the TOM (Top Of Memory) and not conflict with other reserved memory arenas in config.bib. Default for both base and size is zero, indicating a dynamic allocation model. Default behavior disables static memory model.
MaxFbSize	Maximum size of the expected framebuffer. By providing this hint, the display driver can more efficiently organize GART memory, leading to a smaller video memory consumption.	Must be greater than or equal to the expected size of framebuffer. Units are in bytes. Specifying zero causes the default framebuffer reservation sizing. Default: All other chipsets: 16 Mbytes

Table 34. [HKLM\Drivers\Display\Intel\<platform>\<config id>\]Registry Keys (Sheet 2 of 2)

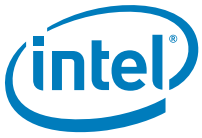
Registry Entry	Description	Possible Ranges
MinVidSurfX MinVidSurfY	In pixels, the minimum width and height of surfaces in order to be acceptable for allocation in Video memory. Due to hardware restrictions that optimize memory access, it is advisable to reserve video memory for larger surfaces and allow GDI and DirectDraw* to allocate small surfaces from system memory.	No limitations. Suggested values for both width and height are 10. Default value for both width and height is 1. Default: MinVidSurfX = 1 MinVidSurfY = 1
SysToVidStretch	Enables system-to-video memory stretch blit operations to take advantage of hardware-accelerated filtering. Normally, it is more efficient to allow GDI to conduct system-to-video stretch blits, but the default filtering used by GDI is Nearest. See the *BlendFilter* key below for hardware accelerated filtering options.	0 = Disabled 1 = Enabled Default: 0
BlendFilter	Provides selection of hardware-accelerated filtering methods for stretch blit operations.	0 = Nearest 1 = Bilinear 2 = Anisotropic Default: 0
TearFB	If enabled, all blit operations to the framebuffer are synchronized with video sync to eliminate any visible tearing or flickering on the display screen. Disabling this feature achieves a performance gain.	0 = Disabled, tearing allowed 1 = Enabled, no visible tearing Default: 0
DisplayConfig	The "DisplayConfig" key sets the display configuration to be in Single, Twin, Clone, or Vertical Extended modes. (Unlike Microsoft Windows XP, Microsoft Windows CE does not support Extended mode). Does not, however, dictate what type of display ports will be used.	1 (single), 2 (clone), 4 (twin), 5 (vertical extended)
DisplayDetect	The "DisplayDetect" key allows the user to enable a display port only if a display device is connected. Displays without EDID will not be detected.	0 = disable 1 = enable Default: 0
PortOrder	The PortOrder setting ensures the correct display port types are used based on user selection.	See PortOrder Information.

6.2.3.1 PortOrder Information

PortOrder specifies the actual ports that are used for the Primary and Secondary display. As shown in the table below, the port numbers are slightly different among the supported chipsets.

Table 35. PortOrder Information

Port Number	Chipsets
1	Integrated TV Encoder
2	sDVO B Port/RGBA Port
3	sDVO C Port
4	Internal LVDS Port
5	Analog Port



The driver attempts to use the ports in the order specified by "PortOrder". For example, "PortOrder" = "5420" will assign the analog port to the primary display and the LVDS port to the secondary display (if any), assuming all the ports are present and detected. Suppose port "5" is not present, in that case the driver tries to assign the next port (4, in this case) in line to the primary display, resulting LVDS port for primary and sDVO B port for secondary.

Setting PortOrder to "00000" causes the driver to use default internal settings.

```
*************
[HKEY_LOCAL_MACHINE\Drivers\Display\Intel\ALL\1\General]
;-------------------------------------
; Select Port Order
;-------------------------------------
"PortOrder"="54320"

;  PortOrder specifies the actual of port
;  that will be taken for the Primary /
;  Secondary ports if there are duplicates
;  of the same type. For example, if both
;  Primary and Secondary are digital, then
;  port order will which sDVO ports will be
;  first and second. The section below gives
;  the port order numbers for various chipsets.
;  Specify value "0000" to use default settings.
;  On i915 chipsets:
;  ================================
;    1 - Integrated TV Encoder
;    2 - sDVO B port/RGBA port
;    3 - sDVO C port
;    4 - Internal LVDS port
;    5 - Analog port
;
```

6.2.3.2 Vertical Extended Mode

The Windows CE IEGD driver supports Vertical Extended display mode, which is one large framebuffer that extends across two displays by doubling the height of resolution. The top half of the framebuffer is on the first pipe and the bottom half is on the second pipe. The Windows CE operating system is unaware of the two displays. This feature is supported only on the dual-pipelined chipsets, which is every supported platform stated in Prerequisites.

This feature is enabled through the `DisplayConfig` key in the `project.reg` file. The resolution, bit depth, and refresh rates of both displays must be the same. Vertical and horizontal panning are *not* supported. DirectDraw is supported on both pipes, but DirectDraw 3D must be disabled when Vertical Extended Display mode is enabled.

6.2.4 Per Port Platform Customization

The Intel Embedded Graphics Drivers provide what is considered the most useful tools to the embedded market — per port platform customizations. This includes the following:

- Defining custom DTD panel timings
  : `PixelClock, HorzActive, HorzSync` etc...
- Customized GPIO pin selection for I^2C and DDC communication with sDVO encoders and panels.
  :`I2cPin, I2cDab, I2cSpeed` etc...
- Flat Panel width and height limitations and power and/or backlight control mechanisms
  :`BkltMethod, BkltT1, BkltT2, GpioPinVdd` etc...
- Port driver specific attribute settings for initialization at boot time.
  : `Brightness, Contrast, H-Position` etc...

All of the above can be set for each individual port depending on the maximum number of ports the chipset supports. Also, you can have multiple instances of these configurations to allow different settings per configuration.

The usage model for this per-config, per-port platform customizations follows after the same options available in the INF registry settings for the Intel Embedded Graphics Drivers for Microsoft Windows XP. Please see Sample iegd.reg File or to the provided registry sample file in the IEGD Windows CE driver package for examples. The following sections provide information on these configurations:

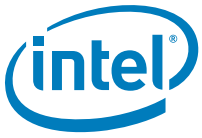
- Per Port Customization – General Port Configuration
- Per Port Customization – Custom DTD Timings
- Per Port Customization — Custom Flat Panel Controls
- Per Port Customization — Attribute Initialization

6.2.4.1 Per Port Customization – General Port Configuration

This section describes port-specific general configuration options. These options are located under
`[HKEY_LOCAL_MACHINE\Drivers\Display\Intel\ALL\1\Port\1\General]`

- Edid
  This boolean key enables (set to 1) or disables (set to 0) the EdidAvail and EdidNotAvail keys.
- EdidAvail and EdidNotAvail
  These two 16-bit keys control the available timings for the display. If an EDID is successfully read from the display device, then IEGD uses the EdidAvail flag to determine what timings are available. Otherwise, if an EDID cannot be read, then IEGD uses the EdidNotAvail key.

Bit #	Value (0 or 1)
0	Disable/Enable driver built-in timings
1	Disable/Enable EDID timings. (Only valid for the EdidAvail flag)
2	Disable/Enable DTD
3-15	Reserved



- CenterOff
  If the selected frame buffer size is smaller than what the IEGD hardware can support, by default the frame buffer will be centered with a black border around it. To explicitly turn off this feature, the user may set the "CenterOff" key to "1".
- Rotation and Flip
  IEGD supports desktop rotation through the "Rotation" key in Single, Twin, and Clone mode. Rotation is not supported in Vertical Extended Mode.
  The "Rotation" key can be set to one of the four follow values.

Degree	Key Value
0	0 (default)
90	5A
180	B4
270	10E

  So, "Rotation"=dword:5A will rotate the frame buffer 90 degrees.
  The "Flip" key flips the desktop horizontally, displaying a mirror image. "Flip" is a boolean value: 1 to enable, 0 to disable.
- Scale
  IEGD can scale the desktop to the output panel using the panel's DTD or EDID (in that order). "Scale" is a boolean value: 1 to enable, 0 to disable.

6.2.4.2 Per Port Customization – Custom DTD Timings

For each configuration, each port can be added with up to 255 customized DTD modes.

The following is an example of adding 800x640 mode to the LVDS port when ConfigId=1 is used.

[HKEY_LOCAL_MACHINE\Drivers\Display\Intel\ALL\1\Port\4\DTD\1]
"PixelClock"=dword:9c40
"HorzActive"=dword:320
"HorzSync"=dword:28
"HorzSyncPulse"=dword:80
"HorzBorder"=dword:0
"HorzBlank"=dword:100
"HorzSize"=dword:0
"VertActive"=dword:280
"VertSync"=dword:1
"VertSyncPulse"=dword:4
"VertBorder"=dword:0
"VertBlank"=dword:1c
"VertSize"=dword:0
"Flags"=dword:1e

6.2.4.3 Per Port Customization — Custom Flat Panel Controls

Similarly, the flat panel native resolution and power and backlight sequencing controls can also be configured here.

```
;[HKEY_LOCAL_MACHINE\Drivers\Display\Intel\ALL\1\Port\1\FPInfo]
      ;  "BkltMethod"=dword:0
      ;  "BkltT1"=dword:0
      ;  "BkltT2"=dword:0
      ;  "BkltT3"=dword:0
      ;  "BkltT4"=dword:0
      ;  "BkltT5"=dword:0
      ;  "GpioPinVdd"=dword:0
      ;  "GpioPinVee"=dword:0
      ;  "GpioPinBklt"=dword:0
      ;  "BkltEnable"=dword:0
      ;  "UseGMCHClockPin"=dword:0
      ;  "UseGMCHDataPin"=dword:0
```

Note: For Per-Config, Per-Port configuration, the subkey path includes the correct "Config" and "Port" numbers

6.2.4.4 Per Port Customization — Attribute Initialization

Attributes are also per config and per port. However, the actual keys are dependant on the port driver being used. Below are examples of registry keys associated with initializing attributes for the Chrontel Port Driver.

For complete information on port driver attributes, refer to Port Driver Attributes.

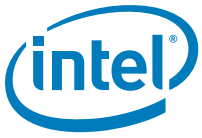
Note: For Per-Config, Per-Port configuration, the subkey path includes the correct "Config" and "Port" numbers.

The following example sets the port driver attributes using the attribute IDs. Please see Internal TV Out Port Driver Attributes (Mobile chipsets only) for a list of attribute IDs and their meanings.

```
[HKEY_LOCAL_MACHINE\Drivers\Display\Intel\ALL\1\Port\1\Attr]
      "0"=dword:32
      "1"=dword:4
      "3"=dword:1
      "8"=dword:1
      "12"=dword:0
      "14"=dword:1
      "19"=dword:1
```

6.2.5 Miscellaneous Configuration Options

This section covers registry settings not in [HKEY_LOCAL_MACHINE\Drivers\Display\Intel].



6.2.5.1 Text Anti-Aliasing

The Microsoft Windows CE driver supports text anti-aliasing. To switch it on, add these registry settings:

```
[HKEY_LOCAL_MACHINE\System\GDI\Fontsmoothing]
[HKEY_LOCAL_MACHINE\System\GDI]
   "ForceGRAY16"=dword:1
```

Note: Text Anti-Aliasing should always be turned on when using a TV display device.

6.2.6 D3D Mobile Support

IEGD 9.0 supports Direct3D* Mobile on Windows CE 5.0 and 6.0. Users need to ensure that their Windows CE target machine platform workspace has been included with D3D Mobile support. This is done by simply dragging the D3D Mobile component from the catalog view to the workspace in Windows CE 5.0 or turning it on via the check box in Windows CE 6.0.

Also, the new IEGD D3D mobile driver binary, "iegd3dg3.dll" (part of the IEGD driver release for Windows CE 5.0 and 6.0) needs to be included in the workspace image. Add this binary into the ".BIB" configuration file of the target platform workspace (see Sample FILES Block from platform.bib File):

```
iegd3dg3.dll <specify_path_here>\iegd3dg3.dll NK
```

No other IEGD registry configuration is necessary for this feature to work.

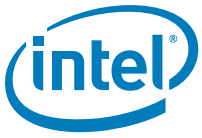
Note: The IEGD Windows CE D3D Mobile feature requires more memory at runtime:

- For Windows CE 5.0, it is recommended that 64 Mbytes is configured via Platform Builder (go to the Platform menu, choose **Settings**, and choose the **Build Options** tab. Ensure the option labeled "Run-time Image Can be Larger than 32 Mbytes (IMGRAM64=1)" is checked).
- For Windows CE 6.0, depending on the number of components built into the target platform workspace, the amount of memory could be significantly more, due in part to the new kernel memory architecture adopted by the OS (assuming all multimedia components are built-in). It is recommended that 84 Mbytes of memory be configured for the target machine workspace image. Refer to the following Microsoft URL for information on how configure more than 64 Mbytes on Windows CE 6.0:
  http://msdn2.microsoft.com/en-us/library/aa909457.aspx

See also the Sample iegd.reg File.

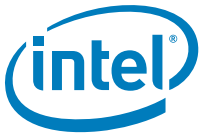
6.2.7 Sample iegd.reg File

```
;****************** BEGIN INTEL DISPLAY DRIVER REGISTRY ENTRY *****************
;******************************************************************************
;*-----------------------------------------------------------------------------
;* Copyright (c) Intel Corporation (2002 - 2008).
;*
;* The source code contained or described herein and all documents
;* related to the source code ("Material") are owned by Intel
;* Corporation or its suppliers or licensors. Title to the Material
;* remains with Intel Corporation or it suppliers and licensors. The
;* Material contains trade secrets and proprietary and confidential
;* information of Intel or its suppliers and licensors. The Material is
;* protected by worldwide copyright and trade secret laws and
;* treaty provisions. No part of the Material may be used, copied,
;* reproduced, modified, published, uploaded, posted, transmitted,
;* distributed, or disclosed in any way without Intels prior express
;* written permission.
;*
;* No license under any patent, copyright, trade secret or other
;* intellectual property right is granted to or conferred upon you by
;* disclosure or delivery of the Materials, either expressly, by
;* implication, inducement, estoppel or otherwise. Any license
;* under such intellectual property rights must be express
;* and approved by Intel in writing.
;*-----------------------------------------------------------------------------
[HKEY_LOCAL_MACHINE\System\GDI\Drivers]
"Display"="ddi_iegd.dll"
;******************************************************************************
; The Following Sections Provide
; General Driver-Wide Registry Settings
;******************************************************************************
[HKEY_LOCAL_MACHINE\Drivers\Display\Intel]
;-----------------------------------------------------------------------------
; Following registry entry for
; pcf version used
; 700 : IEGD version
;-----------------------------------------------------------------------------
"PcfVersion"=dword:700
;-----------------------------------------------------------------------------
; This value dictates the configuration to select for Per-Port settings from
; port specific regsitry. The settings mirror Windows XP IEGD drivers
; implementation. Refer to the User's Guide for details.
;-----------------------------------------------------------------------------
"ConfigId"=dword:1
;-----------------------------------------------------------------------------
; Provide a list of port drivers to attempt to load upon boot time
;-----------------------------------------------------------------------------
"PortDrivers"="lvds sdvo tv"
;******************************************************************************
```



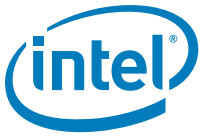
```
; The Following Sections Provide Per-Config configuration. The Platform string in
; the path can be “ALL” for all platforms, or any of the following for
; platform-specific configurations:
; Q35, GM965, Q965, 946GZ, 945G, 945GM, 915GV, and 915GM
;*****************************************************************************
[HKEY_LOCAL_MACHINE\Drivers\Display\Intel\915GV\1\General]
;-----------------------------------------------------------------------------
; Following registry entries for display settings:resolution, bit depth and
; refresh rate
;
; Width & Height values must be hex, for example
; 1400x1050 : 578h x 41Ah
; 1280x1024 : 500h x 400h
; 1024x768 : 400h x 300h
; 800x600 : 320h x 258h
; 640x480 : 280h x 1E0h
; etc...
;
; In vertical extented mode, height is doubled
; 640x960 : 280h x 3c0
; 800x600 : 320h x 4b0h
; etc...;
;-----------------------------------------------------------------------------
"Width"=dword:320
"Height"=dword:258
;-----------------------------------------------------------------------------
; Bit depth must be one of:
; 8bpp : 8
; 16bpp : 10
; 24bpp : 18
; 32bpp : 20
;-----------------------------------------------------------------------------
"Depth"=dword:10
;-----------------------------------------------------------------------------
; Refresh rate must be in hex:
; 60 : 3c
Intel® Embedded Graphics Drivers and Video BIOS v9.0
User’s Guide June 2008
Document Number: 274041-014US
Installing and Configuring Microsoft Windows CE Drivers
; 70 : 46
; 75 : 4b
; 85 : 55
; etc...
; any refresh rate as long as the display port supports it refresh of '0' will
; take the first refresh that matches width, height and bpp
;-----------------------------------------------------------------------------
"Refresh"=dword:3c
;-----------------------------------------------------------------------------
```

```
; Following is registry entry for controlled configuration of video memory
; usage / location
;
; The following settings are for a 64M platform, where the video memory is 14M
; at the top the above settings are assuming there is no system bios / firmware
; that has stolen memory from top of memory. If it does exist reduce
; ReservedMemorySize avoiding overlap exception for ACSFL, memory area is
; reused
;
; NOTE: CURRENTLY THESE SETTINGS ARE REMARKED FOR DYNAMIC VIDEO MEMORY
; CONFIGURATION
;-------------------------------------------------------------------------------
; "ReservedMemoryBase"=dword:03200000
; "ReservedMemorySize"=dword:00E00000
;-------------------------------------------------------------------------------
; Below is Maximum Frame Buffer Size used to limit the maximum size in bytes
; of the main frame buffer
;-------------------------------------------------------------------------------
"MaxFbSize"=dword:300000
;-------------------------------------------------------------------------------
; Page Request Limit is used to control the max allocations of offscreen video
; surfaces, buffers etc.. value is in number of pages (4K).
; this is independant of dynamic or static memory configuration.
;
; The max for 915s, 945s = 256 Mbytes = 0x10000
;-------------------------------------------------------------------------------
"PageReqLimit"=dword:8000
;-------------------------------------------------------------------------------
; Above settings are to define a minimum width and heigh that would allow for
; video surface allocations to succeed, eg: surfaces with width < 16 are
; forced to be in system-mem, surfaces with height < 16 are forced to be in
; system-mem only affects allocations of surfaces with GPE_PREFER_VIDEO_MEMORY
; flag
;-------------------------------------------------------------------------------
"MinVidSurfX"=dword:10
"MinVidSurfY"=dword:10
;-------------------------------------------------------------------------------
; Following are the registry entries for acceleration configuration
;-------------------------------------------------------------------------------
Installing and Configuring Microsoft Windows CE Drivers
Intel® Embedded Graphics Drivers and Video BIOS v9.0
User's Guide June 2008
Document Number: 274041-014US
; Set SysToVidStrech to '1' enables driver to perform System to Video stretch
; blits
;-------------------------------------------------------------------------------
"SysToVidStretch"=dword:1
;-------------------------------------------------------------------------------
```



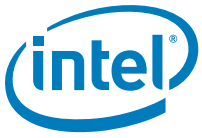
```
; Blend filterting method selection
; The hardware must be capable of
; support, else, emulation is done.
; Possible blend methods are:
; BlendFilter 0 == NEAREST
; BlendFilter 1 == BILINEAR
; BlendFilter 2 == ANISOTROPIC
; BlendFIlter 3 == 4X4
;-------------------------------------------------------------------------
"BlendFilter"=dword:2
;-------------------------------------------------------------------------
; Option for enabling/disabling TEARING - Default is OFF
;-------------------------------------------------------------------------
; Set '1' to enable the NOTEARING option
"TearFB"=dword:1
;-------------------------------------------------------------------------
; Specify whether to enable d3d
; NO_D3D Value: 0(default)
; : 0 --> Enable D3D
; : 1 --> Disable D3D
;-------------------------------------------------------------------------
"NO_D3D"=dword:0
;-------------------------------------------------------------------------
; Select Display configuration, single, twin ...
; Possible Display Config combo:
; DisplayConfig 1 == SINGLE
; (Single is default if none specified)
; DisplayConfig 4 == TWIN
; --> (Twin mode: common timing across ports)
; DisplayConfig 2 == CLONE
; --> (Clone mode: distinct timing per port)
; DisplayConfig 5 == VEXT (vertical extend)
; --> (Vert Extended modes : "Height" )
; ( registry key value must be 2X the )
; ( intended port timings. Both ports )
; ( must use the same timings. For )
; ( example, for port timings of )
; ( 800x600, the DisplayConfig should )
; ( be 5 and the Height=1200 or 0x4b0 )
; ( Overlay wont work in VEXT mode. )
; (915GV doesn't support Vext)
;-------------------------------------------------------------------------
"DisplayConfig"=dword:1
;-------------------------------------------------------------------------
```

```
; Select Port Order
; PortOrder specifies the actual port that will be used for the primary and
; secondary ports. IF specified port is unavailable (port driver failed or
; display detection failed or port is not available on current chipset), then
; the next port in the above order will be used. PortOrder must be set,
; based on chipset specifications:
; On i915 chipsets:
; ================================
; 1 - Integrated TV Encoder
; 2 - sDVO B port/RGBA port
; 3 - sDVO C port
; 4 - Internal LVDS port
; 5 - Analog port
;
; On i865 chipsets:
; ================================
; 1 - sDVO A port
; 2 - sDVO B port/RGBA port
; 3 - sDVO C port
; 4 - Internal LVDS port
; 5 - Analog port
;
; On 835: If RGBA is used (sDVO B & C together), then use sDVO B number
; to specify any parameter for it.
;
; On i81x chipsets:
; =================
; Port numbers:
; 3 - sDVO port
; 5 - Analog port
;-------------------------------------------------------------------------------
"PortOrder"="5432"
;-------------------------------------------------------------------------------
; Set Clone Port resolutions
;-------------------------------------------------------------------------------
; "CloneWidth"=dword:320
; "CloneHeight"=dword:258
; "CloneRefresh"=dword:3c
;-------------------------------------------------------------------------------
; Set "1" to enable Display Detection
; DisplayDetect is to detect display child device before using it
; (panel/tv/etc...).BEWARE, setting this to '1' will mean display for the
; requested port wont be enabled if detection failed. Use this option wisely.
;-------------------------------------------------------------------------------
"DisplayDetect"=dword:0
;-------------------------------------------------------------------------------
```



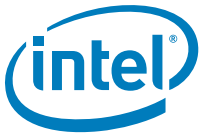
```
; Overlay Color Correction Settings
; Gamma: 32-bit integer in 24i.8f format, ranging from 0.6 - 6.0 decimal
; Brightness: 32-bit integer ranging from 0 to 0xFFFF. 0x8000 = no correction
; Contrast: 32-bit integer ranging from 0 to 0xFFFF. 0x8000 = no correction
; Saturation: 32-bit integer ranging from 0 to 0xFFFF. 0x8000 = no correction
;-------------------------------------------------------------------------------
; "OverlayGammaCorrectR"=dword:100
; "OverlayGammaCorrectG"=dword:100
; "OverlayGammaCorrectB"=dword:100
; "OverlayBrightnessCorrect"=dword:8000
; "OverlayContrastCorrect"=dword:8000
; "OverlaySaturationCorrect"=dword:8000
;*******************************************************************************
; The sections below are for the more detailed per port
; registry configurations. It follows the same usage model and
; key value meanings as the Windows INF registry configuration
; file. Refer to the User's Guide for details.
;*******************************************************************************
;------------------------------------------------------------------------
; Config 1 - sDVO-B Port (For Almador) |
;------------------------------------------------------------------------
; Following are the registry
; entries for port's general config
;------------------------------------------------------------------------
;
;[HKEY_LOCAL_MACHINE\Drivers\Display\Intel\915GV\1\Port\2\General]
;
; Advanced Edid Configuration
; ---------------------------
; "Edid"=dword:1
; "EdidAvail"=dword:7 ; STD TIMINGS + EDID TIMINGS + USER TIMINGS
; "EdidNotAvail"=dword:7 ; STD TIMINGS + USER TIMINGS
;
; EdidAvail and EdidNotAvail: <only 16 bits used>
; ---------- -------------
; These 2 parameters can be used to control the available timings for
; any display. 'EdidAvail' is used when EDID is read from the display
; device. If an attempt to read EDID is failed or 'Edid = 0' then
; driver uses 'EdidNotAvail' flags.
;
; See below bit definitions for both 'EdidAvail' and 'EdidNotAvail'
;
; BIT 0:
; ------
; 0 - Do not use driver built-in standard timings
; 1 - Use driver built-in standard timings
;
```

```
; BIT1: <not applicable to EdidNotAvail>
; ------
; 0 - Do not use EDID block
; 1 - Use EDID block and filter modes
;
; BIT2:
; ------
; 0 - Do not use user-DTDs
; 1 - Use user-DTDs.
;
; BIT3-BIT15
; -----------
; Future use.
;
; Default behavior:
; -----------------
; If user doesn't provide EdidAvail and EdidNotAvail, then
; EdidAvail = Use Std timings + Use EDID block and Filter modes
; EdidNotAvail = Use Std timings
;
; Rotation Configuration
; ----------------------
; "Rotation"=dword:0
; Rotation entries must be at a right
; angle. An invalid entry will be ignored and
; and no rotation will happen for primary.
; In clone or twin modes, the secondary
; port defaults to follow the primary (if set)
; 0 degrees = 0 (not rotated = default)
; 90 degrees = 5A
; 180 degrees = B4
; 270 degrees = 10E
;
; Flip Configuration
; ----------------------
; "Flip"=dword:0
; Flip has a valid entry of 1 to turn on
; and 0 to turn off. When turn on the display
; will be horizontally flip.
;
; Rendered Scaling Configuration
; ------------------------------
; "Scale"=dword:0
; Scale works as a boolean switch. Valid
; entries are zero or 1. When "Scale" = 1,
; IEGD will scale the requested framebuffer
; resolution to the fixed native panel size
; indicated by per-port FPInfo, User-DTD or
; EDID (in that order).
```



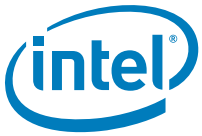
```
; In clone or twin modes, the secondary
; port defaults to follow the primary (if set)
;-------------------------------------------------------------------------
; Following are the registry entries
; for port's sDVO I2C settings
;-------------------------------------------------------------------------
;[HKEY_LOCAL_MACHINE\Drivers\Display\Intel\915GV\1\Port\2\sDVO]
; "I2cPin"=dword:2
; "I2cDab"=dword:70
; "I2cSpeed"=dword:0
; "DdcPin"=dword:0
; "DdcSpeed"=dword:0
;-------------------------------------------------------------------------
; Following are the registry entries
; for port's flat panel's mode-limits,
; power and backlight control
;-------------------------------------------------------------------------
;[HKEY_LOCAL_MACHINE\Drivers\Display\Intel\915GV\1\Port\2\FPInfo]
; Only need Width & Height if Panel cannot accept other timings
; "BkltMethod"=dword:3
; "BkltT1"=dword:1E
; "BkltT2"=dword:4
; "BkltT3"=dword:4
; "BkltT4"=dword:14
; "BkltT5"=dword:1F4
; "GpioPinVdd"=dword:27
; "GpioPinVee"=dword:26
; "GpioPinBklt"=dword:28
; "UseGMCHClockPin"=dword:0
; "UseGMCHDataPin"=dword:0
;-------------------------------------------------------------------------
; Following are the registry entries
; for ports first custom DTD mode to add
;-------------------------------------------------------------------------
;[HKEY_LOCAL_MACHINE\Drivers\Display\Intel\915GV\1\Port\2\DTD\1]
; "PixelClock"=dword:9c40
; "HorzActive"=dword:320
; "HorzSync"=dword:28
; "HorzSyncPulse"=dword:80
; "HorzBorder"=dword:0
; "HorzBlank"=dword:100
; "HorzSize"=dword:0
; "VertActive"=dword:280
; "VertSync"=dword:1
; "VertSyncPulse"=dword:4
; "VertBorder"=dword:0
; "VertBlank"=dword:1c
; "VertSize"=dword:0
; "Flags"=dword:1e
;-------------------------------------------------------------------------
```

```
; Following are the registry entries
; for ports second custom DTD mode to add
; (Up to 255 can be added)
;-------------------------------------------------------------------------
;[HKEY_LOCAL_MACHINE\Drivers\Display\Intel\915GV\1\Port\2\DTD\2]
; "PixelClock"=dword:9c40
; "HorzActive"=dword:320
; "HorzSync"=dword:28
; "HorzSyncPulse"=dword:80
; "HorzBorder"=dword:0
; "HorzBlank"=dword:100
; "HorzSize"=dword:0
; "VertActive"=dword:258
; "VertSync"=dword:1
; "VertSyncPulse"=dword:4
; "VertBorder"=dword:0
; "VertBlank"=dword:1c
; "VertSize"=dword:0
; "Flags"=dword:1e
;-------------------------------------------------------------------------
; Following are the registry
; entries for the port device'
; display attribute parameters
; Use when enabling Port device
; example below is for Conexant
; on Port2 (sDVO-B for almador)
; key names depend on port driver
;-------------------------------------------------------------------------
;[HKEY_LOCAL_MACHINE\Drivers\Display\Intel\915GV\1\Port\2\Attr]
; "Brightness"=dword:32
; "Contrast"=dword:4
; "Flicker Filter"=dword:1
; "Saturation"=dword:4
; "Hue"=dword:32
; "Text Filter"=dword:0
; "Macrovision"=dword:0
; "Overscan ratio"=dword:1
; "TV Format"=dword:1
; "TV Output"=dword:1
; "Composite and S-Video"=dword:1
;-------------------------------------------------------------------------
```



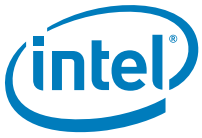
```
; Config 1 - Analog Port (For Any Chipset)
;-------------------------------------------------------------------------
;[HKEY_LOCAL_MACHINE\Drivers\Display\Intel\915GV\1\Port\5\General]
; "Edid"=dword:1
; "EdidAvail"=dword:7 ; STD TIMINGS + EDID TIMINGS + USER TIMINGS
; "EdidNotAvail"=dword:7 ; STD TIMINGS + USER TIMINGS
;[HKEY_LOCAL_MACHINE\Drivers\Display\Intel\Config\1\Port\5\attr]
; GAMMA, BRIGHTNESS, CONTRAST
; "35"=dword:a0a0a0 ; gamma: 3i.5f format for R-G-B, ranging 0.6 to 6
; "36"=dword:808080 ; brightness: 0 to FF, 0x80 is no correction
; "37"=dword:808080 ; contrast: 0 to FF, 0x80 is no correction
;[HKEY_LOCAL_MACHINE\Drivers\Display\Intel\Config\1\Port\5\DTD\1]
; "PixelClock"=dword:9c40
; "HorzActive"=dword:320
; "HorzSync"=dword:28
; "HorzSyncPulse"=dword:80
; "HorzBorder"=dword:0
; "HorzBlank"=dword:100
; "HorzSize"=dword:0
; "VertActive"=dword:280
; "VertSync"=dword:1
; "VertSyncPulse"=dword:4
; "VertBorder"=dword:0
; "VertBlank"=dword:1c
; "VertSize"=dword:0
; "Flags"=dword:1e
;[HKEY_LOCAL_MACHINE\Drivers\Display\Intel\915GV\1\Port\5\DTD\2]
; "PixelClock"=dword:9c40
; "HorzActive"=dword:320
; "HorzSync"=dword:28
; "HorzSyncPulse"=dword:80
; "HorzBorder"=dword:0
; "HorzBlank"=dword:100
; "HorzSize"=dword:0
; "VertActive"=dword:258
; "VertSync"=dword:1
; "VertSyncPulse"=dword:4
; "VertBorder"=dword:0
; "VertBlank"=dword:1c
; "VertSize"=dword:0
; "Flags"=dword:1e
;-------------------------------------------------------------------------
```

```
; Config 1 - Int-LVDS Port (For 915GM) |
;-------------------------------------------------------------------------
;[HKEY_LOCAL_MACHINE\Drivers\Display\Intel\915GV\1\Port\4\General]
; "Edid"=dword:1
; "EdidAvail"=dword:7 ; STD TIMINGS + EDID TIMINGS + USER TIMINGS
; "EdidNotAvail"=dword:7 ; STD TIMINGS + USER TIMINGS
;[HKEY_LOCAL_MACHINE\Drivers\Display\Intel\915GV\1\Port\4\FPInfo]
; Only need Width & Height if Panel cannot except other timings
; "BkltMethod"=dword:0
; "BkltT1"=dword:0
; "BkltT2"=dword:0
; "BkltT3"=dword:0
; "BkltT4"=dword:0
; "BkltT5"=dword:0
; "GpioPinVdd"=dword:0
; "GpioPinVee"=dword:0
; "GpioPinBklt"=dword:0
; "UseGMCHClockPin"=dword:0
; "UseGMCHDataPin"=dword:0
;[HKEY_LOCAL_MACHINE\Drivers\Display\Intel\915GV\1\Port\4\Attr]
; "27"=dword:1
; Attribute "27" = Dual Channel (boolean)
; "18"=dword:1
; Attribute "18" = Panel Fit Upscale (boolean)
;[HKEY_LOCAL_MACHINE\Drivers\Display\Intel\915GV\1\Port\4\DTD\1]
; "PixelClock"=dword:9c40
; "HorzActive"=dword:320
; "HorzSync"=dword:28
; "HorzSyncPulse"=dword:80
; "HorzBorder"=dword:0
; "HorzBlank"=dword:100
; "HorzSize"=dword:0
; "VertActive"=dword:280
; "VertSync"=dword:1
; "VertSyncPulse"=dword:4
; "VertBorder"=dword:0
; "VertBlank"=dword:1c
; "VertSize"=dword:0
; "Flags"=dword:1e
;[HKEY_LOCAL_MACHINE\Drivers\Display\Intel\915GV\1\Port\4\DTD\2]
; "PixelClock"=dword:9c40
; "HorzActive"=dword:320
; "HorzSync"=dword:28
; "HorzSyncPulse"=dword:80
; "HorzBorder"=dword:0
; "HorzBlank"=dword:100
; "HorzSize"=dword:0
; "VertActive"=dword:258
; "VertSync"=dword:1
; "VertSyncPulse"=dword:4
```



```
; "VertBorder"=dword:0
; "VertBlank"=dword:1c
; "VertSize"=dword:0
; "Flags"=dword:1e
;-------------------------------------------------------------------------
; Config 1 - sDVO Port-B (For Napa)
;-------------------------------------------------------------------------
;[HKEY_LOCAL_MACHINE\Drivers\Display\Intel\915GV\1]
; "name"="IEGD sDVO Configuration File"
;[HKEY_LOCAL_MACHINE\Drivers\Display\Intel\915GV\1\Port\2]
; "name"="svga"
;[HKEY_LOCAL_MACHINE\Drivers\Display\Intel\915GV\1\Port\2\FPInfo]
; For a sDVO driver, sample settings for the panel:1400x1050
; Only need Width & Height if Panel cannot except other timings
; "Width"=dword:578
; "Height"=dword:41A
;[HKEY_LOCAL_MACHINE\Drivers\Display\Intel\915GV\1\Port\2\Attr]
; "27"=dword:1
; Attribute "27" = Dual Channel (boolean)
; Optional - Only enable for font anti-aliasing
; Enabling this causes minor performance impact
; Only recommended for TV Output.
;[HKEY_LOCAL_MACHINE\System\GDI\Fontsmoothing]
;
;[HKEY_LOCAL_MACHINE\System\GDI]
; "ForceGRAY16"=dword:1
;****************** INTEL DISPLAY DRIVER REGISTRY ENTRY END ******************
```

This page is intentionally left blank.



7.0 Installing and Configuring Linux OS Drivers

This topic describes the configuration and installation of the IEGD for Linux systems. The IEGD supports X-servers from the X.org* and XFree86* organizations.

The Intel Linux driver is for use with the integrated graphics of Intel chipsets on the Embedded Intel Architecture roadmap. The driver supports 8-, 16- and 24-bit pixel depths, dual independent head configuration on capable hardware, flat panel, hardware 2D acceleration, hardware cursor, the XV extension, and the Xinerama extension.

7.1 Overview

Kernel patches, separate DRM modules, and kernel recompilation were all necessary in previous versions of the Intel Embedded Graphics Driver. In version 9.0, the IEGD Kernel Module (IKM) contains a combination of the agpgart and DRM modules which must be present for the Intel Embedded Graphic Driver. Both modules have been modified for the IEGD architecture and are combined with the Linux kernel.

The IEGD Linux distribution package contains drivers built for the following X-Servers:

- XFree86 version 4.3.0
- X.org version 6.7
- X.org version 6.8
- X.org version 7.0 (Used for X.org 6.9 or 7.0)
- X.org version 7.1
- X.org version 7.2
- X.org version 7.3

The IEGD has been tested with the official version of these servers from the www.Xfree86.org or the www.X.org Web sites and may not operate with other versions of these servers.

During the installation, the `X -version` command returns a result indicating the server version, not the X.org version as was done in earlier versions. The table below shows the corresponding versions for the supported X.org packages.

Table 36. X.org and X-version Conversion Reference

X.org Package Version	X-version Command Result
7.2	1.2 or 1.3
7.3	1.3

See also:

- "Prerequisites"
- "Supported Hardware"
- "Installation"

7.2 Prerequisites

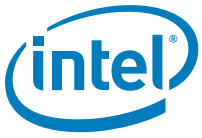
The following lists the prerequisites for installing and configuring the IEGD Linux driver.

- Platform with supported Intel chipset.
- Platform with a minimum of 128 Mbytes.
- Resolution and timing specifications for the display devices that will be configured.
- Driver package consisting of directories and files (see the following reduced samples, which are located under the `IEGD Linux` directory).

Note: In the following, "Xorg-X11R7.0" is an example X server version that should be replaced with the version to be used.

  - `Documents/Relnotes`
    `Documents/UsersGuide.pdf`
    `Documents/Xorg-X11R7.0/iegd.4`
    `Documents/Xorg-X11R7.0/IntelEscape.3x`
  - `License/License.txt`
  - `Driver/<xserver name>/iegd_drv.o (or iegd_drv.so for Xorg 7.0)`
    `Driver/<xserver name>/libXlibXiegd_escape.a`
    `Driver/<xserver name>/libXiegd_escape.so.2.0.0`
    `Driver/<xserver name>/iegd_escape.h`
    `Driver/<xserver name>/lvds.so`
    `Driver/<xserver name>/tv.so`

- Linux kernel source tree for active running kernel.
  - `AGPGART must be enabled as a module.`
  - `Direct Rendering support enabled.`
- XFree86 version 4.2, XFree86 version 4.3, X.org version 6.7.0, X.org version 6.8.2, or X.org version 7.0 X-Server installed and functional.
  - `Directory paths to XFree86 or X.org installation and configuration files.`
- Other system capabilities
  - `Ncurses (SUSE 10), etc.`
  - `IEGD Kernel Module for GART and DRM patches`
- System administration privileges.



7.2.1 Supported Hardware

IEGD supports the following chipsets with integrated graphics:

- Intel® GM45 Express chipset
- Intel® System Controller Hub US15W chipset
- Intel® Q35 Express chipset
- Mobile Intel® GME965 Express chipset
- Intel® Q965 Express chipset
- Mobile Intel® 945GSE Express chipset
- Mobile Intel® 945GME Express chipset
- Intel® 945G Express chipset
- Intel® 915GV Express chipset
- Mobile Intel® 915GME Express chipset
- Mobile Intel® 910GMLE Express chipset

7.3 Installation

Refer to Section 1.7, "Downloading the IEGD and Video BIOS" on page 17 for instructions on obtaining the software. You can then install the IEGD by performing the instructions for your specific distribution in the following sections:

- "Installing Fedora Core 5 or Fedora 7" on page 149
- "Installing SUSE 10" on page 151
- "Installing Novell Linux POS 9 SP3" on page 153
- "Installing Other Kernels" on page 155
- "Enabling Damn Small Linux" on page 184

Note: If you are using a Linux distribution different from those for which these instructions are designed, you may need to adapt the steps for your specific situation.

7.3.1 Installing Fedora Core 5 or Fedora 7

1. Log into to a system administration account.
2. Untar the driver package to a convenient location.

   tar -xvzf <driver package.tgz>

   This creates a directory structure in the directory where you extracted the `.tgz` file and contains the following directories:

   — IEGD_X_x_Linux - Contains the Documents, Driver, License, and Utilities subdirectories, where X is the major version and x is the minor version.

   The Documents subdirectory contains the Xfree86 and Xorg-X11 subdirectory. These directories contain man pages for the IEGD.

   The Driver directory contains subdirectories the supported versions of XFree and Xorg.

   The Utilities directory contains IEGD utilities, including the `iegdgui` runtime configuration utility.

   — IEGD_Patches - Contains files for patching the Linux kernel agpgart module.

3. Check the version of the X-Server your system is running. Type the following command:

   X -version

Note: For Fedora Core 5, the result from this command is 7.0. For Fedora 7, the result from this command is 1.3 which equates to 7.2 for IEGD. Refer to Table 36 on page 147 to convert from the X -version command to IEGD's naming convention.

4. Copy the IEGD driver binary, `iegd_drv.o` (or `iegd_drv.so`), from the `IEGD_X_x_Linux/Driver/<xserver name>` directory to the X-Server's `modules/drivers` directory, where X is the major version and x is the minor version.
   - For FC5 (X.org-7.0 based distribution), the default location is `/usr/lib/X11/lib/modules/drivers`. This location can vary by distribution so check your system for the proper path and replace X and x with the correct version numerals.

     cd IEGD_9_0_Linux/Driver/Xorg-X11R7.0
     cp iegd_drv.so /usr/lib/xorg/modules/drivers
   - For FC7 (X.org-7.2 based distribution), the default location is `/usr/lib/X11/lib/modules/drivers`. This location can vary by distribution so check your system for the proper path and replace X and x with the correct version numerals.

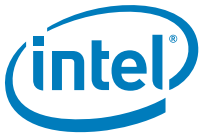
     cd IEGD_9_0_Linux/Driver/Xorg-X11R7.2
     cp iegd_drv.so /usr/lib/xorg/modules/drivers
5. Copy the necessary port driver files (*.so files in the `IEGD_X_x_Linux/Driver/<xserver name>` directory) to the X-Server `lib/modules` directory, where X is the major version and x is the minor version. The default installation location is `/usr/lib/xorg/modules`. This location can vary, so check your system for the proper path. After the required port drivers have been copied, you can specify them in the PortDrivers option in the Device section of the config file. For more information on specifying the PortDrivers option, refer to Table 37, "Supported Driver Options" on page 167. For example, to copy all the port drivers use the following command:

   cp *.so /usr/lib/xorg/modules
6. Copy the escape control library `libXiegd_escape.so.2.0.0` from the `IEGD_X_x_Linux/Driver/<xserver name>` directory to the X-Server library directory, where X is the major version and x is the minor version. The default installation location is `/usr/lib`. For example,

   cp libXiegd_escape.so.2.0.0 /usr/lib
7. In the X-Server library directory, create symbolic links for the escape library aliases:

   cd /usr/lib
   ln -s libXiegd_escape.so.2.0.0 libXiegd_escape.so
   ln -s libXiegd_escape.so.2.0.0 libXiegd_escape.so.2
   ldconfig
8. From the `IEGD_X_x_Linux/Documents` directory, where X is the major version and x is the minor version, copy the driver man page, `iegd.4`, to the `man/man4` directory. The default installation location is `/usr/share/man/man4`. This location can vary by distribution so check your system for the proper path and replace X and x with the correct version numerals. For example:

   cd IEGD_9_0_Linux/Documents/Xorg-X11
   cp iegd.4 /usr/share/man/man4



9. For Fedora Core 5, use the IKM Patch Instructions in Section 7.4 on page 156 which is the preferred method, or use Section 7.6, "GART Patch Instructions (for Fedora Core 5 Only)" on page 160.

   For Fedora 7, proceed to step 10.

10. **For Fedora 7 only**, execute the following commands:

    cd IEGD_Patches/IKM

    ./install.sh

    (Note: if a permissions error is displayed, do a **chmod +x install.sh**)

    depmod -a

    modprobe iegd_mod

11. Modify your xorg.conf file to include a device section for this driver and a Monitor section for your display. See Section 7.7.1, "Configuration Overview" on page 161 for details on the driver configuration and the list of supported options. The default installation location for this file is `/etc/X11`.

12. Reboot

7.3.2 Installing SUSE 10

1. Log into to a system administration account.
2. Untar the driver package to a convenient location.

   tar -xvzf <driver package.tgz>

   This creates a directory structure in the directory where you extracted the `.tgz` file and contains the following directories:

   — `IEGD_X_x_Linux` - Contains the Documents, Driver, License, and Utilities subdirectories, where X is the major version and x is the minor version.

   The Documents subdirectory contains the Xfree86 and Xorg-X11 subdirectory. These directories contain man pages for the IEGD.

   The Driver directory contains subdirectories for the supported versions of XFree and Xorg.

   The Utilities directory contains IEGD utilities, including the `iegdgui` runtime configuration utility.

   — `IEGD_Patches` - Contains files for patching the Linux kernel agpgart module.

3. Check the version of the X-Server your system is running. Type the following command:

   X -version

Note: For SUSE 10 the result from this command is normally Xorg 11 6.8.2. If the command returns a different result, refer to Table 36 on page 147 to convert from the X -version command to IEGD's naming convention.

4. Copy the IEGD driver binary, `iegd_drv.o` (or `iegd_drv.so`), from the `IEGD_X_x_Linux/Driver/<xserver name>` directory, where X is the major version and x is the minor version, to the X-Server's `modules/drivers` directory. For SUSE 10 (X.org-6.8 based distribution), the default location is `/usr/X11R6/lib/modules/drivers`.

   This location can vary by distribution so check your system for the proper path and replace X and x with the correct version numerals.

   cd IEGD_9_0_Linux/Driver/Xorg-X11R6.8

   cp iegd_drv.o /usr/X11R6/lib/modules/drivers

Note: Depending on your installation you may need to copy the IEGD driver to an alternate location:

cp iegd_drv.o /usr/lib/xorg/modules/drivers

5. Copy the necessary port driver files (*.so files in the `IEGD_X_x_Linux/Driver/<xserver name>` directory) to the X-Server `lib/modules` directory, where X is the major version and x is the minor version. The default installation location is `/usr/X11R6/lib/modules`. This location can vary, so check your system for the proper path. Once the required port drivers have been copied, you can specify them in the `PortDrivers` option in the Device section of the config file. For more information on specifying the PortDrivers option, refer to Table 37, "Supported Driver Options" on page 167. For example, to copy all the port drivers use the following command:

   cp *.so /usr/X11R6/lib/modules

6. Copy the escape control library `libXiegd_escape.so.2.0.0` from the `IEGD_X_x_Linux/Driver/<xserver name>` directory to the X-Server library directory, where X is the major version and x is the minor version. The default installation location is `/usr/X11R6/lib`. For example,

   cp libXiegd_escape.so.2.0.0 /usr/X11R6/lib

7. In the X-Server library directory, create symbolic links for the escape library aliases:

   cd /usr/X11R6/lib
   ln -s libXiegd_escape.so.2.0.0 libXiegd_escape.so
   ln -s libXiegd_escape.so.2.0.0 libXiegd_escape.so.2
   ldconfig

8. From the `IEGD_X_x_Linux/Documents` directory, copy the driver man page, `iegd.4`, to the `man/man4` directory, where X is the major version and x is the minor version. The default installation location is `/usr/X11R6/man/man4`. This location can vary by distribution so check your system for the proper path and replace X and x with the correct version numerals. For example:

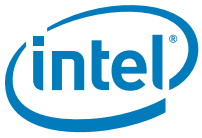
   cd IEGD_9_0_Linux/Documents/Xorg-X11
   cp iegd.4 /usr/X11R6/man/man4

9. Use the IKM Patch Instructions in Section 7.4.3, "Installing IKM with SUSE" on page 158.

Note: You must have your kernel source and GCC installed for this step. If you do not have these installed, you can use the YAST and search for "kernel", select it, then search for "GCC" and select it. At this point you should also ensure that the kernel is set for loadable modules.

At this point, when X Windows starts, it should be using the IEGD driver. To verify this, you can check your Xorg log, or run the `iegdgui` utility (found in the Utilities directory).

Note: You may need to set the `iegdgui` file properties to be executable before it will run.



7.3.3 Installing Novell Linux POS 9 SP3

1. Log into a system administration account.
2. Untar the driver package to a convenient location.

   tar -xvzf <driver package.tgz>

   This creates a directory structure in the directory where you extracted the `.tgz` file and contains the following directories:

   — `IEGD_X_x_Linux` - Contains the Documents, Driver, License, and Utilities subdirectories, where X is the major version and x is the minor version.

   The Documents subdirectory contains the `Xfree86` and `Xorg-X11` subdirectory. These directories contain man pages for the IEGD.

   The Driver directory contains subdirectories for `XFree86-4.3`, `Xorg-6.7`, `Xorg-6.8`, and `Xorg-7.0`.

   The Utilities directory contains IEGD utilities, including the `iegdgui` runtime configuration utility.

   — `IEGD_Patches` - Contains files for patching the Linux kernel agpgart module.

Note: You must have the following installed to successfully install the IEGD gcc, patch, and make.

3. Check the version of the X-Server your system is running. Type the following command:

   X -version

Note: For NLPOS 9 the result from this command is normally XFree86 4.4.0 RC 2. If the command returns a different result, refer to Table 36 on page 147 to convert from the X -version command to IEGD's naming convention.

4. Copy the IEGD driver binary, `iegd_drv.o` (or `iegd_drv.so`), from the `IEGD_X_x_Linux/Driver/<xserver name>` directory, where X is the major version and x is the minor version, to the X-Server's `modules/drivers` directory. For NLPOS 9 (XFree86 based distribution), the default location is `/usr/X11R6/lib/modules/drivers/`.

   This location can vary by distribution so check your system for the proper path and replace X and x with the correct version numerals.

   cd IEGD_9_0_Linux/Driver/Xorg-X11R6.8

   cp iegd_drv.o /usr/X11R6/lib/modules/drivers/

5. Copy the necessary port driver files (`*.so` files in the `IEGD_X_x_Linux/Driver/<xserver name>` directory) to the X-Server `lib/modules` directory, where X is the major version and x is the minor version. The default installation location is `/usr/X11R6/lib/modules`. This location can vary, so check your system for the proper path. After the required port drivers have been copied, you can specify them in the PortDrivers option in the Device section of the config file. For more information on specifying the PortDrivers option, refer to Table 37, "Supported Driver Options" on page 167. For example, to copy all the port drivers use the following command:

   cp *.so /usr/X11R6/lib/modules

6. Copy the escape control library `libXiegd_escape.so.2.0.0` from the `IEGD_X_x_Linux/Driver/<xserver name>` directory to the X-Server library directory, where X is the major version and x is the minor version. The default installation location is `/usr/X11R6/lib`. For example,

   cp libXiegd_escape.so.2.0.0 /usr/X11R6/lib

7. In the X-Server library directory, create symbolic links for the escape library aliases:

   cd /usr/X11R6/lib

   ln -s libXiegd_escape.so.2.0.0 libXiegd_escape.so

   ln -s libXiegd_escape.so.2.0.0 libXiegd_escape.so.2

   ldconfig

8. From the `IEGD_X_x_Linux/Documents` directory, copy the driver man page, `iegd.4`, to the `man/man4` directory, where X is the major version and x is the minor version. The default installation location is `/usr/X11R6/man/man4`. This location can vary by distribution so check your system for the proper path and replace X and x with the correct version numerals. For example:

   cd IEGD_9_0_Linux/Documents/Xorg-X11

   cp iegd.4 /usr/X11R6/man/man4

9. Use the IKM Patch Instructions in Section 7.4 on page 156.

Note: You must have your kernel source and GCC installed for this step. If you do not have these installed, you can use the YAST and search for "kernel", select it, then search for "GCC" and select it. At this point you should also ensure that the kernel is set for loadable modules.

You can use the IKM method described in Section 7.4, "IKM Patch Instructions" on page 156 instead of using the instructions below. You need to use only one of the patching methods, either the IKM or the GART patch.

To patch a Linux 2.6.xxxx kernel with the GART changes:

a. cd into the kernel source directory (e.g., `cd /usr/src/linux-2.x.xxxx`). For NLPOS it is:

   cd /usr/src/linux-2.6.5-7.244

b. Execute the patch command (e.g., `patch -p1 < .../IEGD_Patches/Driver/agpgart.patch-2.x.xxxx`). For NLPOS use:

   patch -p1 < .../IEGD_Patches/Driver/agpgart.patch-2.6.5-7.244

To update the kernel:

a. cd to the kernel source directory (e.g., cd /usr/src/linux-2.x.xxxx)

   cd /use/src/linux-2.6.5-7.244

b. Since the agpgart is typically built-in by default, it needs to be configured to install it as a module before updating the kernel.

   make menuconfig

   From the configuration menu, select `Device Drivers`, then `Character Devices`. Scroll down to `/dev/agpgart`. Change the disposition to `M` (for Module). Exit from the configuration menu and save your changes.

c. Execute the make all modules command:

   make all

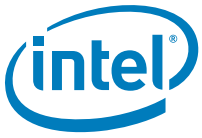
d. Install the modules and build the kernel:

   make modules_install && make install

e. Reboot.

f. If you did not edit the grub.conf file, select the custom kernel.

Note: Make sure that the newly updated kernel is the one being booted.



g. Run the following commands:

modprobe agpgart

modprobe intel-agp

h. To ensure the agpgart modules are loaded after a reboot, add the following line to the `/etc/modprobe.conf` file:

alias char-major-10-175 intel-agp

i. Modify your xorg.conf file to include a device section for this driver and a Monitor section for your display. See Section 7.7.1, "Configuration Overview" on page 161 for details on the driver configuration and the list of supported options. The default installation location for this file is `/etc/X11`.

Note: The CED utility generates generic Linux X config samples using "`Screen0`" and "`Monitor0`" entries but a standard install of NLPOS seems to require "`Screen[0]`" and "`Monitor[0]`" in the `Xorg.conf` file.

j. Reboot.

At this point, when X Windows starts, it should be using the IEGD driver. To verify this, you can check your Xorg log, or run the `iegdgui` utility (found in the Utilities directory).

Note: You may need to set the `iegdgui` file properties to be executable before it will run.

7.3.4 Installing Other Kernels

The instructions for Fedora Core 5 and SUSE 10 describe how to patch and use IEGD with a 2.6.x kernel. If you are using a 2.4.x kernel, the patch step is slightly different:

1. To patch the Linux 2.4.2x kernel with the GART changes:
   a. cd into the kernel source directory (e.g., `cd /usr/src/linux-2.x.xxx`)
   b. Execute the patch command. For example:
      patch -p1 < .../IEGD_Patches/Driver/agpgart.patch-2.x.xxx
2. To update the kernel:
   a. cd to the kernel source directory (e.g., cd /usr/src/linux-2.x.xxx)
   b. Execute the make modules command.
      make modules
   c. Install the modules.
      make modules_install
3. Reboot and if you did not edit the grub.conf file, select the custom kernel.
4. Run the following commands:
   modprobe agpgart
   modprobe intel-agp
5. To ensure the agpgart modules are loaded after a reboot, add the following line to the `/etc/modprobe.conf` file:
   alias char-major-10-175 intel-agp
6. Modify your XF86Config or xorg.conf file to include a device section for this driver and a Monitor section for your display. See Section 7.7.1, "Configuration Overview" for details on the driver configuration and the list of supported options. The default installation location for this file is `/etc/X11`.
7. Reboot.

7.4 IKM Patch Instructions

The IKM process is designed to replace the need to patch your kernel GART and DRM. If you plan to use the patch method of installing IEGD, do NOT use this IKM process. Using the IKM process is the preferred method for installing IEGD.

See also the following topics:

- "Finding and Installing the Kernel Source (Headers)"
- "Installing IKM with Fedora"
- "Installing IKM with SUSE"
- "Using the IEGD Kernel Module"

7.4.1 Finding and Installing the Kernel Source (Headers)

- Building the IKM requires a kernel build tree.
  This should be at least a partial kernel source tree and most importantly, the exact kernel `.config` file used for the build as well as several files generated at kernel build time.
- KERNEL_VERSION is the output of the command **uname -r**
- If you use a kernel from your distribution you will typically have a package with all the files required to build kernel modules for your kernel image.
  - On Fedora and compatibles (e.g., RedHat) this is the `kernel-devel` package. Or if you run `kernel-smp` or `kernel-xen`, you need `kernel-smp-devel` or `kernel-xen-devel`, respectively.
  - On SUSE you need the package kernel-source.
  - In some distributions (e.g., in RHEL or Fedora) the installation of the `kernel-devel` / `kernel-headers` package will be a newer version than the one you currently run. In such a case, you may need to upgrade the kernel package itself and reboot.

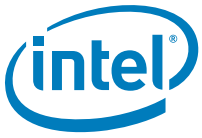
For details, see:

- "Installing IKM with Fedora"
- "Installing IKM with SUSE"

7.4.2 Installing IKM with Fedora

1. Install `kernel-$ARCH-devel`. The version of the package must be the same as the running kernel. Replace `$ARCH` with architecture of the kernel (e.g., smp).
2. Install `Kernel-devel` from the CD/DVD or through the `yum` utility.
3. Install the kernel source for the version you are running by choosing one of the following methods.
   - If you are using a CD or DVD, search for the rpm package for `kernel-$ARCH-devel` and install using

     rpm -ivh kernel-$ARCH-devel



— For the `yum` utility, type the following command:

yum install kernel-devel

This will install `kernel-devel` and resolve dependencies. Note that, as stated earlier, `kernel-devel` that is installed through this method might not be the same version as a running kernel. In this case IKM compilation might be successful, however, when trying to insert it to the running kernel, **modprobe/ insmod** will produce an error. The solution is to upgrade the kernel-package itself and reboot to make the Linux OS run the updated kernel.

4. Hardlink is a utility that consolidates duplicates files in one or more directories by traversing the directories and searching for duplicate files. When Hardlink finds duplicate files, it uses one of them as the master, removes all other duplicates, and places a hardlink for each one pointing to the master file.

   Download Hardlink from the Fedora site:

   http://fedora.secsup.org/linux/core/5/i386/os/Fedora/RPMS/hardlink-1.0-1.21.2.i386.rpm

5. Install Hardlink using one of the methods described below

   — Run the following command:

   rpm -ivh hardlink-1.0-1.21.2.i386.rpm

   — Run the following command through yum utility, making sure the computer is connected to the Internet:

   yum install hardlink

   Hardlink is now installed and will resolve any dependencies.

6. Compile the module using the following commands:

   cd IEGD_Patches/IKM

   ./install.sh

   depmod -a

   modprobe iegd_mod

7. Modify your xorg.conf file to include a device section for this driver and a Monitor section for your display. See Section 7.7.1, "Configuration Overview" on page 161 for details on the driver configuration and the list of supported options. The default installation location for this file is `/etc/X11`.

Note: The CED utility generates generic Linux X config samples using "`Screen0`" and "`Monitor0`" entries but a standard install of NLPOS seems to require "`Screen[0]`" and "`Monitor[0]`" in the `Xorg.conf` file.

8. Reboot.

At this point, when X Windows starts, it should be using the IEGD driver. To verify this, you can check your Xorg log, or run the `iegdgui` utility (found in the Utilities directory).

Note: You may need to set the `iegdgui` file properties to be executable before it will run.

7.4.3 Installing IKM with SUSE

1. Install `kernel-source.$ARCH.rpm`.
2. Change to the `/usr/src/linux` directory.
3. Configure the kernel using

   make cloneconfig
4. Create files required for compiling external modules:

   make scripts && make prepare
5. Change to directory `/lib/modules/<runningkernelversion>`
6. Delete 'build' links by running

   rm -rf build
7. Create a build links file pointing to the kernel source:

   ln -s /usr/src/linux-<kernel_version> build
8. Compile the module using the following commands:

   cd IEGD_Patches/IKM

   ./install.sh

   depmod -a

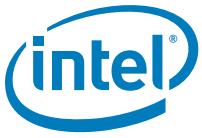
   modprobe iegd_mod
9. Modify your xorg.conf file to include a device section for this driver and a Monitor section for your display. See Section 7.7.1, "Configuration Overview" on page 161 for details on the driver configuration and the list of supported options. The default installation location for this file is `/etc/X11`.

Note: The CED utility generates generic Linux X config samples using "`Screen0`" and "`Monitor0`" entries but a standard install of NLPOS seems to require "`Screen[0]`" and "`Monitor[0]`" in the `Xorg.conf` file.

10. Reboot.

At this point, when X Windows starts, it should be using the IEGD driver. To verify this, you can check your Xorg log, or run the `iegdgui` utility (found in the Utilities directory).

Note: You may need to set the `iegdgui` file properties to be executable before it will run.



7.4.4 Using the IEGD Kernel Module

Note: This needs to be run after adding the IEGD components and updating the symbolic links in the Linux OS.

An installation script is provided to perform the installation. The script generates the Makefile together with the compilation environment for that particular kernel or distro. To install the IKM, run the shell script provided:

cd IEGD_Patches/IKM

./install.sh

The installation script detects the kernel version and points to the proper header files location before creating the Makefile. The script then calls the Make program to start the compilation process. After compilation is complete, the script tries to install the IKM. If the script is run from a normal user, it prompts for the superuser password before copying the generated file and then runs the following command to resolve module dependencies:

depmod -a

To insert the module into the kernel, run

modprobe iegd_mod

This will load all the modules that iegd_mod depends on before loading iegd_mod itself.

IKM installation requires a matching kernel source tree and a working Linux build system. Some of the programs require some additional libraries; for example, a hard link is required by Fedora Core 5.

7.5 Uninstalling the IKM

To uninstall the IKM, run the install script described in Section 7.4.4 on page 159 with following argument:

./install.sh uninstall

This deletes the IKM file from kernel module location and invokes

depmod -a

to resolve dependencies for other module. A reboot might be required because IKM cannot be removed through the `rmmod` utility if a previous agpgart is part of the kernel image.

7.6 GART Patch Instructions (for Fedora Core 5 Only)

Note: You must have your kernel source and GCC installed for this procedure. If you do not have these installed, you will need to follow the kernel install instructions on the Redhat Web site and use your software installer for development tools. At this point you should also ensure that the kernel is set for loadable modules.

You can use the IKM method described in Section 7.4, "IKM Patch Instructions" on page 156 instead of using the instructions below. You need to use only one of the patching methods, either the IKM or the GART patch.

To patch a Linux 2.6.xxxx kernel with the GART changes:

1. cd into the kernel source directory (e.g., `cd /usr/src/linux-2.x.xxxx`). For FC5 it is:

   cd /usr/src/redhat/BUILD/kernel-2.6.15/linux-2.6.15.i686

2. Execute the patch command (e.g., `patch -p1 < .../IEGD_Patches/Driver/agpgart.patch-2.x.xxx`). For FC5 use:

   patch -p1 < …/IEGD_Patches/Driver/agpgart.patch-2.6.15-1.2054

To update the kernel:

1. cd to the kernel source directory (see above).
2. Since the agpgart is typically built-in by default, it needs to be configured to install it as a module before updating the kernel.

   make menuconfig

   From the configuration menu, select `Device Drivers`, then `Character Devices`. Scroll down to `/dev/agpgart`. Change the disposition to `M` (for Module). Exit from the configuration menu and save your changes.

3. Execute the make all modules command:

   make all

4. Install the modules and build the kernel:

   make modules_install && make install

5. Reboot.
6. If you did not edit the `grub.conf` file, select the custom kernel.

Note: Make sure that the newly updated kernel is the one being booted.

7. Run the following commands:

   modprobe agpgart

   modprobe intel-agp

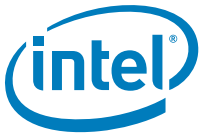
8. To ensure the agpgart modules are loaded after a reboot, add the following line to the `/etc/modprobe.conf` file:

   alias char-major-10-175 intel-agp

9. Modify your `xorg.conf` file to include a device section for this driver and a Monitor section for your display. See Section 7.7.1 for details on the driver configuration and the list of supported options. The default installation location for this file is `/etc/X11`.

Note: You may need to disable SELinux security to allow the IEGD driver to load. Alternatively, you will need to configure SELinux to allow IEGD to operate.

10. Reboot.



At this point, when X Windows starts, it should be using the IEGD driver. To verify this, you can check your Xorg log, or run the `iegdgui` utility (found in the Utilities directory).

Note: You may need to set the `iegdgui` file properties to be executable before it will run.

7.7 Editing the Linux OS Configuration File

This section describes how to edit the Linux configuration files (XF86Config or xorg.conf) for use with the IEGD.

The Intel Linux driver is for use with the integrated graphics of Intel chipsets on the Embedded Intel Architecture roadmap. The driver supports 8-, 16- and 24-bit pixel depths, dual independent head setup on capable hardware, flat panel, hardware 2D acceleration, hardware cursor, the XV extension, and the Xinerama extension.

7.7.1 Configuration Overview

IEGD auto-detects all device information necessary to initialize the integrated graphics device in most configurations. However, you can customize the IEGD configuration for any supported display by editing the X-server's configuration file (`XF86Config` or `xorg.conf`). Please refer to the `XF86Config(5x)` or `Xorg(5x)` man page for general configuration details. This section only covers configuration details specific to the IEGD.

To configure the IEGD for the Linux OS, you must edit the X server's configuration file (`XF86Config` or `xorg.conf`). You can either edit the configuration directly or you can use CED to create configurations that must then be copied into the configuration file. Even if you use CED to create a configuration, you must still edit the Linux configuration file.

The following topics describe methods of configuring the IEGD for the Linux operating system.

- "Linux OS Configuration Using CED"
- "Editing the Linux OS Configuration File Directly"
- "XFree86 and Xorg Configuration Options"
- "Sample Dual Independent Head (DIH) Configuration"
- "Video Memory Management"
- "Configuring Accelerated Video Decode for IEGD driver and US15W"
- "Graphics Port Initialization"
- "OpenGL Support"

See also The Linux OS Configuration File for details on the file structure.

7.7.2 Linux OS Configuration Using CED

You can configure the Linux driver settings using CED as described in Section 3.0, "Platform Configuration Using CED" on page 25.

The output file (`yourbuildnamehere.x`) from CED contains the settings required to configure the IEGD for Linux systems and can be pasted into the appropriate sections of the XF86Config or `xorg_conf` file.

7.7.3 Editing the Linux OS Configuration File Directly

Instead of using the CED, as described in Section 3.0, "Platform Configuration Using CED" on page 25, you can edit the `XF86Config` or `xorg.conf` file directly. The following procedure outlines the steps to follow when editing the Linux configuration file. Section 7.7.4, "The Linux OS Configuration File" on page 162 provides details on each section of the configuration file.

1. Log in as root and open the configuration file for editing. The configuration file is typically located in the `/etc/X11` directory but may be located elsewhere on your system.
2. In the `Device` section of the configuration file, enter the appropriate information for your driver. The configuration file must have at least one `Device` section. The `Device` section lets you define information about IEGD. You can use a single `Device` section for single, twin, or clone configurations. For Dual Independent Head configurations, you must specify a second `Device` section.
3. In the `Screen` section, enter information for each display in your configuration. The configuration file must have at least one `Screen` section. The `Screen` section binds a `Device` with a `Monitor` and lets you define resolution modes for the display. The Screen section is referenced in the `ServerLayout` section of the configuration file.
4. In the `Monitor` section, define monitor specifications and timings that will be used for the display. You must have a `Monitor` section defined for each display in your configuration. The `Monitor` section is referenced by the `Screen` section.
5. Save your changes to the file. For systems booted to run level 3, startx to restart. For systems booted to run level 5, kill X (Alt-backspace) to restart. Reboot if necessary.

7.7.4 The Linux OS Configuration File

To configure the IEGD for use with the Linux OS, you must edit the Linux configuration file (`XF86Config` or `xorg.conf`). There are several sections within the configuration that must be edited or created, including:

- "Device Section"
- "Screen Section"
- "Monitor Section"
- "ServerLayout Section" (when configuring DIH)
- "ServerFlags Section" (when configuring Xinerama)

The above Sections are described following the Sample XF86Config File. Please see the `XF86Config` or `xorg.conf` man pages for complete details.

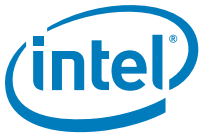
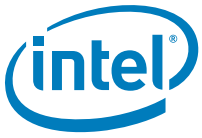


Figure 37. Sample XF86Config File

```
# XFree86 4 configuration created by redhat-config-xfree86
Section "ServerLayout"
        Identifier     "Default Layout"
        Screen      0  "Screen0" 0 0
        InputDevice    "Mouse0" "CorePointer"
        InputDevice    "Keyboard0" "CoreKeyboard"
        InputDevice    "DevInputMice" "AlwaysCore"
EndSection
Section "Files"
# RgbPath is the location of the RGB database.  Note, this is the name of the
# file minus the extension (like ".txt" or ".db").  There is normally
# no need to change the default.
# Multiple FontPath entries are allowed (they are concatenated together)
# By default, Red Hat 6.0 and later now use a font server independent of
# the X server to render fonts.
        RgbPath      "/usr/X11R6/lib/X11/rgb"
        FontPath     "unix/:7100"
EndSection
Section "Module"
        Load  "dbe"
        Load  "extmod"
        Load  "fbdevhw"
        Load  "glx"
        Load  "record"
        Load  "freetype"
        Load  "type1"
EndSection
Section "InputDevice"
# Specify which keyboard LEDs can be user-controlled (eg, with xset(1))
#   Option"Xleds""1 2 3"
# To disable the XKEYBOARD extension, uncomment XkbDisable.
#   Option"XkbDisable"
# To customise the XKB settings to suit your keyboard, modify the
# lines below (which are the defaults).  For example, for a non-U.S.
# keyboard, you will probably want to use:
#   Option"XkbModel""pc102"
# If you have a US Microsoft Natural keyboard, you can use:
#   Option"XkbModel""microsoft"
#
# Then to change the language, change the Layout setting.
# For example, a German layout can be obtained with:
#   Option"XkbLayout""de"
# or:
#   Option"XkbLayout""de"
#   Option"XkbVariant""nodeadkeys"
#
# If you would like to switch the positions of your capslock and
# control keys, use:
#   Option"XkbOptions""ctrl:swapcaps"
```

```
# Or if you just want both to be controled, use:
#   Option"XkbOptions""ctrl:nocaps"
#
        Identifier  "Keyboard0"
        Driver      "keyboard"
        Option    "XkbRules" "xfree86"
        Option    "XkbModel" "pc105"
        Option    "XkbLayout" "us"
EndSection
Section "InputDevice"
        Identifier  "Mouse0"
        Driver      "mouse"
        Option    "Protocol" "IMPS/2"
        Option    "Device" "/dev/psaux"
        Option    "ZAxisMapping" "4 5"
        Option    "Emulate3Buttons" "no"
EndSection
Section "InputDevice"
# If the normal CorePointer mouse is not a USB mouse then
# this input device can be used in AlwaysCore mode to let you
# also use USB mice at the same time.
        Identifier  "DevInputMice"
        Driver      "mouse"
        Option    "Protocol" "IMPS/2"
        Option    "Device" "/dev/input/mice"
        Option    "ZAxisMapping" "4 5"
        Option    "Emulate3Buttons" "no"
EndSection
Section "Monitor"
        Identifier   "Monitor0"
        VendorName   "Monitor Vendor"
        ModelName    "Eizo T550"
        DisplaySize  290220
        HorizSync    30.0 - 82.0
        VertRefresh  50.0 - 160.0
        Option    "dpms"
EndSection
Section "Monitor"
        Identifier   "Monitor1"
        VendorName   "SamSung"
        ModelName    "SyncMaster 192MP"
EndSection
Section "Device"
        Identifier "Intel_IEGD-0"
        Driver     "iegd"
        VendorName "Intel(R) DEG"
        BoardName  "Embedded Graphics"
        BusID      "0:2:0"
        Screen      0
        Option     "PcfVersion"                  "1792"
        Option     "ConfigId"                    "1"
        Option     "ALL/1/name"                  "IEGD_Example"
        Option     "ALL/1/General/PortOrder"     "42000"
```



```
        Option     "ALL/1/General/DisplayConfig"  "1"
        Option     "ALL/1/General/DisplayDetect"  "0"
        Option     "ALL/1/General/CloneRefresh"   "60"
        Option     "ALL/1/General/CloneWidth"     "1280"
        Option     "ALL/1/General/CloneHeight"    "1024"
        Option     "ALL/1/Port/4/name"                    "Sample_LVDS"
        Option     "ALL/1/Port/4/General/EdidAvail"       "3"
        Option     "ALL/1/Port/4/General/EdidNotAvail"    "1"
        Option     "ALL/1/Port/4/General/Rotation"        "0"
        Option     "ALL/1/Port/4/General/Edid"            "1"
        Option     "ALL/1/Port/2/name"                    "Sample_SDVO"
        Option     "ALL/1/Port/2/General/EdidAvail"       "3"
        Option     "ALL/1/Port/2/General/EdidNotAvail"    "1"
        Option     "ALL/1/Port/2/General/Rotation"        "0"
        Option     "ALL/1/Port/2/General/Edid"            "1"
        Option     "PortDrivers"                          "lvds sdvo"
EndSection
Section "Screen"
        Identifier      "Screen0"
        Device          "IntelIEGD-1"
        Monitor         "Monitor0"
        DefaultDepth    16
        SubSection "Display"
            Depth   16
            Modes   "800x600" "640x480"
        EndSubSection
EndSection
Section "DRI"
        Group       0
        Mode        0666
EndSection
```

7.7.4.1 Device Section

The `Device` section provides a description of a graphics device. The Linux configuration file (`XF86Config` or `xorg.conf`) must have at least one `Device` section for the graphics driver. If your chipset supports multiple graphics pipelines, you may have multiple `Device` sections, but in most situations, only one is required. If you are creating a Dual Independent Head (DIH) configuration, you must have at least two `Device` sections.

The `Device` sections in the `XF86Config` and `xorg.conf` configuration files have the following format:

```
Section “Device”
      Identifier “devname”
      Driver “iegd”
      ...
EndSection
```

The `Identifier` field defines the device. This name is used to associate the device with a screen and is referenced in `Screen` sections.

`The Driver` field defines the driver to use and is a required field in the `Device` section. The intel driver, `intel_drv.o`, must be installed in the `/usr/X11R6/lib/modules/drivers` directory.

The remainder of the Device section can contain IEGD-specific options. Please see Table 37 on page 167 for a list and description of IEGD supported options.

7.7.4.2 Screen Section

The Screen section is used to bind a Screen with a Device and a Monitor. It is also used to define resolution modes, color depths, and various other screen characteristics. Please see the XF86Config or xorg man page for detailed information.

The Screen section has the following format:

```
Section "Screen"
      Identifier     "screenname"
      Device         "devname"
      Monitor        "Monitor0"
      DefaultDepth   24
      Subsection "Display"
         Depth      24
         Modes      "1280x1024" "1024x768" "800x600" "640x480"
      EndSubSection
EndSection
```

7.7.4.3 Monitor Section

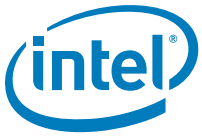
Use the Monitor section to define monitor characteristics and timings for a display. You should have one Monitor section for each display your system supports. The Monitor section is referenced in a Screen section and has the following format.

```
Section "Monitor"
      Identifier "Monitor0"
      VendorName "NEC"
      MonitorName "MEC MultiSync LCD"
      HorizSync      30-60
      VertRefresh    50-75
      ...
EndSection
```

7.7.4.4 ServerLayout Section

The ServerLayout section defines the overall layout of the system configuration. Input devices are specified in the InputDevice fields and output devices usually consist of multiple components (such as a graphics board and a monitor, which are bound together in a Screen section). You typically need to edit this section only when you are using a DIH configuration. You need to add a line to reference the second Screen section and specify its relative location to the first screen. In the following sample, the line beginning with Screen 1... is required for DIH configurations.

```
Section "ServerLayout"
      Identifier "Default Layout"
      Screen 0 "Screen0" 0 0
      Screen 1 "Screen1" RightOf "Screen0"
      InputDevice entries...
EndSection
```



7.7.4.5 ServerFlags Section

If you are configuring the IEGD for Xinerama support, you must set the "Xinerama" option to "True" in the ServerFlags section of the configuration file.

```
Section "ServerFlags"
      Option "Xinerama" "True"
EndSection
```

7.7.5 XFree86 and Xorg Configuration Options

The IEGD provides a format syntax for Linux configuration options. The syntax is similar to the Microsoft Windows* INF file and is as follows:

"All/<ConfigID>/<block name>/<option name>"

The IEGD parses the configuration options and looks for "new-style" 4.0 and later options. If it does not find any, then it falls back to processing old-style options.

The device configuration must contain the "pcfversion" option with value "0x700". This indicates to the driver the options format to use. Earlier pcfversions (0 and 0x400) are supported for backward compatibility.

The IEGD driver supports multiple sets of installed configuration options that may be selected at runtime.

Configuration ID 0 is used unless otherwise specified in the configuration file or supplied by the system BIOS.

The table below shows the supported driver options.

Table 37. Supported Driver Options (Sheet 1 of 3)

Option	Description
Option "PcfVersion" "integer"	This option indicates that the new IEGD format is being used for the Linux Configuration files (XF86Config or xorg.conf). The new format is hierarchical (similar to the Microsoft Windows* INF file) and allows both global and per-configuration information to be stored in the X-server's configuration file (XF86Config or xorg.conf) rather than having per-configuration information stored separately in the EDIDx.bin file. This option is usually set to 0400 hex (1024 decimal) and is required for the new format.
Option "SWCursor" "boolean"	Enable the use of the software cursor. The default is off and the hardware cursor is used.
Option "ShadowFB" "boolean"	Enable or disable double buffering on the framebuffer. The default is that double buffering is disabled.
Option "TearFB" "boolean"	Disable or enable wait for vblank when doing blits. The default is to not wait for vblank when doing blits. This is faster but may cause visible tearing of the display. Set to "1" (default) to not wait for vblank Set to "0" to wait for vblank to reduce tearing
Option "XVideo" "boolean"	Disable or enable XVideo support. In a dual independent head configuration, either the first display or the second display support XVideo. Both displays cannot support XVideo simultaneously. The default is XVideo support is enabled.

Table 37. Supported Driver Options (Sheet 2 of 3)

Option	Description
Option "XVideoBlend" "boolean"	Disable or enable XVideo support using the 3D blend manager. This provides XVideo support in configurations that cannot be supported with overlay. For example, this is supported on both displays in a dual independent head setup. It is also supported when the display is rotated or flipped. Color key is only supported if ShadowFB is enabled and the VideoKey is defined. The default is XVideoBlend support is enabled.
Option "ConfigID" "integer"	This option identifies the configuration.
Option "All/<ConfigID>/Name" "string"	A quoted string used to identify the configuration name.
Option "All/<ConfigID>/Comment" "string"	A quoted string used to identify the configuration file. Comment is a required field for Linux configurations.
Option "PortDrivers" "string"	This option specifies which port driver(s) must be loaded. The list is a space- or comma-separated list of port driver names corresponding to the *.so port driver files included with the Linux OS version of the driver. The port driver for the built-in analog output from the GMCH is always included and does not need to be specified in the PortDrivers option. The port drivers for the built-in LVDS and TV components (on chipsets with such features) are NOT automatically included. The "lvds" and "inttv" port drivers must be specified in order to use those output ports.
Option "All/<ConfigID>/General/PortOrder" "string"	This option can be used to change the default port allocation order. The default order can vary depending on chipset. List the port type numbers in the priority order starting from first to last. The port type numbers are as follows: 1 - Integrated TV Encoder (mobile chipsets only) 2 - sDVO B port 3 - sDVO C port 4 - Integrated LVDS port (mobile chipsets only) 5 - Analog CRT port To set the order as Integrated TV Encoder, ANALOG, LVDS, sDVO C, sDVO B set the PortOrder string to "15432". Zeros can be used to specify don't care. Setting this option incorrectly can result in port allocation failures.
Option "All/<ConfigID>/Port/<port number>/General/Rotation" "integer"	Rotate the display. Valid values are 0, 90, 180, 270.
Option "All/<ConfigID>/Port/<port number>/General/Flip" "boolean"	Invert the display horizontally.
Option "All/<ConfigID>/Port/<port number>/VideoKey" "integer"	This sets the color key for XVideo and XVideoBlend. This value is either a 24-bit value or a 16-bit value, depending on the pixel depth of the screen. The color key is always enabled for XVideo, even when it is not defined. The color key is always disabled for XVideoBlend unless both this option is defined and the ShadowFB option is enabled. The default color key for XVideo is 0x0000ff00. For XVideoBlend, the color key is disabled by default.
Option "All/<ConfigID>/Port/<port number>/CloneWidth" "integer"	This sets the display width for a clone port when CloneDisplay is active. The default is 640.
Option "All/<ConfigID>/Port/<port number>/CloneHeight" "integer"	This sets the display height for a clone port when CloneDisplay is active. The default is 480.

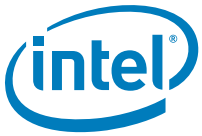


Table 37. Supported Driver Options (Sheet 2 of 3)

Option	Description
Option "XVideoBlend" "boolean"	Disable or enable XVideo support using the 3D blend manager. This provides XVideo support in configurations that cannot be supported with overlay. For example, this is supported on both displays in a dual independent head setup. It is also supported when the display is rotated or flipped. Color key is only supported if ShadowFB is enabled and the VideoKey is defined. The default is XVideoBlend support is enabled.
Option “ConfigID” “integer”	This option identifies the configuration.
Option “All/<ConfigID>/Name” “string”	A quoted string used to identify the configuration name.
Option "All/<ConfigID>/Comment" "string"	A quoted string used to identify the configuration file. Comment is a required field for Linux configurations.
Option "PortDrivers" "string"	This option specifies which port driver(s) must be loaded. The list is a space- or comma-separated list of port driver names corresponding to the *.so port driver files included with the Linux OS version of the driver. The port driver for the built-in analog output from the GMCH is always included and does not need to be specified in the PortDrivers option. The port drivers for the built-in LVDS and TV components (on chipsets with such features) are NOT automatically included. The "lvds" and "inttv" port drivers must be specified in order to use those output ports.
Option "All/<ConfigID>/General/PortOrder" "string"	This option can be used to change the default port allocation order. The default order can vary depending on chipset. List the port type numbers in the priority order starting from first to last. The port type numbers are as follows: 1 - Integrated TV Encoder (mobile chipsets only) 2 - sDVO B port 3 - sDVO C port 4 - Integrated LVDS port (mobile chipsets only) 5 - Analog CRT port To set the order as Integrated TV Encoder, ANALOG, LVDS, sDVO C, sDVO B set the PortOrder string to "15432". Zeros can be used to specify don’t care. Setting this option incorrectly can result in port allocation failures.
Option "All/<ConfigID>/Port/<port number>/General/Rotation" "integer"	Rotate the display. Valid values are 0, 90, 180, 270.
Option "All/<ConfigID>/Port/<port number>/General/Flip" "boolean"	Invert the display horizontally.
Option "All/<ConfigID>/Port/<port number>/VideoKey" "integer"	This sets the color key for XVideo and XVideoBlend. This value is either a 24-bit value or a 16-bit value, depending on the pixel depth of the screen. The color key is always enabled for XVideo, even when it is not defined. The color key is always disabled for XVideoBlend unless both this option is defined and the ShadowFB option is enabled. The default color key for XVideo is 0x0000ff00. For XVideoBlend, the color key is disabled by default.
Option "All/<ConfigID>/Port/<port number>/CloneWidth" "integer"	This sets the display width for a clone port when CloneDisplay is active. The default is 640.
Option "All/<ConfigID>/Port/<port number>/CloneHeight" "integer"	This sets the display height for a clone port when CloneDisplay is active. The default is 480.

Table 37. Supported Driver Options (Sheet 2 of 3)

Option	Description
Option "XVideoBlend" "boolean"	Disable or enable XVideo support using the 3D blend manager. This provides XVideo support in configurations that cannot be supported with overlay. For example, this is supported on both displays in a dual independent head setup. It is also supported when the display is rotated or flipped. Color key is only supported if ShadowFB is enabled and the VideoKey is defined. The default is XVideoBlend support is enabled.
Option "ConfigID" "integer"	This option identifies the configuration.
Option "All/<ConfigID>/Name" "string"	A quoted string used to identify the configuration name.
Option "All/<ConfigID>/Comment" "string"	A quoted string used to identify the configuration file. Comment is a required field for Linux configurations.
Option "PortDrivers" "string"	This option specifies which port driver(s) must be loaded. The list is a space- or comma-separated list of port driver names corresponding to the *.so port driver files included with the Linux OS version of the driver. The port driver for the built-in analog output from the GMCH is always included and does not need to be specified in the PortDrivers option. The port drivers for the built-in LVDS and TV components (on chipsets with such features) are NOT automatically included. The "lvds" and "inttv" port drivers must be specified in order to use those output ports.
Option "All/<ConfigID>/General/PortOrder" "string"	This option can be used to change the default port allocation order. The default order can vary depending on chipset. List the port type numbers in the priority order starting from first to last. The port type numbers are as follows: 1 - Integrated TV Encoder (mobile chipsets only) 2 - sDVO B port 3 - sDVO C port 4 - Integrated LVDS port (mobile chipsets only) 5 - Analog CRT port To set the order as Integrated TV Encoder, ANALOG, LVDS, sDVO C, sDVO B set the PortOrder string to "15432". Zeros can be used to specify don't care. Setting this option incorrectly can result in port allocation failures.
Option "All/<ConfigID>/Port/<port number>/General/Rotation" "integer"	Rotate the display. Valid values are 0, 90, 180, 270.
Option "All/<ConfigID>/Port/<port number>/General/Flip" "boolean"	Invert the display horizontally.
Option "All/<ConfigID>/Port/<port number>/VideoKey" "integer"	This sets the color key for XVideo and XVideoBlend. This value is either a 24-bit value or a 16-bit value, depending on the pixel depth of the screen. The color key is always enabled for XVideo, even when it is not defined. The color key is always disabled for XVideoBlend unless both this option is defined and the ShadowFB option is enabled. The default color key for XVideo is 0x0000ff00. For XVideoBlend, the color key is disabled by default.
Option "All/<ConfigID>/Port/<port number>/CloneWidth" "integer"	This sets the display width for a clone port when CloneDisplay is active. The default is 640.
Option "All/<ConfigID>/Port/<port number>/CloneHeight" "integer"	This sets the display height for a clone port when CloneDisplay is active. The default is 480.

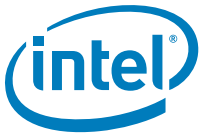


Table 37. Supported Driver Options (Sheet 3 of 3)

Option	Description
Option "All/<ConfigID>/Port/<port number>/CloneRefresh" "integer"	This sets the display vertical refresh rate for a clone port when CloneDisplay is active. The default is 60 Hz.
Option "All/<ConfigID>/Port/<port number>/Active" "boolean"	Make the device ports active and enabled on startup. This is the default. When this option is set to FALSE, the driver will allocate the output ports for this device but will not set the mode or enable the output. To enable the output, the port control extension must be used.
Option "All/<ConfigID>/Port/<port number>/EDID" "boolean"	Enable or disable reading of EDID data from the output port device. Note that if the EDID option is specified in the config file (`XF86Config` or `xorg.conf`), all per-port EDID options in the configuration are overwritten by the EDID option specified in the config file.

7.7.6 Sample Dual Independent Head (DIH) Configuration

For dual independent head operation, several additional options must be set in the `Device` sections for each head. Both `Device` sections must specify the `BusID`, and the `BusID` must be the same for both devices. Each `Device` section must specify the `Screen` section that will associate the device with the `Screen` option.

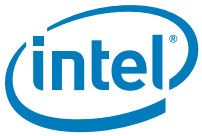
```
BusID  - B:F:S (Bus, Function, Slot)
Screen  - number
```

The example below shows a sample DIH configuration. Only the `Device`, `Screen`, and `Server Layout` sections of the configuration file are shown. For a complete example of a configuration file, see Figure 37 on page 163.

Figure 38. Sample DIH Configuration

```
Section "Device"
      Identifier "IntelEGD-1"
      Driver     "iegd"
      BusID       "0:2:0"
      Screen      0
      VideoRam    32768
EndSection
Section "Device"
      Identifier  "IntelEGD-2"
      Driver      "iegd"
      BusID       "0:2:0"
      Screen      1
      VideoRam    32768
EndSection
Section "Screen"
      Identifier  "Screen 1"
      Device      "IntelEGD-1"
      Monitor     "Monitor1"
      DefaultDepth 24
```

```
        Subsection "Display"
           Depth       8
           Modes       "1280x1024" "1024x768" "800x600" "640x480"
           ViewPort    0 0
        EndSubsection

        Subsection "Display"
           Depth       16
           Modes       "1280x1024" "1024x768" "800x600" "640x480"
           ViewPort    0 0
        EndSubsection

        Subsection "Display"
           Depth       24
           Modes       "1280x1024" "1024x768" "800x600" "640x480"
           ViewPort    0 0
        EndSubsection EndSection
Section "Screen"
        Identifier  "Screen 2"
        Device      "IntelEGD-2"
        Monitor     "Monitor2"
        DefaultDepth 24
        Subsection "Display"
           Depth       8
           Modes       "1280x1024" "1024x768" "800x600" "640x480
           ViewPort    0 0
        EndSubsection

        Subsection "Display"
           Depth       16
           Modes       "1280x1024" "1024x768" "800x600" "640x480"
           ViewPort    0 0
        EndSubsection

        Subsection "Display"
           Depth       24
           Modes       "1280x1024" "1024x768" "800x600" "640x480"
           ViewPort    0 0
        EndSubsection
EndSection
Section "ServerLayout"
        Identifier  "Dual Head Layout"
        Screen "Screen 1"
        Screen "Screen 2" Right Of "Screen 1"
        InputDevice "Mouse1" "CorePointer"
        InputDevice "Keyboard1" "CoreKeyboard"
EndSection
```



7.7.7 Video Memory Management

The Intel integrated graphics controllers have a unified memory architecture that uses system memory for video RAM. The amount of memory used is not fixed and depends on the configuration. Address space for framebuffers and back buffers is always reserved, along with some scratch space for 2D and 3D acceleration. Use of some features such as Video Overlay also affect video memory allocations. The amount of video memory allocated for pixmap surfaces is configurable. By default, 16 Mbytes is used for each screen. You can change this with the `VideoRam` option in the `Device` section of the configuration file. It may be set to any reasonable value up to 32 Mbytes. Increasing this value reduces the amount of system memory available for other applications. This value is in units of 1024 Kbytes (32 Mbytes is represented by 32768).

7.7.8 Configuring Accelerated Video Decode for IEGD driver and US15W

7.7.8.1 Hardware Video Acceleration

Hardware Video Acceleration is the use of a specialized video engine to decode video streams (such as MPEG2, MPEG4, h.264 and VC-1) in order to free up the processor from having to do all of the decoding. Only some chipsets (such as the US15W) support a video engine. The flow of video through the various components generally is as follows:

1. The video player (such as Splay) reads video file and determines the type.
2. Based on type, the proper codec shared library object is loaded.
3. The codec loads the VA library shared library object.
4. The VA library loads the `iegd_drv_video.so` shared library object.
5. The `iegd_drv_video_so` communicates, over the X wire protocol, with the IEGD X driver to send encoded video to the hardware for decoding.

You can either use the provided binary of the `libva` library found in the appropriate `Xorg/Xserver` directory in the IEGD release or build it from the source. To build it, use the following steps:

1. Untar `libva.tgz` included in the IEGD driver package in the Extras folder.
2. Enter the following commands:
   ./autogen.sh --prefix=/usr
   make
   make install

7.7.8.2 IEGD Driver

First install the IEGD Graphics Driver for Linux per the appropriate installation instructions in Section 7.3. IEGD should be fully configured and running X properly before installing the VA Library.

7.7.8.3 VA Library (version 0.29)

Install the VA shared library object on the library path using the steps below.

1. Untar `libva-0.29.tgz` included in the IEGD driver package.
2. Change to the directory where the library file is located (as built above or from the IEGD release) and copy the `libva.so.0.29.0` file to the `/usr/lib` folder using the following command:
   cp libva.so.0.29.0 /usr/lib
3. Create the library aliases:
   ```
   ln -s /usr/lib/libva.so.0.29.0 /usr/lib/libva.so
   ln -s /usr/lib/libva.so.0.29.0 /usr/lib/libva.so.0
   ```
4. Set the Libva environment variable to point to the correct folder:
   Set LIBVA_DRIVERS_PATH to point to the location of the IEGD driver. The following is an example (with the BASH command shell):
   export LIBVA_DRIVERS_PATH=/usr/lib/xorg/modules/

Your driver location may vary, so please use the path to where you installed the IEGD drivers. To make this "sticky" you may want to add this to your `.profile`, `bashrc` or whatever your distribution uses to store these variables.

7.7.8.4 IEGD Video Acceleration Driver

To install the provided Video Acceleration Driver (iegd_drv_video.so):

1. Change to the appropriate Xorg/Xserver directory where you unpacked the IEGD release.
2. Copy the driver to the directory where you installed the main IEGD driver from Section 7.3. For example:
   ```
   cp iegd_drv_video.so /usr/lib/xorg/modules/drivers
   ```

Note: Your actual directory will vary depending on your particular Linux distribution.

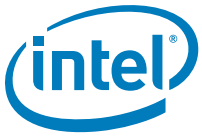
7.7.8.5 Helix Framework Installation

Set up the Helix Framework environment to use accelerated video playback with the IEGD driver using the steps below.

1. Download helix-dbu-server and splay plug-ins from:
   https://helix-client.helixcommunity.org/releases

   You will need to accept the Helix DNA Technology Binary Research Use License agreement and register an account before downloading the software.

   a. Download the latest Splay Plug-ins files.
      Example: `splay-plugin-atlas-01.2.0.tgz`
   b. Download the latest Helix DBUS Server files.
      Example: `helix-dbus-server-0.6.0.tar.bz2`
2. Untar `splay-plugins.tgz` and copy the contents to the directory `/usr/lib/helix/splay`
3. Untar `helix-dbus-server-0.6.0.tar.bz2`
4. Run `make install` from the `helix-dbus-server-0.6.0` directory.



7.7.8.6 Intel® Media Codec Installation

The latest EVALUATION ONLY versions of the VA API enabled hardware accelerated codecs for Helix are available on:

- The Intel VIP site (https://platformsw.intel.com/) or
- The Intel Premier Support site in the IEGD product org (premier.intel.com)

After you have the codec package, follow these steps to install it:

1. Untar the codec package.
2. Copy `libipp_hx_*.so` to `/usr/lib/helix/splay`
3. Remove `/usr/lib/helix/splay/mpgfformat.so`
4. Remove `/usr/lib/helix/splay/h264dec.so`
5. Remove `/usr/lib/helix/splay/mp4vrender.so`
6. Remove `/usr/lib/helix/splay/wmvrender.so`
7. Remove `/usr/lib/helix/splay/wmv9.so`

7.7.8.7 Playing Video

The video player application used must support the Helix plug-in framework. A sample player (splay) is included and known to work. The splay application is located in the `/usr/lib/helix/splay` directory.

To play a video, enter the following command:

- `/usr/lib/helix/splay/splay -l /usr/lib/helix/splay <Video file>` where <*Video file*> is replaced with an actual file name.

7.7.8.8 Troubleshooting

1. If the "splay" application quits silently, try removing the helix configuration files `~/.helix`, `~/.hxplayerrc`, and `~/.realplayerrc`.
2. Check the codec version numbers using the `script get_lib_version.sh` included with the codecs. The output should be:

```
libipp_hx_ac3ad.so    -> ipp_hx_version:20080401:1.8.4.1
libipp_hx_h264vd.so   -> ipp_hx_version:20080401:1.8.4.1
libipp_hx_mp2sp.so    -> ipp_hx_version:20080401:1.8.4.1
libipp_hx_mp2vd.so    -> ipp_hx_version:20080401:1.8.4.1
libipp_hx_mp4vd.so    -> ipp_hx_version:20080401:1.8.4.1
libipp_hx_vc1vd.so    -> ipp_hx_version:20080401:1.8.4.1
```

3. If you receive the following message when you try to play video:

```
libva: Trying to open /usr/X11R6/lib/modules/dri/
iegd_drv_video.so
libva: va_openDriver() returns -1
```

make sure the LIBVA_DRIVERS_PATH environment variable is set to the location where you installed the `iegd_drv_video.so` file. Typically this would be `/usr/lib/xorg/modules/drivers`.

7.7.9 Graphics Port Initialization

When used with a graphic chipset that supports multiple graphics pipelines, the driver supports multiple screens and Xinerama. This support is enabled by creating additional `Device` sections for each additional graphics device on the PCI bus. The driver locates the first device on the bus and associates it with the device section that matches (or one that does not specify a busID). This becomes the primary display. If the chipset supports multiple display pipes, and the config file specifies two `Device` sections and two `Screen` sections, the driver attempts to operate in a dual-independent head mode. After all the graphics devices and device sections have been matched up, the driver attempts to allocate any remaining output ports and attach them to the primary graphics device. For example:

- Two pipes and two ports allows for dual independent displays.
- One pipe and two ports allows for a cloned display (915GV special case).

7.7.10 OpenGL Support

The IEGD supports OpenGL* for the following Intel chipsets:

- Mobile Intel® GME965 Express chipset
- Intel® Q965 Express chipset
- Mobile Intel® 945GSE Express chipset
- Mobile Intel® 945GME Express chipset
- Intel® 945G Express chipset
- Mobile Intel® 915GME Express chipsets
- Intel® 915GV Express chipsets
- Mobile Intel® 910GMLE Express chipset

The OpenGL implementation for IEGD consists of three components.

- `libGL`: This is the shared library that implements the OpenGL and GLX APIs. It is linked by all OpenGL applications.
- `iegd.ko`: This is the Direct Rendering Manager (DRM). It is a kernel module that provides the OpenGL application with the permissions necessary to directly access the DMA buffers used by libGL.
- `X Server`: The existing IEGD X server driver has been enhanced to communicate with `libGL`.

Installing the IEGD OpenGL driver provides a fully hardware accelerated implementation of the OpenGL library to applications. This implementation makes use of a Direct Rendering technology, which allows the client to directly write to DMA buffers that are used by the graphics hardware.

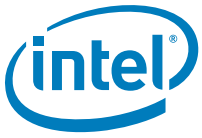
Due to the use of direct rendering technology, system designers should take special care to ensure that only trusted clients are allowed to use the OpenGL library. A malicious application could otherwise use direct rendering to destabilized the graphics hardware or, in theory, elevate their permissions on the system.

A system designer can control the access to the direct rendering functionality by limiting the access to the DRI device file located at:

```
/dev/dri/card0
```

The permissions on this device are set by the X-Server using the information provided in the "DRI" section of the configuration file.

See also "OpenGL Installation".



7.7.10.1 OpenGL Installation

To install the IEGD libGL onto a system, copy the `libGL.so` binary from the package to the standard location. Then compile and install the Direct Rendering Manager (DRM) kernel module from the sources provided. Lastly, enable the DRI option in the X server's configuration file.

Note: The system likely has a version of libGL already installed. You may want to make a backup copy of the existing library before installing the IEGD library. There are two libGL binaries included in the package that each support different hardware types. Only the one supporting target hardware should be installed. Chipsets of generation 3 and earlier (945 and earlier) should use libGLgn3.so and generation 4 chipsets (965 and above) should use libGLgn4.so.

1. cd IEGD_x_Linux/Driver/Xorg-X11R6.8/

Note: This example is specific to Fedora Core 5. The locations and commands may be different for a specific LINUX distribution.

2. cp libGLgn#.so /usr/lib/libGL.so.1.2
   - For 910GMLE, 915GME, 915GV, 945GME, 945G
     Use libGLgn3.so
   - For Q965, GME965
     Use libGLgn4.so
   - Q35 is not supported in 9.0 (no 3D)
3. cd /usr/lib
4. ln -sfv /usr/lib/libGL.so.1.2 libGL.so
5. ln -sfv /usr/lib/libGL.so.1.2 libGL.so.1
6. ldconfig

Note: **Skip steps 7 and 8 if you are using the IKM method detailed in** Section 7.4**.**

7. cd IEGD_Patches/Drm
8. make
   This will build and install the kernel module for the currently running kernel. If another kernel is installed or used, this step must be performed again.
9. depmod -Ae
10. Restart the X-Server or restart the system.

7.7.10.2 OpenGL Use Considerations

Allocation of Mipmaps and Memory Usage

Under normal circumstances the OpenGL driver will allocate all mip levels for a texture at allocation time. This is due to the fact that the OpenGL API allows an application to make use of the mips without first conveying an intention to do so. All mips are therefore available all the time.

The IEGD OpenGL driver has a special-case scenario to prevent the allocation of mips when the application can ensure that they will never be populated or used. On some hardware configurations this can save 50% on texture memory usage. To enable this feature, the application should do the following:

Using glTexParameter*(), set the GL_TEXTURE_MAX_LEVEL parameter to 0 before populating the texture (before any call to glTexImage2D()). This will prevent mips 1-N from being allocated but will not prevent them from being used. If the mips are inadvertently used, the results are undefined.

7.7.11 Sample Advanced EDID Configurations for Linux OS

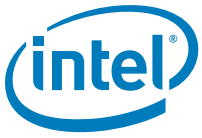
The `edid_avail` and `edid_not_avail` parameters control the available timings for any display. The `edid_avail` parameter is used when EDID information is read from the display. If the driver is unable to read EDID information from the display or if the `edid` parameter in the config file is set to "`0`" (disable), then the settings of the `edid_not_avail` parameter are used. Please see Section 3.0, "Platform Configuration Using CED" on page 25.

There is an `edid` option that can be placed in the XF86Config or xorg.conf files that controls the behavior of the overall driver, and there are also EDID settings within CED that control the behavior on each port (`edid`, `edid_avail`, and `edid_not_avail`). The combination of these settings determines how the driver behaves. The table below shows various configurations and the expected behavior of the driver.

Table 38. Sample Advanced EDID Configurations for Linux OS

Case	XF86Config "edid" option	CED: Per port "edid" option	Expected driver behavior
1.	No "`edid`" option specified	No `edid` flag specified	For every port, driver uses `edid_avail`.
2.	No "`edid`" option specified	`edid`=0 for some ports and `edid`=1 for some ports	For edid=0 ports, driver uses `edid_not_avail` flags. For edid=1 ports driver uses `edid_avail` flags.
3.	"`edid`"=no	Setting does not matter.	For all ports driver will not read `edid` and interprets `edid_not_avail` flags. Driver overrides any per-port `edid` flags, treats all displays as EDID-less displays, and uses `edid_not_avail` flags.
4.	"`edid`"=yes	`edid`=0 for some ports and `edid`=1 for some ports	Same as case 2

Notes: For all cases:
1. If there is not an `edid_not_avail` flag specified for a port, and an EDID-less display is detected, the driver defaults to using the standard built-in timings for that port.
2. If there is not an `edid_avail` flag specified for a port, and an EDID display is detected, the driver defaults to using the EDID data from the display, plus any user specified DTDs.
3. If `edid=1` and the display device is EDID-less, the driver uses `edid_not_avail` flags.



7.7.12 AGP GART Errors

The following are the most common AGP GART errors:

1. Symptom: No "agpgart: " in the system log
   Cause: The IEGD AGP GART patch has not been applied to the system.
2. Symptom: The Xorg.0.log has the following:
   (EE) INTEL(0): gart.c: Acquire IOCTL failed
   Cause: The IEGD AGP GART module has not been loaded.
3. Symptom: When starting the X server, the following message is listed in the X log file.
   "Graphics hardware initialization failed.
   The most likely cause is a missing or incorrect agpgart kernel module.
   module_init returned -1"
   Cause: The agpgart kernel module is not loaded or does not support the chipset being used. Check the kernel messages for the message:
   "agpgart: Detected an Intel xxxx chipset"
   If this message is there, the agpgart is not the problem.

7.8 Using the IEGD Runtime Configuration GUI on Linux Systems

You can change the configuration and runtime attributes of the driver using the iegdgui runtime configuration tool. The `iegdgui` resides in the `/Utilities` directory. The Intel Embedded Graphics Drivers GUI (iegdgui) is a GUI application that is used to view and control the Intel Embedded Graphics Drivers. This tool is used to retrieve status information of the display and driver and also to configure the supported display attributes. The iegdgui also demonstrates multi-monitor support.

7.8.0.1 iegdgui Setup

To run the `iegdgui`, you need to ensure that the XServer has been configured to use the IEGD. See Section 7.7.1, "Configuration Overview" on page 161 for details on configuring and installing the IEGD.

You need GTK+ and libglade, which are part of the Linux distribution and should already be installed.

Ensure that the LD_LIBRARY_PATH environment variable points to the X11R6 library. If it does not, type the following command:

export LD_LIBRARY_PATH=/usr/X11R6/lib

Ensure the `iegdgui` is executable by changing directories to .../`IEGD_X_x_Linux/Utilities` (where X is the major version and x is the minor version) and running the following command:

ls -l iegdgui

Executable permissions should be set for all three Linux groups (user, group, world) and should look like this:

rwxr_xr_x iegdgui ...

If the permissions do not contain an "x" for each group, change the permissions using the following command:

chmod +rwx iegdgui

After you have completed this step, the IEGD can be launched.

7.8.0.2 Using the iegdgui Runtime Configuration Utility

The iegdgui application provides four tabs: **Driver Info**, **Display Config**, **Display Attributes,** and **Color Correction**.

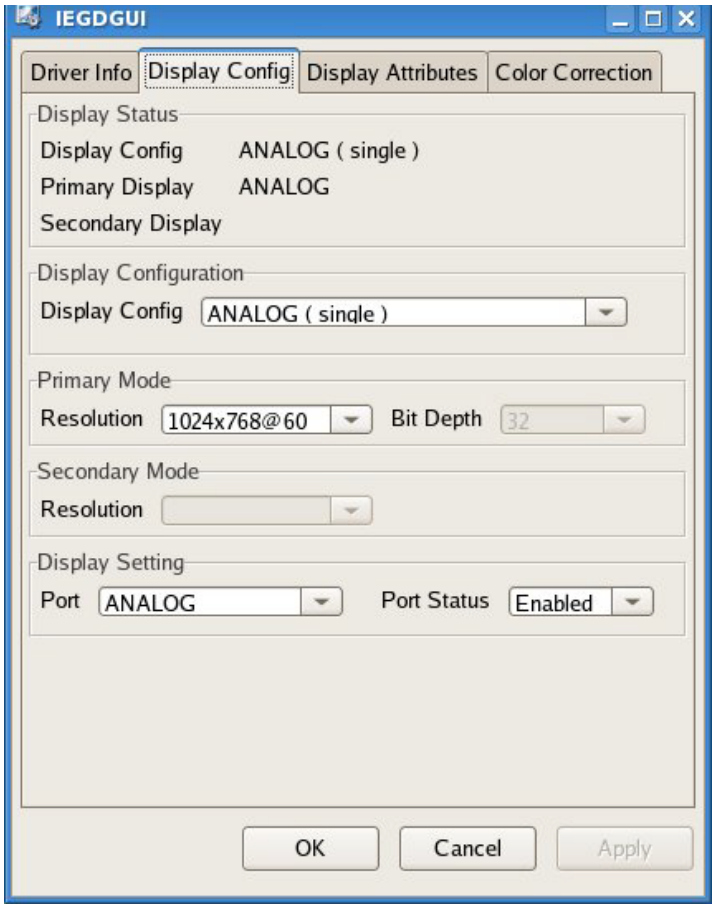
- **Driver Info**: Contains the driver information.
- **Display Config**: Contains current display information and allows configuration of display configurations, display resolutions for primary and secondary displays and enabling/disabling of a specified port.
- **Display Attributes**: Contains the supported Port Driver (PD) attributes and allows configuration of PD attributes.
- **Color Correction** — Contains current color-correction information for the framebuffer and overlay. Using this tab, you can change the framebuffer and overlay color settings.

To view current display information and or to change the current configuration of display configurations, display resolutions of the primary and secondary displays and enabling/disabling of a specified port, click the **Display Config** tab.

Note: If you make any changes to the configuration, click **Apply** for the changes to take effect.

The figure below shows a sample configuration.

Figure 39. Example Linux Runtime Configuration GUI — Display Config Tab



The **Display Status** section of the above dialog shows the current configuration for the **Primary** and **Secondary** displays.

In the **Display Configuration** section of the dialog, select the required display configuration in the **Display Config** drop-down list. This allows the user to choose between Single, Twin, Clone and Extended for all connected ports. A maximum of two ports per display configuration is currently allowed.

In the **Primary Mode** and **Secondary Mode** sections of the dialog, you can change display resolutions via the **Resolution** drop-down list.

In the **Display Settings** section of the dialog, you can view and change the settings for a port and then rotate and flip the display via the appropriate drop-down lists:

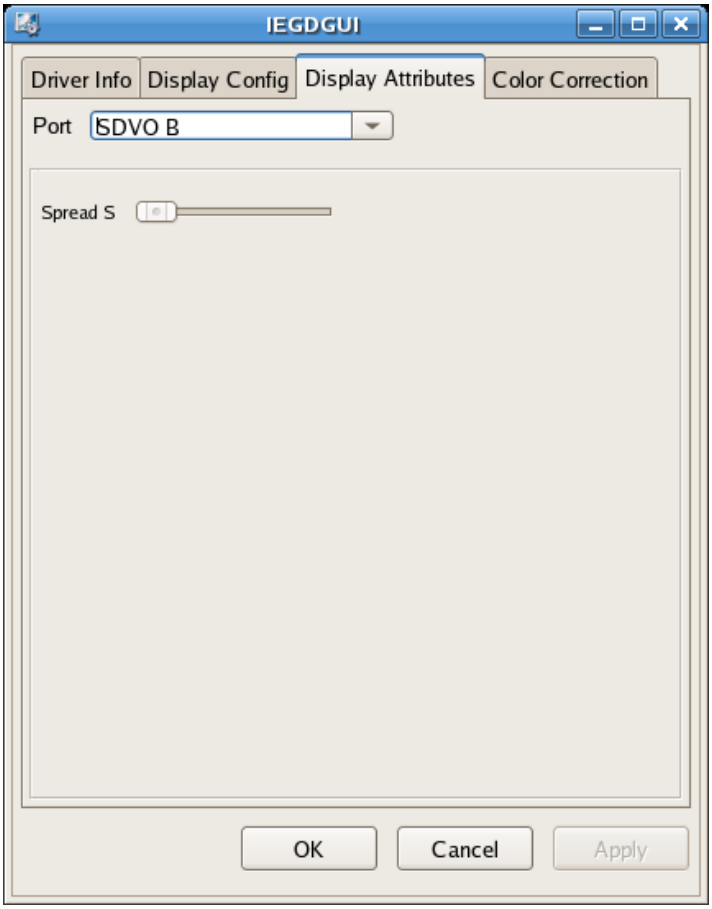
- **Port**: Allows you to select the required port.
- **Port Status**: Allows you to enable or disable the selected port.

Note: If you change any configuration settings in the **Display Config** dialog box, click **Apply** for the changes to take effect.

To view or change the attributes for a port, click the **Display Attributes** tab. The figure below shows a sample configuration. Please see Appendix B for detailed information on port driver attributes.

Note: If you make any changes to the port driver attributes, click **Apply** for the changes to take effect.

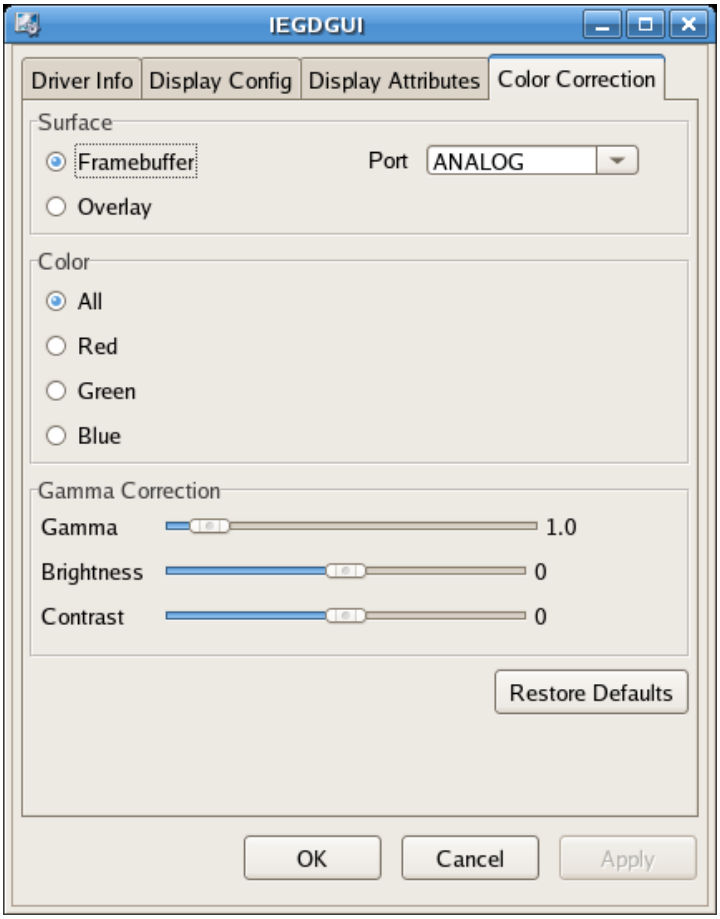
Figure 40. Example Linux Runtime Configuration GUI — Display Attributes Tab



To view and change color corrections, click the **Color Correction** tab. The figures above and below show sample Color Correction tab screens for **Framebuffer** and **Overlay**, color correction values for which are shown in Table 31 and Table 32.

Note: If you make any changes to the color-correction attributes, click **Apply** for the changes to take effect.

Figure 41. Example Linux Runtime Configuration GUI — Color Correction Tab (Framebuffer)



The following steps present an example color-correction procedure:

a. Select **Framebuffer** in the **Surface** section and select the appropriate port for the color correction to be applied to or select **Overlay** in the Surface section for color correction to be applied to the overlay.

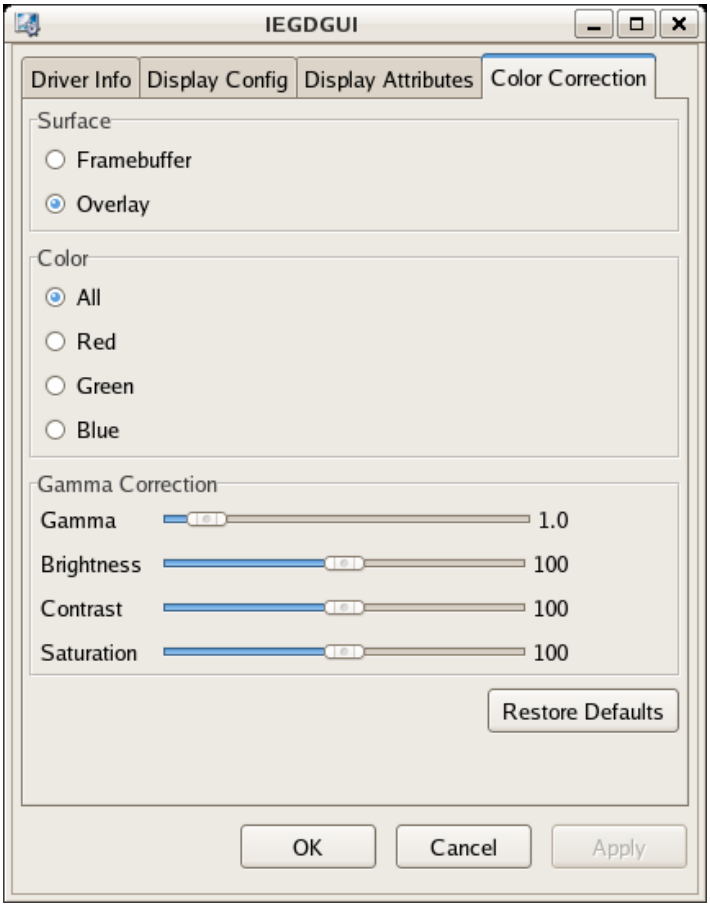
b. Select the required color to be corrected in the **Color** section.

c. Select the required color attribute to be corrected in the **Gamma Correction** section.

d. Click **Restore Defaults** to restore the default values.

Figure 42. Example Linux Runtime Configuration GUI — Color Correction Tab (Overlay)

This



7.9 Enabling Damn Small Linux

This section describes IEGD enablement of Damn Small Linux* (DSL) operating system. For further DSL-specific information and downloads, please see http://www.damnsmalllinux.org.

See also the following IEGD/DSL-related topics:

- Damn Small Linux OS Introduction
- XFree86 Versus TinyX
- Running IEGD on Damn Small Linux OS
- RAM Size Constraint
- Shrinking and Modifying the Extension

7.9.1 Damn Small Linux OS Introduction

DSL is intended for "liveCD" or "frugal" installs, i.e., installed on a USB flash drive for booting, which should only require 50 Mbytes. You can also experiment with a hard-disk installation. Note that a minimal, 50-Mbyte flash drive installation is compressed and loaded at boot time. This "original" DSL filetree is permanent and always gets restored after each reboot. In addition, the compressed filetree is always located in the first partition of the flash drive.

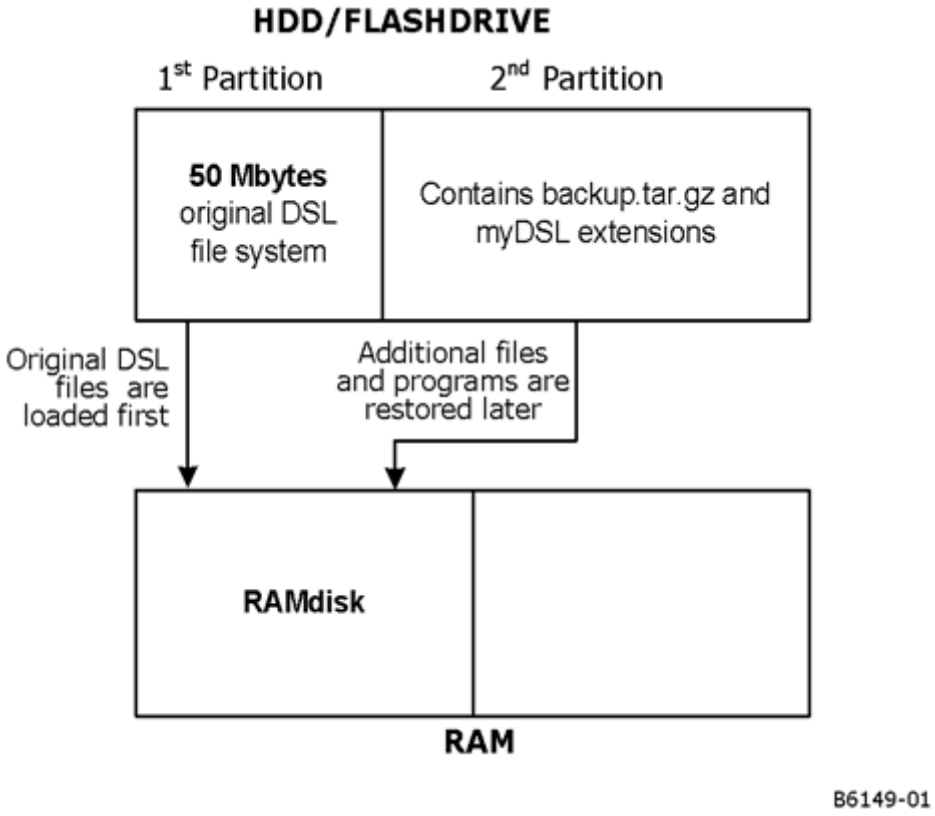
7.9.1.1 Extending Damn Small Linux OS

A second USB-drive partition (sda2) can be used for additional programs or files for use with DSL; see the figure below. An external partition or CD-ROM can also be used for such additional items. Note that for users to write or create new files, DSL uses a RAMdisk, which means that at boot time a certain portion of the RAM is reserved as disk space.

There are two main methods of adding items to DSL:

- myDSL — an extension package that adds new programs to DSL at boot time. Please see the myDSL page (http://distro.ibiblio.org/pub/linux/distributions/damnsmall/mydsl/) for extensions and installation instructions.
- Backup restore script in the original DSL file system.

Figure 43. Damn Small Linux Partition



7.9.2 XFree86 Versus TinyX

DSL uses TinyX* by default. To use XFree86* instead, see "XFree86.dsl.info" in the XFree86.dsl package (http://distro.ibiblio.org/pub/linux/distributions/damnsmall/mydsl/system/XFree86.dsl). Two changes, which are outlined in XFree86.dsl.info, are required to enable XFree86.

Note: IEGD works only with the optional XFree86 X server, not the default TinyX X server.

7.9.3 Running IEGD on Damn Small Linux OS

After the XFree86 package has been installed, you can install and use the IEGD driver. The first thing to do is to try to get the agpgart compiled before running XFree. The current DSL version, 2.3, uses kernel 2.4.26. The agpgart patch 2.4.24 (agpgart.patch-2.4.24) is also used. Kernel source for 2.4.26 can be found on http://www.kernel.org/.

Due to the limited space/resources that a DSL system may have, it is advisable to compile the agpgart in another distro before transferring it into DSL. Remember also to disable version information in make menuconfig when you attempt to compile in an external distro. During enablement of DSL, a RH9 distro is used to patch and compile the agpgart in kernel 2.4.26.

Using RH9 as an example, execute the following commands:

1. Copy linux-2.4.26.tar.gz to /usr/src/
2. cd /usr/src
3. tar xzvf linux-2.4.26.tar.gz
4. cp /usr/src/linux2.4.20-8/configs/kernel-2.4.20-i686.config ./.config
5. cd /usr/src/linux-2.4.26
6. Apply patch agpgart.patch.2.4.24
7. Make clean
8. make mrproper
9. cp ../.config ./.
10. Make menuconfig.

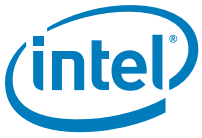
Note: For this step, remember to turn off version information under the kernel loadable module section. Also check that agpgart is being compiled as a module here.

11. make dep make bzImage modules
12. Copy agpgart from /usr/src/linux-2.4.26/drivers/char/agp/agpgart.o
13. Transfer agpgart to DSL /lib/modules/2.4.26/kernel/drivers/char/agp/agpgart.o

After the agpgart has been modprobed, copy all the IEGD files to the proper directory. Startx and IEGD should be running then.

Check that IEGD is running by looking through the XFree logfile. It should be under `/var/log/`. Also, "Intel Embedded Graphics Driver" should display.

After all this is done, you can ensure that IEGD is running after it has been rebooted. Add the appropriate files for IEGD into `.filetool.lst`, include the `agpgart.o` in `/lib/modules/2.4.26/kernel/drivers/char/agp`, and run `filetool.sh` backup to back up the files. Next, change `/opt/bootlocal.sh` to modprobe the agpgart at startup.



7.9.4 RAM Size Constraint

A 128M RAM can be used if a swap file is created in the second partition. To make a swap file:

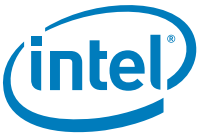
1. Make a blank file of fixed size using dd. Example: dd if=/dev/zero of=knoppix.swp bs=1M count=70
2. Make the file into a swap file. mkswap knoppix.swp
3. Turn on the swap. swapon knoppix.swp

Note: Remember to turn on the swap file every time you start DSL by changing the bootup script in /opt/bootlocal.sh.

7.9.5 Shrinking and Modifying the Extension

There are several types of extensions being used for myDSL. For more information on the different types, how to use them, and their creation, please refer to the myDSL wiki. Basically, myDSL extensions are tarballs that get untarred during bootup. For the purpose of enabling IEGD, the XFree86.dsl extension can be modified and shrunk to help save hard disk space.

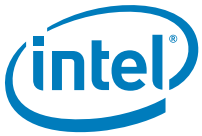
This page is intentionally left blank.



Appendix A Example INF File

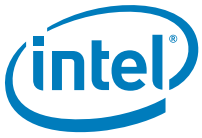
```
;*****************************************************************************
;*****************************************************************************
; Filename: iegd.inf
; $Revision: 1.135 $
; $Id: iegd.inf,v 1.135 2008/01/14 04:46:34 anthill Exp $
; $Source: /nfs/fm/proj/eia/cvsroot/ssigd/ial/dx/install/iegd.inf,v $
;
; Copyright (c) 2006 Intel Corporation.  All rights reserved.
;
;*****************************************************************************

[Version]
Signature="$WINDOWS NT$"
Class=Display
ClassGUID={4D36E968-E325-11CE-BFC1-08002BE10318}
Provider=%Intel%
DriverVer = 01/14/2008,9.0.0.1115


;=============================================================================
[SourceDisksNames]
1=%DiskDesc%,,,""

[SourceDisksFiles]
iegdmini.sys = 1
iegdmkey.vp  = 1
iegdmsys.vp  = 1
iegddis.dll  = 1
iegd3dg3.dll = 1
iegd3dg4.dll = 1
analog.sys   = 1
ch7017.sys   = 1
fs454.sys    = 1
lvds.sys     = 1
ns2501.sys   = 1
ns387.sys    = 1
sii164.sys   = 1
ti410.sys    = 1
th164.sys    = 1
sdvo.sys     = 1
tv.sys       = 1


;=============================================================================
[DestinationDirs]
DefaultDestDir     = 11; System directory
iegd.Display_alm  = 11
iegd.Display_nap  = 11
iegd.Display_gn4  = 11
iegd.Display_plb  = 11
iegd.Miniport_vp  = 11
iegd.Miniport     = 12; Drivers directory
iegd.PortDrvs_alm = 12
iegd.PortDrvs_nap = 12
iegd.PortDrvs_gn4 = 12
```

```
iegd.PortDrvs_plb = 12

;============================================================================
[Manufacturer]
%Intel%=Intel.Mfg

;============================================================================
[Intel.Mfg]
%Intel% %i830m%    = iegd_alm, PCI\VEN_8086&DEV_3577
%Intel% %i865%     = iegd_alm, PCI\VEN_8086&DEV_2572

%Intel% %i915GD0%  = iegd_nap, PCI\VEN_8086&DEV_2582
%Intel% %i915GD1%  = iegd_nap, PCI\VEN_8086&DEV_2782
%Intel% %i915AL0%  = iegd_nap, PCI\VEN_8086&DEV_2592
%Intel% %i915AL1%  = iegd_nap, PCI\VEN_8086&DEV_2792
%Intel% %i945LP0%  = iegd_nap, PCI\VEN_8086&DEV_2772
%Intel% %i945LP1%  = iegd_nap, PCI\VEN_8086&DEV_2776
%Intel% %i945CT0%  = iegd_nap, PCI\VEN_8086&DEV_27A2
%Intel% %i945CT1%  = iegd_nap, PCI\VEN_8086&DEV_27A6
%Intel% %i945WB0%  = iegd_nap, PCI\VEN_8086&DEV_27AE
%Intel% %i35BL0%   = iegd_nap, PCI\VEN_8086&DEV_29C2
%Intel% %i35BL1%   = iegd_nap, PCI\VEN_8086&DEV_29C3
%Intel% %i35BL0A2% = iegd_nap, PCI\VEN_8086&DEV_29B2
%Intel% %i35BL1A2% = iegd_nap, PCI\VEN_8086&DEV_29B3

%Intel% %i965BW0%  = iegd_gn4, PCI\VEN_8086&DEV_2982
%Intel% %i965BW1%  = iegd_gn4, PCI\VEN_8086&DEV_2983
%Intel% %iG9650%   = iegd_gn4, PCI\VEN_8086&DEV_29A2
%Intel% %iG9651%   = iegd_gn4, PCI\VEN_8086&DEV_29A3
%Intel% %iQ9650%   = iegd_gn4, PCI\VEN_8086&DEV_2992
%Intel% %iQ9651%   = iegd_gn4, PCI\VEN_8086&DEV_2993
%Intel% %i946GZ0%  = iegd_gn4, PCI\VEN_8086&DEV_2972
%Intel% %i946GZ1%  = iegd_gn4, PCI\VEN_8086&DEV_2973
%Intel% %i965GM0%  = iegd_gn4, PCI\VEN_8086&DEV_2A02
%Intel% %i965GM1%  = iegd_gn4, PCI\VEN_8086&DEV_2A03
%Intel% %i965GME0% = iegd_gn4, PCI\VEN_8086&DEV_2A12
%Intel% %i965GME1% = iegd_gn4, PCI\VEN_8086&DEV_2A13
%Intel% %ictg0%    = iegd_gn4, PCI\VEN_8086&DEV_2A42
%Intel% %ictg1%    = iegd_gn4, PCI\VEN_8086&DEV_2A43

%Intel% %i900G0%  = iegd_plb, PCI\VEN_8086&DEV_8108

;============================================================================
[iegd_alm.GeneralConfigData]
MaximumNumberOfDevices = 2
MaximumDeviceMemoryConfiguration = 128

[iegd_nap.GeneralConfigData]
MaximumNumberOfDevices = 2
MaximumDeviceMemoryConfiguration = 256

[iegd_gn4.GeneralConfigData]
MaximumNumberOfDevices = 2
MaximumDeviceMemoryConfiguration = 512

[iegd_plb.GeneralConfigData]
MaximumNumberOfDevices = 2
MaximumDeviceMemoryConfiguration = 256

;============================================================================
[iegd_alm]
CopyFiles = iegd.Miniport, iegd.Display_alm, iegd.PortDrvs_alm

[iegd_nap]
CopyFiles = iegd.Miniport, iegd.Display_nap, iegd.PortDrvs_nap, iegd.Miniport_vp

[iegd_gn4]
```



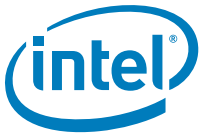
```
CopyFiles = iegd.Miniport, iegd.Display_gn4, iegd.PortDrvs_gn4, iegd.Miniport_vp

[iegd_plb]
CopyFiles = iegd.Miniport, iegd.Display_plb, iegd.PortDrvs_plb, iegd.Miniport_vp

;=============================================================================
[iegd.Miniport]
iegdmini.sys

[iegd.Miniport_vp]
iegdmkey.vp
iegdmsys.vp

[iegd.Display_alm]
iegddis.dll
iegd3dg3.dll

[iegd.Display_nap]
iegddis.dll
iegd3dg3.dll

[iegd.Display_gn4]
iegddis.dll
iegd3dg4.dll

[iegd.Display_plb]
iegddis.dll
iegd3dg3.dll

[iegd.PortDrvs_alm]
analog.sys
ch7017.sys
fs454.sys
lvds.sys
ns2501.sys
ns387.sys
sii164.sys
ti410.sys
th164.sys

[iegd.PortDrvs_nap]
analog.sys
sdvo.sys
lvds.sys
tv.sys

[iegd.PortDrvs_gn4]
analog.sys
sdvo.sys
lvds.sys

[iegd.PortDrvs_plb]
sdvo.sys
lvds.sys

;=============================================================================
[iegd_alm.Services]
AddService = iegdmini, 0x00000002, iegd_Service_Inst, iegd_EventLog_Inst
AddService = analog, ,analog_Service_Inst, iegd_EventLog_Inst
AddService = ch7017, ,ch7017_Service_Inst, iegd_EventLog_Inst
AddService = fs454,  ,fs454_Service_Inst, iegd_EventLog_Inst
AddService = lvds,   ,lvds_Service_Inst, iegd_EventLog_Inst
AddService = ns2501, ,ns2501_Service_Inst, iegd_EventLog_Inst
AddService = ns387,  ,ns387_Service_Inst, iegd_EventLog_Inst
AddService = sii164, ,sii164_Service_Inst, iegd_EventLog_Inst
AddService = ti410,  ,ti410_Service_Inst, iegd_EventLog_Inst
AddService = th164,  ,th164_Service_Inst, iegd_EventLog_Inst
```

```
[iegd_nap.Services]
AddService = iegdmini, 0x00000002, iegd_Service_Inst, iegd_EventLog_Inst
AddService = analog,  ,analog_Service_Inst, iegd_EventLog_Inst
AddService = lvds,    ,lvds_Service_Inst, iegd_EventLog_Inst
AddService = sdvo,    ,sdvo_Service_Inst, iegd_EventLog_Inst
AddService = tv,      ,tv_Service_Inst, iegd_EventLog_Inst

[iegd_gn4.Services]
AddService = iegdmini, 0x00000002, iegd_Service_Inst, iegd_EventLog_Inst
AddService = analog, ,analog_Service_Inst, iegd_EventLog_Inst
AddService = lvds,    ,lvds_Service_Inst, iegd_EventLog_Inst
AddService = sdvo,    ,sdvo_Service_Inst, iegd_EventLog_Inst

[iegd_plb.Services]
AddService = iegdmini, 0x00000002, iegd_Service_Inst, iegd_EventLog_Inst
AddService = lvds,    ,lvds_Service_Inst, iegd_EventLog_Inst
AddService = sdvo,    ,sdvo_Service_Inst, iegd_EventLog_Inst

;=============================================================================
[iegd_Service_Inst]
ServiceType    = 1
StartType      = %SERVICE_DEMAND_START%
ErrorControl   = 0
LoadOrderGroup = Video
ServiceBinary  = %12%\iegdmini.sys

[analog_Service_Inst]
DisplayName    = "analog"
ServiceType    = %SERVICE_KERNEL_DRIVER%
StartType      = %SERVICE_DEMAND_START%
ErrorControl   = %SERVICE_ERROR_IGNORE%
ServiceBinary  = %12%\analog.sys

[ch7017_Service_Inst]
DisplayName    = "ch7017"
ServiceType    = %SERVICE_KERNEL_DRIVER%
StartType      = %SERVICE_DEMAND_START%
ErrorControl   = %SERVICE_ERROR_IGNORE%
ServiceBinary  = %12%\ch7017.sys

[fs454_Service_Inst]
DisplayName    = "fs454"
ServiceType    = %SERVICE_KERNEL_DRIVER%
StartType      = %SERVICE_DEMAND_START%
ErrorControl   = %SERVICE_ERROR_IGNORE%
ServiceBinary  = %12%\fs454.sys

[lvds_Service_Inst]
DisplayName    = "lvds"
ServiceType    = %SERVICE_KERNEL_DRIVER%
StartType      = %SERVICE_DEMAND_START%
ErrorControl   = %SERVICE_ERROR_IGNORE%
ServiceBinary  = %12%\lvds.sys

[ns2501_Service_Inst]
DisplayName    = "ns2501"
ServiceType    = %SERVICE_KERNEL_DRIVER%
StartType      = %SERVICE_DEMAND_START%
ErrorControl   = %SERVICE_ERROR_IGNORE%
ServiceBinary  = %12%\ns2501.sys
```



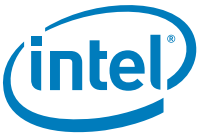
```
[ns387_Service_Inst]
DisplayName    = "ns387"
ServiceType    = %SERVICE_KERNEL_DRIVER%
StartType      = %SERVICE_DEMAND_START%
ErrorControl   = %SERVICE_ERROR_IGNORE%
ServiceBinary  = %12%\ns387.sys

[sii164_Service_Inst]
DisplayName    = "sii164"
ServiceType    = %SERVICE_KERNEL_DRIVER%
StartType      = %SERVICE_DEMAND_START%
ErrorControl   = %SERVICE_ERROR_IGNORE%
ServiceBinary  = %12%\sii164.sys

[ti410_Service_Inst]
DisplayName    = "ti410"
ServiceType    = %SERVICE_KERNEL_DRIVER%
StartType      = %SERVICE_DEMAND_START%
ErrorControl   = %SERVICE_ERROR_IGNORE%
ServiceBinary  = %12%\ti410.sys


[th164_Service_Inst]
DisplayName    = "th164"
ServiceType    = %SERVICE_KERNEL_DRIVER%
StartType      = %SERVICE_DEMAND_START%
ErrorControl   = %SERVICE_ERROR_IGNORE%
ServiceBinary  = %12%\th164.sys

[sdvo_Service_Inst]
DisplayName    = "sdvo"
ServiceType    = %SERVICE_KERNEL_DRIVER%
StartType      = %SERVICE_DEMAND_START%
ErrorControl   = %SERVICE_ERROR_IGNORE%
ServiceBinary  = %12%\sdvo.sys

[tv_Service_Inst]
DisplayName    = "tv"
ServiceType    = %SERVICE_KERNEL_DRIVER%
StartType      = %SERVICE_DEMAND_START%
ErrorControl   = %SERVICE_ERROR_IGNORE%
ServiceBinary  = %12%\tv.sys

;=================================================================================
[iegd_EventLog_Inst]
AddReg = iegd_EventLog_AddReg

[iegd_EventLog_AddReg]
HKR,,EventMessageFile,0x00020000,"%SystemRoot%\System32\IoLogMsg.dll;%SystemRoot%\Syst
em32\drivers\iegdmini.sys"
HKR,,TypesSupported,0x00010001,7

;=================================================================================
[iegd_alm.SoftwareSettings]
AddReg = iegd_SoftwareDeviceSettings_alm

[iegd_nap.SoftwareSettings]
AddReg = iegd_SoftwareDeviceSettings_nap

[iegd_gn4.SoftwareSettings]
AddReg = iegd_SoftwareDeviceSettings_gn4

[iegd_plb.SoftwareSettings]
AddReg = iegd_SoftwareDeviceSettings_plb

;=================================================================================
```

```
[iegd_SoftwareDeviceSettings_alm]
HKR,, InstalledDisplayDrivers, %REG_MULTI_SZ%, iegddis
HKR,, MultiFunctionSupported, %REG_MULTI_SZ%, 1
HKR,, VgaCompatible, %REG_DWORD%, 0
HKR,, PcfVersion,    %REG_DWORD%, 0x0700

HKR,, No_D3D, %REG_DWORD%, 0
HKR,, PortDrivers, %REG_SZ%, "analog ch7017 fs454 lvds ns2501 ns387 sii164 ti410 th164"

;-------------------------------------------------------------------------------
[iegd_SoftwareDeviceSettings_nap]
HKR,, InstalledDisplayDrivers, %REG_MULTI_SZ%, iegddis
HKR,, MultiFunctionSupported, %REG_MULTI_SZ%, 1
HKR,, VgaCompatible, %REG_DWORD%, 0
HKR,, PcfVersion,    %REG_DWORD%, 0x0700

HKR,, No_D3D, %REG_DWORD%, 0

HKR,, PortDrivers, %REG_SZ%, "analog sdvo lvds tv"

;-------------------------------------------------------------------------------
[iegd_SoftwareDeviceSettings_gn4]
HKR,, InstalledDisplayDrivers, %REG_MULTI_SZ%, iegddis
HKR,, MultiFunctionSupported, %REG_MULTI_SZ%, 1
HKR,, VgaCompatible, %REG_DWORD%, 0
HKR,, PcfVersion,    %REG_DWORD%, 0x0700

HKR,, No_D3D, %REG_DWORD%, 0

HKR,, PortDrivers, %REG_SZ%, "analog sdvo lvds"

;-------------------------------------------------------------------------------
[iegd_SoftwareDeviceSettings_plb]
HKR,, InstalledDisplayDrivers, %REG_MULTI_SZ%, iegddis
HKR,, MultiFunctionSupported, %REG_MULTI_SZ%, 1
HKR,, VgaCompatible, %REG_DWORD%, 0
HKR,, PcfVersion,    %REG_DWORD%, 0x0400

HKR,, No_D3D, %REG_DWORD%, 1

HKR,, PortDrivers, %REG_SZ%, "sdvo lvds"

;===============================================================================
[Strings]

;-----------------------------------------------------------------------
; Localizable Strings
;-----------------------------------------------------------------------
Intel="Intel Corporation"
DiskDesc="Embedded Installation"

i865="865 Embedded Graphics Controller"

i915GD0="915G/915GV/910GL Embedded Graphics Controller Function 0"
i915GD1="915G/915GV/910GL Embedded Graphics Controller Function 1"
i915AL0="915GM/915GMS/915GME/910GML/910GMLE Embedded Graphics Controller Function 0"
i915AL1="915GM/915GMS/915GME/910GML/910GMLE Embedded Graphics Controller Function 1"
i945LP0="945G Embedded Graphics Controller Function 0"
i945LP1="945G Embedded Graphics Controller Function 1"
i945CT0="945GM Embedded Graphics Controller Function 0"
i945CT1="945GM Embedded Graphics Controller Function 1"
i945WB0="945GME Embedded Graphics Controller Function 0"
i35BL0="Q35 Embedded Graphics Controller Function 0"
i35BL1="Q35 Embedded Graphics Controller Function 1"
i35BL0A2="Q35 Embedded Graphics Controller Function 0"
i35BL1A2="Q35 Embedded Graphics Controller Function 1"
```



```
i965BW0="965G Embedded Graphics Controller Function 0"
i965BW1="965G Embedded Graphics Controller Function 1"
iG9650="G965 Embedded Graphics Controller Function 0"
iG9651="G965 Embedded Graphics Controller Function 1"
iQ9650="Q963/Q965 Embedded Graphics Controller Function 0"
iQ9651="Q963/Q965 Embedded Graphics Controller Function 1"
i946GZ0="946GZ Embedded Graphics Controller Function 0"
i946GZ1="946GZ Embedded Graphics Controller Function 1"
i965GM0="GM965 Embedded Graphics Controller Function 0"
i965GM1="GM965 Embedded Graphics Controller Function 1"
i965GME0="GME965 Embedded Graphics Controller Function 0"
i965GME1="GME965 Embedded Graphics Controller Function 1"
ictg0="Cantiga Embedded Graphics Controller Function 0"
ictg1="Cantiga Embedded Graphics Controller Function 1"
i900G0="Poulsbo Embedded Graphics Controller Function 0"

;-----------------------------------------------------------------------
; Non Localizable Strings
;-----------------------------------------------------------------------
SERVICE_BOOT_START     = 0x0
SERVICE_SYSTEM_START   = 0x1
SERVICE_AUTO_START     = 0x2
SERVICE_DEMAND_START   = 0x3
SERVICE_DISABLED       = 0x4

SERVICE_KERNEL_DRIVER  = 0x1

SERVICE_ERROR_IGNORE   = 0x0; Continue on driver load fail
SERVICE_ERROR_NORMAL   = 0x1; Display warn, but continue
SERVICE_ERROR_SEVERE   = 0x2; Attempt LastKnownGood
SERVICE_ERROR_CRITICAL = 0x3; Attempt LastKnownGood, BugCheck

REG_EXPAND_SZ = 0x00020000
REG_MULTI_SZ  = 0x00010000
REG_DWORD     = 0x00010001
REG_SZ        = 0x00000000
```

This page is intentionally left blank.



Appendix B Port Driver Attributes

B.1 Standard Port Driver Attributes

Port drivers are modules within the IEGD driver suite that control GMCH-specific modules such as GMCH LVDS, GMCH TV or add-on modules to GMCH. The table below lists the attributes available to port drivers. Some of these standard attributes can be customized for specific port drivers and are detailed in the following topics:

In the attribute tables, device-specific (non-standard) attributes are highlighted in gray.

- "Internal LVDS Port Driver Attributes (Mobile chipsets only)" on page 199
- "CRT (Analog) Port Driver Attributes" on page 199
- "Chrontel CH7307 Port Driver Attributes" on page 201
- "Chrontel CH7308 Port Driver Attributes" on page 202
- "Chrontel CH7315/CH7317/CH7319/CH7320 Port Driver Attributes" on page 202
- "Silicon Image SiI 1362/SiI 1364 Port Driver DVI Attributes" on page 203

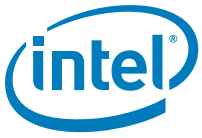
Note: Not all standard attributes are supported by all port drivers. Please see the topics mentioned above for details on the specific attributes supported by each port driver. Flat panel settings are specified via the FPINFO options of the configuration; please see Table 24.

Table 39. Standard Port Driver Attributes

Attribute Name	Attribute ID Number	Description
BRIGHTNESS	0	Brightness adjustment.
CONTRAST	1	Contrast adjustment.
HUE	2	Hue adjustment.
FLICKER	3	Setting to reduce flicker.
HPOSITION	4	Controls the horizontal position of the display.
VPOSITION	5	Controls the vertical position of the display.
HSCALE	6	Horizontal scaling ratio.
VSCALE	7	Vertical scaling ratio.
TVFORMAT	8	TV formats are device-specific.
DISPLAY TYPE	9	Allows selection of different displays for multi-display devices. This attribute is device-specific.
LUMA FILTER	10	TV Luma Filter adjustment.
CHROMA FILTER	11	ChromaFilter adjustment.
TEXT FILTER	12	Text Filter adjustment.
TV OUTPUT TYPE	14	TV output types. This attribute is device-specific.

Table 39. Standard Port Driver Attributes (Continued)

Attribute Name	Attribute ID Number	Description
SATURATION	15	Saturation adjustment.
PANEL FIT	18	Panel fitting. Yes or no.
SCALING RATIO	19	Output Scaling. Device-specific.
FP BACKLIGHT ENABLE	20	Enable flat panel backlight.
PANEL DEPTH	26	Can be either 18 or 24. 18 specifies 6-bit output per color, 24 specifies 8-bit output per color.
DUAL CHANNEL PANEL	27	Is it a dual channel panel or not? Takes 0 or 1.
GANG MODE	28	For achieving a Gang mode output using two digital ports.
GANG MODE EVEN ODD	29	Gang display even or odd. This attribute is to be set along with Gang mode (28). This mode (Gang Mode Even Odd) puts even pixels on one digital port and odd pixels on the other, and needs to be selected based on the display panel used.
SHARPNESS	31	Sharpness.
HWCONFIG	32	Hardware Configuration for sDVO encoders that support multiple configurations.
HORZFILTER	33	Horizontal Filter.
VERTFILTER	34	Vertical Filter.
FRAME BUFFER GAMMA	35	Framebuffer gamma correction.
FRAME BUFFER BRIGHTNESS	36	Framebuffer brightness.
FRAME BUFFER CONTRAST	37	Framebuffer contrast.
2D FLICKER	39	Two-dimension flicker.
ADAPTIVE FLICKER	40	Adaptive flicker.
HORIZONTAL OVERSCAN	41	Horizontal overscan.
VERTICAL OVERSCAN	42	Vertical overscan.
SPREAD SPECTRUM CLOCKING	43	Spectrum Clocking
DOT_CRAWL	44	Dot crawl affects the edges of color and manifests itself as moving dots of color along these edges.
DITHER	45	Dither setting
PANEL PROTECT HSYNC	46	Horizontal sync panel protection
PANEL PROTECT VSYNC	47	Vertical sync panel protection
PANEL PROTECT PIXCLK	48	Pixel clock protection
LVDS PANEL TYPE	49	This is used to select SPWG vs. OpenLDI panel types. 0-SPWG 1-OpenLDI.
VGA 2X IMAGE	57	Controls VGA image in Gang mode.
TEXT ENHANCEMENT	58	Controls text tuning.
MAINTAIN ASPECT RATIO	59	This controls scaled image to match source image aspect ratio or full screen image.
FIXED TIMING	60	This indicates whether the attached display is a fixed timing display.



B.1.1 Internal LVDS Port Driver Attributes (Mobile chipsets only)

Table 40. Internal LVDS Port Driver Attributes

Attribute Name	Attribute ID	Description	Possible Ranges
PANELDEPTH	26	Specify Panel Depth based on connected panel.	Default is 18, however, on some GMCH chipsets 24-bit also is supported. For example, GM965 supports both 18 and 24-bit outputs.
DUALCHANNEL	27	Single or Dual Channel Panel	0 = Single 1 = Dual Default is 0.
DITHER	45	On and off Dithering	0 = on 1 = off Default: • dither on for 18-bit panels • dither off for 24-bit panels.
LVDS Panel Type	49	LVDS panel connector.	0 = SPWG formatted LVDS output (default) 1 = OpenLDI unbalanced color mapping output Default = 0
FIXED TIMING	60	This indicates whether attached display is a fixed timing display.	0 = on 1 = off

B.1.2 CRT (Analog) Port Driver Attributes

Note: The analog port driver is included in the driver by default, unlike other port drivers available for selection as part of the driver configuration.

Table 41. CRT (Analog) Port Driver Attributes

Attribute Name	Attribute ID	Description	Possible Ranges
FIXED TIMING	60	Set this attribute if the attached display supports only one timing.	0 = Not a fixed timing display. 1 = Fixed timing display. Default is 0.
DISPLAY DETECT METHOD	32769	Controls display detection for the CRT.	0 = Uses DDC method first then Analog sense to detect the display. 1 = DDC method only (Digital Sense method) by reading EDID 2 = Analog Sense only Default = 0

B.1.3 HDMI Port Driver Attributes

7.9.5.1 Audio

The IEGD package does not include an HDMI audio driver, so you must obtain and install the driver. The HDMI audio driver needs to support Intel HD Audio to be compatible with IEGD. You must also obtain Microsoft* patch KB888111 to enable HDMI audio. IEGD supports only the Windows HDMI audio driver.

B.1.3.1 SDVO-HDMI (CH7315)

IEGD supports only one type of SDVO-HDMI encoder, which is CH7315. SDVO-B cannot coexist with HDMI-B; SDVO-C cannot coexist with HDMI-C.

SDVO takes precedence over the HDMI port driver. If no SDVO encoder is available HDMI is automatically loaded by default (only in the GM45 Express chipset).

B.1.3.2 Internal HDMI

Internal HDMI is available only for the GM45 Express chipset. Only one HDMI port has audio at any one time. The first port in the port order has audio while the second port would have only display without audio.

Only one HDMI port has HDCP at any one time. The first port to receive a request for HDCP has HDCP enabled only in that port.

B.1.3.3 HDCP

HDCP is supported through the Certified Output Protection Protocol (COPP) interface in Windows.

Table 42. HDMI Port Driver Attributes

Attribute Name	Attribute ID	Description	Possible Ranges
FIXED TIMING	60	This indicates whether attached display is a fixed timing display.	0 = on 1 = off

B.1.4 Internal TV Out Port Driver Attributes (Mobile chipsets only)

Table 43. Internal TV Out Port Driver Attributes

Attribute Name	Attribute ID	Description	Possible Ranges
BRIGHTNESS	0	Screen brightness	0-100. Default is 50.
CONTRAST	1	Color contrast	0-7. Default is 3.
HUE	2	Hue adjustment	0-100. Default is 0.
TV FLICK FILTER	3	TV Flicker Filter. The higher the value, the higher the amount of flicker filtering and text enhancement.	0-1000. Default is 999.
H POSITION	4	Horizontal Position. Increasing the value moves the image to the right and decreasing the value moves the image to the left.	0-511. Default is 64.

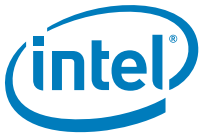


Table 43. Internal TV Out Port Driver Attributes (Continued)

Attribute Name	Attribute ID	Description	Possible Ranges
V POSITION	5	Vertical Position. The value represents the TV line number relative to the VGA vertical sync. Increasing the value moves the image down and decreasing the value moves the image up.	0-511. Default is 0.
TV FORMAT	8	TV formats are device-specific.	Default is NTSC-M (1).
TV OUTPUT	14	TV output types. This attribute is device-specific. ***Note:*** TV output types are limited to S-Video and Composite for the VBIOS.	Default is S-VIDEO (2).
OVERSCAN/SCALING RATIO	19	Output Scaling.	0-1000. Default is 350.

B.1.5 Chrontel CH7307 Port Driver Attributes

The table below shows the attributes for the Chrontel CH7307* port driver.

Note: For flat panel backlight timing settings, please see Table 24.

Table 44. Chrontel CH7307 Port Driver Attributes

Attribute Name	Attribute ID	Description	Possible Ranges
Spread Spectrum Clocking	43	Spectrum clocking	0-15 Default = 0 Step = 1
FIXED TIMING	60	This indicates whether attached display is a fixed timing display.	0 = on 1 = off

B.1.6 Chrontel CH7308 Port Driver Attributes

The table below shows the attributes for the Chrontel CH7308* port driver.

Note: For FPINFO panel width, height, and backlight timing settings, please see Table 24.

Table 45. Chrontel CH7308 Port Driver Attributes

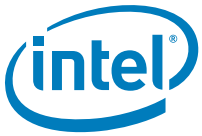
Attribute Name	Attribute ID	Description	Possible Ranges
LVDS Color Depth	26	Panel depth	18 = 18 bits 24 = 24 bits Default = 18
DUAL_CHANNEL	27	Dual-channel pane	Default - 0
Spread Spectrum Clocking	43	Spectrum Clocking	0-15 Default = 0 Step = 1
Dither	45	Dither setting	Default = 0
HSync Panel Protection	46	Horizontal sync panel protection	Default = 0
VSync Panel Protection	47	Vertical sync panel protection	Default = 0
Pixel Clock Protection	48	Pixel clock protection	Default = 0
LVDS Panel Type	49	LVDS panel connector.	0 = SPWG formatted LVDS output (default) 1 = OpenLDI unbalanced color mapping output Default = 0
Text Enhancement	58	Controls text tuning.	0-4.
Fixed Timing	60	This indicates whether attached display is a fixed timing display.	0 = on 1 = off

B.1.7 Chrontel CH7315/CH7317/CH7319/CH7320 Port Driver Attributes

The table below shows the attributes for the Chrontel CH7315/CH7317/CH7319/ CH7320 port driver.

Table 46. Chrontel CH7315/CH7317/CH7319/CH7320 Port Driver Attributes

Attribute Name	Attribute ID	Description	Possible Ranges
VGA Bypass	9	TBD	TBD
Fixed Timing	60	This indicates whether attached display is a fixed timing display.	0 = on 1 = off



B.1.8 Silicon Image SiI 1362/SiI 1364 Port Driver DVI Attributes

Note: For flat panel backlight timing settings, please see Table 24.

Table 47. Silicon Image SiI 1362/SiI 1364 Port Driver DVI Attributes

Attribute Name	Attribute ID	Description	Possible Ranges
FIXED TIMING	60	This indicates whether attached display is a fixed timing display.	0 = on 1 = off

B.2 Default Search Order

Table 48. Default Search Order

Chipset	Default Search Order
Intel® US15W	TBD
Intel® Q35	ANALOG, sDVOB, sDVOC
Intel® GME965	ANALOG, sDVOB, sDVOC, LVDS
Intel® Q965	ANALOG, sDVOB, sDVOC
Intel® 945GSE	TBD
Intel® 945GME	ANALOG, sDVOB, sDVOC, LVDS
Intel® 945G	ANALOG, sDVOB, sDVOC
Intel® 915GV	ANALOG, sDVOB, sDVOC
Intel® 915GME	ANALOG, sDVOB, sDVOC, LVDS
Intel® 910GMLE	ANALOG, sDVOB, sDVOC, LVDS

B.3 Default GPIO Pin Pair Assignments

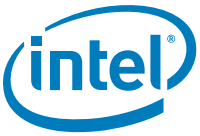
Table 49. Default GPIO Pin Pair Assignments

Chipset	Default GPIO Pin Pair for EDID			
	sDVO/A	sDVOB	sDVOC	LVDS
Intel® Q35	N/A	4	4	N/A
Intel® GME965	N/A	4	4	2
Intel® Q965	N/A	4	4	N/A
Intel® 945GSE	TBD	TBD	TBD	TBD
Intel® 945GME	N/A	4	4	2
Intel® 945G	N/A	4	4	N/A
Intel® 915GV	N/A	4	4	N/A
Intel® 915GM	N/A	4	4	2
Intel® 910GMLE	N/A	4	4	2

B.4 Default I2C Device Address Byte Assignment

Table 50. Default I^{2}C Device Address Byte Assignment

Port Driver	Default Device Address Bytes (DAB)
CH7021, CH7315, CH7317, CH7319, CH7320, CH7021	0x70 (for first sDVO device) 0x72 (for second sDVO device)
CH7307	0x70 (for first sDVO device) 0x72 (for second sDVO device)
CH7308	0x70 (for first sDVO device) 0x72 (for second sDVO device)
SiI 1362	0x70 (for first sDVO device) 0x72 (for second sDVO device)
SiI 1364	0x70 (for first sDVO device) 0x72 (for second sDVO device)



Appendix C Intel® 5F Extended Interface Functions

The BIOS provides a set of proprietary function calls to control operation of the extended features. These function calls all use AH = 5Fh in their designed interface for easy identification as a proprietary function.

These functions are designed to maintain maximum compatibility with the Desktop and Mobile Video BIOS. As such many of the definitions behave identically. When the behavior of the Embedded Video BIOS is not identical to the Desktop and Mobile Video BIOS it is noted.

In addition to these 5F functions, the Video BIOS also supports all 4F functions defined by the *VESA BIOS Extension (VBE) Core Functions Standard, Version 3.0* with the exception of the 0A function (Return VBE Protected Mode Interface). All other functions, from 00 through 09 and 0B are supported by the Video BIOS. Click on the following link to view the VBE 3.0 Core Functions Standard document.

http://www.vesa.org/Public/VBE/vbe3.pdf

The table below provides a summary of the IEGD supported Intel 5F functions.

Table 51. Summary of Intel 5F Extended Interface Functions

Function/Name	Description
BIOS Extended Interface Functions	
"5F01h – Get Video BIOS Information"	Gets VBIOS Build Information.
"5F05h – Refresh Rate"	Sets a new vertical refresh rate for a given mode and returns the current vertical refresh rate
"5F10h – Get Display Memory Information"	Returns information about the linear memory.
"5F1Ch – BIOS Pipe Access"	Sets the BIOS pipe access and returns the BIOS pipe access status.
"5F29h – Get Mode Information"	Returns information on the requested mode.
"5F61h – Local Flat Panel Support Function"	Supports local flat panel features.
"5F68h – System BIOS Callback"	Allows SoftBIOS to do any system callbacks through INT 15h
Hooks for the System BIOS	
"5F31h – POST Completion Notification Hook"	Signals the completion of video POST (Power On Self Test)
"5F33h – Hook After Mode Set"	Allows System BIOS to intercept Video BIOS at the end of a mode set.
"5F35h – Boot Display Device Hook"	Allows System BIOS to override video display default setting.
"5F36h – Boot TV Format Hook"	Allows System BIOS to boot TV in selected TV format state.
"5F38h – Hook Before Set Mode"	Allows the system BIOS to intercept the video BIOS before setting the mode.
"5F40h – Config ID Hook"	Allows System BIOS to supply a configuration ID that is passed to the driver.

C.1 BIOS Extended Interface Functions

The BIOS provides a set of proprietary function calls to control operation of the extended features. These function calls all use AH = 5Fh in their designed interface for easy identification as a proprietary function

These functions are designed to maintain maximum compatibility with the Desktop and Mobile Video BIOS. As such many of the definitions behave identically. When the behavior of the Embedded Video BIOS is not identical to the Desktop and Mobile Video BIOS it is noted.

C.1.1 5F01h – Get Video BIOS Information

This function returns the Video BIOS Build information.

Note: This function is an extension of the Desktop and Mobile Video BIOS. If register ECX does not contain ASCII characters "IEGD" then the VBIOS is not described by this specification.

Calling Register:

AX = 5F01h, Get Video Information function

Return Registers:

AX = Return Status (function not supported if AL != 5Fh):
= 005Fh, Function supported and successful
= 015Fh, Function supported but failed
EBX = 4 bytes Video BIOS Build Number ASCII string, e.g., '1000'
ECX = 4 bytes Embedded Identifier, ASCII string 'IEGD'

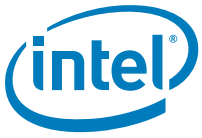
C.1.2 5F05h – Refresh Rate

This function sets a new vertical refresh rate for a given mode and returns the current vertical refresh rate and available refresh rate for a given non-VGA mode.

C.1.2.1 5F05h, 00h – Set Refresh Rate

This sub-function sets a new default refresh rate for the selected pipe. If the mode is currently active, the CRT controller and other registers will be automatically programmed setting the requested refresh rate.

Note: This function is not entirely compatible with the Desktop and Mobile versions. It is not possible to set the refresh rate for a given mode in advance. This function sets the "desired" refresh rate which will be applied to all subsequent mode sets when possible. If the mode provided in BL is the current mode, then a mode change will be automatically performed.



Calling Register:

AX = 5F05h, Refresh Rate function
BH = 00h, Set Refresh Rate sub-function
BL = Mode Number
ECX = Refresh rate (indicated by setting one bit):
Bits 31 - 9 = Reserved
Bit 8 = 120 Hz
Bit 7 = 100 Hz
Bit 6 = 85 Hz
Bit 5 = 75 Hz
Bit 4 = 72 Hz
Bit 3 = 70 Hz
Bit 2 = 60 Hz
Bit 1 = 56 Hz
Bit 0 = 43 Hz (Interlaced - Not supported)

Return Registers:

AX = Return Status (function not supported if AL != 5Fh):
= 005Fh, Function supported and successful
= 015Fh, Function supported but failed

C.1.2.2 5F05h, 01h – Get Refresh Rate

This sub-function returns current vertical refresh rate for the selected pipe and available refresh rates information for a given Non-VGA mode.

Note: This sub-function returns a status of supported but failed (AX = 015Fh) if executed with a standard VGA mode.

Calling Registers:

AX = 5F05h, Refresh Rate function
BH = 01h, Get Refresh Rate sub-function
BL = Mode number

Return Registers:

AX = Return Status (function not supported if AL != 5Fh):
= 005Fh, Function supported and successful
= 015Fh, Function supported but failed
EBX = Available refresh rates (indicated by one or more bits set):
Bits 31 - 9 = Reserved
Bit 8 = 120 Hz
Bit 7 = 100 Hz
Bit 6 = 85 Hz
Bit 5 = 75 Hz
Bit 4 = 72 Hz
Bit 3 = 70 Hz
Bit 2 = 60 Hz
Bit 1 = 56 Hz
Bit 0 = 43 Hz (Interlaced - Not supported)
ECX = Current refresh rate (see EBX for bit definitions)

C.1.3 5F10h – Get Display Memory Information

This function returns information regarding the linear memory starting address, size and memory mapped base address.

Calling Register:

AX = 5F10h, Get Linear Display Memory Information function

Return Registers:

AX = Return Status (function not supported if AL != 5Fh):
= 005Fh, Function supported and successful
= 015Fh, Function supported but failed
ESI = Display memory base address
ECX = Total physical display memory size (in bytes)
EDX = Available display memory size (in bytes)
EDI = Memory Mapped I/O Base Address
EBX = Stride (memory scan line width in bytes)

C.1.4 5F1Ch – BIOS Pipe Access

This function will set the BIOS pipe access or return the BIOS pipe access status.

C.1.4.1 5F1Ch, 00h – Set BIOS Pipe Access

This sub-function will set the currently selected pipe. All 5f functions operate on the currently selected pipe.

When not in clone modes this value cannot be set.

Calling Registers:

AX = 5F1Ch, BIOS Pipe Access function
BH = 00h, Set BIOS Pipe Access sub-function
CH = BIOS Pipe access:
= 00h, Pipe A
= 01h, Pipe B

Return Registers:

AX = Return Status (function not supported if AL != 5Fh):
= 005Fh, Function supported and successful
= 015Fh, Function supported but failed

C.1.4.2 5F1Ch, 01h – Get BIOS Pipe Access

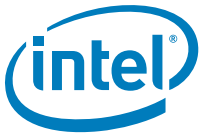
This sub-function will return the currently selected pipe.

Calling Registers:

AX = 5F1Ch, BIOS Pipe Access function
BH = 01h, Get BIOS Pipe Access sub-function

Return Registers:

AX = Return Status (function not supported if AL != 5Fh):
= 005Fh, Function supported and successful
= 015Fh, Function supported but failed
CH = BIOS Pipe access:
= 00h, Pipe A
= 01h, Pipe B



C.1.5 5F29h – Get Mode Information

This function returns the requested mode's resolution, color depth, and maximum required bandwidth using its current refresh rate. This function is applied to extended-graphics modes only. If the mode number is not an extended graphics mode, the function will return failure.

Calling Registers:

AX = 5F29h, Get Mode Information function
BH = Mode To Use:
= 80h, Current Mode
= 00h - 7Fh, Given Mode Number

Return Registers:

AX = Return Status (function not supported if AL != 5Fh):
= 005Fh, Function supported and successful
= 015Fh, Function supported but failed
EBX bits 31 - 16 = Mode horizontal (X) resolution in pixels
EBX bits 15 - 0 = Mode vertical (Y) resolution in pixels
ECX bits 31 - 16 = Maximum bandwidth in megabytes per second
ECX bits 15 - 0 = Color depth in bits per pixel

C.1.6 5F61h – Local Flat Panel Support Function

This function supports local flat panel only features.

Note: Only Subfunction 5h of the 5f61h interface is supported for the Embedded vBIOS.

C.1.6.1 5F61h, 05h – Get Configuration ID

This function is used to return the Configuration ID.

Note: This function is known as "Get Local Flat Panel Number" in the Desktop and Mobile Video BIOS. This function performs a similar purpose however, the configuration IDs have no pre-defined meaning. The Configuration ID is reported to the Embedded Graphics Driver and will be used as described in the *Intel® Embedded Graphics Drivers and Video BIOS User's Guide*.

Calling Registers:

AX = 5F61h, Local Flat Panel Support function
BH = 05h, Get Config ID Subfunction

Return Registers:

AX = Return Status (function not supported if AL != 5Fh):
= 005Fh, Function supported and successful
= 015Fh, Function supported but failed
BL = Config ID

C.1.7 5F68h – System BIOS Callback

This is a generic function that allows SoftBIOS to do any system callbacks through INT 15h. The Input/Output of this function is dependent on the definition of the desired INT 15h hook except for the EAX register.

Calling Registers:

AX = 5F68h, System BIOS Callback Function
EAX bits 31:16 = System BIOS INT 15h Hook Function

Return Registers:

AX = Return Status (function not supported if AL != 5Fh):
= 005Fh, Function supported and successful
= 015Fh, Function supported but failed

C.2 Hooks for the System BIOS

The video BIOS performs several system BIOS interrupt function calls (interrupt 15h hooks). Each function provides the system BIOS with the opportunity to gain control at specific times to perform any custom processing that may be required. After each interrupt hook, the system BIOS must return control to the video BIOS. INT 10h calls could be made within the INT 15h hook calls provided that it is not recursive and thus cause a deadlock.

See the following topics for details on these function calls:

- "5F31h – POST Completion Notification Hook"
- "5F33h – Hook After Mode Set"
- "5F35h – Boot Display Device Hook"
- "5F36h – Boot TV Format Hook"
- "5F38h – Hook Before Set Mode"
- "5F40h – Config ID Hook"

C.2.1 5F31h – POST Completion Notification Hook

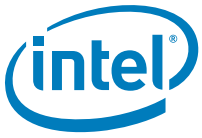
This hook signals the completion of video POST (Power On Self Test). The hook executes after the sign-on message is displayed and PCI BIOS resizing.

Calling Registers:

AX = 5F31h, POST Completion Notification Hook

Return Registers:

AX = Return Status (function not supported if AL != 5Fh):
= 015Fh, Function supported but failed
= 005Fh, Function supported and successful



C.2.2 5F33h – Hook After Mode Set

This hook allows the system BIOS to intercept the video BIOS at the end of a mode set.

Calling Registers:

AX = 5F33h, Hook After Mode Set
BH = Number of character columns
BL = Current mode number
CH = Active display page

Return Registers:

AX = Return Status (function not supported if AL != 5Fh):
= 015Fh, Function supported but failed
= 005Fh, Function supported and successful

C.2.3 5F35h – Boot Display Device Hook

This hook allows the system BIOS to override the video display default setting. The graphics BIOS will set the returned video display during POST (power up initialization).

Note: This function is not entirely compatible with the Desktop and Mobile Video BIOS. The bits in CL have a configurable mapping to the Port Numbers as defined in the *Intel® Embedded Graphics Drivers and Video BIOS User's Guide*. The assigned meanings used in the Desktop specification can be duplicated with a correct configuration. The values below are the default values if no "Common To Port" mapping is provided.

Calling Registers:

AX = 5F35h, Boot Display Device Hook

Return Registers:

AX = Return Status (function not supported if AL != 5Fh);
= 005Fh, Function supported and successful
= 015Fh, Function supported but failed
CL = Display Device Combination to boot (1 = Enable display, 0 = Disable display):= 00h, VBIOS Default
Bit 7 - 6 = Reserved
Bit 5 = Port 5 (or common_to_port[5])
Bit 4 = Port 4 (or common_to_port[4])
Bit 3 = Port 3 (or common_to_port[3])
Bit 2 = Port 2 (or common_to_port[2])
Bit 1 = Port 1 (or common_to_port[1])
Bit 0 = Port 0 (or common_to_port[0])

C.2.4 5F36h – Boot TV Format Hook

This hook allows the system BIOS to boot TV in selected TV format state.

Calling Registers:

AX = 5F36h, Boot TV Format Hook

Return Registers:

AX = Return Status (function not supported if AL != 5Fh):
= 015Fh, Function supported but failed
= 005Fh, Function supported and successful
BL = TV Format requested:
= 00h, No Preference
= 01h, NTSC_M
= 11h, NTSC_M_J
= 21h, NTSC_433
= 31h, NTSC_N
= 02h, PAL_B
= 12h, PAL_G
= 22h, PAL_D
= 32h, PAL_H
= 42h, PAL_I
= 52h, PAL_M
= 62h, PAL_N
= 72h, PAL_60
= 03h, SECAM_L
= 13h, SECAM_L1
= 23h, SECAM_B
= 33h, SECAM_D
= 43h, SECAM_G
= 53h, SECAM_H
= 63h, SECAM_K
= 73h, SECAM_K1

C.2.5 5F38h – Hook Before Set Mode

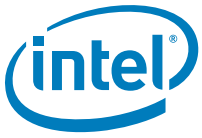
This hook allows the system BIOS to intercept the video BIOS before setting the mode.

Calling Registers:

AX = 5F38h, Hook Before Set Mode
CL = New video mode to be set

Return Registers:

AX = Return Status (function not supported if AL != 5Fh):
= 015Fh, Function supported but failed
= 005Fh, Function supported and successful



C.2.6 5F40h – Config ID Hook

This function is known as "Boot Panel Type Hook" in the Desktop and Mobile Video BIOS. It allows the system BIOS to supply a configuration ID that will eventually be passed to the driver. This configuration ID is unused by the Video BIOS; however, it alters the behavior of the driver as described in the *Intel® Embedded Graphics Drivers and Video BIOS User's Guide*.

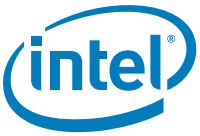
Calling Registers:

AX = 5F40h, Config ID Hook

Return Registers:

AX = Return Status (function not supported if AL != 5Fh):
= 005Fh, Function supported and successful
= 015Fh, Function supported but failed
CL = Configuration ID

This page is intentionally left blank.



Appendix D Intel® OpenGL APIs

The table below presents **supported** IEGD OpenGL* APIs, and presents **non-supported** IEGD OpenGL APIs.

The following OpenGL versions are supported:

- Version 1.3 on all Embedded Intel® Architecture (eIA) chipsets
- Version 1.4 on 915GV, 915GM, 945G, 945GM, Q965, GME965
- Version 1.5 on Q965, GME965
- Version 2.0 on US15W

For general OpenGL information, visit http://www.opengl.org/about/overview/.

Table 52. Supported Intel® OpenGL APIs (Sheet 1 of 2)

Supported API Name(s)
GL_3DFX_texture_compression_FXT
GL_ARB_depth_texture[1]
GL_ARB_fragment_program (965 only)
GL_ARB_multitexture
GL_ARB_occlusion_query (956 only)
GL_ARB_shadow[1]
GL_ARB_textture_env_dot3
GL_ARB_texture_border_clamp
GL_ARB_texture_compression
GL_ARB_texture_cube_map
GL_ARB_texture_env_add
GL_ARB_texture_env_combine
GL_ARB_texture_env_crossbar
GL_ARB_transpose_matrix
GL_ARB_vertex_buffer_object
GL_ARB_vertex_program (965 only)
GL_EXT_abgr
GL_EXT_bgra
GL_EXT_blend_color
GL_EXT_blend_func_separate
GL_EXT_blend_minmax
GL_EXT_blend_subtract
[1] Only supported on Intel 915 series and later chipsets.

Table 52. Supported Intel® OpenGL APIs (Sheet 2 of 2)

Supported API Name(s)
GL_EXT_clip_volume_hint
GL_EXT_compiled_vertex_array
GL_EXT_cull_vertex
GL_EXT_fog_coord
GL_EXT_mulit_draw_arrays
GL_EXT_packed_pixels
GL_EXT_rescale_normal
GL_EXT_secondary_color
GL_EXT_separate_specular_color
GL_EXT_shadow_funcs[1]
GL_EXT_stencil_two_side[1]
GL_EXT_texture_compression_s3tc
GL_EXT_texture_env_add
GL_EXT_texture_filter_anisotropic
GL_EXT_texture_lod_bias (965 only)
GL_IBM_texture_mirrored_repeat
GL_NV_blend_square
GLX_ARB_get_proc_address
[1] Only supported on Intel 915 series and later chipsets.

Table 53. Non-Supported Intel® OpenGL APIs

Non-Supported API Name(s)
GL_ARB_color_buffer_float
GL_ARB_fragment_program_shadow
GL_ARB_point_sprite
GL_ARB_shader_objects
GL_ARB_shading_language_100
GL_ARB_texture_non_power_of_two
GL_EXT_paletted_texture
GL_WIN_swap_hint
WGL_ARB_buffer_region
WGL_ARB_extensions_string
WGL_ARB_make_current_read
WGL_ARB_pbuffer
WGL_ARB_pixel_format
WGL_EXT_swap_control