Graphics Guide for Windows 7

A Guide for Hardware and System Manufacturers

June 12, 2009

Abstract

This guide outlines enhancements to the Windows® desktop and graphics architecture in the Windows 7 operating system. It provides best practices for system design and test configuration recommendations to system and display manufacturers.

This guide is intended for system designers, display driver developers, and test managers who design, deliver, and maintain desktop and mobile PC products that deliver a great customer experience with Windows.

This information applies to the Windows 7 operating system.

References and resources are listed at the end of each numbered part of this paper.

The current version of this paper is maintained on the Web at:   
 <http://www.microsoft.com/whdc/device/display/GraphicsGuideWin7.mspx>

**Queries:** If you have questions that are not addressed by this document, send e-mail to [directx@microsoft.com](mailto:directx@microsoft.com).

Disclaimer: This is a preliminary document and may be changed substantially prior to final commercial release of the software described herein.

The information contained in this document represents the current view of Microsoft Corporation on the issues discussed as of the date of publication. Because Microsoft must respond to changing market conditions, it should not be interpreted to be a commitment on the part of Microsoft, and Microsoft cannot guarantee the accuracy of any information presented after the date of publication.

This White Paper is for informational purposes only. MICROSOFT MAKES NO WARRANTIES, EXPRESS, IMPLIED OR STATUTORY, AS TO THE INFORMATION IN THIS DOCUMENT.

Complying with all applicable copyright laws is the responsibility of the user. Without limiting the rights under copyright, no part of this document may be reproduced, stored in or introduced into a retrieval system, or transmitted in any form or by any means (electronic, mechanical, photocopying, recording, or otherwise), or for any purpose, without the express written permission of Microsoft Corporation.

Microsoft may have patents, patent applications, trademarks, copyrights, or other intellectual property rights covering subject matter in this document. Except as expressly provided in any written license agreement from Microsoft, the furnishing of this document does not give you any license to these patents, trademarks, copyrights, or other intellectual property.

Unless otherwise noted, the example companies, organizations, products, domain names, e-mail addresses, logos, people, places and events depicted herein are fictitious, and no association with any real company, organization, product, domain name, email address, logo, person, place or event is intended or should be inferred.

© 2009 Microsoft Corporation. All rights reserved.

Microsoft, ClearType, DirectX, Direct2D, Direct3D, MSDN,Windows, Windows Media, Windows Server, Windows Vista, and XBox are either registered trademarks or trademarks of Microsoft Corporation in the United States and/or other countries.

The names of actual companies and products mentioned herein may be the trademarks of their respective owners.

Document History

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Date | Change |  |  |  |
| June 12, 2009 | First publication | | | |

Contents

[**INTRODUCTION** 6](#_Toc231891229)

[Key Areas of Innovation for High-Fidelity Graphics in Windows 7 6](#_Toc231891230)

[About This Guide 7](#_Toc231891231)

[System Design Best Practices 7](#_Toc231891232)

[**PART 1: GPU and Display Driver Features and Design** 11](#_Toc231891233)

[Introduction 11](#_Toc231891234)

[Windows 7 Optimized Display Drivers 12](#_Toc231891235)

[Reduced System Memory Footprint 13](#_Toc231891236)

[Improved Gaming Performance 16](#_Toc231891237)

[Improved Diagnosability of Performance and Stability Issues 16](#_Toc231891238)

[IHV Graphics Drivers 17](#_Toc231891239)

[Graphics Hardware IDs 18](#_Toc231891240)

[Driver Distribution 19](#_Toc231891241)

[Unified WDDM Driver Packages 20](#_Toc231891242)

[WDDM v1.1 Driver INF Requirements 21](#_Toc231891243)

[Future of the Windows XP Display Driver Model 23](#_Toc231891244)

[Best Practices for the WDDM v1.1 Driver Model 24](#_Toc231891245)

[Direct3D 10 25](#_Toc231891246)

[Desktop Window Manager 25](#_Toc231891247)

[Direct3D 10 Level -9 26](#_Toc231891248)

[Direct2D API 26](#_Toc231891249)

[DirectX 11 27](#_Toc231891250)

[Compute Shader for Data-Parallel Programming 28](#_Toc231891251)

[About Data-Parallel Programming 28](#_Toc231891252)

[Graphics Processor as a Data-Parallel Processor 29](#_Toc231891253)

[The Compute Shader 29](#_Toc231891254)

[New DirectX 11 Features for the Compute Shader 30](#_Toc231891255)

[Supported Configurations for the Compute Shader 31](#_Toc231891256)

[3‑D Graphics Improvements 32](#_Toc231891257)

[Use of Multiple CPU Cores 33](#_Toc231891258)

[Tessellation 33](#_Toc231891259)

[High-Level Shading Language 33](#_Toc231891260)

[Cross-Platform Development for Xbox 360 34](#_Toc231891261)

[Additional Direct3D 11 Features 34](#_Toc231891262)

[Improved Texture Compression 34](#_Toc231891263)

[Shader Model 5.0 34](#_Toc231891264)

[Stream Output Flexibility 34](#_Toc231891265)

[Depth Buffer Capabilities 35](#_Toc231891266)

[Best Practices for DirectX Technologies 35](#_Toc231891267)

[Video Playback 35](#_Toc231891268)

[DXVA-HD DDI 35](#_Toc231891269)

[GPU Content Protection 37](#_Toc231891270)

[Standardized AES 128-bit Interface 37](#_Toc231891271)

[Driver Software Protection 37](#_Toc231891272)

[Driver Hardware Protection 38](#_Toc231891273)

[Video Overlay DDI in WDDM v1.1 38](#_Toc231891274)

[Best Practices for Video Playback 3](#_Toc231891275)9

[Resources for GPU and Display Driver Features and Design 39](#_Toc231891276)

[**PART 2: Display Features and Design Considerations** 40](#_Toc231891277)

[Introduction 40](#_Toc231891278)

[Display Ecosystem and Hardware 41](#_Toc231891279)

[EDID for Display Devices 43](#_Toc231891280)

[Switchboxes and Extenders 44](#_Toc231891281)

[Docking Stations 46](#_Toc231891282)

[Cables and Connectors 47](#_Toc231891283)

[Best Practices for Display Ecosystem and Hardware 47](#_Toc231891284)

[Display Connection and Configuration 47](#_Toc231891285)

[Obtaining Hardware and Configuration Information 48](#_Toc231891286)

[Display Setting Persistence Database 48](#_Toc231891287)

[Connecting and Configuring Displays API 48](#_Toc231891288)

[Changes to the Windows Graphics Kernel 49](#_Toc231891289)

[Projection Shortcut Key 49](#_Toc231891290)

[Integrating OEM Shortcut Keys with the Projection Shortcut Key 50](#_Toc231891291)

[Mobility Center Connect Display Button 52](#_Toc231891292)

[Display Device Detection 53](#_Toc231891293)

[Default Settings for Display Devices 53](#_Toc231891294)

[Multimonitor Support 54](#_Toc231891295)

[Deprecation of Transient Multimonitor Manager 54](#_Toc231891296)

[Multiple Adapters and Multiple Monitors 55](#_Toc231891297)

[Control Panel Display Application 55](#_Toc231891298)

[Control Panel Display Application User Interface 56](#_Toc231891299)

[Guidelines for Extending the Control Panel Display Application 60](#_Toc231891300)

[Requirements for TV Settings Support 62](#_Toc231891301)

[Requirements for Supporting Aspect Ratio Settings 63](#_Toc231891302)

[WDDM Driver Optimizations for Display Configuration 64](#_Toc231891303)

[Monitor Scaling Mode to Preserve Aspect Ratio 64](#_Toc231891304)

[Additional Display Resolutions 66](#_Toc231891305)

[Synchronization of Control Panel Applications 66](#_Toc231891306)

[Best Practices for Display Connection and Configuration 66](#_Toc231891307)

[Display Color Management 67](#_Toc231891308)

[High Color 67](#_Toc231891309)

[High Color in Windows 7 67](#_Toc231891310)

[Color Management Support for WDDM drivers 68](#_Toc231891311)

[Display Color Calibration Wizard 69](#_Toc231891312)

[Display Color Calibration and the VESA Monitor Control Command Set 70](#_Toc231891313)

[Display Color Calibration Wizard User Interface 71](#_Toc231891314)

[Best Practices for Display Color Management 77](#_Toc231891315)

[High-DPI and Readability 78](#_Toc231891316)

[The Importance of High DPI 78](#_Toc231891317)

[Windows 7 High-DPI Features 80](#_Toc231891318)

[Overriding the Windows DPI Defaults 81](#_Toc231891319)

[Testing Systems at High-DPI Settings 82](#_Toc231891320)

[Best Practices for High DPI and Readability 85](#_Toc231891321)

[Resources for Display Features and Design Considerations 85](#_Toc231891322)

[**PART 3: Multiple-GPU Systems** 87](#_Toc231891323)

[About Multiple GPU Systems 87](#_Toc231891324)

[Linked Display Adapters 88](#_Toc231891325)

[Feature Support for Multi-GPU Systems 88](#_Toc231891326)

[**PART 4: Windows Logo Program Requirements** 89](#_Toc231891327)

[Introduction 89](#_Toc231891328)

[Mandatory WDDM v1.1 Requirements 90](#_Toc231891329)

[GDI Hardware Acceleration 90](#_Toc231891330)

[Extended Color Format Support 91](#_Toc231891331)

[BGRA Color Format 91](#_Toc231891332)

[10-bit XR\_BIAS Desktop/Scan-out 91](#_Toc231891333)

[FP16 Desktop/Scan-out 92](#_Toc231891334)

[Connecting and Configuring Displays 92](#_Toc231891335)

[Support for Additional Target Mode Set on Each VidPN Target 95](#_Toc231891336)

[Support for Aspect Ratio Scaling 96](#_Toc231891337)

[Changes in How the DxgkRecommendVidPnTopology Function Is Called 96](#_Toc231891338)

[Changes in How Windows Handles ACPI Keyboard Shortcuts 97](#_Toc231891339)

[Required Driver Function for Hardware Capabilities 98](#_Toc231891340)

[Logging VSync Information 98](#_Toc231891341)

[Compiler Options 98](#_Toc231891342)

[Optional WDDM v1.1 Requirements 99](#_Toc231891343)

Encryption 99

[GPU-based Content Protection DDI 99](#_Toc231891345)

[DXVA-HD Support 99](#_Toc231891346)

[Overlay DDI 100](#_Toc231891347)

[Best Practices for Windows Logo Requirements 100](#_Toc231891348)

[Resources for Windows Logo Program Requirements 101](#_Toc231891349)

[**Glossary** 102](#_Toc231891350)

INTRODUCTION

Windows® 7 provides a graphically rich desktop experience and supports many advances in graphics hardware and software. To deliver the best desktop graphics experience in Windows 7, a system must take advantage of the graphics hardware paired with robust software in the following key areas:

* **Graphics processing unit (GPU)**. A GPU that supports a minimum of Microsoft® DirectX® 10 and Windows Display Driver Model (WDDM), “Optimized for Windows 7” is guaranteed to exploit most Windows 7 features and is best positioned to take advantage of innovations such as the Direct2D® API.
* **Display**. Advances in display technology greatly increase the options for end users, from tablet PCs to large high-definition displays. Connection and configuration of displays has become correspondingly complex. Original equipment manufacturers (OEMs) and independent hardware vendors (IHVs) should ensure a consistent Windows desktop experience across different innovative display technologies.
* **System Integration and firmware**. Rapid innovations in GPUs, chipsets, and processor technology present interesting opportunities for system manufacturers to integrate and package components to target specific consumer needs. Examples include technologies that take advantage of multiple GPUs in a single system and multiple monitors.

# Key Areas of Innovation for High-Fidelity Graphics in Windows 7

High-fidelity graphics means accurate, high-quality rendering across a range of target displays in many applications. Microsoft focused on the following areas to provide graphics innovations for Windows 7:

* WDDM version 1.1.
* DirectX technologies, including Direct3D® 11, Direct3D 10, and the new Direct2D API.
* Video playback.
* Connecting and configuring displays.
* Color management.
* High dots per inch (DPI) and readability.
* Multi-GPU systems
* Linked display adapters, also called co-rendering.

# About This Guide

This guide describes new features and changes in all areas of the display and graphics subsystem for Windows 7. OEMs, IHVs, and independent software vendors (ISVs) should refer to this document for basic guidelines about software and hardware design, Windows feature support, and proposed Windows Logo Program (WLP) requirements.

The guide is divided into four parts:

* Part 1 provides information about GPU and display driver design, including the WDDM v1.1 driver model DirectX technologies.
* Part 2 focuses on display connection and configuration, color management, and high-DPI and readability enhancements.
* Part 3 covers support for multi-GPU systems.
* Part 4 includes the WLP guidelines for WDDM v1.1 drivers.

Each part contains recommended best practices for design and testing, along with links to resources that provide further information.

# System Design Best Practices

This section summarizes the best practices in this guide for designing a Windows 7 system that delivers the best desktop graphics and connectivity experience.

#### Part 1: GPU and Display Driver Features and Design

##### WDDM v1.1 – support for the “Optimized for Windows 7” driver model.

For WDDM v1.1 drivers:

* Support the new GDI hardware acceleration interfaces and raster operations (ROPs).
* Support a maximum texture size of 8K x 8K.
* Use common industry benchmarks for 2‑D graphics operations to optimize for drawing performance.
* Ensure that OEM and IHV customizations such as Control Panel applications and services do not disable DWM composition.
* Support extended color formats such as blue-green-red-alpha (BGRA) and XR\_BIAS.
* Build WDDM v1.1 kernel-mode driver components with frame pointer optimizations (FPO) disabled.
* Record diagnosability information for TDRs in all possible scenarios.
* Do not cause a reboot or the loss of desktop functionality during driver upgrade.
* Invest in performance analysis and tuning to minimize system boot and resume times.

For GPU and graphics IHVs:

* Include digital connectors on the GPU for attaching displays.

##### DirectX technologies

* At a minimum, support DirectX 10 in the GPU.
* To deliver the best Windows 7 user experience, support DirectX 11 or DirectX 10.1.
* Implement WDDM v1.1 features in drivers.
* Keep the WDDM driver’s device context memory footprint as small as possible.
* Invest time and effort to reduce the WDDM driver’s device startup time.
* Do not use more system resources than a WDDM driver requires.
* Run industry benchmarks for gaming to validate driver performance on a particular system.

##### Video Playback

* Support standardized DXVA-HD if possible.
* Use the new overlay DDI for video presentation if your application requires overlays.
* Validate video playback performance in both windowed and full-screen video modes.
* Ensure a smooth video playback experience of common standard-definition and high-definition content.
* Test local and streaming video playback scenarios.
* Collect and evaluate performance traces if glitches occur during video playback.

#### Part 2: Display Features and Design Considerations

##### Display Ecosystem and Hardware

* Ensure that the display device EDID is valid. If the hardware does not report a valid EDID, provide an updated monitor INF that has EDID overrides.
* Ensure that connector cables do not result in loss of EDID during transmission.
* Build KVM switchboxes to meet the requirements in this document.
* Use digital connectors and cables wherever possible.
* Ensure that internal panels in portable computers and external display devices provide a valid EDID.
* Comply with all VESA display device connectivity standards.
* Provide monitor cables that have little or no signal attenuation.
* Test commonly used monitor switchboxes.
* Work with switchbox vendors to ensure that the EDID is correctly propagated through a switchbox.
* Thoroughly test portable computer docking stations for hot-plug functionality and connectivity.
* Support the PC mode in HDMI monitors if the IT content (ITC) flag is set.

##### Display Connection and Configuration

* Do not automatically apply display settings in value-added software.
* Use the CCD API and algorithms to automatically configure monitors and to save settings in the persistence database. Do not implement private persistence databases in value-added software or drivers for Windows 7.
* Map the OEM projection shortcut key to display the same UI as the new Windows logo key + P key combination.
* Correctly detect monitors on both analog and digital connections.
* Do not perform analog polling to determine display connectivity.
* Perform interruptible HPDs to provide the best connectivity experience.
* In value-added software, do not duplicate functionality that the Windows 7 Control Panel applications provide. For scenarios that Windows 7 does not cover, extend the Control Panel Displayapplication so that all display settings appear in one place.
* Implement overscan compensation for consumer electronics (CE) display devices in GPU software.

##### Display Color Management

* Incorporate display devices that support high-color formats into Windows systems.
* Use the high color technologies in Windows 7 to exploit the full potential of modern display and print hardware.
* Validate display devices for accurate primary colors, white-point, and black-point in the EDID data.
* Test that display LUTs are valid and persist after power transitions such as resume from sleep and hibernation.
* Test value-added display software with LUT loaders that are provided in third-party color calibration tools.
* Use wide color gamut displays to take advantage of the BGRA color format that Windows 7 requires.
* For rich color fidelity, use HDMI v1.3 displays that support xvColor.

##### High DPI and Readability

* Ensure that software and value-added Control Panel applications are high-DPI compliant.
* Use the DPI Configuration Matrix to test DPI settings.
* Report high-DPI issues to the vendors of third-party software that does not appear properly.
* Ensure that the EDID data is valid. The maximum horizontal size and maximum vertical size must be reported correctly in centimeters. Vendors must measure the panel sizes, not including the frame.
* Ensure that the DPI is configured correctly out of the box.
* Use the auto-configuration feature in Windows 7. Manual configuration is required only if the display hardware does not correctly report the EDID configuration.
* For ease of readability, provide displays that support 120 DPI or greater.
* Ensure that the default DPI setting works best with the portable computer’s internal display or with common external displays.
* Validate the resolution value that the EDID provides and ensure that the DPI set through the auto-configuration in OOBE corresponds to that resolution.
* Test setting your own DPI value by using the unattend.xml file.
* Test input devices for correct default speed settings.
* Validate your key Web sites and third-party software at high-DPI settings.

#### Part 4: Windows Logo Program Requirements

System manufacturers:

* For the best Windows user experience, build systems that have DirectX 10 or later graphics adapters and are WDDM v1.1 capable.
* Work with graphics hardware vendors to get certified WDDM v1.1 drivers for validation.
* Refer to the Windows 7 logo requirements kit for validation and testing information.
* Test a variety of hardware configurations on both desktops and mobile systems to ensure a solid end-user experience in Windows 7.
* Graphics hardware vendors:
* Work with Microsoft to develop Windows 7 WDDM v1.1 drivers.
* Test prerelease Windows 7 WDDM v1.1 drivers on the prerelease versions of Windows 7.
* Provide updated WDDM v1.x drivers to Microsoft for deployment through Windows Update.
* In addition to the Windows certification test suite, validate graphics and gaming performance, application compatibility, and various self-host scenarios on each ASIC family.
* Test Windows Vista-certified WDDM v1 drivers on prerelease versions of Windows 7.
* Make the full retail package for WDDM v1.1 drivers available as early as possible.

Independent software vendors (ISVs):

* Test existing and upcoming DirectX games with WDDM v1.1 drivers on prerelease versions of Windows 7.
* Test individual applications on prerelease versions of Windows 7.
* Start to develop Direct2D applications that target Windows.

PART 1:   
GPU and Display Driver Features and Design

In this part:

Introduction

Windows 7 Optimized Display Drivers

Direct3D 10

Direct2D API

DirectX 11

Video Playback

Resources for GPU and Display Driver Features and Design

# Introduction

The Windows 7 graphical desktop harnesses the power of the modern GPU by supporting new features together with key improvements in performance and stability. The desktop uses the two-dimensional (2‑D), three-dimensional (3‑D), and video engines of the modern GPU in the following ways:

* **2‑D engine.** The “Optimized for Windows 7 – WDDM v1.1” driver model introduces graphics device interface (GDI) hardware acceleration of the GPU’s 2‑D engine. This key feature reduces the system memory footprint of the Aero theme.
* **3‑D engine.** Desktop Window Manager (DWM) in Windows 7 uses the latest Direct3D API and can take advantage of the 3‑D engine in a GPU that supports DirectX 10. The new Direct2D API is also based on the high-performance Direct3D 10 API. Gaming innovations led to the development of Direct3D 11, which is a strict superset of Direct3D 10.
* **Video engine.** WDDM v1.1 introduces several optional features for video playback that help standardize the playback of high-definition content through DirectX Video Acceleration – High Definition (DXVA-HD), improve the security of graphics data, and improve the use of overlay presentations.

From a software perspective, the Windows 7 enhancements affect the following components:

* WDDM v1.1, optimized for Windows 7
* Direct3D 10
* Direct3D 11
* Direct2D API
* Video playback

This part of the guide discusses the new features and improvements that the WDDM v1.1 and DirectX technologies make possible.

# Windows 7 Optimized Display Drivers

WDDM v1.1 provides the optimal Windows 7 graphics experience. This updated driver model adds reliability and performance improvements as well as new features to the existing WDDM v1, which was introduced in Windows Vista®.

Figure 1-1 shows the two levels of user experience that are available, depending on the graphics hardware in the Windows 7 system.



Figure 1-1. Graphics performance in Windows 7

As Figure 1-1 shows, Windows 7 makes possible two levels of user experience:

* **Compatible with Windows 7**. A system that has a DirectX 9 graphics adapter and a WDDM v1 driver offers the Aero Glass theme on Windows 7 just as it did on Windows Vista. Other desktop improvements include a redesigned Windows taskbar and new color schemes.

Effective December 2008, the WLP was changed to make it possible for graphics hardware vendors to build WDDM v1.1 drivers for DirectX 9 graphics adapters. Such systems benefit only from the WDDM v1.1 features that reduce the memory footprint for Aero Glass and provide better viewing experience on TVs and widescreen portable computer displays.

Drivers that are based on the Windows XP Display Driver Model (XDDM) are also compatible with Windows 7 with one exception: Windows 7 64-bit operating systems do not allow the installation of signed XDDM display drivers that were certified before Windows Vista.

* **Optimized for Windows 7**. A system that has a DirectX 10 or higher graphics adapter and a WDDM v1.1 driver provides the best Windows 7 experience. Such a system benefits from the advantages of both DirectX 10.x graphics and WDDM v1.1. Users on low-end Windows 7 systems can open more windows simultaneously while using less system memory. The viewing experience on widescreen notebooks and television screens is better and more reliable. Windows 7 includes several gaming improvements that were made especially for WDDM v1.1 drivers on DirectX 10.x adapters. Direct2D applications also benefit from high-performance DirectX 10.x graphics.

For desktop graphics, Windows 7 introduces new features along with reliability and performance improvements that are built upon WDDM v1. WDDM v1.1 requires incremental changes in third-party display drivers to support some of these new features and enhancements.

Although WDDM v1 display drivers that were certified for Windows Vista are compatible with Windows 7, we strongly recommend that OEMs and system builders ship WDDM v1.1 drivers on Windows 7 systems.

WDDM v1.1 enables the following improvements in Windows 7:

* System memory savings in DWM.
* A high-performance Windows desktop that uses DirectX 10.
* High-performance Direct2D API as a Direct3D 10 client.
* Better viewing experience on TVs and widescreen portable computers.
* Performance gains for gaming scenarios that rely heavily on textures.
* Improved diagnosability of performance and stability problems.
* Standardized mechanism for high-definition composition through DXVA-HD.
* GPU-based content protection through standardized solutions.
* Improved video overlay presentation.

## Reduced System Memory Footprint

An important goal for Windows 7 is to improve system responsiveness by reducing the amount of system memory (RAM) that DWM uses. System memory usage significantly affects system responsiveness. Increased system memory usage leads to increased paging activity, which directly leads to *reduced* system responsiveness. Thus, for the best responsiveness, all applications, processes, and operating system components should use as little system memory as possible.

In Windows Vista, every GDI application window accounts for two memory allocations that have identical content: one in video memory and one in system memory, as Figure 1-2 shows. DWM and the graphics hardware use the copy in video memory to compose the window. The software-based GDI uses the CPU and operating system for rendering, without any assistance or acceleration from the graphics hardware. The CPU requires an easily accessible, cacheable copy of the memory allocation for use during rendering.

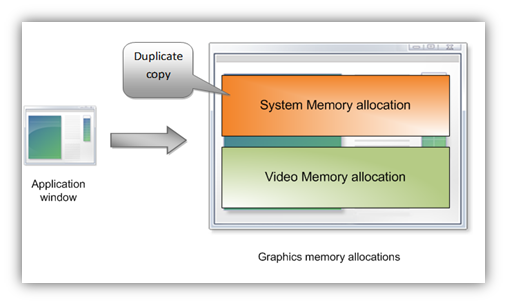
[](http://blogs.msdn.com/blogfiles/e7/WindowsLiveWriter/EngineeringWindows7forGraphicsPerformanc_C8A2/image_12.png)

Figure 1-2. DWM memory allocations in Windows Vista

Because of these two identical allocations, the amount of memory that is required for DWM in Windows Vista scales linearly with the number of windows that are open on the system. Memory pressure is even greater with higher resolution monitors and with multiple monitors.

The Windows 7 architecture eliminates the system memory allocation, as Figure 1-3 on the following page shows. This reduces memory consumption by half for each GDI application window that is visible on the desktop.

Performance studies show that this approach is feasible only if video drivers can accelerate some common GDI operations in hardware. To determine which operations to accelerate, we analyzed the top 100 GDI applications and worked with graphics IHV partners to accelerate the most commonly used GDI operations. For Windows 7, graphics IHVs are required to provide hardware acceleration support for some common GDI DDIs by implementing updated WDDM drivers. These GDI DDIs are a required part of the WDDM v1.1 driver model.

However, the elimination of the duplicate system memory copies involves one trade-off: slightly reduced performance when the CPU reads data back from video memory. An analysis of real-world application statistics showed that these operations are rare and that the memory savings directly result in Windows 7 being more responsive overall than Windows Vista. The improvements are especially noticeable on memory-constrained PCs that have shared memory graphics.

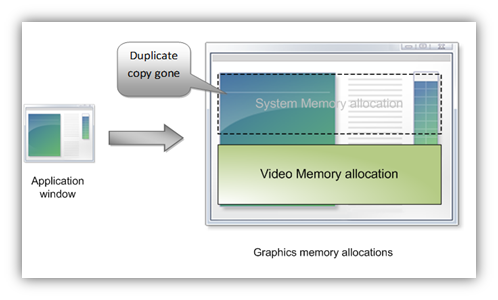
[](http://blogs.msdn.com/blogfiles/e7/WindowsLiveWriter/EngineeringWindows7forGraphicsPerformanc_C8A2/image_14.png)

Figure 1-3. DWM memory allocations in Windows 7

The following summarizes the memory savings for running Aero Glass in Windows 7:

* One system memory surface is saved for each top-level window. The more windows that are open, the higher the savings.

The calculation is as follows:

Table 1-1 lists the amount of system memory that is required to display multiple windows at various screen resolutions.

Table 1-1. System memory requirements for Windows 7

|  |  |  |  |
| --- | --- | --- | --- |
| Resolution | 5 Windows (MB) | 15 Windows (MB) | 30 Windows (MB) |
| 1024x768 | 15.0 | 45.0 | 90.0 |
| 1280x1024 | 25.0 | 75.0 | 150.0 |
| 1600x1200 | 36.5 | 109.5 | 219.0 |

Figure 1-4 on the following page shows the reduction in memory consumption for running Aero Glass in Windows 7 on a typical OEM system that runs Windows Vista. It shows the memory consumption that is measured by using Windows Performance Monitor on a system that has prototype WDDM v1.1 drivers installed. Memory savings increase when more windows are open on the desktop. Because DWM uses less system memory, less paging activity occurs and system responsiveness improves for the same workload.

The reduction in memory resource usage for existing low-end hardware is especially important for business customers who do not buy the most expensive high-end systems. Mobile form factors such as ultramobile PCs (UMPCs) and lightweight notebooks can benefit greatly because they often have 512 megabytes (MB) to

1 gigabyte (GB) of physical memory. With the substantial savings in system memory, users can now do more and their systems seem faster even when many windows are open.

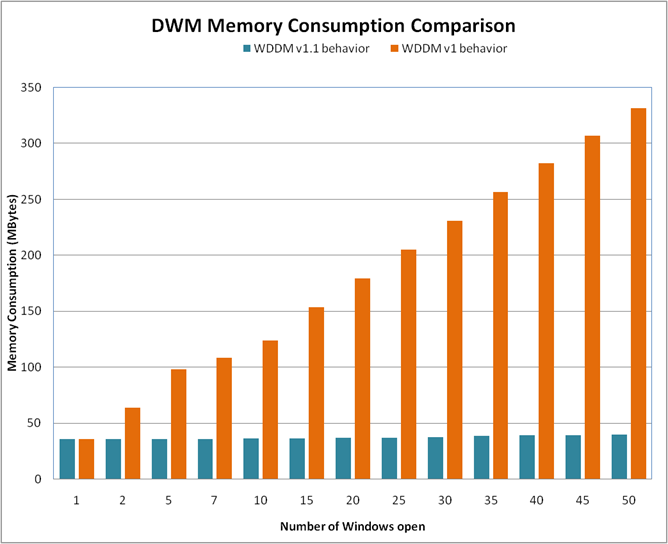


Figure 1-4. Memory consumption by WDDM v1 and WDDM v1.1 drivers

## Improved Gaming Performance

Windows 7 offers several performance improvements for games. After analyzing the performance of several top-tier DirectX games, we improved Video Memory Manager policies for memory allocation, placement, and eviction. WDDM v1.1 uses these improved policies, but WDDM v 1 does not.

Games and other DirectX applications that use large amounts of memory for textures benefit most from these improvements, especially when using a large screen or high-resolution monitor.

## Improved Diagnosability of Performance and Stability Issues

One of the most common stability problems in graphics occurs when the system appears completely “frozen” or “hung” while processing an end-user command or operation. Users generally wait a few seconds and then reboot the system by pressing the Power button. Usually the GPU is busy processing intensive graphical operations, typically during game-play. No screen updates occur, and users assume that their system is frozen.

Both Windows Vista and Windows 7 try to detect these problematic hang situations and recover a responsive desktop dynamically. The system does not reboot, but in most cases the screen flickers as it is redrawn. However, some older DirectX applications render a black screen at the end of recovery, and users must restart these applications. For more details on this process, see “GPU Hang Detection and Recovery” on the WHDC Web site.

These GPU hangs are referred to as time-out detection and recovery errors (TDRs). TDRs are of two types:

* Preemption-based TDRs occur when the GPU cannot preempt in time.
* VSync-based TDRs involve issues that are related to the VSync interrupt.

Windows Vista Online Crash Analysis (OCA) reports show that most TDR reports are classified as Undetermined. After reviewing this classification, we discovered that all the Undetermined TDRs were based on VSync. Unfortunately, Windows provided no additional data to help debug these issues and fix the problems. These errors are difficult to diagnose.

To help improve diagnosability, Windows 7 and WDDM v1.1 enable the logging of VSync-related data. WDDM v1.1 requires that display drivers log data that is related to Monitor VSyncs when TDRs occur. These improvements enable Microsoft and graphics IHVs to quickly respond to and fix any TDR stability issues. For more information about the diagnosability data logging, see “Windows Logo Program Requirements” in Part 4.

## IHV Graphics Drivers

Windows 7 maintains compatibility with older driver models so that XDDM and Windows Vista-certified WDDM v1 drivers install and operate on Windows 7 systems. To take full advantage of the graphics features in Windows 7, however, graphics IHVs must provide WDDM v1.1 drivers.

The graphics adapter that is present in the Windows 7 desktop or portable system determines the display driver that is installed on that system. For example, if an end user installs Vendor A’s DirectX 10 adapter, Windows 7 installs Vendor A’s WDDM v1.1 DirectX 10 - Windows 7 certified display driver. On a system that has Vendor B’s DirectX 9 adapter, Windows 7 installs Vendor B’s Windows Vista-certified WDDM v1 DirectX 9 display driver.

WLP changes that were made in December 2008 make it possible for graphics hardware vendors to build WDDM v1.1 drivers for DirectX 9 graphics adapters.

Figure 1-5 lists the driver types that can be installed on a Windows 7 system, depending upon the graphics adapter.

Figure 1-5. Display driver installation in Windows 7

### Graphics Hardware IDs

A device’s Plug and Play ID (PnP ID)—also called its hardware ID (HWID)—is the key to driver installation and update. In Windows 7, the graphics subsystem uses the hardware ID of the GPU for several purposes:

* To match the device to a driver during installation.
* To determine who is responsible for updating existing drivers.
* To uniquely identify the system that contains the GPU.

The HWID uniquely identifies a device and can also identify the hardware platform. A hardware ID can have four parts or two parts.

A four-part hardware ID indicates a specific implementation of GPU and computer model. It includes the device manufacturer, model, and platform. For example, the following four-part HWID identifies an NVIDIA GeForce 6600 GPU on a Toshiba Tecra M4 Tablet PC:

PCI\VEN\_10DE&DEV\_0146&**SUBSYS\_0010**1179&REV\_A2

The four parts of this ID are as follows:

**PCI\VEN\_10DE** specifies the PCI bus and the bus vendor ID that is assigned by the PCI special interest group (SIG).

**DEV\_0146** specifies the vendor-defined identifier for the device.

**SUBSYS\_00101179** specifies the vendor-defined subsystem identifier (0010) concatenated with the subsystem vendor ID that is assigned by the PCI SIG (1179).

**REV\_A2** specifies the revision number of the ID.

The following four-part HWID identifies the same GPU on a different system, the Toshiba Tecra M3:

PCI\VEN\_10DE&DEV\_0146&**SUBSYS\_0001**1179&REV\_A2

The different subsystem identifier indicates that the GPU is part of a different Toshiba model.

A two-part HWID identifies a generic implementation rather than a specific one, as in the following example:

PCI\VEN\_10DE&DEV\_0146

This HWID identifies the same GPU but does not specify a platform.

The difference between two-part and four-part HWIDs is important for driver installation. If the driver INF lists a two-part HWID, the driver installs on all configurations of the device, thus providing a 1:many mapping of driver to configuration. If the INF lists a four‑part HWID, the driver installs only on that specific configuration, thus providing a 1:1 mapping of driver to configuration. This enables IHVs and OEMs to provide OEM-specific driver packages for graphics hardware.

### Driver Distribution

Windows 7 supports the same driver distribution mechanisms as Windows Vista:

* Windows 7 inbox drivers
* Windows update
* Vendor Web sites
* OEM Web sites

#### Windows 7 Inbox Drivers

Inbox display drivers are included on the Windows distribution media and are automatically installed as part of Windows 7. Inbox drivers are generic and include no Control Panel applications, Help files, OpenGL installable client drivers (ICDs), or other additional software. However, they offer full Windows 7 functionality right out of the box. Both WDDM v1.1 and WDDM v1 drivers can be distributed inbox. The INF for an inbox driver must have a two-part HWID.

The goal in Windows 7 is to enable smooth upgrades for the most popular legacy devices and provide the best support for new features for recently released devices. Table 1-2 lists the vendors and platforms for which Windows 7 includes inbox display drivers.

Table 1-2. Inbox Drivers for Windows 7

|  |  |  |  |
| --- | --- | --- | --- |
| DirectX version | Vendor | Desktop computer | Portable computer |
| DirectX 10 | AMD | Yes | Yes |
| Intel | Yes | Yes |
| NVIDIA | Yes | No |
| Other | No | No |
| DirectX 9 | AMD | Yes | No |
| Intel | Yes | Yes |
| NVIDIA | Yes | No |
| Other | No | No |
| Legacy/XDDM | Any | No | No |

We provide inbox drivers for the most common graphics devices. For some devices, however, an inbox driver is not available. Drivers for some of these devices might be available on Windows Update.

#### Windows Update

Windows Update publishes the latest Windows 7–certified display drivers. Most of the drivers on Windows Update are fully featured, vendor-specific drivers that can have Control Panel applications, OpenGL ICDs, and so on.

WDDM v1.x drivers that are distributed on Windows Update overwrite earlier WDDM v1.x drivers that are provided through inbox distribution. The INF files for drivers on Windows Update generally specify four-part HWIDs. With Microsoft preapproval, vendors can publish generic drivers that have two-part HWIDs to replace previously shipped generic drivers. Such drivers update only previous generic drivers and cannot be installed over installed drivers that have four-part HWIDs.

#### Vendor Web Sites

Graphics IHVs also distribute beta or certified WDDM drivers for Windows 7 on their own driver distribution Web sites. Typically these drivers are not customized for specific OEM configurations and their INF files specify a generic two-part HWID.

#### OEM Web Sites

OEMs can work directly with graphics IHVs to obtain OEM-specific driver packages that are certified for Windows 7 and to post updated drivers on the OEM’s Web site. The INF files for these OEM-specific or customized drivers must supply 4-part HWIDs to indicate that they are the best match for the specific hardware.

### Unified WDDM Driver Packages

Today most graphics IHVs offer unified driver packages to their customers. A unified driver package contains one common driver that works across multiple generations of hardware. For example, when an IHV releases new hardware, the drivers that support this new hardware also support previously released versions of the hardware. This way IHVs can offer the benefits of driver improvements and bug fixes for all variations of the hardware that they ship and support.

To reduce the complexity of testing, the Windows 7 WDDM v1.1 driver interfaces are backward compatible with Windows Vista. Therefore, IHVs that implement Windows 7 WDDM v1.1 driver features can integrate them into common driver source code and build a single driver that runs on both Windows 7 and Windows Vista.

When installed on Windows Vista, such a driver simply acts as a WDDM v1 driver. When the same driver is installed on Windows 7, it can use the new WDDM v1.1 functionality. IHVs are not required to build and maintain multiple source code branches for Windows releases that support the WDDM driver model. Vendors who plan both to provide new drivers for Windows 7 and to update existing ones for Windows Vista also benefit from this approach.

The list of devices that each IHV supports is typically different for Windows 7 and Windows Vista. For this reason, we recommend that IHVs build two INF files for each WDDM driver package that supports both operating systems. One INF can support existing and new hardware for Windows Vista, and the other INF can support devices that meet Windows 7 logo certification.

Although the driver packages are unified, IHVs and OEMs must still pass WLP testing for each operating system version. A Windows Vista WDDM driver must run the required test suite and produce logs on Windows Vista, and a Windows 7 WDDM v1.1 driver must run the required test suite and produce logs on Windows 7.

### WDDM v1.1 Driver INF Requirements

The INF files for WDDM drivers in Windows 7 require several changes in addition to the new requirements that Windows Vista imposed. The most notable change is in the **FeatureScore**. WDDM v1.1 drivers require a higher **FeatureScore** than existing WDDM drivers.

The new requirements for display driver INFs in Windows 7 are as follows:

* Updated **FeatureScore** for all Windows 7 display drivers.
* Updated friendly name for WDDM v1.1 display drivers.
* SKU differentiation directive for all Windows 7 display drivers.
* Unicode encoding for all Windows 7 display drivers.

INF requirements in the following areas remain unchanged since Windows Vista:

* **InstalledDisplayDrivers** directive.
* **FeatureScore** directive.
* COPYFLG\_IN\_USE\_RENAME flag in the **CopyFiles** directive for the user-mode driver binary.
* **StartType** directive for services.
* **CapabilityOverride** settings.
* Friendly string name changes.
* **Version** section directive.
* **SourceDisksNames** and **SourceDisksFiles** directive.
* x64 specifications.

For more information about these requirements, see “Installing Display Miniport and User-Mode Display Drivers” in the Windows Driver Kit (WDK) on MSDN®.

The following sections describe the new INF requirements for Windows 7.

#### Updated FeatureScore Directive

The updated **FeatureScore** directive is a new general installation setting that is required for all WDDM drivers. Table 1-3 shows the values that apply for Windows 7.

Table 1-3. Feature Scores for WDDM versions

|  |  |
| --- | --- |
| Driver model | Feature score |
| Windows 7 WDDM v1.1 WHQL | E6 |
| Windows Vista/Windows 7 Unified WDDM v1.1 drivers certified by using the Windows 7 RC Logo Kit | E6 |
| Windows 7 WDDM v1.1 Inbox | EC |
| Windows Vista/Windows 7 Unified WDDM v1.0 drivers certified by using the Windows 7 RC Logo Kit | F4 |
| Windows 7 WDDM v1.0 Inbox | F4 |
| Windows Vista WDDM v1.0 WHQL | F6 |
| Windows Vista WDDM v1.0 Inbox | F8 |
| XDDM third party | FC |
| XDDM inbox in Windows Vista | FD |
| Video graphics array (VGA) | FE |

For WDDM drivers, the graphics IHV must include the feature score in the **FeatureScore** directive in the *DDInstall* section of the INF.

For XDDM drivers, Microsoft applies the appropriate feature score through the class installer at driver installation or in the INF for inbox XDDM drivers. Vendors must not insert a feature score for an XDDM driver.

An unsigned driver receives a feature score that is equal to FF. This value is the default and indicates no score.

#### Updated Friendly Name

The updated friendly name is a localizable string name that is required in the INF for every Windows 7 inbox display driver. To update the string, append the following to the device name:

(Microsoft Corporation – WDDM v1.1)

For example:

;

; Localizable Strings

;

XDDM Foo Device Name (Microsoft Corporation)

New Driver Model Foo Device Name (Microsoft Corporation – WDDM v1.1)

The updated friendly name specifies the WDDM version that the driver uses.

#### SKU Differentiation Directive

All inbox display driver INFs in Windows Server® 2008, Windows Vista SP1, and later versions must include a value that indicates that the drivers are for Windows Client editions only and that they do not install on Windows Server SKUs.

In Windows 7, Windows Vista SP1, and Windows Server 2008, the **Manufacturer** directive must be followed by a string in the following form:

NT<*platform*>…1

where *platform* is x86 or amd64.

For example, the following value applies for drivers for x86 systems:

[Manufacturer]

%ATI% = ATI.Mfg,NTx86...1

[ATI.Mfg.NTx86...1]

The following value applies for drivers on x64 systems:

[Manufacturer]

%ATI% = ATI.Mfg,NTamd64...1

[ATI.Mfg.NTamd64...1]

For additional information, see “INF Manufacturer Section” in the WDK.

#### General UNICODE Requirement

All inbox INF files must be encoded as Unicode; they must not be ANSI.

To check for UNICODE in INF files

1. Use Notepad to open the INF file.

2. On the **File** menu, click **Save As.**

3. If “ANSI” appears in the **Encoding** field of the dialog box, change the encoding to “Unicode” and save the file under a new name.

Figure 1-6 shows the **Save As** dialog box for a file that has ANSI encoding.



Figure 1-6. Save As Dialog Box that has ANSI encoding

Figure 1-7 shows the correct Encoding value.

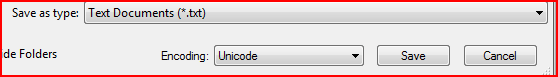


Figure 1-7. Save As Dialog Box that has Unicode encoding

### Future of the Windows XP Display Driver Model

We encourage system manufacturers to ship new systems with WDDM drivers for a stable and reliable Windows experience. Market data that we computed from OCA for Windows client systems shows that WDDM drivers are already available for most systems and only 1 to 2 percent of Windows Vista customers still use XDDM drivers.

Later Windows operating systems will advance the graphics platform based on WDDM. XDDM platform support might not be available in future versions of the Windows operating system.

Windows 7 provides the same level of support for XDDM drivers as Windows Vista, and any signed XDDM driver for Windows Vista installs on Windows 7. Such drivers do not support the Aero experience or other advanced graphics effects. Here is a quick summary of Windows XDDM driver support in Windows 7:

* Windows XDDM drivers cannot obtain a Windows 7 client logo. This is now true for Windows Vista also.
* UMPCs that run Windows Vista or Windows 7 now require WDDM drivers.
* Windows XDDM drivers can receive a Windows Server 2008 logo until June 2011. After then, drivers submitted for logo must support WDDM. The Microsoft WDDM inbox class driver provides support for 2D-only devices that do not meet WDDM requirements.
* Both signed and unsigned 32-bit XDDM drivers can install on 32-bit versions of Windows 7.
* XDDM drivers for Windows Vista migrate during an upgrade to Windows 7, with one exception. Only signed 64-bit XDDM drivers for Windows Server 2008 and Windows Vista can be installed on 64-bit versions of Windows 7. Signed drivers for earlier versions of Windows cannot be installed.

If you have any questions about current or future XDDM support, contact your Microsoft technology account manager (TAM).

## Best Practices for the WDDM v1.1 Driver Model

For WDDM v1.1 drivers:

* Support the new GDI hardware acceleration interfaces and raster operations (ROPs).
* Support a maximum texture size of 8K x 8K.
* Use common industry benchmarks for 2‑D graphics operations to optimize for drawing performance.
* Ensure that OEM and IHV customizations such as Control Panel applications and services do not disable DWM composition.
* Support extended color formats such as blue-green-red-alpha (BGRA) and XR\_BIAS.
* Build WDDM v1.1 kernel-mode driver components with frame pointer optimizations (FPO) disabled.
* Record diagnosability information for TDRs in all possible scenarios.
* Do not cause a reboot or the loss of desktop functionality during driver upgrade.
* Invest in performance analysis and tuning to minimize system boot and resume times.

For GPU and graphics IHVs:

* Include digital connectors on the GPU for attaching displays.

# Direct3D 10

Direct3D 10 is the foundation for current desktop improvements and future graphics innovations. Direct3D 10 provides a powerful API that ensures a consistent, predictable graphics experience across different graphics adapters. Tighter specification means that every Direct3D 10 adapter provides identical feature implementations. Direct3D 10 offers improved programmability, flexibility, and mathematical precision. In particular:

* Shader model 4.0 makes it easier to program sophisticated animations and material shading, complex lighting calculations, environmental effects, and post-processing effects.
* Performance improves because developers can do more on the GPU without involving the CPU.
* Reconfiguring the GPU is significantly faster.

The Direct3D 10 API that was released with Windows Vista represented a re‑architecture and ground-up rewrite of code from earlier Direct3D versions. Direct3D 10.1, which was released with Windows Vista SP1, includes incremental changes. Windows 7 further builds upon the Direct3D 10 infrastructure with the following feature areas:

* DWM
* Direct3D 10 Level 9

## Desktop Window Manager

DWM in Windows 7 uses the DirectX 10 interface that was first available in Windows Vista. On DirectX 10 hardware that has a DirectX 10 WDDM v1.1 driver, DWM uses the native Direct3D 10 API. Without either DirectX 10 or WDDM v1.1, DWM uses the DirectX 10 Level 9 driver that Microsoft provides. The DirectX 10 Level 9 driver is essentially a large subset of Direct3D 10 API that runs on DirectX 9 hardware and uses Direct3D 9 capabilities. For more information, see “Direct3D 10 Level -9” later in this paper.

By using Direct3D 10, DWM gains several benefits:

* Advantages of the Direct3D 10 API and techniques.

The Direct3D 10 API was a radical change from earlier DirectX APIs and provides several benefits to clients. DWM can thus take advantage of this new API.

* Better maintainability of the window manager for later versions of Windows.

DWM uses the DirectX Graphics Interface (DXGI) to discover adapter capabilities. DXGI eliminates common reliability problems and manages low-level tasks that are independent of the Direct3D graphics runtime.

DXGI provides a common framework for several versions of Direct3D. In earlier versions of Direct3D, the 3‑D runtime performed low-level tasks such as enumerating hardware devices, presenting rendered frames to an output, controlling gamma, and managing a full screen. These tasks are now implemented in DXGI. For more information, see “DXGI” in the DirectX Software Developers Kit (SDK) on MSDN.

* Ease of supportability.

Use of the Direct3D API and a clean rewrite of DWM automatically provide the benefit of great long-term supportability and the ability to make fixes if necessary.

* Performance.

Use of the Direct3D 10 API improves performance in some DWM micro-benchmark scenarios. For example, it increases the speed of glass blur and window update and at the same time reduces the memory footprint.

For DWM to run on Direct3D 10, the display driver must support BGRA color format as described in Part 4 of this paper. BGRA color format support is included in the WDDM v1.1 requirements, so that a WDDM v1.1 driver offers the premium experience for desktop graphics. In the absence of a WDDM v1.1 driver, DWM uses the DirectX 10 Level 9 driver.

## Direct3D 10 Level -9

Direct3D 10 applications can simultaneously target DirectX 9 and DirectX 10 hardware in Windows 7 by using the Microsoft Direct3D 10 Level 9 driver. The following is a summary of benefits:

* Unified source code for DirectX 10 and DirectX 9 hardware.

Game developers can maintain a single source code tree by using Direct3D 10 on both DirectX 10 and DirectX 9 hardware.

* Higher performance.

Direct3D 10 Level 9 is a specialized driver for the Direct3D 10 runtime. Therefore, game developers obtain the higher performance of the Direct3D 10 runtime (versus the Direct3D 9 runtime) but can still target the Direct3D 9 hardware.

* Positioning for the future.

Direct3D 11 runs on Direct3D 10 hardware, including Direct3D 10 Level 9. Extending a Direct3D 10 Level 9 code path to use Direct3D 11 features is straightforward, whereas extending a Direct3D 9 code path requires a complicated abstraction layer.

# Direct2D API

The Direct2D API enables applications to render 2‑D graphics with better performance and quality than GDI or GDI+. Direct2D is a new user-mode library that is based on the Direct3D 10.1 API. It interoperates easily with other Windows technologies, including the Windows Imaging Component (WIC), Direct3D 10, and the Windows Color System (WCS). Figure 1-8 on the following page shows how the Direct2D API layers over other graphics APIs.



Figure 1-8. Direct2D API layering

As Figure 1-8 shows, the Direct2D API is layered over the Direct3D 10 API and DXGI. These components enable Direct2D applications to run on Direct3D 10 hardware directly or on Direct3D 9 hardware by using the Direct3D 10 Level 9 driver.

To achieve the best performance, Direct2D requires a WDDM v1.1 driver. Such drivers support DirectX 10 hardware and the BGRA color format. If a WDDM v1.1 driver is not available, Direct2D uses Direct3D 10 Level 9.

Direct2D uses the Direct3D 10.1 API, which supports hardware-accelerated rendering on DirectX 10 GPUs. On Direct3D 9 hardware, Direct2D uses Direct3D 10 Level 9 rendering to achieve hardware acceleration. This combination provides excellent performance on the graphics hardware on most Windows PCs.

Direct2D includes a high-performance software rasterizer that is used if hardware acceleration is either not possible or not required. Applications that use Direct2D instead of GDI+ to render content in software perform much better with similar visual quality.

# DirectX 11

Windows 7 introduces the next-generation Direct3D 11 API, which is a strict superset of Direct3D 10. The Direct3D 11 API enables Windows 7 to take advantage of DirectX 11 hardware. Direct3D 11 adds features to the existing DirectX 10 (and 10.1) pipeline to improve 3‑D performance and support data-parallel computing on the GPU.

The Direct3D 11 API significantly advances graphics technology in two important areas:

* Compute shader for data-parallel computing.
* 3-D graphics.

In addition, it includes numerous other improvements that are based on extensive feedback from third-party hardware and software vendors. The following sections provide details about the advances in Direct3D 11.

## Compute Shader for Data-Parallel Programming

As the processing power of the GPU increases, it becomes a viable processor not only for games but also for general computing applications. DirectX 11 introduces a new compute shader API that enables the GPU to be used for data-parallel computing.

Data-parallel programming is a way to target parallel processors with code that scales to any number of processor cores. Direct3D is based on this programming model, but until Direct3D 11, various restrictions limited developer options and flexibility. The compute shader in DirectX 11 enables a broader set of algorithms to target the tera floating-point operations per second (TFLOPS) and gigabytes per second (GB/s) of the graphics processor.

### About Data-Parallel Programming

In 2003, silicon fabricators began to reach limits in chip clock rates because of power and thermal effects. Since then, manufacturers have focused on adding processor cores, instead of making individual cores run faster. Unfortunately as applications required better performance developers often were forced to substantially restructure their code base.

Developers approached the problem from two different directions. The multicore approach splits a code base in half and runs each portion on a separate core. However, synchronizing the two parts of the code base can be extremely complicated. Deadlocks and live locks make development difficult, and synchronization commonly limits performance boosts to well under 2X. Software developers must split the code base again when the number of cores again doubles.

The other approach is to parallelize each separate data element that the application processes. This involves identifying key inner loops and assigning them to separate cores. Vector supercomputers have used this approach, which has resulted in the introduction of many single-instruction, multiple-data (SIMD) parallel processors.

The data-parallel model has several advantages for developers. The primary benefit is that the number of data elements that the system can process scales automatically with the number of cores, provided that the number of data elements processed is large. Processing large numbers of small data elements is a common problem for many classes of applications. Media applications must handle audio samples or video/photo pixels and tiles. Database applications must manage millions of records. Financial, technical, and scientific applications often process huge numbers of floating point values.

In general, performing tasks in parallel is natural for hardware, whereas performing tasks serially is natural for software. The data-parallel abstraction isolates the application from hardware details, such as the number of cores. Therefore, the application code is the same and behaves the same regardless of the number of cores. Because the application does not require recompilation, the need for software developers to test and debug on different configurations is reduced. At the same time, hardware developers can innovate on the number of cores without breaking application compatibility. The DirectX 11 compute shader provides an easy way to use data-parallel programming to help applications achieve scalable performance for the next several generations of silicon performance growth.

### Graphics Processor as a Data-Parallel Processor

Today, every PC already contains a dedicated data-parallel processor: the GPU. The graphics processor has been designed to target 3-D rendering as a data-parallel workload for the past 15 years. The GPU can already deliver several times the performance of the CPU in 3-D rendering applications. It can also deliver as much as a TFLOP of general computing capability to other application areas.

A current high-end GPU can deliver about 4 TFLOPS of power for graphics applications, and one of those TFLOPS provides full 32-bit single-precision floating-point math that is generally useful. In addition, graphics processors often include a dedicated memory subsystem that has roughly 10 times the bandwidth of the CPU’s interface to main memory. Many applications other than graphics have already achieved substantial performance improvements by using the higher processing power and the greater memory bandwidth of the GPU.

The field of general-purpose computation on GPUs (GPGPU) has shown that algorithms for fast Fourier transforms (FFT), sort, and linear algebra—to name a few—can all attain higher performance levels by using Direct3D than by running on the CPU. Additional graphics-related applications that are not 3‑D, such as image and media processing and composition, are common uses for Direct3D 9 and Direct3D 10.

### The Compute Shader

The DirectX 11 compute shader provides a general mechanism for applications to access the computing power and bandwidth of SIMD cores such as those that are used in graphics processors. By using this flexible technology, many more applications can use the data-parallel processor capability of the GPU. To promote the development of compute-intensive applications, we encourage system manufacturers to work with graphics vendors and Microsoft to plan the move to DirectX 11–based graphics hardware in the Windows 7 timeframe.

The compute shader was designed to meet the following requirements:

* An API that is simpler than Direct3D.

The compute shader does not require an application to set up the parameters and state that are required for 3-D rendering. To send work to the GPU in previous DirectX versions, even algorithms that are totally unrelated to graphics had to draw a triangle. With the compute shader, an application can launch a thread by explicit request.

* Explicit separation between data-parallel code and serial code.

Data-parallel code usually runs on SIMD processors and typically implements substantially different algorithms than serial code. Application developers can completely control the programming model that each code segment uses.

* A single consistent programming model that spans hardware implementations.
* Automatic conversion of data types for read and write operations.

Media applications often use data types that are smaller than 32-bit floating-point values. Explicit conversion means that applications can avoid unnecessarily wasting bandwidth on 32-bit data.

* Interactive display of computation results, ideally updated at the monitor refresh rate.

Therefore, the computational work must be tightly integrated with the graphics tasks to meet the total refresh time constraint of 10 to 20 milliseconds (ms) (which results in 50 to 100 frames per second). For this reason, the DirectX compute shader is tightly integrated with Direct3D. This integration includes both the ability of compute shaders to read and write the array and surface objects that DirectX typically uses and the ability of the graphics pipeline to use scattered writes to update the more general data structures on which compute-oriented algorithms rely.

To further increase flexibility, polymorphism, and opportunities for optimization by specialization, the compute shader supports runtime compilation and runtime data binding.

Some algorithms are difficult to implement because Direct3D is pure data-parallel with no shared memory constructs or atomic operations. Because the compute shader is more general and contains the core data-parallel constructs for shared memory accesses with atomic operations, Direct3D 11 can execute a broader set of algorithms than previous versions, and some of these might be faster than the pure data-parallel versions.

### New DirectX 11 Features for the Compute Shader

In Windows 7, DirectX 11 includes the following new features to support general processing in the compute shader:

* Explicit thread dispatch.
* Random access I/O (scatter).
* Interthread communication by using locally shared registers.
* Ability to read and sample DirectX data objects.
* Atomic operators on shared memory locations.

The following sections provide details about these features. For more information, see “Supporting Direct3D Version 11” in the WDK.

#### Explicit Thread Dispatch

The compute shader provides an explicit **Dispatch**() entry point to clarify how many threads are launched for a given algorithm.

In earlier DirectX releases, an application was required to draw two triangles to begin work in multiple execution threads. Developers required a good understanding of polygon fill rules and texture sampling rules to guarantee that the number of threads that the application started matched the data set to be processed.

#### Random Access I/O (scatter)

By using the compute shader, applications can write data to arbitrary destinations. This feature enables developers to implement new types of algorithms that have output sizes that are determined at runtime. For example, variable-length compression and encoding schemes require this capability.

Earlier versions of Direct3D required the developer to pre-allocate the output destination of all units of execution before runtime.

#### Interthread Communications That Use Locally Shared Registers

The compute shader enables threads to read and write from a new class of registers that are visible to multiple threads. Threads can use these registers to cache previous results or to swap data across thread boundaries.

#### Ability to Read and Sample DirectX Data Objects

The compute shader can also use the standard resource types that the graphics APIs support for graphics-like image sampling and filtering and for random-access read and write operations.

#### Atomic Operators on Shared Memory Locations

When multiple threads try to access the same shared locations in video memory, conflicts and race conditions can occur. The compute shader includes a special set of intrinsic operators that enable atomic access to shared locations. An application can use these operators to ensure that its computations produce the same result regardless of the order in which threads execute.

### Supported Configurations for the Compute Shader

The DirectX 11 compute shader ships with DirectX 11 hardware and is available on Windows Vista and later versions of Windows. The Direct3D 11 API on Windows 7 supports the DirectX 11 computer shader.

A core subset of compute shader features is available on recent DirectX 10-class hardware so that developers can start to create DirectX 11 compute shader applications now on a hardware-accelerated development platform. Table 1-4 lists the shader features that DirectX 10 and DirectX 11 hardware support. Note that a “group” is the name for a collection of threads that run in the same instance of the compute shader.

Table 1-4. Compute Shader Features

|  |  |  |
| --- | --- | --- |
| Feature | DirectX 10 hardware | DirectX 11 hardware |
| Number of 32-bit shared registers | 4 K | 8 K |
| Shared register access | Private write/shared read | Fully indexed |
| Maximum group dimensions | (768, 768, 1) | (1024, 1024, 64) |
| Maximum group threads | 768 | 1024 |
| Atomic operators | None | Full set |
| Double precision | None | Check feature |
| DispatchIndirect() | None | Supported |

#### Target Applications

Many applications can benefit from the compute shader technology. The following list describes several such applications, ordered by similarity to current Direct3D applications.

* Photo and imaging.

Although most imaging tasks are similar enough to graphics that Direct3D is a good solution, some operations such as FFT benefit substantially from the random access I/O capabilities of the compute shader.

* Video.

Video compression and encoding are difficult to accomplish without random access I/O. The compute shader enables many key algorithms that are used in video encoding that simply were not feasible previously. Currently, applications typically use fixed-function cores to decode video. The compute shader adds a TFLOP of flexible computing power, which enables applications to perform many additional tasks in real time instead. Such tasks include super-resolution scaling, fast noise removal, and de-ringing.

* Advanced rendering.

Games and visualizations benefit from A-Buffer techniques for order-independent transparency (OIT) and real-time radiosity that uses a global illumination (GI) model to render realistic, diffuse lighting effects.

* Search, sort, and query.

Databases often handle large numbers of small records that data-parallel programming can target efficiently. Many search, sort, and query algorithms can easily be parallelized at a fine-grained level. Some of them benefit from processing models that do not enforce ordering of results, such as datasets that will be sorted later.

* Technical and scientific.

Academic researchers who required increased performance were the first to program graphics processors for applications beyond 3-D rendering. Scientific and technical computing can benefit greatly by using the compute shader, especially when single-precision math is often used.

* Cryptography.

Since the release of Direct3D 10, GPUs have supported internal computations on integer data types. Combined with the new flexibility of the compute shader, integer arithmetic enables client PCs to treat their GPU as an encryption and decryption accelerator and enables server-clustered PCs to process large data sets.

## 3‑D Graphics Improvements

Direct3D 11 addresses limitations in the performance and visual quality of today’s graphics. It supports new and improved features for the following:

* Use of multiple CPU cores.
* Tessellation.
* High-level shading language.
* Cross-platform development with the Microsoft XBox® 360 platform.

### Use of Multiple CPU Cores

Direct3D 11 increases the throughput of rendering calls by distributing the application, runtime, and driver workloads across multiple cores. Parallel application threads can generate precompiled rendering commands without synchronization. In addition, background threads can efficiently create and initialize device objects such as textures. On a multicore CPU, multithreading can significantly improve end-user frame rates and 3‑D graphics application performance.

The multithreading feature is also available on Direct3D 10 hardware. The Direct3D 11 runtime communicates with existing Direct3D 10 drivers to enable this feature. Although Direct3D 11 supports multithreading on all Direct3D 10 drivers, OEMs should work with their graphics IHV partners to obtain updated Direct3D 10 drivers that are fully optimized for multithreading.

### Tessellation

Direct3D 11 implements tessellation on the GPU to calculate a smoother curved surface from a coarse, less detailed, input patch. Tessellation subdivides each quad or triangle patch face into smaller triangular faces that better approximate the desired surface.

By using tessellation, an application can render a single model at several levels of detail. This technique shrinks the gap between higher quality offline content creation and faster real-time rendering. For example, a model that is far away can be rendered with low detail, but when seen up close, the same model can be refined to be smoother and more attractive, while maintaining consistent frame rates.

In effect, tessellation acts as a form of compression. The coarse base model results in faster load times from disk to main memory and from main memory to the GPU.

Direct3D 11 hardware supports tessellation of geometry and higher-order surfaces such as Non-Uniform Rational [B-Spline](http://en.wikipedia.org/wiki/B-spline) (NURBS), N-patches, bicubic patches, and Catmull-Clark subdivision, which digital content creators commonly use. Even more specifically, Direct3D 11 tessellation supports:

* Triangle and quad patch primitives.
* Adaptive and continuous refinement of the domain topology.
* Programmable evaluation of higher-order surfaces with displacement.

By using tessellation in these models, DirectX 11 enables applications to render graphics that are truer to the original artist’s creative vision.

### High-Level Shading Language

The high-level shading language (HLSL) has new subroutines and better support for double-precision floating-point computation. These features provide a more expressive, more flexible programming model that enables ISVs to better express their vision in code.

Software developers sometimes design programmable shaders by combining multiple subroutines. This approach often results in a large number of custom shaders, each of which uses a combination of subroutines to realize slight variations. However, these custom shaders have the penalty of expensive state changes. The alternate approach—sometimes called “one shader to rule them all”—uses flow control statements to choose the optimal shader. However, a single shader causes unnecessary register pressure because the compiler must allocate registers based on the worst-case path. Poor register allocation can seriously reduce runtime performance.

To address this problem, subroutines extend the Direct3D 11 runtime with a simple, expressive programming model that provides an abstract layer over the combinatoric complexity but still achieves the performance of the custom precompiled shaders. As a result, programming complexity migrates from the application level to the driver level.

### Cross-Platform Development for Xbox 360

Many Direct3D 11 features improve cross-platform development between PC games and Xbox 360 games. In particular, multithreading and tessellation use and extend the existing functionality of Xbox 360. The increased ease of cross-platform development helps vendors more easily create games for both XBox 360 and the PC.

## Additional Direct3D 11 Features

In addition to 3‑D improvements and support for data-parallel computing, Direct3D 11 incorporates several other improvements that are based on extensive ISV and IHV feedback.

### Improved Texture Compression

Direct3D 11 introduces the first high-dynamic range, compressed hardware texture format and a new higher quality compression format for standard dynamic range textures. These new formats result in better looking textures and imagery in 3‑D applications.

### Shader Model 5.0

Shader model 5.0 adds many new features, including the following:

* Support for high-precision math for scientific and CAD applications.
* Bit reversal instructions for FFTs.
* Shader conversion instructions for FP16 to FP32 and vice versa.

### Stream Output Flexibility

Direct3D 11 can perform multi-element stream output to multiple independent streams, including the rasterizer. This reduces the need for expensive CPU involvement in multipass GPU rendering and computation.

### Depth Buffer Capabilities

Direct3D 11 accelerates techniques that often use depth buffers, which are common in the current generation of games. Direct3D 11 can:

* Write depth from the shader conservatively, maintaining early-z acceleration.
* Declare a read-only depth view that can be simultaneously bound as depth and texture, allowing z-comparison, z-rejection, and texture read without copying the depth buffer. Scenarios include soft particles and volumetric fog effects.

Best Practices for DirectX Technologies

* At a minimum, support DirectX 10 in the GPU.
* To deliver the best Windows 7 user experience, support DirectX 11 or DirectX 10.1.
* Implement WDDM v1.1 features in drivers.
* Keep the WDDM driver’s device context memory footprint as small as possible.
* Invest time and effort to reduce the WDDM driver’s device startup time.
* Do not use more system resources than a WDDM driver requires.
* Run industry benchmarks for gaming to validate driver performance on a particular system.

# Video Playback

Windows 7 includes enhancements to the video pipeline that enable standardized solutions for media center and third-part video IHVs. The video playback enhancements include the following:

* Common DXVA-HD device driver interface (DDI)
* GPU content protection.
* Video overlay DDI in WDDM v1.1.

Historically, video source data that is compressed in one of the popular video codec formats—such as H.264, MPEG2, or RVC-1—and encoded. The video pipeline receives such data, processes it, and presents it over an output connector to the monitor. Because the video source is usually encrypted, the decoder must perform a key exchange with the display driver to decrypt the video stream and encrypt the compressed data before it sends the data to the GPU for further video processing. Video data is ready for presentation after it is de-interlaced (if necessary), color converted, composited, and possibly stretched.

## DXVA-HD DDI

WDDM v1.1 provides a common DXVA-HD DDI that can efficiently handle high-definition composition for DVD playback. ISVs can use the new public interface instead of inventing a private interface.

Figure 1-9 shows how ISV applications can layer over the DXVA-HD support that Windows 7 provides.



Figure 1-9. DXVA-HD layering

As Figure 1-9 shows, third-party applications can use the Direct3D 9 API to represent a graphic context and to compose the final presentation. The DXVA2 API provides hardware acceleration for video decoding—for example, MPEG2, VC-1, and H.264 audio/video control (AV/C). The new DXVA-HD API provides hardware acceleration for video processing and composition.

The DXVA-HD DDI supports the Blu-ray format, which has a highly flexible composition model that is controlled by Blu-ray Disc Movie Mode (HDMV) or Java for Blu-ray Disk (BD-J). The new digital high-definition broadcast standards—such as the Japanese Integrated Services Digital Broadcasting-Terrestrial (ISDB-T) with broadcast markup language (BML)—use similar technology. Currently, some applications use Direct3D shaders instead of DXVA2 to perform video processing and composition. This model is inefficient and does not work for low-end desktop and portable systems. Such applications should instead use the DXVA-HD API.

The DXVA-HD API/DDI also provides support for video adapters that have dedicated video processing hardware for low-end and mid-end application-specific integrated circuits (ASICs). Development of interfaces for this dedicated hardware is difficult. The unified DXVA-HD API/DDI removes the need for each ISV to invent its own private interface.

The new DXVA-HD DDI allows all input streams to be either red-green-blue (RGB) or YCbCr (YUV) and composes them all in a single pass. All input streams share the same capabilities; there are not separate main and substreams. The DDI provides a flexible mechanism for manipulating and processing streams. Each stream can be associated with variable stream states, which provides an efficient way to modify the composition of a stream “on the fly” during processing. The DDI also provides enhancements to the following features:

* Multi-interlaced video deinterlacing.
* Luma keying for Blu-ray.
* Background image for Blu-ray (BD-J).
* Flexible alpha blending and alpha channel data handling.
* Scenario-based color parameter and flexibility.
* Static and dynamic performance optimization.
* Frame rate conversion and inverse telecine.
* State-based video process bit-block transfer (BLT).
* Extensibility for proprietary techniques.

## GPU Content Protection

WDDM v1.1 supports standardized Advanced Encryption Standard (AES) 128-bit encryption in addition to hardware and software content protection. AES 128-bit encryption is a standard solution for Media Center and third-party ISVs.

### Standardized AES 128-bit Interface

WDDM v1.1 exposes a new interface that provides standardized AES 128-bit encryption of a video stream.

In Windows Vista, most of the process of encrypting the input video stream is performed by private libraries that IHVs develop for video decoders. The decoders call Direct3D 9 to get the PCI ID of the GPU, and then call the appropriate IHV library to encrypt the stream. This technique is not only cumbersome, but also means that the decoder itself cannot directly control or guarantee the level of encryption.

In Windows 7, applications can use the authenticated channel and the crypto session interface to achieve AES-128 bit encryption for protecting premium video content. The decoder knows what types of encryption the driver supports and the content protection capabilities that are associated with the encryption. A key exchange mechanism enables the decoder to establish a session key for the crypto session.

### Driver Software Protection

A display driver can provide defense-in-depth protection against bad code in a process. For example, the driver can ensure that protected content cannot be locked or copied to unprotected surfaces. An authenticated channel between the application and the driver can provide additional protection against shared surface attacks.

Although display driver software protections provide more defense-in-depth than the Direct3D 9 protections, they can never meet the protection level that hardware can provide.

### Driver Hardware Protection

Some graphics adapters have content protection built into hardware, which makes the content very secure. However, an authenticated channel still communicates with the display driver, which provides only software protection. Although such channels are quite secure, they are less secure than an authenticated channel to a secure processor within the GPU.

## Video Overlay DDI in WDDM v1.1

WDDM v1.1 includes a new DDI for video overlay presentation in Windows 7. This DDI simplifies the overlay presentation model because it targets the use of overlays purely for presentation and not for de-interlacing or composition effects.

Windows 7 supports the ability to keep DWM on while presenting overlays and reduces per-frame composition time to improve the performance of overlay video playback. To take advantage of this support, user-mode WDDM v1.1 drivers require only minor changes.

Features of the overlay presentation model include the following:

* The overlay model is simple and does not support shrinking, mirroring, de‑interlacing, source color-keying, or alpha blending. Most of this support has also been removed from current overlay hardware because the GPU performs these tasks well without software support.
* The overlay DDI allows for stretching but only if the actual overlay hardware supports stretching.
* The overlay model always uses destination color-keying. The application cannot use overlays to composite text or player controls. The WDDM v1.1 DDI draws the color key and Windows determines the color-key value.
* The overlay presentation model uses an HWND window handle and a client rectangle to track where the data is presented on the screen. Direct3D 9 detects and handles all window movements at presentation time rather than in response to a WM\_SIZE message. The application cannot display an overlay independently of an HWND.
* Each device can have only one overlay.

Best Practices for Video Playback

* Support standardized DXVA-HD if possible.
* Use the new overlay DDI for video presentation if your application requires overlays.
* Validate video playback performance in both windowed and full-screen video modes.
* Ensure a smooth video playback experience of common standard-definition and high-definition content.
* Test local and streaming video playback scenarios.
* Collect and evaluate performance traces if glitches occur during video playback.

# Resources for GPU and Display Driver Features and Design

#### MSDN Web site:

DirectX Developer Center

<http://msdn.microsoft.com/en-us/directx/default.aspx>

DXGI

<http://msdn.microsoft.com/en-us/library/cc835730(VS.85).aspx>

Supporting Direct3D Version 11

<http://msdn.microsoft.com/en-us/library/dd434682.aspx>

#### Windows Driver Kit (WDK):

INF Manufacturer Section

<http://msdn.microsoft.com/en-us/library/ms794359.aspx>

Installing Display Miniport and User-Mode Display Drivers

<http://msdn.microsoft.com/en-us/library/ms797609.aspx>

#### WHDC Web Site:

Display and Graphics home page

<http://www.microsoft.com/whdc/device/display/default.mspx>

GPU Hang Detection and Recovery

<http://www.microsoft.com/whdc/device/display/GPUHang_Det-Rec.mspx>

#### Questions and Comments

Desktop and Graphics Technology Team

[directx@microsoft.com](mailto:directx@microsoft.com)

PART 2:   
Display Features and Design Considerations

#### In this part:

[Introduction](#_Toc230488703)

[Display Ecosystem and Hardware](#_Toc230488704)

[Display Connection and Configuration](#_Toc230488710)

[Display Color Management](#_Toc230488719)

[High-DPI and Readability](#_Toc230488723)

[Resources for Display Features and Design Considerations](#_Toc230488725)

# Introduction

Windows 7 helps users connect and configure display devices more easily and reliably on both desktop and portable systems. Windows 7 user interface (UI) enhancements—such as the Projection shortcut key—aid mobility and flexibility. The result is a more predictable and consistent experience when users connect an external display device to any desktop or portable PC.

Advances in display device support include the following:

* **Improved multimonitor support.** To improve the multimonitor experience, Windows 7 tracks the arrival and removal of display devices. It automatically configures each device based on user-applied settings or optimal defaults.
* **New and improved UI.** The Windows logo key (Picture of the Windows logo key)+P is defined as a shortcut key with which users can easily choose portable projections, extended desktop on multiple monitors, or portable-only or projection-only displays. The improved Control Panel **Display** application adds support for monitor rotation, desktop duplicate mode, display detection, and live representation of connected display devices.
* **Streamlined user experience.** Windows 7 optimizes defaults for common scenarios and provides appropriate fallbacks for corner cases. Defaults are the same for desktop and portable systems except when users add a monitor or projector. In that case, the user’s desktop display is duplicated on the portable system and extended on the desktop system. Fewer unique UI controls are required to configure display devices. Windows 7 also exposes an API and a persistence database so that OEMs and IHVs can more easily integrate their hardware features and value-added software with Windows capabilities.
* **Improved reliability and robustness**. Windows 7 display management better handles faulty and legacy hardware, improves reliability during display detection and is less susceptible to timing issues.
* **Increased support for color**.Windows 7 provides additional support for high-color hardware and includes a color calibration wizard so that graphics professionals can easily match displayed colors to printed or natural colors.
* **Improved readability**. Automatic DPI setting and per-user DPI settings make it easier for users to set the display at a native resolution that renders readable text. Manufacturers can override the Windows defaults to provide device-specific settings if required.

Part 2 provides details about the new features and enhancements as they affect these areas:

* Display ecosystem and hardware.
* Display connection and configuration.
* Color management.
* High DPI and readability.

# Display Ecosystem and Hardware

Windows 7 display management offers a hardware-agnostic user experience. Users have a consistent experience across PCs that are manufactured by different system manufacturers and contain different graphics hardware. All the display connectivity and configuration features are available on both desktops and portable computers. Windows 7 display management is designed to operate correctly in the following user scenarios:

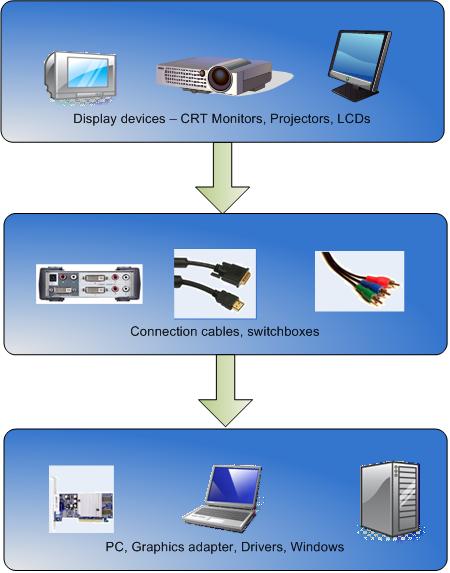
* Connecting or disconnecting a display device when the PC is in the working state and in a low power state.
* Connecting one or more displays and configuring the screen resolution, orientation, and display arrangement by using the Control Panel **Display** application.
* Transitioning the PC through different power states, including sleep/resume and shutdown/reboot.
* Docking and undocking a portable computer, or connecting a portable computer to a docking station that has multiple display devices.

To ensure a good user experience, OEMs and hardware vendors should test their systems, hardware, and drivers in all these scenarios.

The large number of connectivity options and external and internal display devices offers a challenge and an opportunity for PC and device vendors to make it easy for end users to use the display device of their choice in a way that best suits their needs. The display ecosystem must provide the following:

* Defaults that enable the system and display device to work together appropriately without user intervention when the display device is connected or disconnected.
* Flexibility in configuring systems for advanced users, especially in multimonitor configurations.
* Standards that enable devices from multiple vendors to communicate seamlessly.

To achieve these goals, all the elements of the display ecosystem must work together. Figure 2-1 shows these elements.

Figure 2-1. Display ecosystem

As Figure 2-1 shows, the display ecosystem consists of several elements:

* **Display devices**. Any monitor, projector, or TV (LCD, plasma, or CRT) is considered a display device that can be connected to a PC for viewing the desktop. This can also be extended to form factors such as digital signage, UMPCs, and other such devices.
* **Connection cables, switchboxes, and extenders**. A cable connects the display device to a PC. Types of cable interfaces and connectors include Component, Composite, S‑video, VGA, Digital Video Interface (DVI), High-Definition Multimedia Interface (HDMI), DisplayPort, and so forth. These cables connect the display device directly to the graphics adapter in the PC. A switchbox or port replicator can connect multiple PCs to a single display device. The PC and the display device connect to the switchbox by using cables. An extender passes an unchanged signal from one cable to another, so that users can connect two short cables to make one longer cable. In addition, a portable PC might be connected to a docking station, which passes a signal from the graphics card to a connected display device.
* **PC hardware and software**. The display device is connected to the PC. The PC renders the desktop that is projected on the display device. The graphics adapter in the PC renders the image. This document does not differentiate between desktop and portable PCs, except as specifically noted.

Vendors should follow Video Electronics Standard Association (VESA) standards and pay extra attention to the requirement for Extended Display Identification Data (EDID). The following sections present guidelines for display hardware, switchboxes, docking stations, and cables.

## EDID for Display Devices

To ensure that Windows can correctly set the display resolution, the display device must implement and transmit a well formed EDID structure that contains all required fields, as defined in VESA Enhanced Extended Display Identification Data Standard (E‑EDID), Release A, Section 3, and mark the native panel resolution as the “preferred” resolution. The following guidelines apply to the display EDID:

* The EDID must provide a monitor name that is suitable for presentation to the user.
* The EDID must provide a unique ID, manufacturer name, and product code IDs.
* The EDID must provide a unique monitor serial number in either the serial number field or in a Detailed Timing Block.
* The EDID must set the preferred mode bit and include timing data for the preferred display mode in Detailed Timing #1.

For an LCD or other fixed-format display, the preferred display mode must be the native mode of the panel.

For other display types, the preferred display mode is the optimal display mode, which is based on the size and capabilities of the device. It must meet the requirements for refresh rates.

The preferred mode must be a progressive mode.

* The EDID must implement monitor descriptors, including monitor range limits and monitor name.
* The EDID must implement monitor range limits unless the display device supports only discrete timings.
* The EDID must provide accurate physical dimensions of the display device, unless the device is a projector.
* Display devices that support more than 8 bits per pixel must use EDID 1.4 to describe the color capabilities.

If the display device EDID is not available, Windows applies a default resolution that is based on the connector with which the device is attached. The default resolution for a VGA connector is 1024 x 768 at 60 Hertz (Hz).

If the display hardware does not provide a correctly formed EDID, the monitor INF should supply the correct information. If a display device contains an invalid EDID, Windows 7 applies display device defaults that might not be the best for the device. An OEM or IHV can override the incorrect EDID and the Windows 7 defaults by supplying the correct information in a monitor INF. For details, see “How to Use an INF to Override the Monitor EDID” on the WHDC Web site.

## Switchboxes and Extenders

Advanced users of PCs often use keyboard, video, and mouse (KVM) switchboxes to connect multiple PCs to a single display device. Some users also join video cables by using extenders. For VGA and DVI switchboxes, standards have not been explicitly defined and unpredictable behavior often occurs. The primary challenges have been inaccurate detection of connected display devices and missing or faulty EDID data. New standards such as HDMI and DisplayPort clearly define expected behavior for such devices.

Manufacturers of switchboxes can ensure that their customers have a good user experience with Windows 7 by following the guidelines in this section and testing their hardware with Windows 7.

Figure 2-2 shows a typical KVM switchbox configuration.



Figure 2-2. Typical KVM switchbox configuration

In Figure 2-2, three PCs (PC1, PC2, and PC3) are connected to a switchbox (S). The switchbox is connected to one external monitor (M1). The switch is set so that M1 displays the PC1 desktop.

To provide a good user experience with Windows 7, switchboxes should follow the guidelines in this section for the following tasks:

* Detect a physical monitor.
* Provide the accurate EDID.
* Respond to PC queries.
* Support monitor switching.
* Pass inter-integrated circuit (I2C) calls from the PC to the monitor.
* Use VESA-specified cable.
* Indicate power state.

The following guidelines refer to the configuration in Figure 2-2.

**Detect a physical monitor.** The switchbox must always be aware of the presence or absence of a physical monitor that is connected to it.

To detect a monitor when the switchbox first obtains power

When S first obtains power, it must detect the presence or absence of M1.

1. On a digital connector, the switchbox can query the hot plug detect (HPD) pin. On an analog connector the switchbox can perform load sensing.

2. If a monitor is present, the switchbox must notify each PC by generating an HPD event as specified in the DVI specification or—for an analog connector—by setting a load on the line that is the same as the load that M1 generated. This enables the graphics driver to detect the presence of the monitor.

For example, the switchbox can pull the I2C line high and then terminate the RGB line. This causes the graphics driver to determine that a monitor is present on the analog connector.

To detect a monitor while the switchbox is already powered on

While S is powered on, it must detect when a user physically connects or disconnects the monitor in the same way as at power-up.

1. If the connection state of M1 changes, the switchbox must notify each PC (PC1, PC2, and PC3) in the same way that it notified the PCs at power-up: by generating an HPD event or by setting a load on the line that is equivalent to the load that M1 generated.

2. The graphics driver can then detect the presence of the monitor.

**Provide the accurate EDID. The switchbox must always provide the accurate EDID from the connected monitor. Whenever S becomes aware of the presence of M1 as described in the guidelines for detecting a monitor, it must make the EDID from the monitor available to each connected PC (PC1, PC2, and PC3).**

In addition, any PC can query S for the EDID by using I2C commands whether or not that PC is currently connected to the monitor. S must return the accurate EDID from M1 and must not provide any default EDID. If M1 does not have an EDID, S must not return an EDID.

For more information about monitor EDID, see “EDID for Display Devices” later in Part 2.

**Respond to PC queries.** Any PC can query S for the presence of a monitor at any time by checking the HPD pin on a DVI connector or by load sensing on an analog connector. S must always be able to determine whether M1 is connected to S. The result of this query should be the same regardless of which PC is currently connected to the monitor.

**Support monitor switching.** If users switch the monitor from one PC to another, the switchbox must not give the impression that a monitor has been connected or disconnected. Specifically:

* For a digital connector, the switchbox must not generate an HPD event.
* For an analog connector, the line must not have signal noise that could cause a graphics driver to determine that M1 has been connected or disconnected.

**Pass I2C calls from the PC to the monitor.** For the PC that is currently connected to the monitor, the switchbox must pass through all I2C calls so that all VESA Monitor Control Command Set (MCCS) and display data channel (DDC) commands work as expected.

For PC1, S should pass through all I2C commands to M1 with no modifications. For PC2 and PC3, this is not required, until they are switched to M1.

However, the EDID must always be available to all PCs, as described previously in “Provide the EDID.”

**Use VESA-specified cable.** The cable that is provided with the switchbox must conform to VESA specifications for cable length, signal integrity, presence of all required data lines, and so on.

**Indicate power state.** The switchbox should clearly indicate when it has sufficient power to be fully functional, typically by providing a “Power on” LED.

## Docking Stations

In Windows 7, display settings for the docking station are stored in the display persistence database along with other display settings. Each time a display device is connected to the computer either directly or through the docking station, Windows 7 applies the persistent settings. For more information about the persistence database, see “Display Setting Persistence Database” later in Part 2.

To ensure that the graphics driver can keep track of the current display devices, the OEM BIOS must signal each dock or undock event to the graphics driver. In response, the graphics driver can query for a change in the state of connected display devices and then notify Windows. BIOS support is critical for the success of this feature.

Docking station display settings are no longer stored in the hardware profiles.

## Cables and Connectors

To complement the robust software solutions that are part of Windows 7, we strongly recommend that partners implement modern digital connectors on GPUs and integrated graphics. Vendors should follow VESA standards for cables and connectors.

Best Practices for Display Ecosystem and Hardware

* Ensure that the display device EDID is valid. If the hardware does not report a valid EDID, provide an updated monitor INF that has EDID overrides.
* Ensure that connector cables do not result in loss of EDID during transmission.
* Build KVM switchboxes to meet the requirements in this document.
* Use digital connectors and cables wherever possible.
* Ensure that internal panels in portable computers and external display devices provide a valid EDID.
* Comply with all VESA display device connectivity standards.
* Provide monitor cables that have little or no signal attenuation.
* Test commonly used monitor switchboxes.
* Work with switchbox vendors to ensure that the EDID is correctly propagated through a switchbox.
* Thoroughly test portable computer docking stations for hot-plug functionality and connectivity.
* Support the PC mode in HDMI monitors if the IT content (ITC) flag is set.

# Display Connection and Configuration

Windows 7 incorporates new logic for connecting and configuring displays (CCD) that supports the following features:

* Control Panel Display application.
* Projection shortcut key.
* Multimonitor support.
* Default settings.
* Deprecated transient multimonitor manager.
* CCD API.
* Better detection of display devices.

## Obtaining Hardware and Configuration Information

The UI enhancements in Windows 7 are based on fundamental changes that make it easier for the system and graphics drivers to obtain and coordinate information about the graphics hardware and display settings. These changes include the following:

* A display setting persistence database.
* A new CCD API.
* Changes to the graphics kernel.

### Display Setting Persistence Database

Windows 7 stores the display settings and re-applies them each time the same display device is reconnected. A display setting persistence database retains important display settings, such as the most recently used topology, display resolution, display orientation, refresh rate, bits per pixel, display layout, and aspect ratio. The persistence database is always available and the display settings are applied even before the logon screen appears, which reduces the number of screen flashes after users log on.

The persistence database stores display settings based on the EDID of the display device itself and not on the graphics card to which the device is connected. This means that if users disconnect a display device from one adapter and connects it to another, Windows 7 applies the same display settings. To differentiate between different devices that have the same EDID, Windows uses the serial number that is stored in the EDID. Therefore, it is critical for the success of this feature that all display devices have well-formed EDIDs and unique serial numbers. For more information on the EDID requirements, see “EDID for Display Devices” earlier in Part 2.

With one exception, the persistence database stores settings at the system level and not the user level because display settings primarily reflect hardware capabilities and position rather than user preference. Because CCD by default configures the display device to its optimum settings, per-user settings are not required. The only setting that persists on a per-user basis is the readability setting, which makes the text appear bigger or smaller. For more information about this setting, see “High DPI and Readability” later in this document.

Third-party applications should use the system’s persistence database to store configuration settings instead of maintaining a separate persistence database. This ensures an optimal user experience with few screen flashes.

### Connecting and Configuring Displays API

Control Panel applications that OEMs and IHVs supply must be synchronized with each other and with the Windows 7 inbox Control Panel Display application. Third-party Control Panel applications should not duplicate functionality that Windows provides. Windows 7 exposes a new CCD API that applications can use to configure display devices. The CCD API and the persistence database provide a unified way for Windows, drivers, and applications to query, set, store, and retrieve monitor configurations.

The CCD API provides access to the persistence database. Third-party drivers and applications should use the API to prevent conflicts between their settings and the settings that Windows uses, and to ensure that their Control Panel applications use the same information as the Windows Control Panel Display application. For additional details, see “Connecting and Configuring Displays” in the WDK for Windows 7.

### Changes to the Windows Graphics Kernel

The graphics kernel incorporates many changes that improve reliability and robustness and enable some features to work even with older or faulty hardware. Key improvements in this area include the following:

* The monitor EDID is always available to the CCD logic.

Windows 7 eliminates errors that occurred on earlier versions when the monitor EDID data was not available for the CCD logic in time to configure the display. On systems with faulty hardware, EDID processing was sometimes delayed behind slow Plug and Play operations. In such situations, configuration proceeded without the EDID and the CCD logic could not re-apply the appropriate persisted settings. This problem has been fixed in Windows 7.

* Long faulty VGA cables cause less disruption.

Long VGA cables between a projector and a portable computer can sometimes generate noise, particularly in large conference rooms with long cables and a lot of electronic equipment. Noise on the cable can prevent a driver from reliably determining whether a projector is connected and can result in frequent screen flashes.

Windows 7 detects such conditions and prevents the screen from flashing so that users can continue the presentation. In addition, for the best user experience, large conference rooms should use cables that have adequate noise shielding and also use good repeaters that transmit a strong signal on all VGA signal lines.

## Projection Shortcut Key

In Windows 7, the Windows logo key (Picture of the Windows logo key)+P is defined as a shortcut key to select projection options for portable computers. When users press the shortcut key, Windows displays the dialog box that is shown in Figure 2-3.

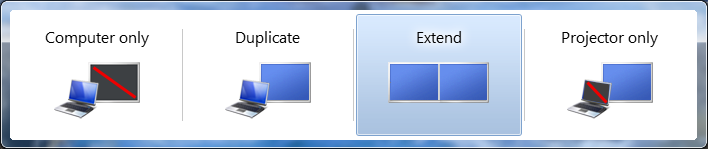


Figure 2-3. Projection shortcut key dialog box

The Picture of the Windows logo key+P shortcut always provides four connection options in the following order:

* Computer only  
  Shows the desktop only on the portable computer screen.
* Duplicate  
  Mirrors the desktop on the computer screen and on the projector.
* Extend  
  Extends the desktop to appear on both the computer screen and the projector, so that users have more desktop space.
* Projector only  
  Shows the desktop only on the projector.

The Projection shortcut key is optimized for portable computers, but it can be used on desktop computers as well. Each time users select a projection option, Windows 7 detects display hardware to ensure that the option applies to the current configuration. The Projection shortcut key is always available. Even at the logon screen, the shortcut key can be used to start and stop projection.

Although the dialog box does not change if more than one display device is present, the user’s selection applies to all the display devices that are connected to the PC. That is, the Extended desktop and Projector only options display the desktop on as many display devices as are present. This feature is particularly useful on a docking station that has two connected display devices.

The Projection shortcut key also provides a solution for faulty hardware that the system cannot detect. If users press the shortcut key, the system automatically forces the output to the analog connector if the system cannot detect a display device.

### Integrating OEM Shortcut Keys with the Projection Shortcut Key

To ensure a consistent user experience and eliminate possible confusion, the system should have only one Projection shortcut key. OEMs should map their shortcut keys to the Windows projection shortcut key, so that both shortcut keys display the Windows Projection shortcut key dialog box.

Windows 7 enables a more consistent user experience for OEM shortcut keys than Windows Vista. In Windows 7, the CCD API records information in the display setting persistence database so that both the operating system and OEM applications can retrieve it. OEMs should use the database to apply display settings so that both Windows and OEM Control Panel applications use the same information. Third-party drivers and applications use the CCD API to request a particular display topology. In response, Windows 7 applies the most recently saved display settings for the requested topology.

For Windows Vista, the OEMs and the graphics IHVs worked together to implement ACPI-based OEM specific shortcut keys. Windows queried the graphics driver for information about the topology and display settings to apply. The operating system was unaware of the display settings that were applied.

#### Implementation Guidelines for OEM Shortcut Keys

OEM shortcut keys are typically combinations of the Fn key and a function key (F1 to F12). They can be implemented through the BIOS or by a user-mode application.

To map an OEM shortcut key to the Projection shortcut key, the system BIOS must inject keystrokes into the Windows input stack. If the OEM shortcut key is an Fn+Fx key combination, the BIOS must report the keystrokes as listed in Table 2-1.

Table 2-1. OEM Shortcut Key Behavior

| User action | Key reported |
| --- | --- |
| User presses and holds the Fn key. | Do nothing. |
| The Fn key is down, AND user presses down and holds the Fx key. | “Windows Logo key” Down followed by “P key” Down |
| User releases the Fx key. | “P key” Up |
| User presses the Fx key. | “P key” Down |
| User releases the Fx key. | “P key” Up |
| User releases the Fn key to apply the settings | “Windows Logo key” Up |

The state diagram in Figure 2-4 shows which keystrokes the BIOS should report for each possible state transition.

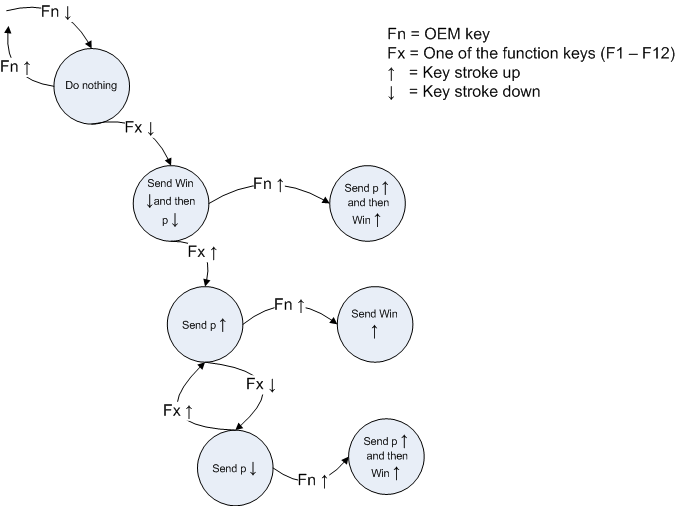


Figure 2-4. State diagram for Projection shortcut key

The following additional guidelines apply:

* If the keyboard has two Windows logo keys, the BIOS should report the left Windows logo key.
* The BIOS must ensure that settings made by using the OEM shortcut key (Fn+Fx) are synchronized with those made by the Windows logo key. Some users might use both the Picture of the Windows logo keykey and the Fn+Fx keys. OEMs that define an OEM shortcut key are responsible for inserting the correct keyboard scan codes to keep the Fn+Fx and Picture of the Windows logo key key presses synchronized with each other.

After the system displays the Projection dialog box, subsequent keystrokes should work as follows:

* The Fn+Fx key moves from one option to the next.
* The P key moves from one option to the next.
* A left mouse click selects an option.
* The arrow keys move from one option to the next.
* The ESC key dismisses the dialog box.
* While the user holds down the Fn key, additional presses of the Fx key move the selection to the next option in the dialog box.

An OEM shortcut key can cause a change in display topology only if the driver provides the required topology. Windows applies display settings to the new topology from the persistence database or by using the algorithms that define the defaults.

An OEM shortcut key can change the display settings only if the driver provides exactly the same topology as is currently applied, but with the required new display settings. Windows applies the setting and updates the display persistence database accordingly.

#### Testing Guidelines for OEM Shortcut Keys

Test OEM shortcut key integration in the following scenarios:

* Test all four available options with the Projection shortcut key during normal operation with a logged-on user.

Either the settings must be re-applied from the database or new settings should be applied as described in “Default Settings for Display Devices” later in Part 2.

* Test all four options while the user is at the logon screen. The Projection shortcut key is designed to work before a user logs on.
* Test stress conditions to ensure that the operating system and BIOS do not become unsynchronized if the user uses both the Fn+Fx combination and the Picture of the Windows logo keykey+P combination.
* If the system receives a monitor HPD notification, ensure that the dialog box hides itself.

### Mobility Center Connect Display Button

In Windows 7, the Mobility Center continues to provide an easy way for users to configure the settings of a portable computer. Figure 2-5, on the following page shows the **Windows Mobility Center** UI. The **Connect Display** button is still available to users, but it simply launches the Projection shortcut key UI. Users can then select a projection mode.

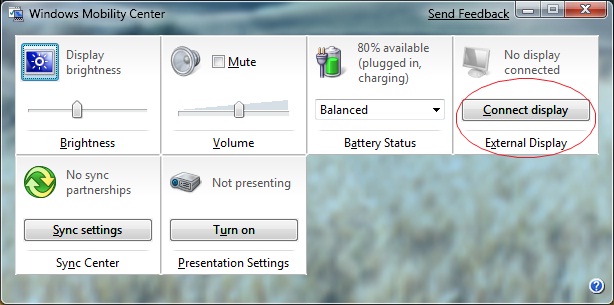


Figure 2-5. Mobility Center

## Display Device Detection

Windows 7 detects display devices more reliably than previous versions. Although Windows 7 can perform detection in several ways, it now relies exclusively on the most reliable detection mechanism, and all WDDM graphics drivers use the same mechanism. Windows 7 performs detection in the following situations:

* All system transitions from low to high power state: resuming from sleep, resuming from hibernation, and initial boot.
* The user presses the new Projection shortcut key or an OEM shortcut key that is mapped to the Projection shortcut key.
* The user clicks **Detect** on the Control Panel **Display** application.
* The user selects **Scan for hardware changes** in Device Manager.

In addition, all digital connectors (DVI, HDMI, and DisplayPort) raise interrupts to signal the connection or disconnection of a display device. Vendors should pursue hardware solutions for raising similar interrupts for display devices that are connected to analog (VGA) connectors.

The CCD logic tracks interrupts from all display connectors. If such an interrupt occurs, Windows checks the persistence database for the last known good settings to configure the device. If the persistence database does not contain settings for this device, Windows applies the optimal defaults, as described in “Default Settings for Display Devices” later in Part 2.

To provide a good user experience, third-party applications must not automatically apply other display settings that override the system’s persistent settings. However, applications can apply changes that are based on user request.

## Default Settings for Display Devices

Windows 7 optimizes defaults for common scenarios and provides appropriate fallbacks for corner cases. Defaults are the same for desktop and portable systems except when users add a monitor or projector. In that case, the desktop display is duplicated on the portable PC system and extended on the desktop system.

As soon as a new display device is connected, Windows 7 attempts to enable the display and set it to its optimum settings. The most important settings are the display topology and the display resolution.

For the display topology, the key decision is whether to enable the display in Duplicate (mirrored) mode or Extended desktop mode. For portable computers, the default is Duplicate mode. For desktops, the default is Extended mode.

For the display resolution, Windows 7 sets the display to its native resolution as specified in EDID. LCD panels can natively support only one resolution, so it is critically important to set the display in that resolution for a rich desktop experience. If an LCD display is not set to its native resolution, the desktop requires scaling, which could negatively affect the desktop color and the readability of text. To increase the text size, users should use the Readability feature in Control Panel and described in “High DPI and Readability” later in Part 2.

## Multimonitor Support

To improve the multimonitor experience, Windows 7 tracks the arrival and removal of display devices. It automatically configures the device, based on user-applied settings or optimal defaults.

### Deprecation of Transient Multimonitor Manager

Because of the projection shortcut key and the new Control Panel Display application, the Windows Vista Transient Multimonitor (TMM) feature is no longer required and has been deprecated. WDDM drivers for Windows 7 are not required to support the TMM interfaces.

When users connect a display device, Windows 7 does not display the TMM dialog box. Instead, the system applies default settings and users can change the defaults by using the Projection shortcut key or the Control Panel Display application.

Figure 2-6 shows the deprecated TMM dialog box.

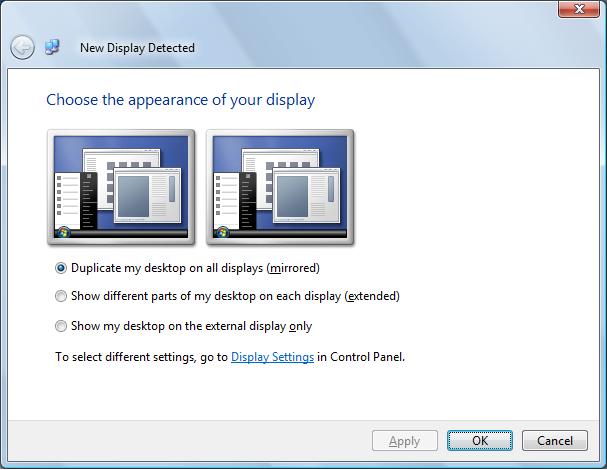


Figure 2-6. Deprecated Windows Vista New Display Detected dialog box

The TMM feature in Windows Vista periodically polled the analog (VGA) connectors to determine whether the display device had been connected or disconnected. This approach enabled the system to automatically detect analog devices, but it introduced challenges that were related to performance, power consumption, and reliability of detection.

Windows 7 does not periodically poll the analog connector. The optimal implementation of analog detection can be done only in hardware. Therefore, we recommend that the graphics adapter or the system connector implement automatic detection on analog connectors, either by load sensing or by using I2C-based commands.

### Multiple Adapters and Multiple Monitors

Windows 7 considers all graphics adapters that are connected to the PC along with all display devices that are connected to these graphics cards. If users connect a new graphics adapter to multiple display devices, Windows 7 enables all display devices without user intervention. If users disconnect a display device from one adapter and connect it to another adapter, Windows 7 applies the existing display settings with the new adapter. In addition, the Projection shortcut key works with multiple adapters.

## Control Panel Display Application

The improved Control Panel Display application adds support for monitor rotation, desktop duplicate mode, and display detection. When users connect a new display device, the system updates Control Panel immediately to show the new device. In Windows 7, the Control Panel Display application supports the following features:

* Rotation.
* Duplicate desktop mode.
* Showing all connected monitors.
* Scanning for newly connected monitors.
* Showing the monitor name.
* Recommendations for native display resolution.

For the best user experience, a single UI should be able to configure all display settings. Some features are available in both the Control Panel Display application and in third-party value-added Control Panel applications. OEMs and IHVs should analyze the functionality that the Display application provides and limit their value-added Control Panel applications to scenarios that are not covered in the Windows 7 inbox UI. To cover specialized scenarios, we encourage OEMs and IHVs to extend the **Advanced Settings** tab of the Control Panel Display application.

Vendors commonly extend Control Panel to provide a UI for the following:

* Changing the TV broadcast format.
* Providing overscan compensation.
* Setting the aspect ratio.

This section describes the UI of the Control Panel Display application and explains how to extend it to control TV format, overscan compensation, and aspect ratio.

### Control Panel Display Application User Interface

The UI for the Control Panel Display application enables users to configure all the commonly used settings. Users can reach it by right-clicking the Desktop and selecting **Screen resolution** or by using Control Panel. It is also available through Windows search.

In Windows 7, the Tablet PC Settings dialog box no longer offers users an option to change the display orientation. Instead, users are provided a convenient link to the Control Panel Display application, which now supports display orientation configuration.

Figure 2-7 shows the Windows 7 Display UI.

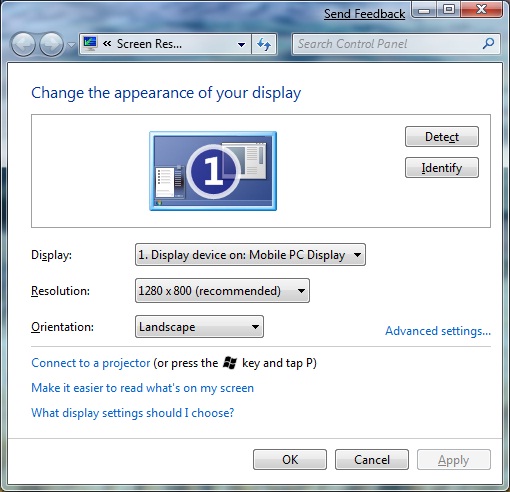


Figure 2-7. Windows 7 Control Panel Display application

The following sections describe some of the key features of the new Control Panel Display application.

#### Number of Display Devices

The Control Panel Display application shows both active and inactive display devices that are connected to the PC.

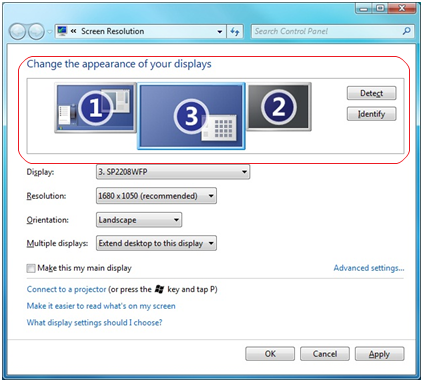


Figure 2-8. Better Multimonitor support

#### Friendly Monitor Names

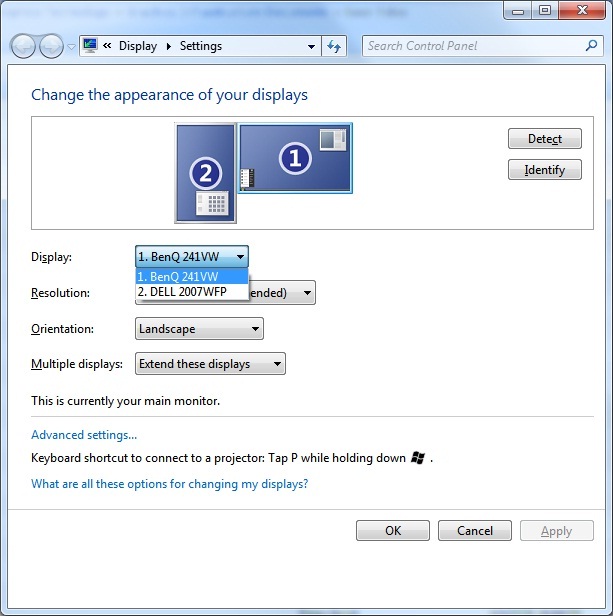


Figure 2-9. Monitor names in the Control Panel Display application

#### Highlighted Native Resolution and Aspect Ratio

The resolution slider highlights the native resolution of the display device, along with other resolutions that match the native aspect ratio, to help users select the best settings. This way, even if users choose a resolution that is not native to the device, the desktop does not appear stretched or compressed.

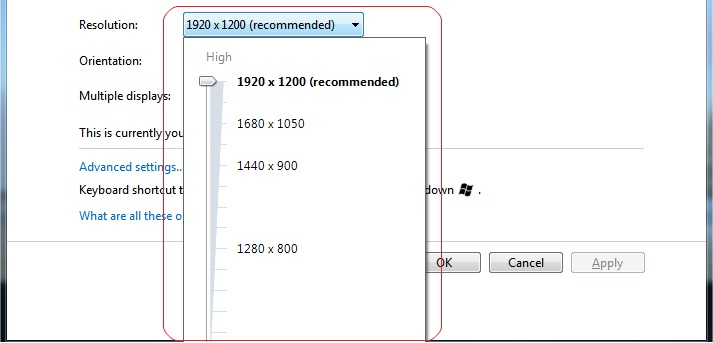


Figure 2-10. Recommended native resolution

#### Display Orientation

Users can set the desktop orientation of the active display devices, as shown in Figure 2-11.



Figure 2-11. Display orientation

#### Multiple Monitor Settings

Users can set common multimonitor settings for all active and inactive display devices, as shown in Figure 2‑12.



Figure 2-12. Multimonitor settings

#### Readability Settings

The Control Panel **Display** application encourages users to change the Readability settings instead of the display resolution to make the text bigger or smaller, as shown in Figure 2‑13.



Figure 2-13. Link to High-DPI dialog box for better readability

#### Analog TV Detection

A **Detect** button enables users to request that the system detect any analog display devices. Additionally, even if the system does not detect an analog display device, users can choose **Try to connect anyway on: VGA** to force the display on the analog connector.

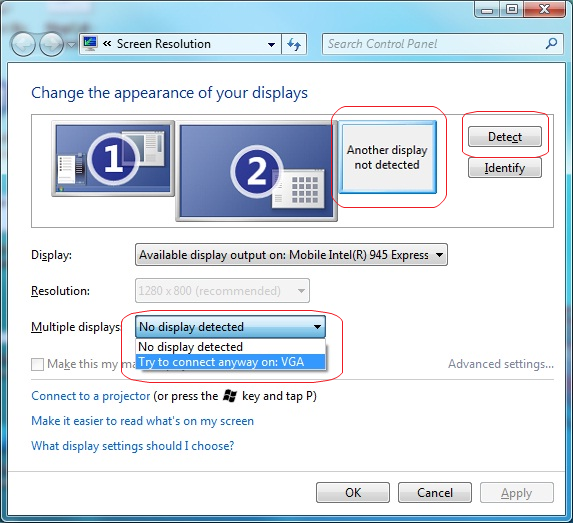


Figure 2-14. Support for analog display devices

### Guidelines for Extending the Control Panel Display Application

Third-party drivers are not required to extend the Control Panel Display application. However, any such extensions should follow these guidelines:

* If the driver provides TV format, overscan compensation, or aspect ratio settings, the driver package must provide a UI that adds tabs to the **Advanced Settings** page of the Control Panel Display application as described in “Requirements for TV Settings Support” and “Requirements for Aspect Ratio Support” later in Part 2.
* The Control Panel Display application extension must be installed as part of the driver package and cannot be included in the inbox driver.
* The UI must be designed in accordance with the guidelines for extending the Control Panel UI, which are described in the OEM Preinstallation Kit (OPK). For more information, see “Microsoft OEM Preinstallation Kits” on the Microsoft Web site.
* The UI must contain the elements that are described in this section and must not contain any additional functionality. It can optionally have one button that starts the full IHV Control Panel application.
* The extension must use the new CCD API. Drivers should use private interfaces only for cases that the CCD API does not cover.
* Upon starting, the UI must accurately reflect the current state of the display.
* When display settings change, the UI must automatically update itself by handling the WM\_DISPLAYCHANGE and WM\_DEVICECHANGE messages.

To display the UI, users must:

1. Right-click the desktop and select **Display Settings**.

2. In the Control Panel **Display** application, select **Advanced Settings** .

Figure 2-15 shows a typical **Advanced Settings** page.

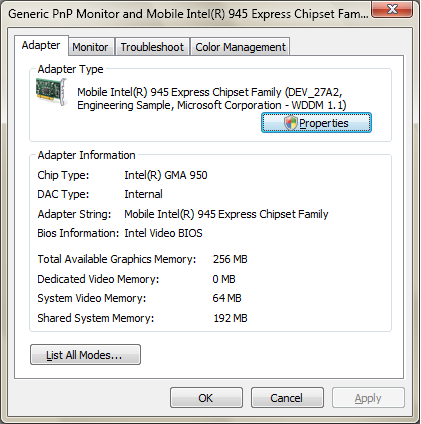


Figure 2-15. Typical Advanced Settings page

IHVs can add two additional tabs to this page, as shown in red on Figure 2-16.

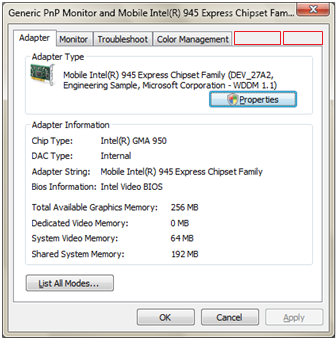


Figure 2-16. Extending the Control Panel Display application

### Requirements for TV Settings Support

Graphics IHV partners can extend the Control Panel Display application with a UI to control TV format and overscan compensation. Drivers must use the new CCD API as indicated to send requests and settings to Windows.

#### Requirements for Supporting TV Format

Broadcast television standards differ in various parts of the world. The TV format control enables users to change display settings to project images on a TV that does not support the same settings as their computer. For example, a user who travels from one geographic zone to another might connect a portable computer to the TV in the hotel room. This TV does not support National Television System Committee (NTSC), so the user must change the settings to a phase alternating line (PAL) to project images on the TV.

The following are the requirements for supporting TV format:

* Provide a simple drop-down menu that enables users to choose one of the geographic settings—such as PAL, NTSC, Sequential Color with Memory (SECAM), and so forth—that the driver supports. Windows does not provide a function to set TV format, so the driver must use a private interface.
* Apply the changes when the user selects **Apply** or **OK**.
* After applying the changes, use the CCD API to send a request to Windows to re‑enumerate the display resolutions that match the selected geographic setting.
* After the re-enumeration is complete, use the CCD API to set a resolution that is appropriate for the selected geographic setting.
* Update the slider on the main tab of the Control Panel Display application to reflect the resolutions that are now available.

#### Requirements for Supporting Overscan Compensation

Overscan compensation enables users to change the size of the desktop so that it is completely visible on a TV. For example, when users connect a portable computer to a TV to watch a movie, the Start menu might not be visible because of TV overscan. Similarly, when users connect a portable computer to a projector in a conference room, part of the desktop might be cut off.

The following are the requirements for supporting overscan compensation:

* Provide one slider to expand and shrink the desktop vertically.
* Provide one slider to expand and shrink the desktop horizontally.
* Update the desktop on the fly as users interact with the slider.
* After users commit the settings, use the CCD API to set the final mode and save it in the persistence database.
* Optionally, provide a check box to force the sliders to move together to maintain a particular aspect ratio.

Figure 2-17 shows a sample TV Settings tab.

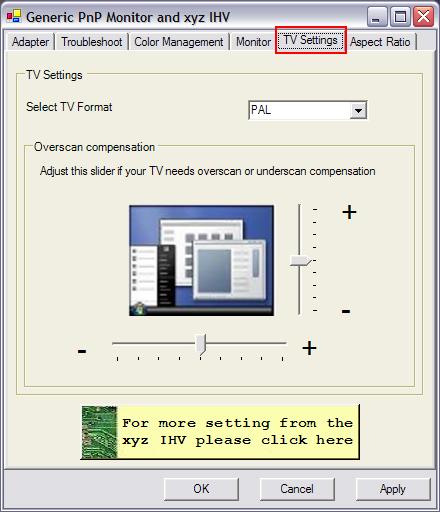


Figure 2-17. Sample TV Settings tab

### Requirements for Supporting Aspect Ratio Settings

Aspect ratio settings provide additional user control over the desktop image. For example, if users connect a widescreen portable computer to a 4:3 projector, the system by default stretches the desktop, which changes the aspect ratio. By supporting aspect ratio settings, a vendor can enable users to preserve the aspect ratio.

The requirements for supporting aspect ratio settings are as follows:

* Make the following options available to users:

Centered

Stretched

Aspect ratio preserving

No scaling (identity)

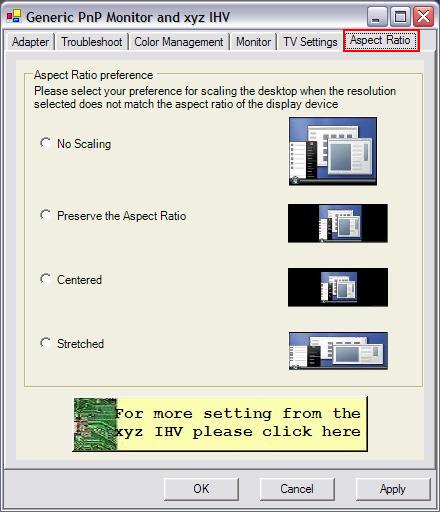
For more information, see “Connecting and Configuring Displays” and “**SetDisplayConfig”** in the WDK.

* When users select an option, present a preview of the display in either of the following ways:

Static radio buttons with previews, as shown in Figure 2-18.

Dynamic drop-down with a preview below it.

* If the computer has a single active monitor, the settings apply to that monitor. If the user has two monitors that are configured in extend mode, the settings apply to the monitor that the user selected before opening the **Advanced Settings** page.
* In Duplicate Desktop mode, use the CCD API to assign the selected setting to each monitor. Windows 7 then applies the appropriate setting, as shown in Figure 2‑18.



**Figure 2-18. Sample Aspect Ratio tab**

## WDDM Driver Optimizations for Display Configuration

Although most of the display configuration improvements are backward compatible with WDDM v1.0 drivers, WDDM v1.1 provides optimizations especially for the following features:

* Monitor scaling mode to preserve aspect ratio.
* Additional display resolutions.
* Synchronization of Windows and third-party Control Panel applications.

### Monitor Scaling Mode to Preserve Aspect Ratio

Widescreen portable computers are common, but typical projectors are still 4:3 mode. Consequently, when the computer is in Duplicate Desktop mode, the aspect ratios of the portable computer screen and the projector screen often do not match, so the system must scale the desktop on one of the display devices.

In Windows 7, a new option stretches the desktop image to the maximum but maintains the aspect ratio. Therefore, the image occupies the maximum screen but is not distorted. This setting is the default.

Windows 7 supports the following modes when a WDDM v1.1 driver is installed:

* Centered
* Stretched
* Aspect ratio-preserving stretched
* No scaling (identity)

The scaling mode setting for each attached display device is stored in the persistence database and reapplied along with all the other display settings. Figures 2-19, 2-20, and 2-21 show the centered, stretched, and aspect ratio-preserving stretched modes, respectively. In each figure, the scaled display is on the left and the original display is on the right.

**Centered mode**: The desktop appears with no scaling at all. Black bands might be visible above and below or on either sides.



Figure 2-19. Centered mode

**Stretched mode**: The desktop is horizontally and vertically stretched to ensure that the entire screen is used. No black bands are visible, but the desktop might look distorted.



Figure 2-20. Stretched mode

**Aspect ratio-preserving stretched mode (new):** The desktop is stretched horizontally and vertically as much as possible while maintaining the aspect ratio. Black bands might appear above and below the desktop or to the left and right, but not both. We expect this to be the most popular user preference, so it is the Windows 7 default.

****

Figure 2-21. Stretching with aspect ratio preservation

### Additional Display Resolutions

In Windows 7, users can choose the display resolution from the Control Panel Display application. However, some display devices cannot communicate their capabilities to the PC. As a result, the desired resolution might not appear in the Display application. Such problems can occur if users connect the PC to a TV, to a monitor without an EDID, or to a switchbox that does not transmit the EDID to the PC.

To address these issues, WDDM v1.1 drivers must always make available a set of common resolutions, based on the type of connector. This set includes widescreen resolutions as well. These resolutions appear under the **Advanced Settings** tab of the Control Panel Displayapplication.

### Synchronization of Control Panel Applications

Windows 7 introduces many infrastructure optimizations to improve the user experience in connecting and configuring display devices, and it fixes problems that existed in Windows Vista. WDDM v1.1 enables drivers to take advantage of these improvements, which results in fewer screen flashes and better integration with the persistence database.

WDDM v1.1 drivers should use the CCD API to query, set, store, and retrieve monitor configurations and to ensure that their settings synchronize with those that Windows uses.

Best Practices for Display Connection and Configuration

* Do not automatically apply display settings in value-added software.
* Use the CCD API and algorithms to automatically configure monitors and to save settings in the persistence database. Do not implement private persistence databases in value-added software or drivers for Windows 7.
* Map the OEM projection shortcut key to display the same UI as the new Windows logo key + P key combination.
* Correctly detect monitors on both analog and digital connections.
* Do not perform analog polling to determine display connectivity.
* Perform interruptible HPDs to provide the best connectivity experience.
* In value-added software, do not duplicate functionality that the Windows 7 Control Panel applications provide. For scenarios that Windows 7 does not cover, extend the Control Panel Displayapplication so that all display settings appear in one place.
* Implement overscan compensation for consumer electronics (CE) display devices in GPU software.

# Display Color Management

Windows 7 introduces two significant new features for display color management:

* Support for high color technology, which is often implemented in high-definition display devices.
* A display calibration wizard, which enables users to visually calibrate any monitor that they can attach to a PC.

This section describes these features.

## High Color

High color is a set of capabilities that enable applications to render content with more than 8 bits per pixel. The graphical image appears in a color space that is larger than the standard red-green-blue (sRGB) gamut, with much higher precision and a higher dynamic range.

Today’s wider gamut display devices support more colors that the human eye can recognize than the sRGB gamut. Such devices can deliver a better image experience by using laser, LED, or even updated fluorescent backlights. Many wide-gamut content sources support more colors than sRGB, including HD-Photo, VC1-FX, and Blu-ray. Protocols to send such content from the PC to the display device include HDMI 1.4 xvYcc and DisplayPort.

Higher bit depths provide higher precision within a particular gamut of color and help to reduce banding artifacts that interfere with imaging, video, medical, and editing applications.

High dynamic range support is provided for the support of larger brightness values.

### High Color in Windows 7

Windows 7 has three pixel formats for high-color capabilities:

* 10-bit per channel sRGB (30-bit) high-precision format.
* 10-bit per channel xRGB (30-bit) extended-range (XR) format.
* 16-bit per channel scRGB (48-bit) high-precision and wide-gamut format.

Table 2-2 compares how these formats and 8-bit per channel sRGB suit the requirements of various applications and devices.

Table 2-2. Comparison of Color Formats

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Use | 8-bit per channel sRGB | High precision (10-bit per channel sRGB) | Extended range (10-bit per channel xRGB) | High precision and wide gamut (16 -bit per channel scRGB) |
| Appl­­i-  cation | Inadequate precision and image quality for professional and high-end consumer applications. | Adequate precision and image quality for medical and technical applications. | Inadequate precision and image quality for professional and high-end consumer applications. | Covers all gamut requirements for foreseeable future media. |
| Display and camera gamut support | Does not cover gamut of current cameras (such as AdobeRGB) or upcoming media formats. Does not cover gamut of current and upcoming display devices, such as monitors and TVs. | Does not cover gamut of current cameras (such as AdobeRGB) or upcoming media formats. Does not cover gamut of current and upcoming display devices, such as monitors and TVs. | Covers gamut of current cameras (such as AdobeRGB) or upcoming media formats. Covers gamut of current and upcoming display devices, such as monitors and TVs. | Covers gamut of foreseeable future cameras and upcoming media formats along with most foreseeable future monitors and TVs. |
| Band-  width estimate | Efficient bandwidth and power con-sumption for por-table computers. | Efficient bandwidth and power con-sumption for por-table computers. | Efficient bandwidth and power con-sumption for por-table computers. | 2X bandwidth and power consumption as other formats. |
| Miscel-  -laneous | Current standard on all PCs, GPUs, and monitors. | Trivial implemen-tation even on most DirectX 9 GPUs. | New implementa-tion. Current GPUs do not support transparent compositing. | Enables high-dynamic range) (HDR) display devices. |

Direct3D 10 lets developers create a 10-bit, 10-bit XR, or 16-bit-per-channel render target, so that they can preserve content that has values beyond the traditional 8-bit formats.

The following DirectX formats support high color formats:

* DXGI\_FORMAT\_R10G10B10A2\_UNORM
* DXGI\_FORMAT\_R10G10B10A2\_UNORM with XR Bias Swap Chain
* DXGI\_FORMAT\_R16G16B16A16\_FLOAT

The support for high color in Direct3D 10 is limited to full-screen mode only.

### Color Management Support for WDDM drivers

Some WDDM drivers require special color management support to implement the following in desktop and scan-out:

* 10-bit XR\_BIAS
* FP16

This section provides implementation guidelines for driver writers.

#### Implementing 10-bit XR\_BIAS Desktop/scan-out

If the GPU supports 10-bit color scan-out, no additional hardware changes are required to support 10-bit XR format in Windows 7.

When a WDDM v1.1 driver enables 10-bit XR desktop mode, the driver must apply the specified bias and scale to the look-up table (LUT) contents. Upon exit from 10‑bit XR desktop mode, the driver must restore the previous unmodified LUT contents. GPUs must enable—not clip—extended gamut and headroom or toeroom values within the range from -0.75 to 1.25.

To verify correct behavior, use an output device that supports extended color range such as HDMI 1.3 xvColor or on any device that supports extended headroom and toeroom.

#### Implementing FP16 Desktop/Scan-Out

WDDM drivers must make FP16 linear, not gamma-corrected.

Unlike integer formats, no gamma correction has been applied to any content that is provided into the frame buffer in FP16 mode. If a driver assumes that FP16 is gamma-corrected, the display—especially a low-end display—looks quite dark. When the driver places the display into a desktop scan-out mode that has a float format, the driver must modify the gamma ramps by adding the difference between sRGB gamma and linear to the ramps. Upon exit from FP16 mode, the driver must remove this bias.

In modes that support extended gamut color values, the GPU must allow—not clip—such values within the range of the color format that is being scanned out. For 10-bit desktop, the range is [0 to 1]. For 10-bit XR the range is [-0.75 to 1.25]. For FP16 scan-out the range is [-4 to 4], as was used in scRGB.

To verify correct behavior of any changes to the display gamma ramp, it is not sufficient to read back the contents of the frame buffer. Instead, use either visual comparison with a reference display device or a digital signal capture device that can record the actual values that are sent out through the display connector.

Table 2-2. Summary of Color Format Requirements

|  |  |  |  |
| --- | --- | --- | --- |
| DirectX version | 10-Bit | 16-Bit | XRBIAS |
| DirectX 9 | Optional | Optional | Optional |
| DirectX 10 | Required | Required | Required |
| DirectX 11 | Required | Required | Required |

## Display Color Calibration Wizard

Graphics design applications in industries such as computer-aided design, automobile design, fashion industry, film-making, and printing rely on accurate color rendition and its translation into the print form. The success of these applications depends on how closely the colors on the screen match those designed in software and, more importantly, the printed outcome on paper or cloth. To aid this difficult task, Windows 7 introduces the Display Color Calibration Wizard.

The Display Color Calibration Wizard provides a fast and simple way to visually calibrate any monitor that can be connected to a PC. In addition to the standard brightness and contrast controls, the Display Color Calibration Wizard helps users adjust the gamma and the white point of the display. It calibrates the display graphics pipeline to target the sRGB color space gamma and white point and saves the calibration state to a color profile.

The Display Color Calibration tool serves a much broader purpose than the Windows Media® Center TV calibration wizard, which uses a series of video clips to optimize the video viewing experience on the user’s television set.

### Display Color Calibration and the VESA Monitor Control Command Set

The long-term direction of the Display Color Calibration wizard is to use the MCCS to calibrate the display. We strongly encourage display device vendors to support MCCS in their display devices and to contact us for further information.

If the display device does not support MCCS, Windows 7 typically asks users to simply use the on-screen menu (OSM) to adjust display controls such as Brightness and Contrast. In the absence of MCCS, Windows 7 adjusts the gamma and color balance by manipulating the display card gamma LUTs according to the user’s settings of the on‑screen slider controls. Portable computers seldom provide contrast adjustment and typically increase and decrease brightness by using a pair of Fn+function key combinations.

### Display Color Calibration Wizard User Interface

The following sections show how users interacts with the Display Color Calibration Wizard.

To start the Display Color Calibration wizard, users navigate to the **Advanced** tab of the Control Panel Color Management application, as shown in Figure 2-22.



Figure 2-22. Starting the Display Color Calibration Wizard

#### Gamma Adjustment

The wizard provides a self-explanatory guide to obtain the best gamma on a display device, as Figure 2-23 shows.

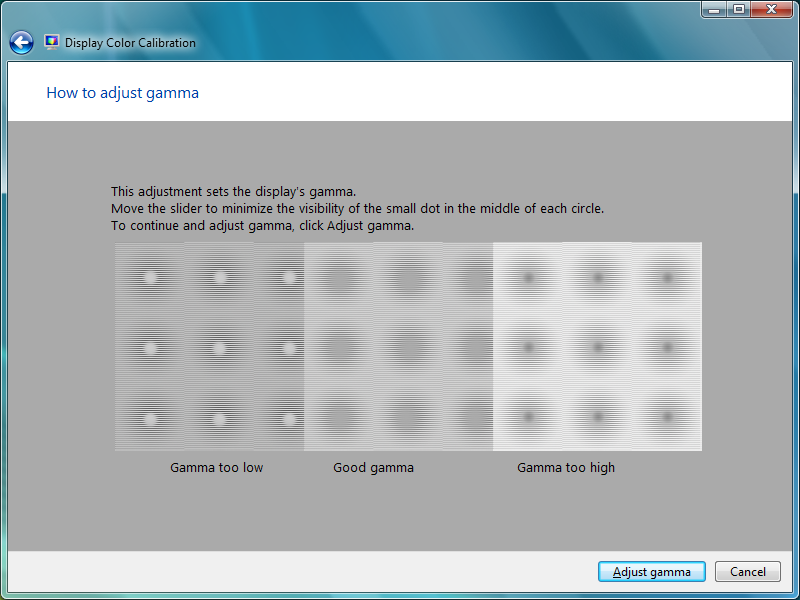


Figure 2-23. Gamma adjustment guide

When users click **Adjust gamma**, the wizard displays a simple slider to control the gamma, as shown in Figure 2-24.

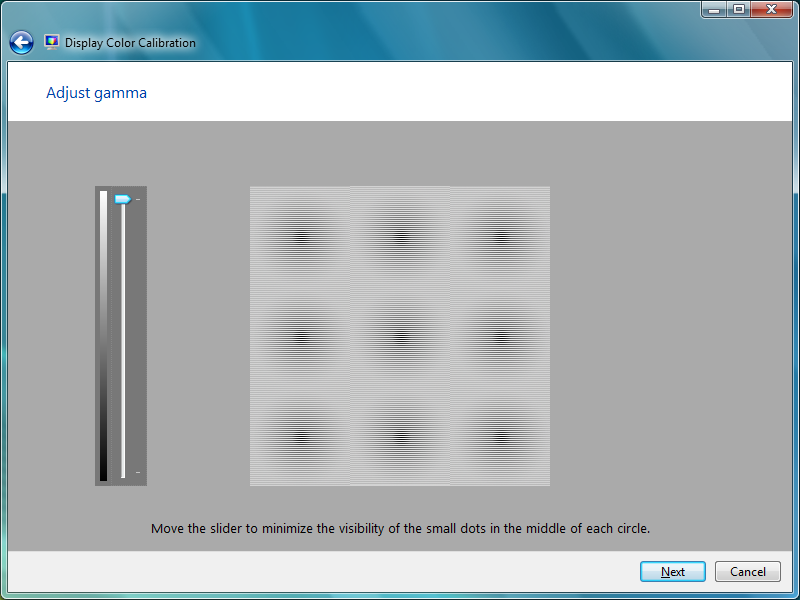


Figure 2-24. Gamma adjustment control

#### Brightness

When users click **Next**, the wizard displays a dialog box that directs them to the monitor’s OSM to adjust brightness, as seen in Figure 2-25.

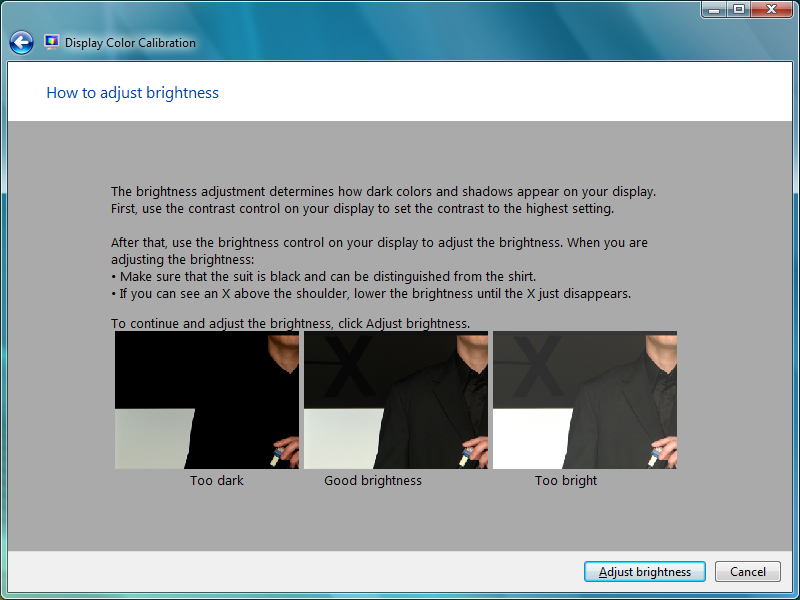


Figure 2-25. Brightness adjustment guide

#### Contrast

Similar to the Brightness adjustment dialog box, the Contrast adjustment dialog box guides users to the monitor’s OSM to adjust the contrast for the image in the screen, as shown in Figure 2-26.

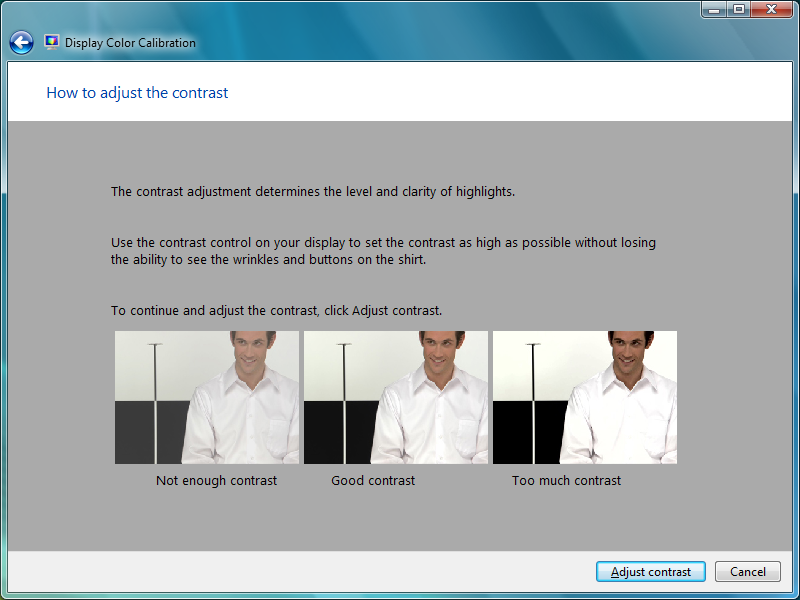


Figure 2-26. Contrast adjustment guide

#### Color Balance

Finally, the Color Balance dialog box contains a simple guide to optimize the balance of red, green, and blue components, as shown in Figure 2-27.

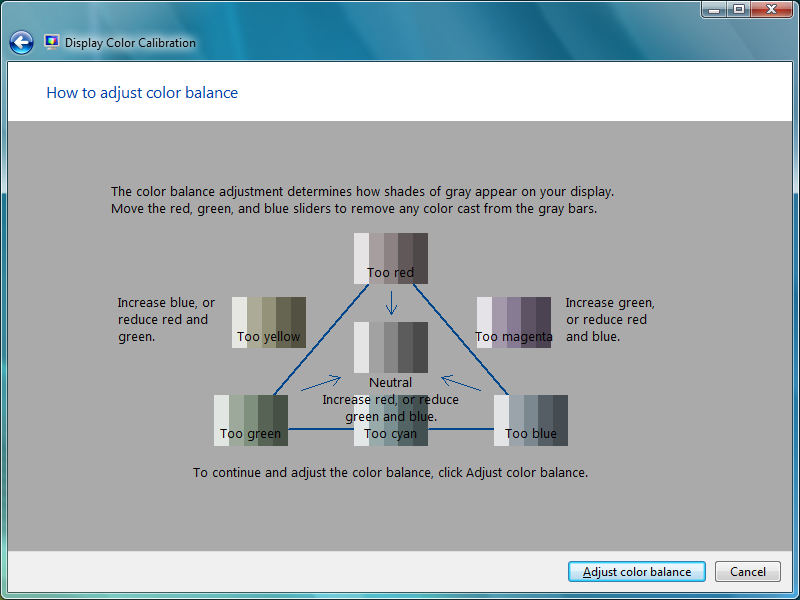


Figure 2-27. Color balance adjustment guide

When users click **Adjust color balance**, the wizard displays a sample image with a slider to adjust the color, as shown in Figure 2-28.

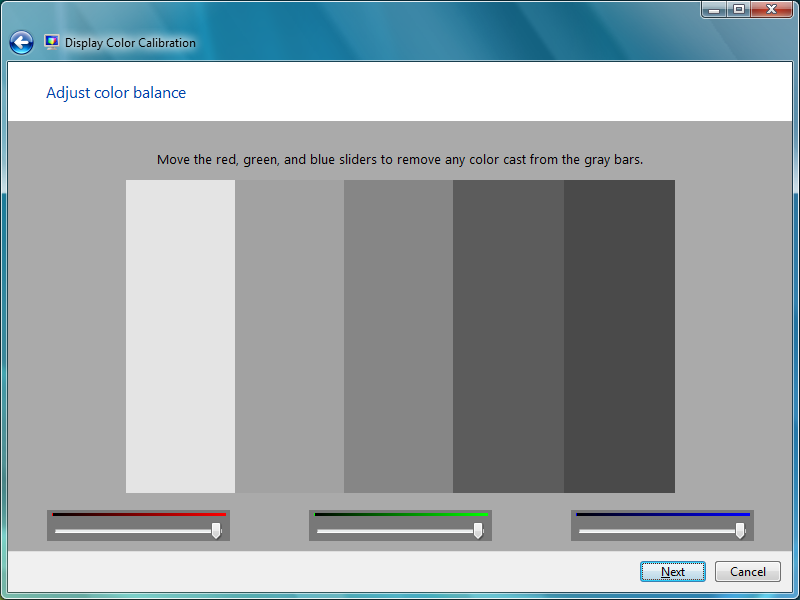


Figure 2-28. Color balance adjustment controls

Best Practices for Display Color Management

* Incorporate display devices that support high-color formats into Windows systems.
* Use the high color technologies in Windows 7 to exploit the full potential of modern display and print hardware.
* Validate display devices for accurate primary colors, white-point, and black-point in the EDID data.
* Test that display LUTs are valid and persist after power transitions such as resume from sleep and hibernation.
* Test value-added display software with LUT loaders that are provided in third-party color calibration tools.
* Use wide color gamut displays to take advantage of the BGRA color format that Windows 7 requires.
* For rich color fidelity, use HDMI v1.3 displays that support xvColor.

# High-DPI and Readability

Several new Windows 7 features improve readability and support high-DPI settings:

* Automatic configuration of the recommended DPI setting for compatible display devices. Manufacturers can override the defaults to provide device-specific settings if required.
* Redesigned Control Panel DPI application that guides users to choose a native display resolution.
* Per-user DPI settings in the persistence database.
* Better display quality at high-DPI settings.
* Support for high-DPI settings in Microsoft Internet Explorer and the Windows shell.

Research shows that about 45 percent of users do not configure their display to the native resolution. Microsoft believes that most of these users select a lower screen resolution because the default text is too small at the native resolution.

Windows 7 addresses these issues by automatically configuring the recommended DPI for applicable machines, while still enabling manufacturers to override this default by setting their own DPI preference. To guide users to choose a native resolution, Windows 7 introduces a more accessible and informative Control Panel DPI application. Users can also change the DPI without rebooting—although logoff and logon are required—and can choose their own DPI setting. Finally, Windows 7 provides greatly improved display quality at high DPI. Internet Explorer is now fully DPI-aware, and all Windows shell programs (Explorer, Taskbar, System Tray, and so on) support up to 144 DPI.

### The Importance of High DPI

Customer data from Windows users reveals that roughly half of users do not configure their PC to use the full native screen resolution. Figure 2-29 shows the percentage of users who set their displays to the maximum resolution. The detail box in the figure shows that for users whose displays can support a resolution of 1600x1200, only 32 percent use that resolution. 3 percent use 1280x1024, 57 percent use only 1024x768, and another 8 percent use even lower resolutions.

Figure 2-29. User display resolution data

Many of the comments we have seen support our hypothesis that a common reason for using a lower resolution is the difficulty in reading default text, which can appear small on high-resolution displays.

Regardless of the reason, the result is larger but blurry text that can significantly increase eye fatigue when users read a PC screen for a long period of time. On LCD displays, much of the blurriness is caused by the hardware’s fixed pixels. At non-native resolution settings, the system must render fractional pixels across fixed units, which causes a blurred effect. Another reason for the relative blurriness is that when the display is not set to native resolution, Windows cannot take advantage of Microsoft ClearType® text rendering, as described in the “ClearType FAQ” on the Microsoft Web site.

The loss of fidelity at non-native screen resolutions is less pronounced on a CRT display than on an LCD display because CRTs do not have fixed pixels. However, because of advantages in cost, size, and energy consumption and the popularity of the portable PC, LCD displays are quickly gaining market share.

Another problem with a non-native screen resolution is that many users inadvertently configure the display to a non-native aspect ratio as well. This results in an image that is both blurry and skewed, which further exacerbates eyestrain.

Non-native screen resolutions can also cause poorly rendered media content. Even with capable hardware, many user configurations prevent them from seeing native high-definition 720p or 1080p TV content, which corresponds to 1280x720 or 1920x1080 screen resolutions, respectively. The PC monitor has traditionally been the high-definition display device, but without addressing this problem the computer industry risks trailing the TV industry in high-definition display. Although only about 10 percent of users have a truly 1080p-capable PC screen today, the installed base is likely to grow as these displays decrease in price. In the future, users will want to take advantage of even higher fidelity content. As an example, at 400 DPI, displays will be almost indistinguishable from looking at printed text on paper. Even the current generation of eBook readers with a DPI of about 170 look very much like a piece of paper behind a piece of glass.

### Windows 7 High-DPI Features

Windows 7 improves on the high-DPI support that in Windows Vista and earlier releases by using the monitor EDID information and enhancing the existing **High-DPI** **Settings** page. The improvements are discussed in the following sections.

#### Change of DPI without Requiring a Reboot

Windows 7 does not require reboot for changes in DPI setting to take effect, although users must log off and log on again. In Windows Vista, changing the DPI setting required a system reboot.

#### Per-User DPI Setting

Windows 7 supports per-user DPI settings so that users who share a PC can have independent DPI settings for a personalized viewing experience. In earlier Windows versions, the DPI setting was per-machine only.

#### Automatic Configuration of DPI

Customer data reveals that many monitors already support high DPI. However, relatively few are configured for the optimal DPI and native screen resolution. Windows 7 automatically configures the native resolution and corresponding DPI setting during Setup, so that users are no longer required to guess at the correct configuration. Windows 7 never configures the DPI to produce an effective resolution lower than 1024x768. Table 2-3 shows how Windows 7 maps the screen size to the DPI.

Table 2-3. Mapping Screen Size to DPI Selection

| Description | Horizon-tal DPI | Vertical DPI | Width  (inches) | Panel  DPI | Operating system DPI | Scale level  (%) |
| --- | --- | --- | --- | --- | --- | --- |
| 17" WXGA+ | 1440 | 900 | 17.0 | 100 | 96 | 100 |
| 15.4" WXGA+ | 1440 | 900 | 15.4 | 110 | 96 | 100 |
| 15.4" WXGA | 1280 | 768 | 15.4 | 97 | 96 | 100 |
| 14.1" WXGA | 1280 | 768 | 14.1 | 106 | 96 | 100 |
| 13.3" WXGA | 1280 | 768 | 13.3 | 112 | 96 | 100 |
| 17" WUXGA | 1920 | 1200 | 17.0 | 133 | 120 | 125 |
| 17" WSXGA+ | 1680 | 1050 | 17.0 | 117 | 120 | 125 |
| 15.4" WSXGA+ | 1680 | 1050 | 15.4 | 129 | 120 | 125 |
| 14.1" WXGA+ | 1440 | 900 | 14.1 | 120 | 96 | 125 |
| 13.3" WXGA+ | 1440 | 900 | 13.3 | 127 | 96 | 125 |
| 12.1" WXGA | 1280 | 768 | 12.1 | 123 | 96 | 125 |
| 15.4" WUXGA | 1920 | 1200 | 15.4 | 147 | 144 | 150 |

The Scale Level column represents the default configuration that is based on the Maximum Horizontal Image Size and Maximum Vertical Image Size parameters in the EDID. Windows 7 can choose the default correctly only if the EDID contains valid data for these parameters. For more information, see the VESA E-EDID Implementation Guide, which is listed in “Display Features and Design Considerations Resources.”

### Overriding the Windows DPI Defaults

System manufacturers can override the Windows automatic configuration of DPI and instead choose their own recommended DPI. Windows 7 provides two ways for manufacturers to set a custom DPI:

* By editing the Unattend.xml file.
* By using the Windows System Image Manager.

To set DPI by using the Unattend.XML setup utility

Provide values for the following elements of the Display setting in the Unattend.xml answer file:

* ColorDepth
* DPI
* HorizontalResolution
* RefreshRate
* VerticalResolution

For example:

<Display>

<ColorDepth>32</ColorDepth>

<DPI>120</DPI>

<HorizontalResolution>1600</HorizontalResolution>

<RefreshRate>60</RefreshRate>

<VerticalResolution>1050</VerticalResolution>

</Display>

This sample code snippet from the Unattend.xml file sets the display resolution to 1600 x 1050, with 32-bit color depth at 120 DPI and a 60‑Hz refresh rate.

To set DPI by using the Windows System Image Maker

Another way to configure the automatic DPI setting is by using the Windows System Image Maker (SIM) to create an Unattend.xml answer file. SIM is part of the Windows Automated Installation Kit (WAIK), which is available for download from the Microsoft Web site. To change settings by using SIM:

1. Start SIM.

2. Open the corresponding Windows image file or catalog file.

3. Select **Display** in the **Windows Image** pane on the lower left and add to the answer file in one of the following passes:

Specialize

AuditSystem

AuditUser

OobeSystem

These sections will appear in the **Answer File** pane in the center of the window.

4. Edit the **DPI** value in the **Display** **Properties** pane on the upper right.

Figure 2-30 shows the SIM UI.

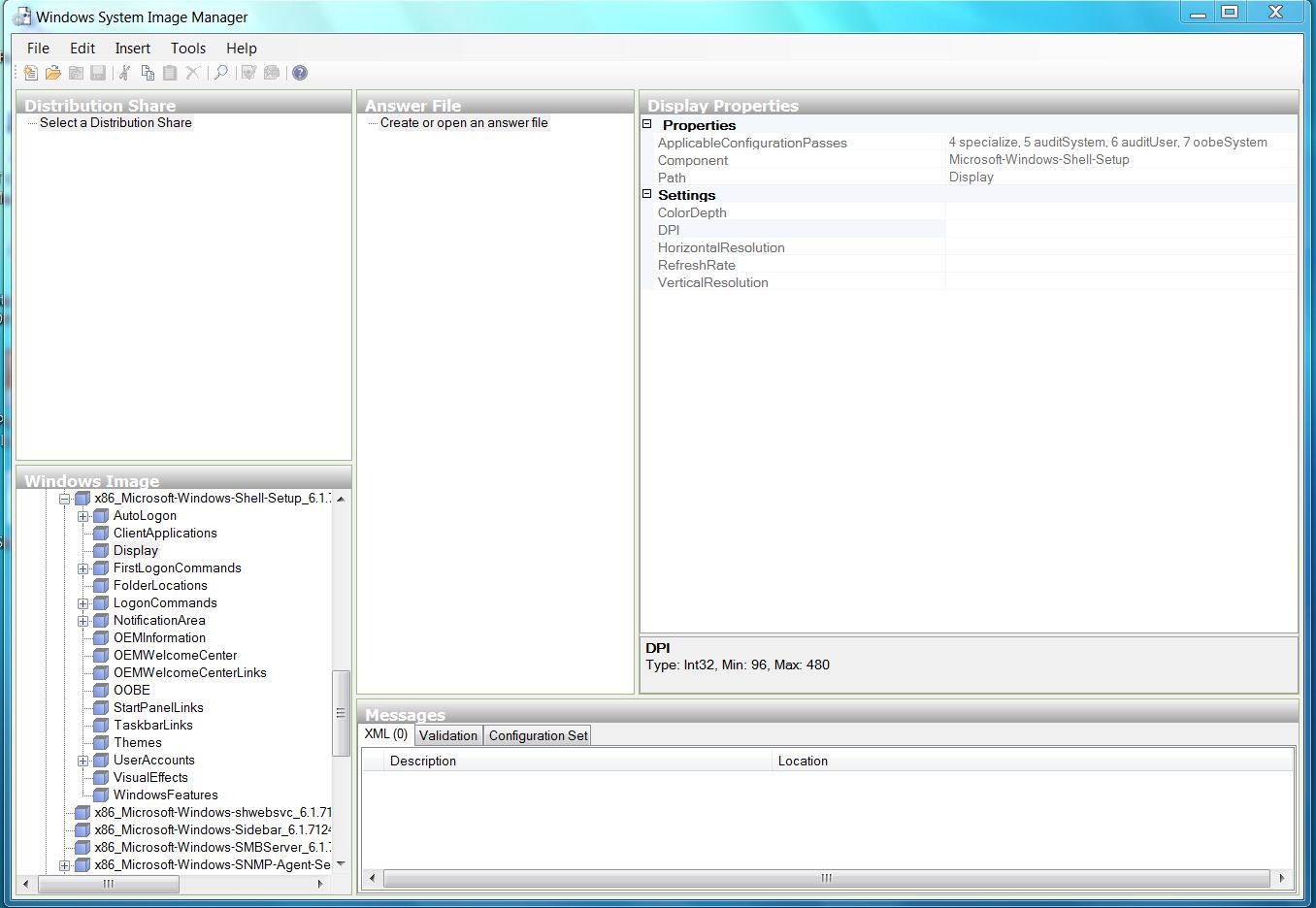


Figure 2-30. Windows System Image Manager UI

For more information, see the “Windows System Image Manager Technical Reference” on the TechNet Web site.

### Testing Systems at High-DPI Settings

For the best display quality and user experience, you should test systems at high-DPI settings to ensure that the following work as expected:

* Manual changes to DPI.
* Automatic DPI configuration.
* DPI override.
* Speed of input devices at high-DPI settings.
* Inbox applications and important Web sites.

Most problems are caused by applications that do not properly handle high-DPI settings, so ensure that such testing allows adequate time to investigate and patch the applications to work correctly.

Table 2-4 shows the test matrix that you should use to test your inbox applications and key Web sites.

Table 2-4. High-DPI Test Matrix

| DPI setting  (Windows Vista/Windows 7) | Minimum  resolution | Recommended  resolution |
| --- | --- | --- |
| 96 / 100% | 800x600 | 1024x768 |
| 120 / 125% | 1024x768 | 1280x960 |
| 144 / 150% | 1200x900 | 1600x1200 |
| 196 / 200% | 1600x1200 | 2500x1600 |

Windows Vista shows DPI settings as absolute dots per inch, and Windows 7 shows the same values as percentages to indicate the scaling factor.

To test manual changes to DPI

This test requires that you change the per-user DPI setting and then ensure that the change affects only a single user and remains in effect after the user logs off and logs on again. The test involves these steps:

1. Create two different users on the system.

2. Navigate to the Control Panel **Display** application and select **Make text and other items larger or smaller**.

3. Select **Medium – 125%** and then click **Apply**.

4. Click the **Start** button and then click the arrow to the right of the **Lock** button.

5. Select **Switch User** from the menu. When the logon screen appears, log on as a different user.

6. Confirm that the DPI setting for this user has not changed.

7. Log off and log on again as the original user. Confirm that the DPI setting is 125%.

To test automatic configuration

1. Configure a system with a monitor that is capable of high DPI and perform a clean installation of Windows.

2. Validate that the DPI is set to the correct default according to Table 2-5.

Table 2-5. Validation Matrix for Default DPI Settings

|  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Monitor size [inch]** | | | **11** | **13** | **15** | **17** | **19** | **20** | **22** | **24** | **30** | **40** |
| **Resolution [W x H]** | | | **DPI** | | | | | | | | | |
| SVGA | 800 | 600 | 96 | 96 | 96 | 96 | 96 | 96 | 96 | 96 | 96 | 96 |
| WSVGA | 1024 | 600 | 96 | 96 | 96 | 96 | 96 | 96 | 96 | 96 | 96 | 96 |
| XGA | 1024 | 768 | 96 | 96 | 96 | 96 | 96 | 96 | 96 | 96 | 96 | 96 |
|  | 1152 | 768 | 96 | 96 | 96 | 96 | 96 | 96 | 96 | 96 | 96 | 96 |
| HD 720 | 1280 | 720 | 96 | 96 | 96 | 96 | 96 | 96 | 96 | 96 | 96 | 96 |
| WXGA | 1280 | 768 | 96 | 96 | 96 | 96 | 96 | 96 | 96 | 96 | 96 | 96 |
| WXGA | 1280 | 800 | 96 | 96 | 96 | 96 | 96 | 96 | 96 | 96 | 96 | 96 |
|  | 1280 | 854 | 96 | 96 | 96 | 96 | 96 | 96 | 96 | 96 | 96 | 96 |
|  | 1280 | 960 | 120 | 120 | 96 | 96 | 96 | 96 | 96 | 96 | 96 | 96 |
| SXGA | 1280 | 1024 | 120 | 120 | 96 | 96 | 96 | 96 | 96 | 96 | 96 | 96 |
| **Monitor size [inch]** | | | **11** | **13** | **15** | **17** | **19** | **20** | **22** | **24** | **30** | **40** |
| **Resolution [W x H]** | | | **DPI** | | | | | | | | | |
| SXGA+ | 1400 | 1050 | 120 | 120 | 120 | 96 | 96 | 96 | 96 | 96 | 96 | 96 |
|  | 1440 | 900 | 96 | 96 | 96 | 96 | 96 | 96 | 96 | 96 | 96 | 96 |
|  | 1440 | 960 | 120 | 120 | 120 | 96 | 96 | 96 | 96 | 96 | 96 | 96 |
| UGA | 1600 | 1200 | 144 | 144 | 120 | 120 | 96 | 96 | 96 | 96 | 96 | 96 |
| WSXGA+ | 1680 | 1050 | 120 | 120 | 120 | 120 | 96 | 96 | 96 | 96 | 96 | 96 |
| HD 1080 | 1920 | 1080 | 120 | 120 | 120 | 120 | 120 | 96 | 96 | 96 | 96 | 96 |
| WUXGA | 1920 | 1200 | 144 | 144 | 144 | 120 | 120 | 96 | 96 | 96 | 96 | 96 |
| 2K | 2048 | 1080 | 120 | 120 | 120 | 120 | 120 | 120 | 96 | 96 | 96 | 96 |
| QXGA | 2048 | 1536 | 144 | 144 | 144 | 144 | 120 | 120 | 120 | 96 | 96 | 96 |
| WQXGA | 2560 | 1600 | 144 | 144 | 144 | 144 | 144 | 144 | 120 | 120 | 96 | 96 |
| QSXGA | 2560 | 2048 | 144 | 144 | 144 | 144 | 144 | 144 | 144 | 120 | 96 | 96 |
| QUXGA | 3200 | 2400 | 144 | 144 | 144 | 144 | 144 | 144 | 144 | 144 | 120 | 96 |
| WQUXGA | 3840 | 2400 | 144 | 144 | 144 | 144 | 144 | 144 | 144 | 144 | 144 | 96 |

To test input devices for correct default speed settings

The sensitivity of the touch pad and eraser head input devices varies on some machines at high-DPI settings. To test this:

* Ensure that both devices operate at the same speed when running a machine on high DPI. Follow the validation matrix for DPI and resolution configurations that appears in Table 2-5.

To test the DPI override

If you set a custom DPI value in response to a user request or to work around a display that has an invalid EDID configuration, follow these steps to check whether your value overrides the defaults:

1. Configure the Unattend.xml file to set the desired DPI for the target machine.

2. Install Windows 7, and then right-click the desktop and select **Display**.

3. Verify that the scale percentage is set to the desired value.

To test inbox applications and key Web sites

Third-party inbox applications, value-added Control Panel applications, and important Web sites—such as your driver and product information sites—must work properly at high DPI. To test this, follow these steps:

1. Right-click the desktop and select **Screen resolution**.

2. Select one of the scenarios, such as **Medium – 125%**, and then click **Apply**.

3. Select **Log off Now**.

4. Run any value-added Control Panel applications and inbox applications and look for incorrect fonts, layout problems, clipped text, and pixilated bitmaps, icons, and artwork.

5. Navigate to the important Web sites and ensure that they operate properly at high-DPI settings. In particular, look for incorrect fonts, layout problems, clipped text, and pixilated bitmaps, icons, and artwork.

Best Practices for High DPI and Readability

* Ensure that software and value-added Control Panel applications are high-DPI compliant.
* Use the DPI Configuration Matrix to test DPI settings.
* Report high-DPI issues to the vendors of third-party software that does not appear properly.
* Ensure that the EDID data is valid. The maximum horizontal size and maximum vertical size must be reported correctly in centimeters. Vendors must measure the panel sizes, not including the frame.
* Ensure that the DPI is configured correctly out of the box.
* Use the auto-configuration feature in Windows 7. Manual configuration is required only if the display hardware does not correctly report the EDID configuration.
* For ease of readability, provide displays that support 120 DPI or greater.
* Ensure that the default DPI setting works best with the portable computer’s internal display or with common external displays.
* Validate the resolution value that the EDID provides and ensure that the DPI set through the auto-configuration in OOBE corresponds to that resolution.
* Test setting your own DPI value by using the unattend.xml file.
* Test input devices for correct default speed settings.
* Validate your key Web sites and third-party software at high-DPI settings.

# Resources for Display Features and Design Considerations

Information on the Microsoft Web site

ClearType FAQ  
<http://www.microsoft.com/typography/ClearTypeFAQ.mspx>

Microsoft OEM Preinstallation Kits  
<http://www.microsoft.com/oem/sblicense/OPK/default.mspx>

How to Use an INF to Override the Monitor EDID  
<http://www.microsoft.com/whdc/device/display/edid_over.mspx>

MSDN

How to Write High-DPI Applications  
<http://msdn.microsoft.com/en-us/library/ms969894.aspx>

Engineering Windows 7 Blog

Engineering Windows 7 Graphics Performance  
<http://blogs.msdn.com/e7/archive/2009/04/25/engineering-windows-7-for-graphics-performance.aspx>

Follow-up on High DPI resolution  
<http://blogs.msdn.com/e7/archive/2008/09/13/follow-up-on-high-dpi-resolution.aspx>

More Follow up to discussion about High DPI  
<http://blogs.msdn.com/e7/archive/2008/09/16/more-follow-up-to-discussion-about-high-dpi.aspx>

Specifications

VESA E-EDID Implementation Guide  
<http://www.vesa.org/Public/EEDIDguideV1.pdf>

Tools and references on TechNet

Windows Automated Installation Kit (Windows AIK)  
<http://technet.microsoft.com/en-us/library/cc748933.aspx>

Windows System Image Manager Technical Reference  
<http://technet.microsoft.com/en-us/library/cc722301.aspx>

Unattended Windows Setup Reference  
<http://technet.microsoft.com/en-us/library/cc722187.aspx>

Windows Vista Deployment Step-by-Step Guide  
<http://technet.microsoft.com/en-us/library/cc721929.aspx>

WDK

Content Protection DDI  
<http://msdn.microsoft.com/en-us/library/dd434674.aspx>

Connecting and Configuring Displays   
<http://msdn.microsoft.com/en-us/library/dd835165.aspx>

Connecting and Configuring Displays (CCD)  
<http://msdn.microsoft.com/en-us/library/dd434691.aspx>

SetDisplayConfig   
[http://msdn.microsoft.com/en-us/library/dd163239.aspx](http://msdn.microsoft.com/en-us/library/dd163239.aspx%20)

#### Questions and Comments about High DPI

If you have additional questions or comments about high dpi, please send e-mail to   
[disup@microsoft.com](mailto:ditsup@microsoft.com)

PART 3:   
Multiple-GPU Systems

**In this part:**

About Multiple-GPU Systems

Linked Display Adapters

Feature Support for Multiple GPU Systems

The end-to-end Windows desktop experience depends not only on individual GPU and display design and connections but also on how the GPUs, firmware, and related hardware components work together. Enthusiasts and power users demand systems that provide increased gaming performance and large desktop real estate to improve productivity and the media experience. Many users set up multimonitor systems, which have two or more display devices that are connected to a single PC, which in turn can have one or more GPUs.

# About Multiple GPU Systems

Windows 7 supports multiple GPUs that run simultaneously. Multi-GPU support falls into two broad categories:

* Homogeneous multi-adapter.
* Heterogeneous multi-adapter.

In a homogeneous multi-adapter configuration, the PC has more than one graphics adapter but all adapters use the same graphics driver. Here are two examples:

* Two identical cards from the same graphics hardware vendor, such as two PCI‑Express (PCIe) ATI Radeon x600 cards, each in an x16 PCIe slot.
* Two different cards from the same graphics hardware vendor, such as a PCIe NVIDIA GeForce 7600 in an x16 slot and a PCIe NVIDIA GeForce 6600 in a second x16 slot.

The bus type—PCIe, Accelerated Graphics Port (AGP), or PCI—is irrelevant. The graphics adapters can all be on the same bus or they can be on different buses. The key point is that all graphics adapters use the same graphics driver.

In a heterogeneous multi-adapter configuration, the PC has more than one graphics adapter and uses more than one graphics drivers. A common example is the use of graphics adapters from two different manufacturers, each of which requires a different graphics driver from the respective manufacturer.

Windows 7 supports heterogeneous multi-adapter configurations, whereas Windows Vista did not. In Windows 7, a system can have a heterogeneous multi-adapter configuration, with multiple GPUs that require different WDDM drivers. The WDDM model for Windows Vista required that all display adapters use the same driver.

# Linked Display Adapters

A linked display adapter (LDA) is a mode of operation in which multiple physical adapters appear merged into a single logical adapter. From an application’s point of view, only the single logical adapter is visible. However, the underlying WDDM driver uses all the physical adapters in the link to improve the performance of the logical adapter. The logical adapter typically offers much higher performance than any individual physical adapters can provide.

The driver can link physical adapters that have identical capabilities and identical memory configurations. Every physical adapter that is part of a logical adapter must be visible on the PCIe bus as an independent device. WDDM assumes that all nodes in a given physical adapter have uniform access to the physical adapter’s memory.

LDAs are supported only on the PCIe bus. LDA configuration is not supported on AGP systems.

# Feature Support for Multi-GPU Systems

Windows 7 supports nearly all WDDM v1.1 graphics features on multi-GPU systems as on single-GPU systems. However, Windows 7 disables the new system memory savings for the Aero Glass theme if two or more active graphics adapters are present.

When a system supports multiple monitors by using multiple GPUs that are not linked, Windows disables device bitmaps if the desktop extends across more than one of the monitors. Because system memory savings for Aero Glass depend on the device bitmaps, this feature is not available on such systems.

PART 4:   
Windows Logo Program Requirements

#### In this part:

Introduction

Mandatory WDDM Version 1.1 Requirements

Optional WDDM Version 1.1 Requirements

Best Practices for Windows Logo Requirements

Resources for Windows Logo Program Requirements

# Introduction

This section provides information about the WLP requirements for WDDM v1.1 drivers. Some of these requirements are mandatory, and others are optional. For details about WLP and WDDM v1.1 requirements, see the Windows Logo Program Web site.

WLP requires the following DirectX and WDDM versions for graphics adapters on Windows 7 systems:

###### For ultramobile PC (UMPC) systems

* DirectX 9 and WDDM v1.0 at minimum.

###### For client systems

* Until December 1, 2009: DirectX 9 and WDDM v1.0 at minimum. If the adapter supports DirectX 10, the driver must support WDDM v1.1.
* Effective December 1, 2009: DirectX 10 and WDDM v1.1 at minimum.

###### For server systems

* XDDM at minimum until June 2011.

Table 4-1 lists the WDDM v1.1 features that are relevant for the logo program, the corresponding logo program requirement in this document, and whether the feature is mandatory or optional.

Table 4-1. Mandatory and Optional WDDM v1.1 Features

|  |  |  |
| --- | --- | --- |
| WDDM v1.1 feature description | Logo program requirement | Mandatory or optional? |
| System memory savings for running Aero Glass | GDI hardware acceleration | Mandatory |
| DirectX 10 color format support | Extended color format | Mandatory |
| Direct2D API as a Direct3D 10 client | Extended color format | Mandatory |
| A better viewing experience on TVs and widescreen portable computers | Connecting and configuring displays | Mandatory |
| Improved diagnosability of TDR stability problems | Logging VSync information | Mandatory |
| Improved diagnosability of performance problems | Compiler options | Mandatory |
| Method of encryption to perform for premium content over user-accessible bus (UAB) | Encryption | Optional |
| Standardized mechanism for high-definition composition by using DXVA-HD | DXVA-HD DDI support | Optional |

# Mandatory WDDM v1.1 Requirements

To be certified as a WDDM v1.1 driver, a Windows 7 display driver must support the features that are described in the following subsections.

## GDI Hardware Acceleration

A WDDM v1.1 driver must report to Windows that it supports GDI hardware acceleration.

The WDDM v1.1 driver must report support for kernel-mode command buffers by setting DXGK\_DRIVERCAPS::**PresentationCaps.SupportKernelModeCommandBuffer** to TRUE.

A driver must report the support for GDI acceleration only if the cache-coherent GPU aperture segment exists and there is no significant performance penalty when the CPU accesses the memory.

The WDDM v1.1 driver must speed up the following 2‑D operations:

* BitBlt
* ColorFill
* StretchBlt
* AlphaBlend
* Cleartype font support
* TransparentBlt

The WDDM v1.1 driver must support the following ROPs:

* ROP support for the BitBlt operation:

DXGK\_GDIROP\_SRCCOPY = 1, (Dst = Src)

DXGK\_GDIROP\_SRCINVERT = 2, (Dst = Dst ^ Src)

DXGK\_GDIROP\_SRCAND = 3, (Dst = Dst & Src)

DXGK\_GDIROP\_SRCOR = 4, (Dst = Dst | Src)

DXGK\_GDIROP\_ROP3 = 5, (The high word contains the GDI Rop3 code)

* ROP support for the ColorFill operation:

DXGK\_GDIROPCF\_PATCOPY = 1, (Dst = Color)

DXGK\_GDIROPCF\_PATINVERT = 2, (Dst = Dst ^ Color)

DXGK\_GDIROPCF\_PDXN = 3, (Dst = ~(Color ^ Dst) )

DXGK\_GDIROPCF\_DSTINVERT = 4, (Dst = ~Dst)

DXGK\_GDIROPCF\_PATAND = 5, (Dst = Dst & Color)

DXGK\_GDIROPCF\_PATOR = 6, (Dst = Dst | Color)

DXGK\_GDIROPCF\_ROP3 = 7, (The high word contains the GDI Rop3 code.)

Other related requirements:

* Support for texture size of 8K x 8k
* Support for linear heaps (**PresentationCaps.AlignedShift** > 0)

For details about how to implement this feature in WDDM v1.1 drivers, see “GDI Hardware Acceleration” in the WDK.

## Extended Color Format Support

Logo program requirements apply to the following aspects of extended color format:

* BGRA color format
* 10-bit XR\_BIAS desktop/scan-out
* FP16 desktop/scan-out

### BGRA Color Format

WDDM v1.1 drivers for Windows 7 must support BGRA color format. Both the Desktop Windows Manager and Direct2D require BGRA format. The BGRA color format was not required as part of the Direct3D 10 specification:

* WDDM v1.1 drivers must support blue-green-red (BGR) texturing and rendering.
* WDDM v1.1 drivers must support back buffer casting.
* WDDM v1.1 drivers can also support high color.

### 10-bit XR\_BIAS Desktop/Scan-out

If the hardware supports 10-bit color scan-out, hardware changes are not required to support 10-bit XR format in Windows 7.

When a WDDM v1.1 driver enables 10-bit XR desktop mode, the driver must apply the specified bias and scale to the LUT contents. Upon exit from XR desktop mode, the driver must restore the previous unmodified LUT contents.

GPUs are required to allow—not clip—extended gamut and headroom and toeroom values within the range [-0.75 to 1.25]. This can be verified on an output device that supports extended color range, such as HDMI 1.3 xvColor, or on any device that supports extended headroom and toeroom.

### FP16 Desktop/Scan-out

WDDM drivers must make FP16 linear, not gamma-corrected.

Unlike integer formats, no gamma correction has been applied to any content that is provided into the frame buffer in FP16 mode. If a driver assumes that FP16 is gamma-corrected, the display—especially a low-end display—looks quite dark. When the driver puts the display into a desktop scan-out mode that has a float format, the driver must modify the gamma ramps by adding the difference between sRGB gamma and linear to the ramps. Upon exit from FP16 mode, the driver must remove this bias.

In modes that support extended gamut color values, the GPU must allow—not clip—extended such values within the range of the color format that is being scanned out. For 10-bit desktop, the range is [0 to 1]. For 10-bit XR the range is [-0.75 to 1.25]. For FP16 scan-out the range is [-4 to 4], as was used in scRGB.

To verify correct behavior of any changes to the display gamma ramp, it is not sufficient to read back the contents of the frame buffer. Instead, use either visual comparison with a reference display device or a digital signal capture device that can record the actual values that are sent out through the display connector.

## Connecting and Configuring Displays

To achieve the design goals of the CCD feature, the WDDM driver must comply with the requirements in this section.

The driver package must not override the Windows 7 default display settings for a new configuration or the saved display settings from the persistence database:

* Windows automatically applies display settings by using its built-in algorithm or by retrieving settings from the persistence database. The driver must not automatically override such settings with its own settings. “Display settings” refers to the topology of sources and targets, resolution, refresh rate, bits per pixel, and aspect ratio.
* The driver must not persist the same settings that Windows saves in the system persistence database and reapplies.
* The driver package must not try to read or write the persistence database by directly accessing the system registry.

If the graphics adapter exposes analog targets, the driver must support forced projection on it as follows:

* The driver must be able to display the desktop on an analog target if the operating system has requested that the driver not check the connectivity status of the target.
* This requirement applies to VGA (including DVI-A), S-Video, Composite, and Component targets.

If possible, the driver should also support the setting of the ITC bit for a display device that is connected by using an HDMI connector and provide a user controllable way to set/reset this bit.

To modify the mode list, the driver must not request a display mode reenumeration.

The driver can request reenumeration of display modes only in the following situations:

* A DisplayPort target requires link training.
* The user requests the addition of a custom mode that has not previously been enumerated.
* The user requests a change to the TV broadcast format, such as from PAL to SECAM.

The driver must accurately report the target and the connectivity state to the operating system, based on the currently connected display devices:

* The operating system must be able to drive any target from any source, although the driver can constrain which targets can be driven in combination.
* If an integrated display device does not contain its own EDID, the driver must create one based on information from sources such as the system Video BIOS. The created EDID must be complete according to the requirements for EDIDs for external monitors with the exception that no serial number is required.
* For each DVI-I connector on the system, the driver must expose one target of D3DKMDT\_VOT\_DVI type and one target of D3DKMDT\_VOT\_HD15 type.
* The driver must not use D3DKMDT\_VOT\_OTHER as the target type for any target type that is already defined in D3DKMDT\_VIDEO\_OUTPUT\_TECHNOLOGY.
* The driver must respond accurately to a request for the connection state of a target. The response must reflect the state of the display device that is physically connected. If the driver does not have this information, it must return STATUS\_GRAPHICS\_UNKNOWN\_CHILD\_STATUS from the *DxgkDdiQueryChildStatus* function.
* The driver must not enumerate more than one target of the D3DKMDT\_VOT\_INTERNAL type on any adapter.
* The driver must not enumerate a display device as “Connected” to more than one adapter at the same time.

The driver must exactly apply the scaling that the operating system requests as defined in D3DKMDT\_VIDPN\_PRESENT\_PATH\_SCALING.

When Windows calls the driver’s *DxgkDdiCommitVidPn* function, it specifies the scaling for each path by using one of the enumeration values that is defined in D3DKMDT\_VIDPN\_PRESENT\_PATH\_SCALING. For each possible value, the driver must apply the scaling exactly as follows:

* D3DKMDT\_VPPS\_UNINITIALIZED

This flag is never passed to a WDDM v1.1 driver.

* D3DKMDT\_VPPS\_IDENTITY

The source mode and target mode are exactly equal. The driver must not perform any scaling.

* D3DKMDT\_VPPS\_CENTERED

The source mode is vertically and horizontally no greater than the target mode and is smaller than the target mode either vertically, horizontally, or both. The contents of the source must be centered on the target, as shown in Figure 4-1A. The driver must not perform any scaling. The driver must use black pixels to fill the region of the target that does not contain desktop data.

* D3DKMDT\_VPPS\_STRETCHED

The source mode is not the same size as the target mode either vertically, horizontally, or both. The target can be smaller than the source in either dimension only if the smaller target dimension is less than 768 pixels. The driver must scale the contents of the source to fit the entire target surface. The driver must not try to preserve the aspect ratio of the source mode.

Figure 4-1B shows the effects of scaling the source to smaller and larger targets.

* D3DKMDT\_VPPS\_ASPECTRATIOCENTEREDMAX

The source mode is not the same size as the target mode either vertically, horizontally, or both. The target can be smaller than the source in either dimension only if the smaller target dimension is less than 768 pixels. The driver must scale the contents of the source until either the horizontal or vertical source resolution is equal to the horizontal or vertical target resolution. The driver must maintain the aspect ratio of the source mode during the scaling. The driver must use black pixels to fill the region of the target that does not contain any desktop data.

Figure 4-1C shows the effects of scaling the source to smaller and larger targets.

* D3DKMDT\_VPPS\_CUSTOM

The driver can apply any scaling based on its own settings. However, the scaling that the driver applies must not result in a scaling mode that is already defined by the preceding four items.

* D3DKMDT\_VPPS\_UNPINNED

This value is used in the same way as in Windows Vista.

* D3DKMDT\_VPPS\_NOTSPECIFIED

This flag is never passed to a WDDM v1.1 driver.

The driver must support the new display mode manager (DMM) functions together with the new scenarios in which Windows calls the existing DMM functions. In addition, IHVs must support the new CCDs functions, which provide many different improvements to connecting and configuring monitors, projectors, and other display devices.



Figure 4-1. Centered, stretched, and aspect ratio-preserving centered scaling

### Support for Additional Target Mode Set on Each VidPN Target

*DxgkDdiGetAdditionalMonitorModeSet* provides a dynamic and scalable way to add target modes on video present network (VidPn) targets. The driver queries Windows for a list of additional modes to support on a per-connector-type basis. The driver then validates the modes and can veto support for any mode. The extra modes are exposed only if users choose **Show all modes** from the Control Panel Display application.

The driver must use the following interface to request a list of additional target modes to add:

// DXGK\_MONITOR\_INTERFACE\_VERSION\_V2 supported functions

DXGKDDI\_MONITOR\_GETADDITIONALMONITORMODESET

pfnGetAdditionalMonitorModeSet;

DXGKDDI\_MONITOR\_RELEASEADDITIONALMONITORMODESET

pfnReleaseAdditionalMonitorModeSet;

### Support for Aspect Ratio Scaling

A WDDM v1.1 driver must support the new aspect ratio-preserving mode for scaling in addition to the scaling modes that were required for Windows Vista. The following definitions show all the modes that a driver must support:

#define D3DKMDT\_SCALING\_SUPPORT\_MASK 0xf;

// Purpose: Specifies the scaling modes that are supported given

// by the current path configuration.

typedef struct \_D3DKMDT\_VIDPN\_PRESENT\_PATH\_SCALING\_SUPPORT

{

UINT Identity : 1;

UINT Centered : 1;

UINT Stretched : 1;

UINT AspectRatioCenteredMax : 1;

} D3DKMDT\_VIDPN\_PRESENT\_PATH\_SCALING\_SUPPORT;

// Purpose: VidPN present path scaling type.

typedef enum \_D3DKMDT\_VIDPN\_PRESENT\_PATH\_SCALING

{

D3DKMDT\_VPPS\_UNINITIALIZED = 0,

// For the following transformation, spatial resolutions must be

// equal on both the source and the target.

// Source content is not modified in any way.

D3DKMDT\_VPPS\_IDENTITY = 1,

// For the following three transformations, spatial resolution of

// the source differs from that of the target.

// Source content is centered on the target.

D3DKMDT\_VPPS\_CENTERED = 2,

// Source content is scaled to fit the target. No aspect ratio

// preserving.

D3DKMDT\_VPPS\_STRETCHED = 3,

// Source content is scaled to fit the target. Aspect ratio

// preserving.

D3DKMDT\_VPPS\_ASPECTRATIOCENTEREDMAX = 4,

// Source content scaling mode is not pinned.

D3DKMDT\_VPPS\_UNPINNED = 254,

// OS does not specify the scaling mode. Miniport should decide

// based on its own settings.

D3DKMDT\_VPPS\_NOTSPECIFIED = 255

}

D3DKMDT\_VIDPN\_PRESENT\_PATH\_SCALING;

For modes greater than or equal to 1024 x 768 or the rotated version of 768 x 1024:

* If the driver supports D3DKMDT\_VPPS\_STRETCHED, it must also support D3DKMDT\_VPPS\_ASPECTRATIOCENTEREDMAX and vice versa.

### Changes in How the DxgkRecommendVidPnTopology Function Is Called

In Windows 7, the CCD logic determines the correct topology, based on the user’s expectations. Therefore, Windows no longer calls the driver’s ***DxgkRecommendVidPnTopology* function** to determine the topology. This behavior differs from that of Windows Vista, where Windows asks the driver to recommend a topology to apply.

Table 4-2 summarizes the conditions under which Windows calls this function for each possible value of DXGK\_RECOMMENDVIDPNTOPOLOGY\_REASON.

Table 4-2. ***DxgkRecommendVidPnTopology*** Calls on Windows 7

|  |  |  |
| --- | --- | --- |
| Reason | WDDM v1 driver | WDDM v1.1d driver |
| DXGK\_RVT\_INITIALIZATION \_NOLKG | Never called because the CCD persistency database replaces the last known good (LKG) value. | Never called because the CCD persistency database replaces the LKG. |
| DXGK\_RVT\_INITIALIZATION \_LKGOVERRIDE | Never called because the CCD persistency database replaces the LKG. | Never called because the CCD persistency database replaces the LKG. |
| DXGK\_RVT\_AUGMENTATION \_NOLKG | Never called because the CCD persistency database replaces the LKG. | Never called because the CCD persistency database replaces the LKG. |
| DXGK\_RVT\_AUGMENTATION \_LKGOVERRIDE | Called as in Windows Vista by legacy GDI functions.  Not called by new Windows 7 APIs. | Never called because the CCD persistency database replaces the LKG. |

### Changes in How Windows Handles ACPI Keyboard Shortcuts

In Windows 7, OEMs must use the persistence database to apply display settings so that the operating system and any OEM Control Panel applications are synchronized. To achieve this, the driver requests a particular topology and Windows applies the saved display settings for the requested topology.

WDDM v1 and WDDM v1.1 drivers must expect the following behavior from Windows 7:

* DXGK\_ACPI\_CHANGE\_DISPLAY\_MODE

If the IHV’s Advanced Configuration and Power Interface (ACPI) notification callback returns DXGK\_ACPI\_CHANGE\_DISPLAY\_MODE, the DMM calls the driver’s *DxgkDdiRecommendFunctionalVidPn* to get the VidPn and compares it to the current client VidPn. If the two VidPns have the same topology, the DMM sets the new VidPn without modifying it.

If the topologies are different, DMM strips the mode information from the VidPn and leaves just the topology. The persistence database determines the modes for the specific topology. The DMM then sets the resulting display configuration.

This behavior applies to both WDDM v1 and WDDM v1.1 drivers.

* D3DKMTInvalidateActiveVidPn

For WDDM v1 drivers, this callback is supported in Windows 7 with the same behavior as in Windows Vista.

For WDDM v1.1 drivers, this callback is not supported. Windows 7 returns STATUS\_NOT\_SUPPORTED without checking with the driver.

### Required Driver Function for Hardware Capabilities

WDDM v1.1 drivers must support the new *DxgkDdiQueryVidPnHwCapability* function, which returns information about the capabilities of the display device. This function has the following prototype:

typedef struct \_D3DKMDT\_VIDPN\_HW\_CAPABILITY

{

    UINT DriverRotation             : 1;

    UINT DriverScaling              : 1;

    UINT DriverCloning              : 1;

    UINT DriverColorConvert         : 1;

    UINT DriverLinkedAdapaterOutput : 1;

    UINT DriverRemoteDisplay        : 1;

    UINT Reserved                   : 26;

}

D3DKMDT\_VIDPN\_HW\_CAPABILITY;

typedef struct \_DXGKARG\_QUERYVIDPNHWCAPABILITY

{

    IN   D3DKMDT\_HVIDPN                        hFunctionalVidPn;

    IN   D3DDDI\_VIDEO\_PRESENT\_TARGET\_ID        hTargetId;

    OUT  D3DKMDT\_VIDPN\_HW\_CAPABILITY           VidPnHWCaps;

}

DXGKARG\_QUERYVIDPNHWCAPABILITY;

typedef \_\_inout DXGKARG\_QUERYVIDPNHWCAPABILITY\*

INOUT\_PDXGKARG\_QUERYVIDPNHWCAPABILITY;

NTSTATUS

APIENTRY

DXGKDDI\_QUERYVIDPNHWCAPABILITY(

    IN\_CONST\_HANDLE                             i\_hAdapter,

    INOUT\_PDXGKARG\_QUERYVIDPNHWCAPABILITY       io\_pVidPnHWCaps

    );

## Logging VSync Information

WDDM v1.1 drivers must report certain information about VSync in the TDR debug data. For each VidPn target, the WDDM v1.1 driver must log the following:

* Whether the VidPn target is enabled or disabled.
* Resolution and refresh rate of the VidPn target.
* Whether VSync interrupts are enabled or disabled on that target.
* The physical address that is currently being scanned out.
* Whether a flip is pending, and the address of the pending flip if one exists.
* The SetVidPnSourceAddress most recently received from the WDDM scheduler.
* The notification most recently reported to the WDDM scheduler.
* Register dump for the display.

## Compiler Options

WDDM v1.1 kernel-mode drivers must be built with the FPO disabled. This requirement does not apply to a user-mode driver binary.

# Optional WDDM v1.1 Requirements

If a WDDM driver implements the following features, the implementation must follow the guidelines in the relevant specifications.

## Encryption

If the driver supports encryption, the following guidelines apply:

* The driver should support AES 128 encryption by specifying D3DCRYPTOTYPE  
  \_AES128\_CTR for compressed content sent to the GPU over the UAB.
* To negotiate the session key for the encryption operation, the driver should use a RSAES-OAEP key exchange by specifying D3DKEYEXCHANGE\_RSAES\_OAEP.

## GPU-based Content Protection DDI

The GPU-based content protection DDI enables Windows Media Center and third-party video players in windowed mode with DWM on. The feature support scales from driver/software to driver/hardware protections.

This DDI consists of support for the following features:

* Encryption BLT—Video-to-system BLT with encryption.
* Driver protections of shared surfaces.
* Encryption on eviction—Encryption of video memory resources when evicted from video memory.
* Decryption BLT—System-to-video BLT with decryption.

For more details about these features, see “GPU-based Content Protection” on MSDN and “Content Protection DDI” in the WDK.

## DXVA-HD Support

These feature requirements provide video pipeline enhancements that enable HD video processing and composition. A driver that implements DXVA-HD exposes its support by setting the D3DCAPS3\_DXVAHD bit in the D3DCaps9.Caps3 field. A WDDM v1.1 driver that supports DXVA-HD must support the input and output formats in Table 4-3.

Table 4-3. Required DXVA-HD Input and Output Formats

|  |  |
| --- | --- |
| Input format | Output format |
| D3DDDICAPS\_DXVAHD\_GETVPINPUTFORMATS | D3DDDICAPS\_DXVAHD\_GETVPOUTPUTFORMATS |
| YUY2 AYUVX8R8G8B8 A8R8G8B8  All decode render target formats supported in DXVA Decode DDI. | X8R8G8B8 A8R8G8B8 |

## Overlay DDI

Support of the overlay DDI is optional for WDDM v1.1 drivers. This DDI enables Direct3D 9-based applications to use video overlays while Aero Glass is running—that is, when DWM is turned on. This feature improves the security and the performance of video overlay presentation by eliminating per-frame composition.

The video overlay presentation requirements are as follows:

* A WDDM v1.1 driver must set the D3DCAPS\_OVERLAY bit in the **Caps** member of the D3DCAPS9 structure.
* A WDDM v1.1 driver must support the D3DDDICAPS\_CHECKOVERLAYSUPPORT query type in the user-mode *pfnGetCaps* DDI call.
* A WDDM v1.1 driver must support overlays in at least one valid configuration (Displaymode, OverlayFormat, Width, and Height) in the DDICHECKOVERLAYSUPPORTDATA function for the supported overlay. The maximum width and height of the supported overlay must be greater than zero.

Best Practices for Windows Logo Requirements

System manufacturers:

* For the best Windows user experience, build systems that have DirectX 10 or later graphics adapters and are WDDM v1.1 capable.
* Work with graphics hardware vendors to get certified WDDM v1.1 drivers for validation.
* Refer to the Windows 7 logo requirements kit for validation and testing information.
* Test a variety of hardware configurations on both desktops and mobile systems to ensure a solid end-user experience in Windows 7.

Graphics hardware vendors:

* Work with Microsoft to develop Windows 7 WDDM v1.1 drivers.
* Test prerelease Windows 7 WDDM v1.1 drivers on the prerelease versions of Windows 7.
* Provide updated WDDM v1.x drivers to Microsoft for deployment through Windows Update.
* In addition to the Windows certification test suite, validate graphics and gaming performance, application compatibility, and various self-host scenarios on each ASIC family.
* Test Windows Vista-certified WDDM v1 drivers on prerelease versions of Windows 7.
* Make the full retail package for WDDM v1.1 drivers available as early as possible.

Independent software vendors (ISVs):

* Test existing and upcoming DirectX games with WDDM v1.1 drivers on prerelease versions of Windows 7.
* Test individual applications on prerelease versions of Windows 7.
* Start to develop Direct2D applications that target Windows.

# Resources for Windows Logo Program Requirements

#### WHDC Web site

How to Use an INF to Override the Monitor EDID  
<http://www.microsoft.com/whdc/device/display/edid_over.mspx>

Display and Graphics  
<http://www.microsoft.com/whdc/device/display/default.mspx>

Windows Logo Program  
<http://www.microsoft.com/whdc/winlogo/default.mspx>

#### Specifications

VESA Enhanced Extended Display Identification Data - Implementation Guide

<http://www.vesa.org/Public/EEDIDguideV1.pdf>

#### Windows Driver Kit

GDI Hardware Acceleration

<http://msdn.microsoft.com/en-us/library/dd434692.aspx>

Content Protection DDI

<http://msdn.microsoft.com/en-us/library/dd434674.aspx>

#### MSDN

DirectX Developer Center  
<http://msdn.microsoft.com/en-us/directx/default.aspx>

GPU-Based Content Protection  
<http://msdn.microsoft.com/en-us/library/dd318792(VS.85).aspx>

#### Engineering Windows 7 Blog

Follow-up on High DPI resolution  
<http://blogs.msdn.com/e7/archive/2008/09/13/follow-up-on-high-dpi-resolution.aspx>

More Follow up to discussion about High DPI  
<http://blogs.msdn.com/e7/archive/2008/09/16/more-follow-up-to-discussion-about-high-dpi.aspx>

E-mail Aliases

If you have questions about high DPI, send e-mail to [disup@microsoft.com](mailto:ditsup@microsoft.com)

If you have other questions that are not addressed by this document, please send e-mail to [directx@microsoft.com](mailto:directx@microsoft.com).

Glossary

ARIB

A Japanese broadcast agreement similar to OCUR, but worded more strictly. It requires that uncompressed data that cross the bus are encrypted with the strength of DES56 or higher.

BD-J

An interactive layer written in Java that is used by Blu-ray Disc.

BGRA

Color format required by the DWM to run on Direct3D drivers.

BML

Broadcast XML used in ISDB-T.

CDD

Canonical Display Driver. A standard component that implements GDI driver entry points, delivers GDI content to the primary surface, and provides DX/GDI interoperability when DWM is off.

Direct2D

Microsoft’s 2‑D API that uses the rendering power of the GPU through DirectX graphics.

DWM

Desktop Window Manager. The Windows component that presents the contents of the desktop.

DxgKrnl.sys

The Windows kernel portion of the WDDM graphics stack. It resides in the kernel and provides a variety of services, including communications with the kernel-mode WDDM video driver.

DXVA

DirectX Video Acceleration.

Direct3D 10 Level 9 (also called Feature Level 9)

The Microsoft Direct3D 10 driver that uses the DirectX 10 API on DirectX 9 hardware.

HDMV

The application layer that Blu-ray Disc uses.

ISDB-T

HDTV broadcast standard in Japan.

RGBA

The color format used by DWM in Windows Vista.

OCUR

Open Cable Unconditional Receiver. A cable agreement that requires various forms of content protection, including the protection of compressed and uncompressed content as it crosses the bus and protection from other attacks.

Screen scraping

Attacks on the screen contents, such as print screen and GetDC.

WDDM

Windows Display Driver Model. The display driver model that was introduced in Windows Vista.

WDDM v1

The first version of the Windows Display Driver Model. WDDM v1 drivers from various graphics hardware vendors shipped with Windows Vista.

WDDM v1.1

WDDM optimized for Windows 7. It contains several enhancements to the first version of WDDM. WDDM v1.1 drivers support additional features on Windows 7 that WDDM v1 drivers do not support.

XDDM

The Windows 2000/XP display driver model. Legacy XDDM display drivers are still supported in Windows Vista and Windows 7. New drivers should use WDDM instead of XDDM.