CoreDetection

<https://github.com/walbourn/directx-sdk-samples>

This is the DirectX SDK's Direct3D 11 sample updated to use the Windows 10 SDK without any dependencies on legacy DirectX SDK content. This sample is a Win32 desktop DirectX 11.0 application for Windows 10, Windows 8.1, Windows 8, and Windows 7.

**This is based on the legacy DirectX SDK (June 2010) Win32 desktop sample. This is not intended for use with Windows Store apps, Windows RT, or universal Windows apps.**

# Description



This sample demonstrates how to obtain details about the physical and logical core processor layout for Windows x86 and x64 systems. Robust handling of this detection is required for Win32 desktop applications making use of the [SetThreadAffinityMask](http://msdn.microsoft.com/en-us/library/windows/desktop/ms686247.aspx) API.

On Windows XP Service Pack 3, Windows Vista, Windows 7, and Windows 8.x the preferred method for detecting CPU information is using the [GetLogicalProcessorInformation](http://msdn.microsoft.com/en-us/library/windows/desktop/ms683194.aspx) API. Prior to that, the CPUID information must be properly decoded, handling various vendor-specific differences. The CPUID implementation can be removed if the application only supports Windows Vista or later (such as a Direct3D 11 only application).

This sample also works around some known issues with the GetLogicalProcessorInformation API. See [KB 932370](http://support.microsoft.com/kb/932370).

See the [Coding For Multiple Cores on Xbox 360 and Microsoft Windows](https://docs.microsoft.com/en-us/windows/win32/dxtecharts/coding-for-multiple-cores) article for more information.

Both SetThreadAffinityMask and GetLogicalProcessorInformation are desktop only APIs. Therefore, this sample does not apply to Windows Store apps.

# More Information

[Where is the DirectX SDK (2021 Edition)?](https://aka.ms/dxsdk)

[Games for Windows and DirectX SDK blog](https://walbourn.github.io/)