

# Writing WDM Kernel Mode Drivers for Windows<sup>®</sup>

## (Lecture with Lab Edition)

### Overview

The seminar covers both the architectural background and the practical details of implementing real, useful, WDM drivers. Of course, like all our seminars, this seminar is taught by experienced Windows driver writers who've "been there."

The overall goal of this seminar is to provide students with the necessary knowledge and hands-on experience to design, develop, and test many types of Windows WDM drivers. The key to reaching this goal is gaining a solid knowledge of the underlying architecture and operating principals of the Windows I/O, PnP, and Power subsystems. The seminar also specifically discussed basic USB concepts and the Windows USB stack.

Note that unlike other operating systems, drivers for different types of devices under Windows differ significantly (for example, SATA disk drivers have almost no implementation details in common with LAN communications drivers). Therefore, every type of driver cannot be covered in detail during the limited time available in this seminar.

The seminar focuses on the fundamentals and over-arching concepts that are applicable to all types of Windows kernel mode drivers. The seminar specifically discusses the implementation details (including numerous practical hints and tips) of software drivers, filter drivers, drivers for USB devices, and (to a lesser extent) drivers for PCI-based Programmed I/O devices. The seminar does not cover specific implementation of printer, video, SCSI, ATA, NDIS, 1394, Bluetooth, or (of course) user-mode drivers.

### Seminar Formats

This seminar is available as a 3 day lecture or 5 day lecture with interspersed hands on labs, which can be customized with additional or different topics for presentation at your location. Please contact an OSR Seminar Coordinator for details on arranging an on-site seminar.

### Target Audience

This seminar is designed for engineers who need to design, develop, or test Windows WDM drivers. Engineers who have experience with the Windows Driver Foundation Kernel Mode Driver Framework (KMDF) will also benefit from this seminar, as WDM provides the underlying basis for KMDF. Developers who want to develop file systems or file system filter drivers will also need the information presented in this seminar, as it is an absolute prerequisite to understanding those topics.

## Prerequisites

Students attending this seminar must have solid knowledge of general operating systems concepts (user mode versus kernel mode, virtual memory concepts, concurrency issues), device concepts (registers, interrupts), and the basics of Windows O/S Architecture. Previous experience developing device drivers (on any operating system) will definitely be an advantage.

While certainly not mandatory, students who wish to be especially well-prepared for this seminar may wish to read as much as they can of the Kernel-Mode Driver Architecture Design Guide (from the Windows Driver Kit), with special attention to the first five sections (Introduction to Windows Drivers, Windows Driver Model, Device Objects and Device Stacks, Kernel-Mode Driver Components, and Handling IRPs).

Note that almost all of this material will be discussed in detail during the seminar. However, having a basic understanding of these topics prior to attending the seminar will aid students in understanding the details of some of the more complex topics.

Due to the hands-on orientation of this seminar, attendees will be assumed to be able to use Windows at a user level, including using Microsoft Visual Studio (VC++). Working knowledge of the C programming language, and how to read and write to a file using Win32 are also assumed. Previous experience developing device drivers (on any operating system), or solid knowledge of general operating system and device driver concepts is necessary.

## Seminar Outline

### 1. The Windows Operating System

A brief review of the general architecture of the Windows operating system, as specifically applicable to driver writers.

### 2. Building the Device Tree

A detailed discussion of how the PnP Manager works with drivers to build stacks of devices through the enumeration process. The role of Function Drivers, Bus Drivers, and Filter Drivers is discussed. PDOs, FDOs, and Filter Device Objects are defined. IRPs, I/O Stack Locations, and how requests are sent from driver to driver using IoCallDriver are all discussed. Synchronous and asynchronous IRP completion. A brief discussion of completion routines is also included.

### 3. Driver Installation

In this section we discuss how to create installation control files for WDM and standard kernel mode device drivers. The Ten Most Frequently Used .INF File Sections are discussed in detail.

### 4. Building and Debugging Kernel-Mode Drivers

This section describes how WDM drivers are built using the WDK build environment as well as the basics of how to setup and use the Windows kernel mode debugger, WinDbg.

## **5. Interrupt Request Levels and Deferred Procedure Calls**

Windows synchronizes kernel mode activity by using a set of Interrupt Request Levels (IRQLs). This section covers how IRQLs are used to achieve synchronization within the OS. Also the processing that occurs at these IRQLs - including Deferred Procedure Calls (DPCs) and dispatching - are discussed.

## **6. The DriverEntry and AddDevice Entry Points**

The two most common entry points are described. We also provide a description of the functions typically called, and do a walk-through of code from a sample driver for each of these entry points. PnP and WDM issues.

**Lab** Setup and add a device interface

## **7. I/O Function Codes and Buffer Methods**

A specific discussion of I/O function codes (major and minor), as well as defining custom Device Control requests (IOCTLs). Direct I/O, Buffered I/O, and "Neither I/O" are also discussed.

## **8. Dispatching and Completing Requests**

This module describes Dispatch Routines including I/O request validation, as well as the basics of request queuing and completion.

## **9. PNP and START\_DEVICE**

In this module, we take our first overall look at the Windows PnP state machine. Building on what we learned when we discussed the Device Tree, we describe how the PnP subsystem interacts with Function and Filter drivers. We describe how to process "bus first" and "function first" requests, and walk-through example code for handling IRP\_MN\_START\_DEVICE requests.

**Lab** Add an IOCTL; Read/Write; Filter

## **10. Programmed I/O**

The basics of hardware registers (both memory mapped and those in port I/O space) are discussed, as well as the HAL functions used by Windows drivers to access them. Review of the flow of an I/O Request for drivers that directly handle device interrupts.

## **11. USB Concepts**

An introduction to basic USB concept. Device, configuration and endpoint descriptors are discussed as are control, bulk, interrupt and isochronous endpoints. The overall model of the Windows USB stack is presented.

## **12. USB Implementations**

Add an IOCTL; Read/Write; Filter

**Lab** The details of how to write drivers for simple USB devices, concentrating on the interactions between your function driver and the Windows USB stack. Retrieving descriptors, choosing a configuration and interface, interacting with control, bulk, and interrupt endpoints.

### **13. Serialization**

What it means to be "thread safe", and fully re-entrant. Serialization in kernel mode. Spin locks are defined, and the different types of spin locks are described.

**Lab** USB

### **14. Driver Verifier, Prefast, SDV**

In this section, we explain how to use Windows Driver Verifier to test your driver is explained. Also covered are the roles of PreFAST (PFD) and Static Driver Verifier. A number of driver debugging hints and tips are presented.

### **15. Cleanup, Close, and Cancel**

Cleanup, close and request cancellation are compared and contrasted. When one might need to implement support for each in a typical WDM driver is discussed. Guidelines for supporting request cancellation, including Cancel Safe Queues and in-progress request handling are presented.

### **16. Power Management**

Basic Windows power management concepts, as well as the responsibilities of the power policy owner, are discussed in this section. Proper handling of power transitions, interactions with PnP, starting and completing D-IRPs, and dealing with various failure situations are also discussed.

**Lab** Cancel; Power