Writing WDM Kernel Mode Drivers for Windows®

The Windows Driver Modem (WDM) is the native interface that serves as the base for all other Windows driver models. While most new general-purpose Windows drivers are now (rightly) written using the Windows Driver Foundation (WDF), understanding WDM is still important for many engineers. These include engineers who are maintaining older drivers that were written in WDM, as well as those who use driver models other than WDF that are based on WDM.

The overall goal of this seminar is to provide students with a solid understanding of the WDM core concepts. This knowledge will allow them to design, develop, and test many types of Windows WDM drivers, if that is their goal. However, this knowledge will also provide them with deep knowledge of native I/O interface used by Windows, and with them allow them go on to learn the implementation details other WDM-based Miniport architecture (such as StorPort, AVStream, or other types of devices).

Details

Length: 5 days

Format: Lecture and Lab

Target Audience
Engineers who need to understand how to design, develop, maintain or test Windows drivers using the Windows Driver Model (WDM).

Developers who seek a thorough understanding of the workings of the Windows I/O subsystem, either in preparation for writing/maintaining a WDM driver or for gaining a stronger understanding of Windows architecture.

Important, Please Read: This seminar deals strictly with the Windows Driver Model (WDM) and does not prepare attendees for writing drivers using the Windows Driver Foundation. Most new driver development for Windows is done using the Windows Driver Foundation and not WDM. Therefore, most students who need to design or develop new Windows drivers should not select this seminar, but should rather attend OSR's *Writing WDF Drivers: Core Concepts* seminar.

Students who want to learn about writing File Systems for Windows are probably best served by attending OSR's *Windows Internals and Software Driver Development seminar*, and not this seminar.

See "Which OSR Seminar is Right for Me" http://www.osr.com/seminar_choice.html -- If you have any questions about which seminar will best fit your needs, please contact an OSR Seminar Consultant by email or phone. We're here to help!

Prerequisites

Students attending this seminar must have a good working knowledge of O/S concepts in general (user mode versus kernel mode, virtual memory concepts, and concurrency issues) and Windows O/S architecture concepts in particular. 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 Visual Studio (VC++). Working knowledge of the C programming language, and how to read and write to a file using Win32 (CreateFile, ReadFile, WriteFile) are also assumed.

After This Seminar

This seminar provides you with the necessary information to attend OSR's Developing File Systems for Windows seminar.

Seminar Outline

- **Introduction**
  Welcome remarks, seminar goals and objectives, and a brief introduction to Windows driver architectures.

- **Windows Architecture Overview**
  A brief review of Windows operating system architecture, focused specifically on the details needed by a WDM driver writer.

- **The Windows Device Tree**
  A description of how the Windows PnP subsystem discovers and enumerates drivers, on both dynamically enumerable buses (such as PCI and USB) and non-dynamically enumerable Simple Peripheral Buses (SPBs, such as SPI, GPIO, and high speed serial buses). 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.

- **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.

- **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.

- **The DriverEntry and AddDevice Entry Points**
  The two initial entry points in a driver are described. We also provide a description of the

functions a driver typically calls within each of these routines, and do a walk-through of code from a sample driver for each of these entry points. PnP and WDM issues.

Lab: Building and Debugging, Driver Initialization (DriverEntry, AddDevice, etc).

- **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.

- **Interrupt Levels & Deferred Procedure Calls**
  In this module, we discuss the all-important concept of Interrupt Request Levels (IRQLs), and the specific uses that Windows makes of various IRQLs. We also discuss Deferred Procedure Calls (DPCs) and how they're used in Windows for Interrupt Service Routine completion (DPCforISR). We also discuss passive-level interrupts and its associated Work Item for ISR.

- **Buffer Methods and IOCTL Definitions**
  In this section, the different ways that requestor data buffers can be described are discussed. Direct I/O, Buffered I/O and "Neither I/O" are described, compared, and contrasted. Also discussed is how to define custom Device IO Control Codes (IOCTLs), and how the previously described buffering methods apply to IOCTLs.

- **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.

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

  Lab: Request Processing, Buffering and Queuing

- **Case Study: Programmed I/O Driver**
  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.

- **Tools for Driver Quality**
  In this section, we explain the tools and techniques for verifying and testing your driver. Windows Driver Verifier, SDV, and Code Analysis (PreFast for Drivers) are all discussed.

- **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

  Lab: Static Tools, Driver-to-Driver Communication

- **Introduction to Power Management**
  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.

- **I/O Completion Details**
  A very detailed and important discussion of how Windows handles I/O completion. Why it is necessary to call IoMarkIrpPending and return STATUS_PENDING. When you must either return STATUS_PENDING or block in the dispatch routine if a completion routine reclaims an IRP with STATUS_MORE_PROCESSING_REQUIRED.

- **Introducing WDF**
  This module provides a short (about 2 hour) brain-dump of the basics of writing Windows drivers using the Windows Driver Foundation. General architectural concepts, taxonomy, key WDF Objects are discussed. The basics of how a WDF driver is constructed is discussed, including Event Processing Callbacks. WDF driver initialization, including an overview of WDF's unified PnP and Power Management implementation. An overview of how I/O requests are received and processed (WDFQUEUEs, dispatching types, and WdfCompleteRequest). Includes a brief walk-through of a simple WDF driver.

  Lab: Cancel, Power, KMDF Driver Initialization (optional)