

## Writing Windows Virtual Device Drivers 2nd Edition

If you're a Windows & developer, this book will tell you how to use virtual device drivers (VxDs) to write programs that have direct access to hardware devices, can interface with vital CPU functions, and can take over parts of the operating system. Fully-commented, complete working source code examples demonstrate how to write a VxD to talk to any hardware device, and show the wealth of tricks you can perform with VxDs, including interprocess communication. An accompanying disk contains VxD-Lite, Microsoft's toolkit for building generic virtual device drivers.

There is nothing like the power of the kernel in Windows - but how do you write kernel drivers to take advantage of that power? This book will show you how. The book describes software kernel drivers programming for Windows. These drivers don't deal with hardware, but rather with the system itself: processes, threads, modules, registry and more. Kernel code can be used for monitoring important events, preventing some from occurring if needed. Various filters can be written that can intercept calls that a driver may be interested in.

PCMag.com is a leading authority on technology, delivering Labs-based, independent reviews of the latest products and services. Our expert industry analysis and practical solutions help you make better buying decisions and get more from technology.

An exhaustive technical manual outlines the Windows NT concepts related to drivers; shows how to develop the best drivers for particular applications; covers the I/O Subsystem and implementation of standard kernel mode drivers; and more. Original. (Intermediate).

The quick way to learn Windows 10 This is learning made easy. Get more done quickly with Windows 10. Jump in wherever you need answers--brisk lessons and colorful screenshots show you exactly what to do, step by step. Discover fun and functional Windows 10 features! Work with the new, improved Start menu and Start screen Learn about different sign-in methods Put the Cortana personal assistant to work for you Manage your online reading list and annotate articles with the new browser, Microsoft Edge Help safeguard your computer, your information, and your privacy Manage connections to networks, devices, and storage resources

Develop the software and hardware you never think about. We're talking about the nitty-gritty behind the buttons on your microwave, inside your thermostat, inside the keyboard used to type this description, and even running the monitor on which you are reading it now. Such stuff is termed embedded systems, and this book shows how to design and develop embedded systems at a professional level. Because yes, many people quietly make a successful career doing just that. Building embedded systems can be both fun and intimidating. Putting together an embedded system requires skill sets from multiple engineering disciplines, from software and hardware in particular. Building Embedded Systems is a book about helping you do things in the right way from the beginning of your first project: Programmers who know software will learn what they need to know about hardware. Engineers with hardware knowledge likewise will learn about the software side. Whatever your background is, Building Embedded Systems is the perfect book to fill in any knowledge gaps and get you started in a career programming for everyday devices. Author Changyi Gu brings more than fifteen years of experience in working his way up the ladder in the field of embedded systems. He brings knowledge of numerous approaches to embedded systems design, including the System on Programmable Chips (SOPC) approach that is currently growing to dominate the field. His knowledge and experience make Building Embedded Systems an excellent book for anyone wanting to enter the field, or even just to do some embedded programming as a side project. What You Will Learn Program embedded systems at the hardware level Learn current industry practices in firmware development Develop practical knowledge of embedded hardware options Create tight integration between software and hardware Practice a work flow leading to successful outcomes Build from transistor level to the system level Make sound choices between performance and cost Who This Book Is For Embedded-system engineers and intermediate electronics enthusiasts who are seeking tighter integration between software and hardware. Those who favor the System on a Programmable Chip (SOPC) approach will in particular benefit from this book. Students in both Electrical Engineering and Computer Science can also benefit from this book and the real-life industry practice it provides.

This book and companion disk are designed for accomplished programmers who understand the Windows environment and want to optimize their files. The text will especially benefit tool developers, multimedia developers, and graphic tool developers.

Writing Windows Virtual Device Drivers Addison Wesley

Developing Windows NT Device Drivers: A Programmer's Handbook offers programmers a comprehensive and in-depth guide to building device drivers for Windows NT. Written by two experienced driver developers, Edward N. Dekker and Joseph M. Newcomer, this book provides detailed coverage of techniques, tools, methods, and pitfalls to help make the often complex and byzantine "black art" of driver development straightforward and accessible. This book is designed for anyone involved in the development of Windows NT Device Drivers, particularly those working on drivers for nonstandard devices that Microsoft has not specifically supported. Because Windows NT does not permit an application program to directly manipulate hardware, a customized kernel mode device driver must be created for these nonstandard devices. And since experience has clearly shown that superficial knowledge can be hazardous when developing device drivers, the authors have taken care to explore each relevant topic in depth. This book's coverage focuses on drivers for polled, programmed I/O, interrupt-driven, and DMA devices. The authors discuss the components of a kernel mode device driver for Windows NT, including background on the two primary bus interfaces used in today's computers: the ISA and PCI buses. Developers will learn the mechanics of compilation and linking, how the drivers register themselves with the system, experience-based techniques for debugging, and how to build robust, portable, multithread- and multiprocessor-safe device drivers that work as intended and won't crash the system. The authors also show how to call the Windows

NT kernel for the many services required to support a device driver and demonstrate some specialized techniques, such as mapping device memory or kernel memory into user space. Thus developers will not only learn the specific mechanics of high-quality device driver development for Windows NT, but will gain a deeper understanding of the foundations of device driver design.

Master the new Windows Driver Model (WDM) common to Windows 98 and Windows 2000. You get theory, instruction and practice in driver development, installation and debugging. Addresses hardware and software interface issues, driver types, and a description of the new 'layer' model of WDM. ;

A Guide to Kernel Exploitation: Attacking the Core discusses the theoretical techniques and approaches needed to develop reliable and effective kernel-level exploits, and applies them to different operating systems, namely, UNIX derivatives, Mac OS X, and Windows. Concepts and tactics are presented categorically so that even when a specifically detailed vulnerability has been patched, the foundational information provided will help hackers in writing a newer, better attack; or help pen testers, auditors, and the like develop a more concrete design and defensive structure. The book is organized into four parts. Part I introduces the kernel and sets out the theoretical basis on which to build the rest of the book. Part II focuses on different operating systems and describes exploits for them that target various bug classes. Part III on remote kernel exploitation analyzes the effects of the remote scenario and presents new techniques to target remote issues. It includes a step-by-step analysis of the development of a reliable, one-shot, remote exploit for a real vulnerability a bug affecting the SCTP subsystem found in the Linux kernel. Finally, Part IV wraps up the analysis on kernel exploitation and looks at what the future may hold. Covers a range of operating system families — UNIX derivatives, Mac OS X, Windows Details common scenarios such as generic memory corruption (stack overflow, heap overflow, etc.) issues, logical bugs and race conditions Delivers the reader from user-land exploitation to the world of kernel-land (OS) exploits/attacks, with a particular focus on the steps that lead to the creation of successful techniques, in order to give to the reader something more than just a set of tricks

The Microsoft® Windows® driver model (WDM) supports Plug and Play, provides power management capabilities, and expands on the driver/minidriver approach. Written by long-time device-driver expert Walter Oney in cooperation with the Windows kernel team, this book provides extensive practical examples, illustrations, advice, and line-by-line analysis of code samples to clarify real-world driver-programming issues. And it's been updated with the latest details about the driver technologies in Windows XP and Windows 2000, plus more information about how to debug drivers. Topics covered include: Beginning a driver project and the structure of a WDM driver; NEW: Minidrivers and class drivers, driver taxonomy, the WDM development environment and tools, management checklist, driver selection and loading, approved API calls, and driver stacks Basic programming techniques; NEW: Safe string functions, memory limits, the Driver Verifier scheme and tags, the kernel handle flag, and the Windows 98 floating-point problem Synchronization; NEW: Details about the interrupt request level (IRQL) scheme, along with Windows 98 and Windows Me compatibility The I/O request packet (IRP) and I/O control operations; NEW: How to send control operations to other drivers, custom queue implementations, and how to handle and safely cancel IRPs Plug and Play for function drivers; NEW: Controller and multifunction devices, monitoring device removal in user mode, Human Interface Devices (HID), including joysticks and other game controllers, minidrivers for non-HID devices, and feature reports Reading and writing data, power management, and Windows Management Instrumentation (WMI) NEW: System wakeup, the WMI control for idle detection, and using WMIMOFCK Specialized topics and distributing drivers; NEW: USB 2.0, selective suspend, Windows Hardware Quality Lab (WHQL) certification, driver selection and loading, officially approved API calls, and driver stacks COVERS WINDOWS 98, WINDOWS ME, WINDOWS 2000, AND WINDOWS XP! CD-ROM FEATURES: A fully searchable electronic copy of the book Sample code in Microsoft Visual C++® A Note Regarding the CD or DVD The print version of this book ships with a CD or DVD. For those customers purchasing one of the digital formats in which this book is available, we are pleased to offer the CD/DVD content as a free download via O'Reilly Media's Digital Distribution services. To download this content, please visit O'Reilly's web site, search for the title of this book to find its catalog page, and click on the link below the cover image (Examples, Companion Content, or Practice Files). Note that while we provide as much of the media content as we are able via free download, we are sometimes limited by licensing restrictions. Please direct any questions or concerns to [booktech@oreilly.com](mailto:booktech@oreilly.com).

Provides advice for Visual Basic programmers attempting to interface hardware through standard ports.

Expert coverage of the cornerstone of Windows. Thielen and Woodruff, experts in this field, introduce the concepts of virtual device drivers and provide fully commented samples of the main types of drivers currently being written. A bonus disk provides programmers with VxDlite, Microsoft's toolkit for building generic virtual device drivers.

The Definitive Guide to Windows API Programming, Fully Updated for Windows 7, Windows Server 2008, and Windows Vista Windows System Programming, Fourth Edition, now contains extensive new coverage of 64-bit programming, parallelism, multicore systems, and many other crucial topics. Johnson Hart's robust code examples have been updated and streamlined throughout. They have been debugged and tested in both 32-bit and 64-bit versions, on single and multiprocessor systems, and under Windows 7, Vista, Server 2008, and Windows XP. To clarify program operation, sample programs are now illustrated with dozens of screenshots. Hart systematically covers Windows externals at the API level, presenting practical coverage of all the services Windows programmers need, and emphasizing how Windows functions actually behave and interact in real-world applications. Hart begins with features used in single-process applications and gradually progresses to more sophisticated functions and multithreaded environments. Topics covered include file systems, memory management, exceptions, processes, threads, synchronization, interprocess communication, Windows services, and security. New coverage in this edition includes Leveraging parallelism and maximizing performance in multicore systems Promoting source code portability and application interoperability across Windows, Linux, and UNIX Using 64-bit address spaces and ensuring 64-bit/32-bit portability Improving performance and scalability using threads, thread pools, and completion ports Techniques to improve program reliability and

performance in all systems Windows performance-enhancing API features available starting with Windows Vista, such as slim reader/writer locks and condition variables A companion Web site, [jmhartsoftware.com](http://jmhartsoftware.com), contains all sample code, Visual Studio projects, additional examples, errata, reader comments, and Windows commentary and discussion.

Start developing robust drivers with expert guidance from the teams who developed Windows Driver Foundation. This comprehensive book gets you up to speed quickly and goes beyond the fundamentals to help you extend your Windows development skills. You get best practices, technical guidance, and extensive code samples to help you master the intricacies of the next-generation driver model—and simplify driver development. Discover how to: Use the Windows Driver Foundation to develop kernel-mode or user-mode drivers Create drivers that support Plug and Play and power management—with minimal code Implement robust I/O handling code Effectively manage synchronization and concurrency in driver code Develop user-mode drivers for protocol-based and serial-bus-based devices Use USB-specific features of the frameworks to quickly develop drivers for USB devices Design and implement kernel-mode drivers for DMA devices Evaluate your drivers with source code analysis and static verification tools Apply best practices to test, debug, and install drivers PLUS—Get driver code samples on the Web

\* Includes a complete QuickBasic compiler with source code. We cannot overstate that this is a huge marketing hook. Virtually every experienced programmer today started out with some version of Basic or QuickBasic and has at some point in their career wondered how it worked. The sheer nostalgia alone will generate sales. The idea of having QuickBasic for them to play with (or let their kids play with) will generate sales. \* One of a kind book – nothing else comes close to this book. \* Demystifies compiler technology for ordinary programmers – this is a subject usually covered by academic books in a manner too advanced for most developers. This book is pitched at a level accessible to all but beginners. \* Teaches skills used in many other types of programming from creation of macro/scripting languages to file parsing.

The only book available on networking device drivers, this book describes the various network device driver architectures and covers the most common ones in great detail—including NDIS, 3COM and Microsoft; ODI from Novell; Packet Driver from Ftp Software; and DLPI from USL, Inc. Popular network operating systems are also covered from the device driver standpoint.

Provides information on writing a driver in Linux, covering such topics as character devices, network interfaces, driver debugging, concurrency, and interrupts.

Delve inside Windows architecture and internals—and see how core components work behind the scenes. Led by three renowned internals experts, this classic guide is fully updated for Windows 7 and Windows Server 2008 R2—and now presents its coverage in two volumes. As always, you get critical insider perspectives on how Windows operates. And through hands-on experiments, you'll experience its internal behavior firsthand—knowledge you can apply to improve application design, debugging, system performance, and support. In Part 2, you'll examine: Core subsystems for I/O, storage, memory management, cache manager, and file systems Startup and shutdown processes Crash-dump analysis, including troubleshooting tools and techniques

InfoWorld is targeted to Senior IT professionals. Content is segmented into Channels and Topic Centers. InfoWorld also celebrates people, companies, and projects.

Software developer and author Karen Hazzah expands her original treatise on device drivers in the second edition of *Writing Windows VxDs and Device Drivers*. The book and companion disk include the author's library of wrapper functions that allow the progr

An authoritative guide to Windows NT driver development, now completely revised and updated. The CD-ROM includes all source code, plus Microsoft hardware standards documents, demo software, and more.

This book explains device drivers and how to write them for the Windows environment. It explains the differences between DOS and Windows drivers, then details the different Windows operating modes and the three types of Windows device drivers--system, printer, and virtual.

This e-book has been written for embedded software developers by Apriorit experts. It goes in-depth on how to save time when developing a Windows device driver by emulating a physical device with QEMU and explores the details of device driver emulation based on QEMU virtual devices. Key Highlights ? Develop a Windows device driver without a physical device ? Learn to securely test device drivers, find and fix defects ? Leverage the QEMU machine emulator for your needs Book Description Developing a Windows driver for an embedded device is a very specific task, as this software should ensure stable and secure communication between operating system and a highly specialized device. Besides, in the world of ever increasing competitiveness, manufacturers tend to force developers to start working on embedded software before hardware is manufactured. Fortunately, developers can emulate a physical device using virtualization technologies like QEMU. After creating a virtual device in QEMU, developers can use it not only for a device driver development but also for its testing and fixing defects. In this e-book, the author shares his practical experience on developing Windows drivers using a QEMU virtual device. You'll dive deep into driver implementation process and learn about all the benefits and limitations of device emulation in QEMU. The e-book includes detailed steps to establish communication between a device and its driver. It also shows how to use QEMU for building running, testing, and debugging the whole environment. This approach was tested by Apriorit team for quite a long time, so it has already confirmed its value and effectiveness. What you will learn ? Dive into driver implementation stages ? Weigh up the pros and cons of using a QEMU virtual device ? Initialize the device in QEMU ? Develop your own Windows driver for a QEMU virtual hardware ? Establish communication between a device and its driver ? Process requests from a user mode application ? Use QEMU for building running, testing, and debugging embedded software About the Author Artem Kotovsky is a Software Analyst at Apriorit Inc. Apriorit Inc. is a software development service provider headquartered in the Dover, DE, US, with several development centers in Eastern Europe. With over 350 professionals, it brings high-quality services on software consulting, research, and development to software vendors and IT companies worldwide. Apriorit's main specialties are cybersecurity and data management projects, where system programming, driver and kernel level development, research and reversing matter. The company has an independent web platform development department focusing on building cloud platforms for business. Table of Contents Introduction Why do we use QEMU? Pros and cons of using a QEMU virtual device Driver implementation stages Communication between a device and its driver I/O address space Interrupts Line-based interrupts Message-signaled interrupts Bus mastering Test device specifications Structure of the device I/O memory Interrupts Device description in QEMU Initializing the device in QEMU Working with the I/O

memory space Working with interrupts Working with DMA memory Processing requests QEMU device Implementing a WDF driver for the test device The minimum driver Initializing device resources Working with I/O memory Interrupt handling Working with DMA Sending requests to the device Processing requests from a user mode application Testing and debugging Quality control of driver code Driver installation Driver communication Implementing driver unit tests Implementing driver autotest Driver verification with Driver Verifier and WDF Verifier References

The Cloud Computing and Services Science book comprises a collection of the best papers presented at the International Conference on Cloud Computing and Services Science (CLOSER), which was held in The Netherlands in May 2011. In netting papers from the conference researchers and experts from all over the world explore a wide-ranging variety of the emerging Cloud Computing platforms, models, applications and enabling technologies. Further, in several papers the authors exemplify essential links to Services Science as service development abstraction, service innovation, and service engineering, acknowledging the service-orientation in most current IT-driven structures in the Cloud. The Cloud Computing and Services Science book is organized around important dimensions of technology trends in the domain of cloud computing in relation to a broad scientific understanding of modern services emerging from services science. The papers of this book are inspired by scholarly and practical work on the latest advances related to cloud infrastructure, operations, security, services, and management through the global network. This book includes several features that will be helpful, interesting, and inspirational to students, researchers as well as practitioners. Professionals and decision makers working in this field will also benefit from this book

For developers who must know and understand the fundamentals to be able to apply the more advanced aspects that will emerge with NT 5, here is an in-depth book to the rescue, covering the core techniques of programming NT device drivers.

[Copyright: 964a4d443c69c333382a82cfbf18662c](#)