

Porting Windows® Device Drivers to AMD64 Platforms

March 2003

ADVANCED MICRO DEVICES, INC.

One AMD Place

Sunnyvale, CA 94088

Contents

Contents	2
Introduction.....	4
AMD64 technology primer	4
Porting to AMD64	5
Maintaining compatibility with 32-bit applications	5
Tools for Porting your Driver	5
Checklist for Porting your Driver.....	6
Windows 64 Helper Functions.....	7
Intrinsic Functions	7
Resources and Call to Action.....	8
Acronyms and Terms.....	9

Windows Hardware Engineering Conference

Author's Disclaimer and Copyright:
© 2003 Advanced Micro Devices, Inc.
All rights reserved.

The contents of this document are provided in connection with Advanced Micro Devices, Inc. ("AMD") products and technology. AMD makes no representations or warranties with respect to the accuracy or completeness of the contents of this publication and reserves the right to make changes to specifications and product and technology descriptions at any time without notice. No license, whether express, implied, arising by estoppel or otherwise, to any intellectual property rights is granted by this publication. AMD assumes no liability whatsoever, and disclaims any express or implied warranty, relating to its products including, but not limited to, the implied warranty of merchantability, fitness for a particular purpose, or infringement of any intellectual property right.

AMD, the AMD Arrow logo, AMD Athlon, AMD Opteron, and combinations thereof and 3DNow! are trademarks of Advanced Micro Devices, Inc. Windows is a registered trademark of Microsoft Corporation. Other product names used in this publication are for identification purposes only and may be trademarks of their respective companies.

WinHEC Sponsors' Disclaimer: The contents of this document have not been authored or confirmed by Microsoft or the WinHEC conference co-sponsors (hereinafter "WinHEC Sponsors"). Accordingly, the information contained in this document does not necessarily represent the views of the WinHEC Sponsors and the WinHEC Sponsors cannot make any representation concerning its accuracy. THE WINHEC SPONSORS MAKE NO WARRANTIES, EXPRESS OR IMPLIED, WITH RESPECT TO THIS INFORMATION.

Microsoft, Windows, and Windows NT are trademarks or registered trademarks of Microsoft Corporation in the United States and/or other countries. Other product and company names mentioned herein may be the trademarks of their respective owners.

Introduction

The AMD64 technology is designed to enable existing 32-bit application software to run unchanged and co-exist with 64-bit applications running on Microsoft® Windows® operating system for AMD64.

While there may be compelling reasons to extend high performance applications to a 64-bit version of the Microsoft Windows family of operating systems, there is a large base of 32-bit application software which can run with full compatibility. A 64-bit Operating System is required for utilizing the innovations of the AMD64 processor family. Device drivers are part of a 64-bit Operating System and thus are also required to operate in 64-bit Mode. Making your device drivers 64-bit safe will allow your device to work with the benefits of a 64-bit OS while maintaining compatibility with existing 32-bit software applications.

To support your devices under the AMD64 based Windows OS, you will need an understanding of the potential problem areas for a device driver. This paper will present those areas and provide assistance to address each area.

This white paper is divided into two sections: It will first present the features of the AMD64 processors that are enabled on when running 64-bit Windows®. Second, it will discuss the tools and present a checklist for porting your driver to 64-bit Windows for the AMD64 platform.

AMD64 technology primer

AMD64 processors operate in either Long Mode or Legacy Mode. The AMD Athlon™ 64 processor and AMD Opteron™ processors are examples of an AMD64 processor. Legacy Mode enables existing 32-bit and 16-bit operating systems to operate unchanged. This paper will describe the changes that affect a device driver with the system in Long Mode. A 64-bit device driver is required for use on a 64-bit version of Microsoft Windows®.

Long Mode consists of a 64-bit Mode and Compatibility Mode. A 64-bit operating system and 64-bit device drivers are required in Long Mode. A 32-bit application can operate in Compatibility Mode under a 64-bit OS. If the 32-bit application has an associated device driver for privileged instruction access, the device driver will need porting.

AMD64 supports the following new features:

- 64-bit virtual addresses (implementations can have less)
- Register extensions through a new prefix (REX):
 - Adds eight GPRs (R8–R15)
 - Widens GPRs to 64-bits
 - Adds eight 128-bit Streaming SIMD Extension (SSE/SSE2) registers (XMM8–XMM15)
- 64-bit instruction pointer (RIP)
- New RIP-relative data addressing mode
- Flat address space with single code, data, and stack space

Operating Mode		OS Required	Application Recompile Required	Defaults		Register Extensions	Typical GPR Width
				Address Size (bits)	Operand Size (bits)		
Long Mode	64-bit Mode	New 64-bit OS	yes	64	32	yes	64
	Compatibility Mode		no	32		no	32
				16	16		16
Legacy Mode	Protected Mode	Legacy 32-bit OS	no	32	32	no	32
	Virtual-8086 Mode			16			
	Real Mode	Legacy 16-bit OS		16	16		16

An understanding of the registers available in AMD64 based platforms will allow you to further optimize your code. High-performance hand optimized software can fully utilize these extra registers and other architectural features of the CPU. To take advantage of all these features, please refer to the AMD Athlon 64 and AMD Opteron Software Optimization Guide, available along with other technical resources at <http://developer.amd.com/>.

Porting to AMD64

Maintaining compatibility with 32-bit applications

A system call made to 64-bit Windows by a 32-bit application will require translation of its arguments such as address pointers. This translation layer is integral to the 64-bit operating system. 64-bit Windows has a layer called Windows on Windows 64 (WOW64). WOW64 is the emulation layer that enables a 32-bit application to application to operate in Compatibility Mode when under 64-bit Windows.

WOW64 exists as a Dynamically Linked Library (DLL) that is integral to the operating system. The OS creates a separate 32-bit process to run each 32-bit application, and the thunking DLL resides within each of these 32-bit processes context. The 32-bit application is dynamically linked to the thunking layer. Each time the application executes a system call to the operating system, the thunking layer is invoked and performs the following sequence of operations:

- Translates parameters if necessary
- Transfers control to (Calls) the 64-bit kernel
- Translates the results if necessary
- Returns the results to the application

Tools for Porting your Driver

Before embarking on porting your driver, you will need to obtain updated tools for this effort. The Windows Device Driver Development Kit has been updated for developing your device

drivers for the AMD64 platform. The Windows DDK has new compiler functionality enabled for the Long Mode in AMD64 processors. It is recommended that you review the MSDN DDK documentation titled “*Compiler for AMD 64-Bit Environments*” and “*Calling Convention for AMD 64-Bit Environments*” at <http://msdn.microsoft.com/library/default.asp>. The updated compiler also includes increased conformance to the ANSI/ISO standard. This may result in new build errors or warnings from previous compiler builds.

Windbg can also be used for remotely debugging your AMD64 driver from either a 32 or 64-bit system. The AMD64 platform has natively extended the registers from an x86 32-bit system. This extension enables you to begin the porting adventure with your existing device driver source code and maintain familiarity while debugging.

Checklist for Porting your Driver

After obtaining updated tools, your driver may require some modifications. It is recommended that all device driver source files are reviewed and modified accordingly. These guidelines below will enable your driver to be ported while following sound practices and using a single source code base for 32-bit and 64-bit versions of your driver. Include in review all source code, header/include, resource files and makefiles.

- **Use `#if defined (__AMD64__)` for any AMD64 specific references.**
- **Locate all inline C assembly code** and convert to use intrinsic functions or native assembly subroutine.
- **Review existing assembly** for location of parameters not on stack. The compiler for AMD64 uses the `FASTCALL` calling convention exclusively.
- **Know your adapter and OS capability.** If your adapter supports DMA, identify its physical addressing capability in the `DEVICE_DESCRIPTION` structure. There are many adapters in the market today that only support a 32-bit address space. If your device falls into this category, it is physically un-able to directly access memory above the 4 GB range. By setting the field `Dma64BitAddresses` to `FALSE` in the `DEVICE_DESCRIPTION` structure, the OS will double-buffer for the device driver. This may result in lower system performance but will achieve 64-bit compatibility. If your device supports 64-bit addresses, the driver may benefit the larger address range. The Device Driver can identify if the OS is 64-bit capable through the `Mm64BitPhysicalAddress` system variable.
- **64-bit Windows follows the LLP64 programming model.** In an LLP64 platform, pointers and long longs are 64-bit values. Note this is different from 64-bit Unix operating systems in which the `long` data type is 64-bit. The integer data type remains 32-bit.
- **Review references to pointers and integers.** In 64-bit Windows all pointers are now 64-bit. Use `POINTER_64` and `POINTER_32` to specify a pointer with a desired size if necessary. When 64-bit integer value is required, use `ULARGE_INTEGER`. Use Helper Functions when converting between data types.
- **Use `PHYSICAL_ADDRESS` when referencing a physical address.** The `PHYSICAL_ADDRESS` data type has the same definition of a `LARGE_INTEGER`. It is a union of either a `QuadPart` or structure members `LowPart` and `HighPart`. In 32-bit mode fill in the `LowPart` field and zero out the `HighPart`. In 64-bit mode, you can use the 64-bit field, `QuadPart`.
- **Review use of any pointer arithmetic and confirm result.** Indexing into an array using pointer arithmetic may give you unexpected results from the value being used to calculate the index.
- **Review all IOCTL's.** An IOCTL allows an application to send a control code to a specified device driver. The content and control are device driver specific.

A Windows application calls `DeviceIoControl()` to send a control code to a device driver. This call contains parameters for passing data into and out of the driver through buffers. Although the pointers to these input and output buffers are converted to 64-bit, the contents pointed to the input and output buffers are not thunked. If the IOCTL passes data through the buffers, the driver must correctly identify the caller's context to interpret and write data to the buffers. This can be accomplished by calling the `IoIs32bitProcess()` kernel mode function, or by updating the definition of IOCTL's with a new set of IOCTL's that mirror the original 32-bit set.

- **Convert media instruction to SSE/SSE2 Instructions**

Microsoft Windows for AMD64 will not context switch x87, 3DNow!, MMX for 64-bit native threads. This code may be converted to SSE/SSE2 through the use of intrinsic functions.

- **Update sources makefile to build either 32 or 64-bit driver.** This is a simple update that is designed to make your support easier.

- **Update INF to use amd64 platform directive when supporting multiple platforms**

```
[install-section-name.ntamd64]
```

- **Identify and remove use of BIOS callbacks.** These are not allowed once the AMD64 platform is running in 64-bit Long Mode.

- **Beware of legacy API's which could have been removed from OS.** These are documented in the MSDN Library.

The MSDN Library section called "Porting Issues Checklist" has further details on this subject.

Windows 64 Helper Functions

64-bit Windows has Helper Functions for converting between data types. Using these inline Windows 64 helper functions allows the safe conversion between pointer and integer types by preserving the sign attribute of the source value. Windows 64 helper functions do not rely on assumptions about the relative sizes of these types.

```
HandleToUlong()  
HandleToLong()  
LongToHandle()  
PtrToUlong()  
PtrToUint()  
PtrToUshort()  
PtrToLong()  
PtrToInt()  
PtrToShort()  
IntToPtr()  
UIntToPtr()  
LongToPtr()  
ULongToPtr()
```

Intrinsic Functions

The compiler for AMD64 has intrinsic functions, which provide you functionality that you previously may have used inline assembly for. It is recommended that intrinsic functions be

used over a native AMD64 assembly subroutine. Intrinsic functions should make your code more portable to other platforms supported by the Microsoft Windows family of operating systems.

- Interlocked Intrinsic Functions
- String MOV Intrinsic Functions
- Arithmetic Intrinsic Functions
- Intrinsic Functions to Read and Write Registers
- TEB/PCR Access Intrinsic Functions
- Special and Privileged Instruction Intrinsic Functions
- Port I/O Intrinsic Functions
- Compiler Helper Intrinsic Functions

See the MSDN Library for detailed documentation on intrinsic functions.

<http://msdn.microsoft.com>

Resources and Call to Action

Call to Action:

- For system manufacturers: Obtain 64-bit device drivers for peripherals
- For device manufacturers: Urge your driver developers to port drivers to 64-bit
- For driver developers: Port your 32-bit drivers to 64-bit using a single source code base

Contacts:

- For Microsoft Driver Development Kit see <http://www.microsoft.com/ddk/>
- If you have AMD64 hardware, e-mail x64info@microsoft.com to join the WindowsBeta for AMD64 program
- If you would like physical or remote access to AMD64 hardware, assistance with porting or performance optimization of drivers, please contact the AMD Developer Center through <http://www.developwithamd.com/devcenter>.

Resources:

AMD64 Technology Programmer's Manuals at <http://www.amd.com>

- AMD x86-64 Architecture Programmer's Manual Volume 1: Application Programming
- AMD x86-64 Architecture Programmer's Manual Volume 2: System Programming
- AMD x86-64 Architecture Programmer's Manual Volume 3: General-Purpose and System Instructions
- AMD x86-64 Architecture Programmer's Manual Volume 4: 128-Bit Media Instructions
- AMD x86-64 Architecture Programmer's Manual Volume 5: 64-Bit Media and x87 Floating-Point Instructions

Acronyms and Terms

AMD64 AMD's 64-bit platform that extends x86 and defines a new class of computing. AMD's enhancements to the venerable x86 architecture are designed to allow users of personal computer clients and servers to migrate seamlessly to the superior performance of 64-bit technology.

DDK Microsoft® Windows® Driver Development Kit