ACPI BIOS Guideline for Linux

Thomas Renninger - Copyright SUSE Linux GmbH, 2008

August 7, 2008

Abstract

This specification is intended for PC hardware vendors and PC BIOS developers. It documents and describes ACPI implementations of the Linux kernel which are important for BIOS developers. Irregularities to the ACPI specification are discussed. Problems that may occur when Linux is used with ACPI driven BIOSes are outlined, with explanations on how to avoid them.

Contents

1 Introduction

Many PC hardware vendors have recently started to offer Linux pre-loaded on their hardware. Linux fully supports version 2.0 of the ACPI specification and partly supports version 3.0. There are still some pitfalls for vendors, which can easily be avoided. This paper describes problems that can occur with ACPI implementations on Linux. Input and feedback from vendors and programmers is welcomed. If you have any ideas for improving or expanding this documentation, please send suggestions to trenn@suse.de or to the linux-acpi mailing list [8].

2 Vendor specific ACPI implementations

Linux supports most ACPI specified devices perfectly (e.g. "battery", "battery vs. plugged-in status", "lid", "cpufreq frequency scaling (P-states)", "processor sleep states (C-states)" and much more). Vendors often implement devices through ACPI which are not included in the general ACPI or other specifications. Examples include wireless LAN on/off switches, and buttons for volume up/down and mute.

Vendors provide proprietary drivers for Windows for their specific devices and in many cases there also is a re-engineered Linux driver (e.g. asus_acpi, sony_acpi, thinkpad_acpi). Those Linux drivers are often not complete, and are hard to maintain. It is possible, for example, for parts of the undocumented interface to change from one model to another, or even worse, through a BIOS update. Vendors should:

1. Use devices described in the ACPI specification whenever possible.
2. If new devices or functions are introduced, document how to use them. A short specification or a request for comments (RFC) can form the basis of a new standard which follows your needs.
3. Use a unique Hardare ID to describe the device. This makes it easy for the corresponding Linux driver to match and register for the device.
4. Support mainline developers. Open source developers are often not bound to a company. Most of the drivers implemented by open source developers are for private use only; many do not fit other machines or models. Many open source developers appreciate incentives such as a trip to the next Linux symposium, a machine from their favorite hardware vendor and such, and this can be a quick, inexpensive way to get your driver into the shape you would like it to be.
5. Provide an input channel (such as a mailing list) to get feedback. This can enormously help you get informed about problems such as planned breakages in future Linux code. ACPI BIOS bugs are also likely to affect your supported Microsoft operating system, or an upcoming version also of it, and it may happen that these problems have already been analyzed in detail and debugged by open source developers. People may discuss a specific firmware bug, assisting BIOS developers in their search to identify the problem and potentially saving precious time producing an update for customers.

3 Avoid the use of the _OSI function if possible

3.1 What is _OSI and how is it used

_OSI is an ACPI method provided by the operating system that can be invoked by ACPI BIOS code. It is used by BIOS developers to detect which operating system is running. The method that should be used (cmp. ACPI spec[1], chapter 5.7.2 and 5.7.3) is _OS, but for various reasons, _OSI is used to identify recent operating systems.

The intent of the _OSI function is to identify features provided by the OS. For example some BIOSes check for Vista which supports and demands the latest ACPI backlight functions (compare ACPI spec Appendix B).

Vendors sometimes update BIOS to use _OSI to work around specific operating system problems. This is very dangerous and should always be avoided.

Here is an example of how a vendor wrongly fixed his ACPI BIOS implementation trying to work around a Linux bug. It then broke when newer kernels were fixed after the bug got identified: http://bugzilla.kernel.org/show_bug.cgi?id=7787

3.2 How _OSI is implemented on Linux

Since version 2.6.23 the mainline kernel no longer returns true for _OSI("Linux") BIOS invocations. The intent is to prevent BIOS providers and kernel developers from a maintenance nightmare. Linux specific implementations should never be needed.

The Linux kernel returns true when _OSI is invoked with any known Windows OS string (compare with drivers/acpi/utilities/uteval.c in the kernel sources for the recent list):

- "Windows 2000", /* Windows 2000 */
- "Windows 2001", /* Windows XP */
- "Windows 2001 SP1", /* Windows XP SP1 */
- "Windows 2001 SP2", /* Windows XP SP2 */
- "Windows 2001.1", /* Windows Server 2003 */
- "Windows 2001.1 SP1",
  /* Windows Server 2003 SP1 - Added 03/2006 */
- "Windows 2006", /* Windows Vista - Added 03/2006 */

According to Windows Hardware Developer Central[6], Windows operating systems behave similar and also return true for all or at least most, previous introduced, above Windows _OSI strings.

Therefore it is currently not possible for BIOS developers to identify that the machine is running on Linux.

The goal is to be compatible with the latest Microsoft operating system. It is currently tried to adopt or be compatible with the bugs of these other operating systems.

3.3 BIOS providers have to take care about _OSI on Linux

It may happen that vendors must add a BIOS hotfix which could break other supported operating systems. Vendors should be aware that adding operating system specific _OSI hooks are as dangerous regarding maintenance as any other ACPI code.

Windows Vista specific code for example, will also be processed on the latest Linux kernels returning true for Vista. Thus it may happen that you break supported Linux distributions with a Windows _OSI hook. Be aware that the code will also be processed on the next Windows operating system version.

If vendors have to add BIOS fixes for other operating systems which potentially do break supported Linux distributions, they may exclude them by checking for a distribution specific _OSI string. In general it is always a good idea to not make use of distribution specific _OSI strings. This code will not be executed in future kernel versions and should only be used in emergency cases as described above.

4 WMI - Windows Management Instrumentation

WMI is a Microsoft specific service. A small part of it describes possible ACPI WMI implementations provided by the BIOS. This is not part of the official ACPI specification and BIOS developers should avoid using it. The Linux kernel driver supports basic WMI ACPI functionality (since 2.6.25), but it is marked experimental. ACPI functionality should not depend on the WMI interface.

5 Post Video BIOS after Suspend to Ram

Graphics drivers on Linux are located in userspace, therefore they cannot initialize the graphics device in the early resume phase. There are efforts to move necessary parts of the graphics device drivers into the kernel, but this is complex and needs maturity for it to be stable on all recent graphics devices. Therefore BIOS vendors still have to provide the legacy way for resuming graphics cards and have to make sure the BIOS does "post" the video BIOS when resuming, or at least make sure the operating system can do so (by issuing "lcall $0xc000, 3"). Also, regular software interrupt calls ("int $0x10") must work during resume from suspend to ram.

6 Check ACPI operation region declarations

Sanity check ACPI operation region declarations and PNP resources. ACPI operation region declarations define the IO port, memory and other resources to control devices in BIOS through the ACPI language. PNP resource declarations are bound to an ACPI device and reserve resources to be exclusively used by an operating system driver which serves and registers this ACPI device. Sometimes several region declarations exist for the same device, or they partly overlap. This must not happen. Resources must be declared or used exclusively by either

ACPI BIOS parts or system drivers. Neither Operation Region declarations nor PNP resources may overlap.

It is expected that some hardware vendors do get ACPI BIOS parts from several external sources. ACPI BIOS templates for specific devices may be added to the BIOS. This makes it difficult for vendors to know if a device is addressed through ACPI parts themselves or whether its resources are exported to an external driver via PNP resources. If both are done, the device may be accessed through two instances without access coordination, which can lead to severe and very hard to identify system stability problems. The linuxfirmwarekit discussed below should soon be able to identify most such issues, and could be of great help to vendors to smoothly glue several ACPI parts together into one integrated, sanity checked, ACPI aware BIOS.

7 Miscellaneous

7.1 Smart Battery

The Smart Battery specification should be avoided. There were some hardware vendors, e.g. Acer, using the more complex battery specification called "Smart Battery" (compare with ACPI specification 10.1). Linux provides a driver for it, but because few BIOS implementations use it, the driver is not well tested. Instead of the Smart Battery Interface, make use of the "Control Method Batteries" interface (compare with ACPI specification 10.2).

7.2 Thermal Zones

Make use of ACPI thermal zones. Thermal control is important, and Linux can do quite a good job in this area. Provide thermal zones when you can (that will mean Linux can monitor temperatures inside the case) and provide reasonable passive trip points and polling intervals as specified by the ACPI specification. With properly set passive trip points, the machine can continue working even with a failed fan. This is very important for servers.

7.3 Always return valid values if possible

Make sure sane figures are returned for all specified values of an implemented ACPI device. For example the battery voltage is sometimes wrong or the wrong unit of current (mAh vs mW) is used. While other applications may not need these values, Linux applications could make use of them, and wrong values can be shown, or the system could even be wrongly shutdown or suspended when the remaining battery capacity is not calculated correctly.

ACPI lacks the possibility to return error values. This is a general problem for BIOS developers. When an error code path must be covered and it makes no sense to return a valid value to the OS for the invoked function, there is no possibility to tell the OS about the error. We hope this will change in the future; for now it is best to ask on the ACPI kernel developers list[6] what value would be best to return for specific problems.

8 Get used to Intel's BIOS tools

8.1 ACPICA - ACPI Component Architecture

While Microsoft uses its own proprietary, closed source ACPI compiler, Linux uses Intel's ACPI Component Architecture. The code base is used as ACPI parser and interpreter inside the kernel, and also provides a lot of easy-to-use tools for general ACPI development and stability testing.

Most important for vendors is the iasl binary which can disassemble and recompile raw ACPI tables provided by the BIOS. It uses the same code base as the ACPI parser in the kernel. Vendors should check whether their ACPI implementation sticks to the ACPI specification and works with the ACPICA tools. (For a quick, easy test, see 8.2 below.)

The Intel compiler is more strict. Warnings often lead to general ACPI BIOS errors that may also affect Microsoft Windows or other operating systems. Some may just point to ACPI specification violations which the Microsoft compiler allows. The Intel parser may also be able to cope with this code, but fixing such violations is easy in most cases and makes the ACPI BIOS implementation more robust against future operating system changes. You may get help if you are unsure whether a compiler warning is serious or how to fix it, by subscribing to and asking on the ACPICA developer mailing list.

8.2 Linuxfirmwarekit

Intel provides a tool to check the BIOS for Linux compatibility. The tool is distribution independent. A bootable CD image can be downloaded from linuxfirmwarekit.org. Once the CD image is booted, the BIOS tests are started automatically. One test is to disassemble the ACPI tables provided by the BIOS and recompile them again with Intel's iasl ACPI compiler. It may happen that there are misleading warnings. If in doubt, ask on the acpica or linuxfirmwarekit mailing list (see chapter 10).

OpenSuSE and SLES provide the same test application on their installation media. The kernel used for booting and starting the application is the same one used by the SuSE distribution. The BIOS test can be chosen in the boot loader when booting the installation media or invoked at runtime when the firmwarekit package is installed.

9 Summary

This section summarizes the above discussed topics and describes key points that vendors should take care over to ensure proper ACPI Linux support.

- Avoid the use of ACPI WMI implementations.
- Avoid _OSI workarounds whenever possible.
- If the supported Linux kernel is transparent to Windows, patch it so that it returns true for the specific OS the vendor claims to support. Only use this hook to not break the supported Linux distribution by Microsoft Windows specific bug workarounds.

- Report any Linux bugs to the linux-acpi mailing list. Fix the bug in the source code of the supported Linux distribution (ask for help, this is open source software): do not fix such bugs in the BIOS or it will fail on future, fixed Linux kernels.
- Avoid the Smart Battery ACPI interface, use the more common Control Method Batteries interface.
- Implement the ACPI specification strictly.
- Use Intel's ACPICA compiler tools to detect ACPI Source Language syntax errors.
- Use Intel's linuxfirmwarekit to detect general and known BIOS errors.

10 Useful Links and Mailing Lists

10.1 Links

1 ACPI Specification (Used for this paper: version 3.0b, 2006)
http://www.acpi.info

2 ACPI Component Architecture
http://acpica.org

3 Linuxfirmwarekit
http://linuxfirmwarekit.org

4 Linux bug workaround in BIOS via _OSI - A fix in the kernel broke it
http://bugzilla.kernel.org/show_bug.cgi?id=7787

5 Kernel bug tracking system - Report problems there if you think you hit a kernel bug
http://bugzilla.kernel.org

6 Windows Hardware Development Center How to Identify Windows Version in ACPI Using _OSI

10.2 Mailing Lists

6 ACPICA developer list
http://www.acpica.org/mailman/listinfo/devel

7 Firmwarekit Developer and Discussion List
http://www.bughost.org/mailman/listinfo/firmwarekit-discuss

8 ACPI kernel developer list Send a mail with "subscribe linux-acpi" in the body to majordomo@vger.kernel.org.
Further details can be found here: http://vger.kernel.org/majordomo-info.html

11 Credits

Thanks to all who have reviewed the paper or contributed contents to it. I only list people who did some significant work, thanks to all who sent minor corrections like a corrected link or word.

- **Andi Kleen**

  Initial review, lots of spelling corrections and technical advises.

- **Pavel Machek**

  Initial review, lots of spelling corrections. Contributed the section "Post Video BIOS after Suspend to Ram"

- **Esther Renninger**

  Thanks darling for going through this boring computer things, I love you!

- **David Newall**

  The first native speaker going through this. A full review, 544 lines diff, thanks a lot!