

[ANNOUNCE] ACPI BIOS Guideline for Linux

Source: <http://linux.derkeiler.com/Mailing-Lists/Kernel/2008-07/msg10756.html>

- *From:* Thomas Renninger <trenn@xxxxxxx>
 - *Date:* Thu, 24 Jul 2008 17:32:22 +0200
-

Hi,

while arguing about current _OSI Linux implementation, I realized this is complex and absolutely undocumented.

I started to write this up properly. This was the point when I realized adding some major general ACPI things OEMs should consider on Linux is urgently missing and should get published.

Many thanks to Andi Kleen and Pavel Machek who already reviewed an early version some weeks ago. I still changed a lot since then (also corrected a lot considering your corrections, thanks it probably was a lot work going through this that detailed!).

I like to get as much feedback for this as possible. Be sure I will go through this attentively and add or modify whatever makes sense.

I mainly try to get feedback from OEMs and BIOS developers, I am going to and I also like you to point to this. They are the main audience for this paper. Currently OEMs start to take care or at least look at Linux for the future even in the laptop area (this is why every latest ACPI BIOS checks whether Linux is running...). This should give them a documentation and an overview for what they should take care for. Until a BIOS hits the customer takes a lot time, so they need stability and a general policy they can count on.

You find the paper here:

ftp://ftp.suse.com/pub/people/trenn/ACPI_BIOS_on_Linux_guide/acpi_guideline_for_vendors.pdf

At the end are the sources of the original document for better commenting of special parts of the document.

I am looking forward for a lot feedback.

[ANNOUNCE] ACPI BIOS Guideline for Linux

Thanks,

Thomas

```
\documentclass[10pt,a4paper]{article}
```

```
\usepackage{hyperref}
```

```
\usepackage[numbers]{natbib}
```

```
\title{ACPI BIOS Guideline for Linux}
```

```
\author{Thomas Renninger – Copyright SUSE Linux GmbH, 2008}
```

```
\begin{document}
```

```
\maketitle
```

```
\begin{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. It outlines problems that may happen on ACPI driven BIOSes on Linux and how to avoid them.

```
\end{abstract}
```

```
\newpage
```

```
\tableofcontents
```

```
\newpage
```

```
\section{Introduction}
```

Recently a lot PC hardware vendors have been started to offer Linux pre-loaded on their hardware. Linux fully supports the ACPI specification version 2.0 and partly 3.0. There are still some pits vendors can fall in, which can be avoided easily. This paper describes problems that could occur with ACPI implementations on Linux.

It is intended to get input and feedback from vendors and programmers.

If you have any ideas to improve or expand this documentation, please send suggestions to <mailto:trenn@xxxxxxx> {trenn@suse.de} or to the linux-acpi mailing list [8].

```
\section{Vendor specific ACPI implementations}
```

Linux perfectly supports most ACPI specified devices (e.g. ``battery'', ``battery vs. plugged-in status'', ``lid'', ``cpufreq frequency scaling (P-states)'', ``processor sleep states (C-states)'' and a lot more).

Vendors often implement devices through ACPI which are not included in the general ACPI or other specifications. For example buttons like wireless LAN on/off switches, volume up/down and mute buttons and more.

[ANNOUNCE] ACPI BIOS Guideline for Linux

Vendors provide a proprietary driver for Windows for such vendor specific devices and in many cases there also is a re-engineered Linux driver (e.g. asus_acpi, sony_acpi, thinkpad_acpi and more). Those drivers are often not complete and hard to maintain, so 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:

```
\begin{enumerate}
```

```
\item
```

Use devices described in the ACPI specification whenever possible.

```
\item
```

If new devices or functions are introduced, document how to use them.

A short specification or a request for comments (RFC) can be the basis of a new standard which is following your needs.

Also do make use of a unique Hardware ID to describe the device and thus make it easy for the corresponding Linux driver to match and register for the device.

```
\item
```

Support mainline developers. Open source developers are often not bound to a company. Most of the drivers implemented by open source developers are made for private use only, many of them do not fit other machines or models.

Incentives like a trip to the next Linux symposium, a machine of their favorite hardware vendor or something like that is often a quick and cheap option to get the driver into a shape you would like to see it.

```
\item
```

Provide an input channel (a mailinglist for example) to get feedback. This can be of big help to get informed about problems like breakage in upcoming Linux code. ACPI BIOS bugs likely affect your supported Microsoft operating system or an upcoming version also. It may happen that problems are already analyzed and debugged very detailed by open source developers. People may already point to a specific firmware bug which can ease up the search for BIOS developers to identify the problem and can save precious time until an update can be provided to the customer.

```
\end{enumerate}
```

```
\section{Avoid the use of the \_OSI function if possible}
```

```
\subsection{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 to check for recent operating systems _OSI is used to identify the running OS for various reasons.

[ANNOUNCE] ACPI BIOS Guideline for Linux

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).

But feature identification is an exception. In reality BIOS developers use the `_OSI` functionality to work around operating system errors. If the supported OS is a closed source one like Microsoft Windows, vendors rely on the use of `_OSI` to work around possibly upcoming OS specific errors (for example through a Service Pack) via a BIOS update.

Vendors must not do this on Linux. If bugs in the Linux OS are identified, vendors must make sure that the cause is fixed in the Linux kernel. Linux does not have a strict OS versioning. The root cause might get fixed in the latest upstream kernel and the workaround could break there. Such fixes are often backported to distributions. Therefore an identification (as it is common for Windows) whether the underlying Linux operating system is affected by a specific bug is not possible.

Here is an example of how a vendor wrongly fixed his ACPI BIOS implementation and tried to workaround a Linux bug. It then broke with more recent kernels after the bug got identified and fixed:

http://bugzilla.kernel.org/show_bug.cgi?id=7787

\subsection{How `_OSI` is implemented on Linux}

Since version 2.6.23 the mainline kernel does not return true for `_OSI("Linux")` BIOS invocations anymore.

The intent is to prevent BIOS providers and kernel developers from a maintenance nightmare. Linux specific implementations should never be needed.

The second ACPI specification violation is that the Linux kernel returns true for all known Windows OS `_OSI` strings (compare with `drivers/acpi/utilities/uteval.c` in the kernel sources for the recent list):

```
\begin{itemize}
\item
``Windows 2000", /* Windows 2000 */
\item
``Windows 2001", /* Windows XP */
\item
``Windows 2001 SP1", /* Windows XP SP1 */
\item
``Windows 2001 SP2", /* Windows XP SP2 */
\item
``Windows 2001.1", /* Windows Server 2003 */
\item
``Windows 2001.1 SP1",
```

[ANNOUNCE] ACPI BIOS Guideline for Linux

/* Windows Server 2003 SP1 – Added 03/2006 */

\item

``Windows 2006", /* Windows Vista – Added 03/2006 */

\end{itemize}

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

The idea is to be compatible with the latest Microsoft operating system. If people report bugs which come out to be operating system specific bug workarounds, it is currently tried to adopt or better, be compatible with these bugs of other operating systems.

There are several drawbacks with this implementation:

\begin{itemize}

\item

Microsoft versions will change and can be fixed in future releases (this only applies for `_OSI` specific workarounds) while the Linux implementation has to be compatible with all bugs that ever appeared in any Microsoft ACPI implementation.

\item

Microsoft does return true for only one specific version string. This can lead to undefined code paths being executed on Linux where for all known Windows strings true is returned. For example the BIOS checks at initialization time for Windows XP and sets a specific variable, then it checks for Windows Vista and sets another variable.

Now later ACPI code paths executed on Linux will be unwanted as Windows is probably only returning true for either Vista or XP.

Vendors should check with else if clauses or use the same variable to store the results of `_OSI` execution at initialisation time.

Make sure the latest Windows versions are tested in the end and the latest Windows version (e.g. ``Windows 2006", cmp. with above strings) returning true is used.

\item

It is nearly impossible for vendors to follow all kernel versions in mainline and in distributions and to check what Windows strings are returned in which kernel version.

\item

Vendors who care about Linux cannot guarantee proper support of Linux and Windows with the same BIOS without modifying the kernel slightly(see below for details). Especially if they are forced to implement Microsoft erratas through BIOS hot-fixes. Whether this is the case cannot be proven, but comparing a lot recent ACPI BIOS implementations harden the assumption that this is rather common.

\end{itemize}

\subsection{BIOS providers have to take care about `_OSI` on Linux }

[ANNOUNCE] ACPI BIOS Guideline for Linux

It is expected that vendors must add `_OSI` hooks to workaround bugs in other operating systems. If this is not the case, the use of `_OSI` should simply be avoided and everything works out fine.

On Linux, vendors must (instead of adding workarounds into the ACPI BIOS) fix the Linux kernel or at least discuss and help to identify and fix the problem for the latest mainline kernel. Then the needed necessary code patches which fix the problem for the latest Linux kernel must be backported to the kernels of the supported distributions. Like that vendors can expect support from kernel developers and a quick solution for the problem and they can make sure they do not run into unmanageable maintenance problems (which are very likely to happen if Linux kernel bugs get fixed in the BIOS).

Unfortunately vendors seem to depend on the `_OSI` functionality and it looks like as they have to provide OS specific BIOS hot-fixes to guarantee support for Microsoft operating systems. Because the Linux `_OSI` implementation is currently transparent to the Windows one, BIOS developers cannot distinguish whether their code is running on Linux or on a Microsoft OS. Hopefully this will change soon, but it needs a testing phase and will take some time until such a change find its way into distributions. Goal for Linux is to not return true for Windows OS strings.

The `Linux` OS keyword as stated in the ACPI specification ([1], cmp. with chapter 5.7.2) already is and will stay obsolete in the future. It got removed in kernel version 2.6.23. Vendors should not rely on Linux returning true for this string.

For now, while Linux still is transparent to Microsoft Windows, vendors have to patch the kernel of the distribution they claim to support. (For example `Linux` string is used for SUSE Linux Enterprise Server and Desktop version 11).

The above described `os="Supported Linux Distribution"` string must only be used to not execute operating system bug workarounds on a Linux system. If for example a BIOS hotfix is required for Vista SP2, the Linux kernel currently might also execute this code (depending on whether the OS string has already been added to the kernel). In this case add the condition not to execute this BIOS hotfix on your supported Linux distributions to prevent them from breaking through the BIOS update.

It is very important that vendors do not mis-use the ability to distinguish between Linux and Microsoft Windows and workaround Linux kernel bugs. Once a problem is identified to be a kernel bug, report the problem on the linux-acpi mailing list (see links at the end). ACPI kernel developers will provide and commit a fix for the next mainline kernel. This fix must then eventually be backported to the kernel of the supported distributions (if not already done).

Vendors must still make sure that their BIOS runs fine, even if `_OSI("Windows XY")` calls do not return true.

The problems outlined above show that for long-term, the only maintainable solution for Linux is to not return true for Windows OS strings. While such an interface change needs testing and some time to stabilize, BIOS developers should keep in mind that this change could happen in the future.

`\section{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.

`\section{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 to run stable on all recent graphics devices.

Therefore the BIOS vendors still have to provide the legacy way of graphics cards resuming 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(0, 3)`). Also regular software interrupt calls (`_int(0x10)`) must work during resume from suspend to ram.

`\section{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 bind to an ACPI device and reserve resources to exclusively be used by an operating system driver which serves and registers for this ACPI device.

Sometimes there exist several region declarations 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 nor both may overlap.

It is expected that some hardware vendors do get ACPI BIOS parts from several external sources. ACPI BIOS templates for specific devcies may be added to the BIOS. This makes it difficult for vendors to keep an

[ANNOUNCE] ACPI BIOS Guideline for Linux

overview whether a device is addressed through ACPI parts themselves or whether its resources are already be exported to an external driver via PNP resources. If both is done, the device may be accessed through two instances without any access coordination, which can lead to sever and very hard to identify system instability problems.

The linuxfirmwarekit discussed below should be able to identify most of such issues soon and could be of great help for vendors to smoothly glue several ACPI parts together into one integrated, sanity checked, ACPI aware BIOS.

`\section{Miscellaneous}`

`\subsection{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 there were not much BIOS implementations using 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).

`\subsection{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 case) and provide reasonable passive trip points and polling intervals as specified by the ACPI specification. With properly set up passive trip points, the machine can continue working even with failed fan. This is very important for servers.

`\subsection{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 entity for the current (mAh vs mW) are mixed up. While other applications may not need these values, Linux applications could make use of them and show wrong values or even shutdown or suspend the system wrongly 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. If an error code path has to be covered and it does not make sense to return a valid value to the OS for the invoked function, then there is no possibility to tell the OS about the error. Hopefully 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 the specific problem.

\section{Get used to Intel's BIOS tools}

\subsection{ACPICA – ACPI Component Architecture}

\label{acpica}

While Microsoft uses its own proprietary, closed source ACPI compiler, Linux does make use of Intel's \href{<http://acpica.org>} {ACPI Component Architecture}. The code base is used as ACPI parser and interpreter inside the kernel, but it 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. Therefore vendors should check whether their ACPI implementation sticks to the ACPI specification and works with the ACPICA tools (for an easy quick test, see \ref{linuxfirmwarekit} 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. If in doubt whether a compiler warning is serious and how to fix it, you may get help if you subscribe and ask on the \href{<http://www.acpica.org/mailman/listinfo/devel>} {ACPICA developer mailing list}.

\subsection{Linuxfirmwarekit}

\label{linuxfirmwarekit}

Intel provides a tool to check the BIOS for Linux compatibility.

The tool is distribution independent.

A \href{<http://www.linuxfirmwarekit.org/download.php#bootcd>} {bootable CD image} can be downloaded from linuxfirmware.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 \ref{links}).

OpenSuSE and SLES provide the same test application on their installation media. But the kernel used for booting and starting the application is the same which is 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.

[ANNOUNCE] ACPI BIOS Guideline for Linux

\section{Summary}

This section summarizes above discussed topics and shortly describes keynotes, vendors should take care for to be able to ensure proper ACPI Linux support.

\begin{itemize}

\item

Avoid the use of ACPI WMI implementations.

\item

Avoid _OSI workarounds whenever possible.

\item

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.

\item

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 you end up in not being able to support future fixed Linux kernels with this BIOS.

\item

Avoid the Smart Battery ACPI interface, use the more common Control Method Batteries interface.

\item

Strictly implement the ACPI specification.

\item

Use Intel's ACPICA compiler tools to detect ACPI Source Language syntax errors.

\item

Use Intel's linuxfirmwarekit to detect general and already known BIOS errors.

\end{itemize}

\section{Useful Links and Mailing Lists}

\label{links}

\subsection{Links}

\begin{itemize}

\item

[1] ACPI Specification (Used for this paper: version 3.0b, 2006)

\href{<http://www.acpi.info>} {<http://www.acpi.info>}

\item

[2] ACPI Component Architecture

\href{<http://acpica.org>} {<http://acpica.org>}

\item

[3] Linuxfirmwarekit

[ANNOUNCE] ACPI BIOS Guideline for Linux

<http://linuxfirmwarekit.org> {<http://linuxfirmwarekit.org>}

\item

[4] Linux bug workaround in BIOS via _OSI – A fix in the kernel broke it

http://bugzilla.kernel.org/show_bug.cgi?id=7787 {http://bugzilla.kernel.org/show_bug.cgi?id=7787}

\item

[5] Kernel bug tracking system – Report problems there if you think you hit a kernel bug

<http://bugzilla.kernel.org> {<http://bugzilla.kernel.org>}

\end{itemize}

\subsection{Mailing Lists}

\begin{itemize}

\item

[6] ACPICA developer list

<http://www.acpica.org/mailman/listinfo/devel> {<http://www.acpica.org/mailman/listinfo/devel>}

\item

[7] Firmwarekit Developer and Discussion List

<http://www.bughost.org/mailman/listinfo/firmwarekit-discuss>

{<http://www.bughost.org/mailman/listinfo/firmwarekit-discuss>}

\item

[8] ACPI kernel developer list

<mailto:linux-acpi@vger.kernel.org>

{<mailto:linux-acpi@vger.kernel.org>}

\end{itemize}

\Large

Disclaimer:

\normalsize

Trademarks and trade names may be used in this document to refer to either the entities claiming the marks and names or their products. The author of this document disclaims any proprietary interest in trademarks and trade names other than its own.

\end{document}

—
To unsubscribe from this list: send the line "unsubscribe linux-kernel" in the body of a message to majordomo@xxxxxxxxxxxxxxxxxx

More majordomo info at <http://vger.kernel.org/majordomo-info.html>

Please read the FAQ at <http://www.tux.org/lkml/>