

# [RFC] dev\_acpi: device driver for userspace access to ACPI

**Source:** <http://linux.derkeiler.com/Mailing-Lists/Kernel/2004-08/0587.html>

---

**From:** Alex Williamson ([alex.williamson\\_at\\_hp.com](mailto:alex.williamson_at_hp.com))

**Date:** 08/03/04

To: [acpi-devel@lists.sourceforge.net](mailto:acpi-devel@lists.sourceforge.net)  
Date: Tue, 03 Aug 2004 11:00:26 -0600

This is by no means ready for release, but I wanted to get a sanity check. I'm still stuck on this idea that userspace needs access to ACPI namespace. Manageability apps might use this taking inventory of devices not exposed by other means, things like X can locate chipset components that don't live in PCI space, there's even the possibility of making user space drivers.

Populating the sysfs tree didn't seem to generate as much interest as I'd hoped and I don't think it kept with the spirit of sysfs very well. So, now I present dev\_acpi (name suggestions welcome). The link below is a tarball with a first stab at the driver as well as a simple proof of concept application. It should build against any 2.6 kernel as long as you have the include files available. There are no kernel changes required, thus it doesn't expose anything not already exposed as a symbol.

The basic concept of operation is that the ioctl operates on the ACPI path passed into the ioctl call. The ioctl may return the result of the operation either in the status field of the argument or use that to indicate the number of bytes available to read(2) for the result. The header file included describes the input and output for each operation. If the status field indicates a byte count to read, the calling application can easily size buffers, and call read(2) on the device file to get the results. I've also include support for write(2) that could allow writing arguments for method calls that take input (completely untested). I've limited some of the output (for instance in GET\_NEXT) to try to only print out standard ACPI objects, but the filter is pretty simple (objects beginning w/ '\_'). I know the completely open interface from the sysfs implementation scared some people. Non-standard objects can still be operated on, but you've got to know what to look for.

Many of the ioctls mimic the behavior of the acpi calls that are already exported. What I have now is only a start at what could be provided. The sample, proof-of-concept app, is called acpitree. It's much like the tree app for listing files and directories. It does

## Linux-Kernel: [RFC] dev\_acpi: device driver for userspace access to ACPI

evaluate and print \_HIDs represented by integers, but that's about it.  
Here's some sample output (sorry anyone not using fixed width fonts):

>From an rx4640 ia64 server:

```
\\
|-- _GPE
|-- _PR_
|-- _SB_
| |-- SBA0
| | |-- _HID (HWP0001)
| | |-- _CID
| | |-- _CRS
| | |-- _INI
| | |-- _UID
| | |-- MIO_
| | | |-- _HID (IPI0001)
| | | |-- _UID
| | | |-- _STA
| | | `-- _CRS
| | |-- PCIO
| | |-- _UID
| | |-- _STA
| | |-- _BBN
| | |-- _HID (HWP0002)
| | |-- _CID
| | |-- _PRT
...

```

>From an nc6000 laptop:

```
\\
|-- _GPE
|-- _PR_
|-- _SB_
| |-- _INI
| |-- C00C
| | |-- _HID (PNP0C01)
| | `-- _CRS
| |-- C046
| | |-- _HID (PNP0A03)
| | |-- _ADR
| | |-- C047
| | | |-- _ADR
| | | |-- C0D1
| | | | |-- _ADR
| | | | |-- _REG
| | | | |-- _S3D
| | | | |-- _S4D
| | | | |-- _DOS
| | | | |-- C0DD
| | | | |-- _ADR
...

```

## Linux-Kernel: [RFC] dev\_acpi: device driver for userspace access to ACPI

You can find the driver and sample app here:

[http://free.linux.hp.com/~awilliam/acpi/dev\\_acpi/dev\\_acpi-20040803.tar.bz2](http://free.linux.hp.com/~awilliam/acpi/dev_acpi/dev_acpi-20040803.tar.bz2)

There's a brutally short README there. Caveat: the driver is hardcoded to use an experimental major number, you'll have to mknod it, see the README.

Please try it out, let me know if it sucks. I make no guarantees it won't kill your system, but it shouldn't unless you start evaluating dangerous objects (ie, if you don't know what it does, don't do it). And of course, if you have any suggestions, I welcome feedback. Thanks,

Alex

--

Alex Williamson

HP Linux & Open Source Lab

-

To unsubscribe from this list: send the line "unsubscribe linux-kernel" in the body of a message to [majordomo@vger.kernel.org](mailto:majordomo@vger.kernel.org)

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

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