



# Fixing USB Autosuspend

Serge Y. Strobandt

Copyright 2015–2017, licensed under [Creative Commons BY-NC-SA](#)

## Problem



The Linux kernel automatically suspends USB devices when there is driver support and the devices are not in use.<sup>1</sup> This saves quite a bit of power. However, some USB devices are not compatible with USB autosuspend and will misbehave at some point. Affected devices are most commonly USB mice and keyboards.<sup>2</sup> Personally, I had only one Microsoft™ *Wheel Mouse Optical* affected, despite owning other models of the same brand.

In essence, this is not really a USB hardware problem, but perhaps more a Linux problem. The actual fault lies with a misinterpretation of the [eXtensible Host Controller Interface \(xHCI\)](#) specification. This issue previously did not exist with the older [Enhanced Host Controller Interface \(EHCI\)](#) specification. A «[Sharp](#)» [explanation](#) is available online.<sup>3</sup>

The Linux kernel patch for this problem will probably one day automatically trickle in downstream and onto my affected Xubuntu LTS 14.04 system. Nonetheless, with a crashing computer mouse at hand, things cannot wait. However, patching my current 3.13.0-35-generic x86\_64 kernel is out of the question! We will rather grab this opportunity to learn a bit about writing rules for `udev`, the device manager for the Linux kernel. I am writing “for the Linux kernel”, because `udev` executes entirely in [user space](#).

## Identify the device

First, one needs to properly identify the affected USB device by its vendor and product ID. Run the following command and look for the device description.

```
$ sudo lsusb -v
```

We take note of the Bus and Device numbers —yours will be different!— as well as the idProduct and idVendor hex numbers.

```
...
Bus 008 Device 002: ID 045e:0040 Microsoft Corp. Wheel Mouse Optical
Device Descriptor:
  bLength                18
  bDescriptorType        1
  bcdUSB                  1.10
  bDeviceClass            0 (Defined at Interface control)
  bDeviceSubClass        0
  bDeviceProtocol        0
  bMaxPacketSize0       8
  idVendor                0x045e Microsoft Corp.
  idProduct              0x0040 Wheel Mouse Optical
  bcdDevice              3.00
  iManufacturer          1 Microsoft
  iProduct               3 Microsoft 3-Button Mouse with IntelliEye(TM)
  iSerial                0
  bNumConfigurations    1
...
```

## Test

If the affected system is a laptop computer, first eliminate `laptop-mode-tools`, `powertop` and similar tools as a possible cause of the problem. Since my affected system is a desktop, I assured myself that these tools are not installed.

Once that is cleared off, a simple test suffices to know whether the suspect USB device is really affected by autosuspend. Beware, kernel device numbering is different from `lsusb`. In the `/sys/bus/usb/devices/usb8/` subdirectory, device numbering starts with `8-0`. Hence, the second device listed by `lsusb` is here actually called `8-1`. In case of doubt, check the values of `idVendor` and `idProduct`.

```
$ cat /sys/bus/usb/devices/usb8/8-1/idVendor
045e

$ cat /sys/bus/usb/devices/usb8/8-1/idProduct
0040
```

If the device driver supports it, the USB `power/control` attribute will default to `auto`. There may also be a file named `level`. As of the 2.6.35 kernel, this file is depreciated and now replaced by `control`.<sup>1</sup>

```
$ cat /sys/bus/usb/devices/usb8/8-1/power/control
auto
```

The `auto` state is the normal state in which the kernel is allowed to autosuspend and autoresume the device. In kernels up to 2.6.32, one could also specify `suspend`, meaning that the device should remain suspended and autoresume was not allowed. This setting is no longer supported.<sup>1</sup>

In a command terminal, enter now the following command:

```
$ echo 'on' |sudo tee /sys/bus/usb/devices/usb8/8-1/power/control
```

The `on` state means device autosuspend is not allowed. Of course, system suspends are still allowed. If, **without rebooting**, your device now works flawlessly, USB autosuspend is the problem at hand. However, above command only works temporarily. More needs to be done to solve the problem indefinitely.

## Permanent solution

A permanent solution can only be achieved by creating a device-specific rules file for `udev`, the device manager for the Linux kernel:

```
$ sudo $EDITOR /etc/udev/rules.d/usb-power.rules
```

Please, note: While `lsusb` reports the hex vendor and product id with a `0x` prefix, **the syntax for `udev` rules does *not* allow the leading `0x`**, even though the id numbers are still specified in hex.<sup>4</sup> My `usb-power.rules` file contains only one comment line and one line of code:

```
# Microsoft Corp. Wheel Mouse Optical
ACTION=="add", SUBSYSTEM=="usb", ATTR{idVendor}=="045e",
ATTR{idProduct}=="0040", TEST=="power/control", ATTR{power/control}="on"
```

## Reboot & check

Finally, reboot your machine and check whether the pertaining `power/control` attribute remained set to `on`.

```
$ cat /sys/bus/usb/devices/usb8/8-1/power/control
on
```

# References

1. Alan Stern. Power management for USB. Published 2014.  
<https://www.kernel.org/doc/Documentation/usb/power-management.txt>
2. ArchWiki. Power saving. [https://wiki.archlinux.org/index.php/Power\\_saving#USB\\_autosuspend](https://wiki.archlinux.org/index.php/Power_saving#USB_autosuspend)
3. Sarah Sharp. Update: Looks like this is an xHCI specific issue, and probably not the cause of the USB device disconnects under EHCI. Published 2013.  
<https://plus.google.com/+SarahSharp/posts/RZpndv4BCCD>
4. Jeff Norden. Personal communication. Published online 2017.



This work is licensed under a [Creative Commons Attribution-NonCommercial-ShareAlike 4.0 International License](#).

Other licensing available on request.

Unattended [CSS](#) typesetting with [Prince](#).

This work is published at <https://hamwaves.com/usb.autosuspend/en/>.

Last update: Monday, March 1, 2021.