Red Hat Enterprise Linux Release 9.2 Manual Pages on 'lspci.8' command

$ man lspci.8

lspci(8)   The PCI Utilities   lspci(8)

NAME

    lspci - list all PCI devices

SYNOPSIS

    lspci [options]

DESCRIPTION

    lspci is a utility for displaying information about PCI buses in the

    system and devices connected to them.

    By default, it shows a brief list of devices. Use the options described

    below to request either a more verbose output or output intended for

    parsing by other programs.

    If you are going to report bugs in PCI device drivers or in lspci it?

    self, please include output of "lspci -vvx" or even better "lspci

    -vvxxx" (however, see below for possible caveats).

    Some parts of the output, especially in the highly verbose modes, are

    probably intelligible only to experienced PCI hackers. For exact defi?

    nitions of the fields, please consult either the PCI specifications or

    the header.h and /usr/include/linux/pci.h include files.

    Access to some parts of the PCI configuration space is restricted to

    root on many operating systems, so the features of lspci available to

    normal users are limited. However, lspci tries its best to display as

    much as available and mark all other information with <access denied>

    text.

OPTIONS

  Basic display modes

    -m   Dump PCI device data in a backward-compatible machine readable form. See below for details.

    -mm   Dump PCI device data in a machine readable form for easy parsing by scripts. See below for details.

    -t   Show a tree-like diagram containing all buses, bridges, devices and connections between them.

  Display options

    -v   Be verbose and display detailed information about all devices.

    -vv   Be very verbose and display more details. This level includes everything deemed useful.

    -vvv Be even more verbose and display everything we are able to parse, even if it doesn't look interesting at all (e.g., unde? fined memory regions).

    -k   Show kernel drivers handling each device and also kernel modules capable of handling it. Turned on by default when -v is given in the normal mode of output. (Currently works only on Linux with kernel 2.6 or newer.)

    -x   Show hexadecimal dump of the standard part of the configuration space (the first 64 bytes or 128 bytes for CardBus bridges).

    -xxx Show hexadecimal dump of the whole PCI configuration space. It is available only to root as several PCI devices crash when you try to read some parts of the config space (this behavior proba? bly doesn't violate the PCI standard, but it's at least very stupid). However, such devices are rare, so you needn't worry much.

    -xxxx Show hexadecimal dump of the extended (4096-byte) PCI configura? tion space available on PCI-X 2.0 and PCI Express buses.

    -b   Bus-centric view. Show all IRQ numbers and addresses as seen by the cards on the PCI bus instead of as seen by the kernel.

    -D   Always show PCI domain numbers. By default, lspci suppresses them on machines which have only domain 0.

-P   Identify PCI devices by path through each bridge, instead of by

      bus number.

-PP   Identify PCI devices by path through each bridge, showing the

      bus number as well as the device number.

Options to control resolving ID's to names

-n   Show PCI vendor and device codes as numbers instead of looking

      them up in the PCI ID list.

-nn   Show PCI vendor and device codes as both numbers and names.

-q   Use DNS to query the central PCI ID database if a device is not

      found in the local pci.ids file. If the DNS query succeeds, the

      result is cached in ~/.pciids-cache and it is recognized in sub?

      sequent runs even if -q is not given any more. Please use this

      switch inside automated scripts only with caution to avoid over?

      loading the database servers.

-qq   Same as -q, but the local cache is reset.

-Q   Query the central database even for entries which are recognized

      locally. Use this if you suspect that the displayed entry is

      wrong.

Options for selection of devices

-s [[[[<domain>]:]<bus>]:][<device>][.[<func>]]

      Show only devices in the specified domain (in case your machine

      has several host bridges, they can either share a common bus

      number space or each of them can address a PCI domain of its

      own; domains are numbered from 0 to ffff), bus (0 to ff), device

      (0 to 1f) and function (0 to 7). Each component of the device

      address can be omitted or set to "*", both meaning "any value".

      All numbers are hexadecimal. E.g., "0:" means all devices on

      bus 0, "0" means all functions of device 0 on any bus, "0.3" se?

      lects third function of device 0 on all buses and ".4" shows

      only the fourth function of each device.

-d [<vendor>]:[<device>][:<class>]

      Show only devices with specified vendor, device and class ID.

      The ID's are given in hexadecimal and may be omitted or given as

"*", both meaning "any value".

Other options

  -i <file>

      Use <file> as the PCI ID list instead of /usr/share/hw?

      data/pci.ids.

  -p <file>

      Use <file> as the map of PCI ID's handled by kernel modules. By

      default, lspci uses /lib/modules/kernel_version/modules.pcimap.

      Applies only to Linux systems with recent enough module tools.

  -M   Invoke bus mapping mode which performs a thorough scan of all

      PCI devices, including those behind misconfigured bridges, etc.

      This option gives meaningful results only with a direct hardware

      access mode, which usually requires root privileges.   Please

      note that the bus mapper only scans PCI domain 0.

  --version

      Shows lspci version. This option should be used stand-alone.

PCI access options

  The PCI utilities use the PCI library to talk to PCI devices (see

  pcilib(7) for details). You can use the following options to influence

  its behavior:

  -A <method>

      The library supports a variety of methods to access the PCI

      hardware. By default, it uses the first access method avail?

      able, but you can use this option to override this decision. See

      -A help for a list of available methods and their descriptions.

  -O <param>=<value>

      The behavior of the library is controlled by several named pa?

      rameters. This option allows to set the value of any of the pa?

      rameters. Use -O help for a list of known parameters and their

      default values.

  -H1   Use direct hardware access via Intel configuration mechanism 1.

      (This is a shorthand for -A intel-conf1.)

  -H2   Use direct hardware access via Intel configuration mechanism 2.

(This is a shorthand for -A intel-conf2.)

-F <file>

Instead of accessing real hardware, read the list of devices and values of their configuration registers from the given file pro? duced by an earlier run of lspci -x. This is very useful for analysis of user-supplied bug reports, because you can display the hardware configuration in any way you want without disturb? ing the user with requests for more dumps.

-G   Increase debug level of the library.

MACHINE READABLE OUTPUT

If you intend to process the output of lspci automatically, please use one of the machine-readable output formats (-m, -vm, -vmm) described in this section. All other formats are likely to change between versions of lspci.

All numbers are always printed in hexadecimal. If you want to process numeric ID's instead of names, please add the -n switch.

Simple format (-m)

In the simple format, each device is described on a single line, which is formatted as parameters suitable for passing to a shell script, i.e., values separated by whitespaces, quoted and escaped if necessary. Some of the arguments are positional: slot, class, vendor name, device name, subsystem vendor name and subsystem name (the last two are empty if the device has no subsystem); the remaining arguments are option-like:

-rrev Revision number.

-pprogif

Programming interface.

The relative order of positional arguments and options is undefined. New options can be added in future versions, but they will always have a single argument not separated from the option by any spaces, so they can be easily ignored if not recognized.

Verbose format (-vmm)

The verbose output is a sequence of records separated by blank lines.

Each record describes a single device by a sequence of lines, each line containing a single `tag: value' pair. The tag and the value are sepa‐ rated by a single tab character. Neither the records nor the lines within a record are in any particular order. Tags are case-sensitive.

The following tags are defined:

Slot	The name of the slot where the device resides ([domain:]bus:de‐ vice.function). This tag is always the first in a record.

Class	Name of the class.

Vendor Name of the vendor.

Device Name of the device.

SVendor

Name of the subsystem vendor (optional).

SDevice

Name of the subsystem (optional).

PhySlot

The physical slot where the device resides (optional, Linux only).

Rev	Revision number (optional).

ProgIf Programming interface (optional).

Driver Kernel driver currently handling the device (optional, Linux only).

Module Kernel module reporting that it is capable of handling the de‐ vice (optional, Linux only). Multiple lines with this tag can occur.

NUMANode

NUMA node this device is connected to (optional, Linux only).

IOMMUGroup

IOMMU group that this device is part of (optional, Linux only).

New tags can be added in future versions, so you should silently ignore any tags you don't recognize.

Backward-compatible verbose format (-vm)

In this mode, lspci tries to be perfectly compatible with its old ver‐ sions. It's almost the same as the regular verbose format, but the De‐

vice tag is used for both the slot and the device name, so it occurs twice in a single record. Please avoid using this format in any new code.

FILES

/usr/share/hwdata/pci.ids

A list of all known PCI ID's (vendors, devices, classes and sub? classes). Maintained at https://pci-ids.ucw.cz/, use the update-pciids utility to download the most recent version.

/usr/share/hwdata/pci.ids.gz

If lspci is compiled with support for compression, this file is tried before pci.ids.

~/.pciids-cache

All ID's found in the DNS query mode are cached in this file.

BUGS

Sometimes, lspci is not able to decode the configuration registers com? pletely. This usually happens when not enough documentation was avail? able to the authors. In such cases, it at least prints the <?> mark to signal that there is potentially something more to say. If you know the details, patches will be of course welcome.

Access to the extended configuration space is currently supported only by the linux_sysfs back-end.

SEE ALSO

setpci(8), pci.ids(5), update-pciids(8), pcilib(7)

AUTHOR

The PCI Utilities are maintained by Martin Mares <mj@ucw.cz>.

pciutils-3.7.0   31 May 2020   lspci(8)