NAME

`cpucontrol` — control utility for the `cpuctl`(4) device

SYNOPSIS

`cpucontrol` [`-v`] `-m` *msr device*
`cpucontrol` [`-v`] `-m` *msr=value device*
`cpucontrol` [`-v`] `-m` *msr&=mask device*
`cpucontrol` [`-v`] `-m` *msr|=mask device*
`cpucontrol` [`-v`] `-i` *level device*
`cpucontrol` [`-v`] `-i` *level,level_type device*
`cpucontrol` [`-vn`] [`-d` *datadir*] `-u` *device*
`cpucontrol -e` *device*

DESCRIPTION

The `cpucontrol` utility can be used to read and write arbitrary machine-specific CPU registers via the `cpuctl`(4) special device. It can also be used to apply CPU firmware updates.

The following options are available:

`-d` *datadir*
: Directory paths where to look for microcode images. The option can be specified multiple times. The paths are added in order of the options appearance on the command line, default directories are appended after the user-supplied paths.

`-n`
: Do not look for the microcode images in the standard directories. Currently standard directory to look for the microcode update files is `/usr/local/share/cpucontrol`.

`-m` *msr*
: Show value of the specified MSR. MSR register number should be given as a hexadecimal number. The high word is printed first, then the low word is printed second.

`-m` *msr=value*
: Store the *value* in the specified MSR register. The *value* argument can be prefixed with ~ operator. In this case the inverted value of argument will be stored in the register.

`-m` *msr&=mask*
: Store the result of bitwise AND operation between *mask* and the current MSR value in the MSR register. The *mask* argument can be prefixed with ~ operator. In this case the inverted value of mask will be used.

`-m` *msr|=mask*
: Store the result of bitwise OR operation between *mask* and the current MSR value in the MSR register. The *mask* argument can be prefixed with ~ operator. In this case the inverted value of mask will be used.

`-i` *level*
: Retrieve CPUID info. Level should be given as a hex number.

`-i` *level,level_type*
: Retrieve CPUID info. Level and level_type should be given as hex numbers.

`-u`
: Apply CPU firmware updates. The `cpucontrol` utility will walk through the configured data directories and apply all firmware updates available for this CPU.

`-e`
: Re-evaluate the kernel flags indicating the present CPU features. This command is typically executed after a firmware update was applied which changes information reported by the `CPUID` instruction.

Only execute the `-e` command after the microcode update was applied to all CPUs in the system. The kernel does not operate correctly if the features of processors are not identical.

`-v` Increase the verbosity level.

`-h` Show help message.

EXIT STATUS

The `cpucontrol` utility exits 0 on success, and >0 if an error occurs.

EXAMPLES

The command

"`cpucontrol -m 0x10 /dev/cpuctl0`"

will read the contents of TSC MSR from CPU 0.

To set the CPU 0 TSC MSR register value to 0x1 issue

"`cpucontrol -m 0x10=0x1 /dev/cpuctl0`".

The following command will clear the second bit of TSC register:

"`cpucontrol -m 0x10&=~0x02 /dev/cpuctl0`".

The following command will set the forth and second bit of TSC register:

"`cpucontrol -m 0x10|=0x0a /dev/cpuctl0`".

The command

"`cpucontrol -i 0x1 /dev/cpuctl1`"

will retrieve the CPUID level 0x1 from CPU 1.

To perform firmware updates on CPU 0 from images located at `/usr/local/share/cpuctl` use the following command:

"`cpucontrol -nd /usr/local/share/cpuctl -u /dev/cpuctl0`"

SEE ALSO

`cpuctl`(4)

HISTORY

The `cpucontrol` utility first appeared in FreeBSD 7.2.

AUTHORS

The `cpucontrol` utility and this manual page was written by Stanislav Sedov <`stas@FreeBSD.org`>.