NAME

perfmon - CPU performance-monitoring interface

SYNOPSIS

cpu I586_CPU
cpu I686_CPU
options PERFMON

DESCRIPTION

The **perfmon** driver provides access to the internal performance-monitoring capabilities of the Intel Pentium and Pentium Pro CPUs. These processors implement two internal counters which can be configured to measure a variety of events for either count or duration (in CPU cycles), as well as a cycle counter which counts clock cycles. The **perfmon** driver provides a device-style interface to these capabilities.

All access to the performance-monitoring counters is performed through the special device file "*/dev/perfmon*". This device supports a number of ioctl(2) requests, defined in <*machine/perfmon.h*> along with the definitions of the various counters for both Pentium and Pentium Pro processors.

NOTA BENE: The set of available events differs from processor to processor. It is the responsibility of the programmer to ensure that the event numbers used are the correct ones for the CPU type being measured.

The following ioctl(2) requests are defined:

PMIOSETUP (struct pmc) Set up a counter with parameters and flags defined in the structure. The following fields are defined in struct pmc:

int pmc_num	the number of the counter in question; must be less than NPMC (currently 2).
u_char pmc_event	the particular event number to be monitored, as defined in <*machine/perfmon.h*>.
u_char pmc_unit	the unit mask value, specific to the event type (see the Intel documentation).
u_char pmc_flags	flags modifying the operation of the counter (see below).
u_char pmc_mask	the counter mask value; essentially, this is a threshold used to

restrict the count to events lasting more (or less) than the specified number of clocks.

The following pmc_flags values are defined:

PMCF_USR	count events in user mode
PMCF_OS	count events in kernel mode
PMCF_E	count number of events rather than their duration
PMCF_INV	invert the sense of the counter mask comparison

PMIOGET (struct pmc) returns the current configuration of the specified counter.

PMIOSTART

PMIOSTOP (int) starts (stops) the specified counter. Due to hardware deficiencies, counters must be started and stopped in numerical order. (That is to say, counter 0 can never be stopped without first stopping counter 1.) The driver will *not* enforce this restriction (since it may not be present in future CPUs).

PMIORESET (int) reset the specified counter to zero. The counter should be stopped with PMIOSTOP before it is reset. All counters are automatically reset by PMIOSETUP.

PMIOREAD (struct pmc_data) get the current value of the counter. The pmc_data structure defines two fields:

int pmcd_num	the number of the counter to read
quad_t pmcd_value	the resulting value as a 64-bit signed integer

In the future, it may be possible to use the RDPMC instruction on Pentium Pro processors to read the counters directly.

PMIOTSTAMP (struct pmc_tstamp) read the time stamp counter. The pmc_tstamp structure defines two fields:

int pmct_rate	the approximate rate of the counter, in MHz
quad_t pmct_value	the current value of the counter as a 64-bit integer

It is important to note that the counter rate, as provided in the pmct_rate field, is often incorrect because of calibration difficulties and non-integral clock rates. This field should be considered more of a hint or sanity-check than an actual representation of the rate of clock ticks.

FILES

/dev/perfmon	character device interface to counters
/usr/include/machine/perfmon.h	include file with definitions of structures and event types
/usr/share/examples/perfmon	sample source code demonstrating use of all the **ioctl**() commands

SEE ALSO

ioctl(2), hwpmc(4)

Intel Corporation, *Pentium Pro Family Developer's Manual*, vol. 3, January 1996, Operating System Writer's Manual.

HISTORY

The **perfmon** device first appeared in FreeBSD 2.2.

AUTHORS

The **perfmon** driver was written by Garrett A. Wollman, MIT Laboratory for Computer Science.