NAME

hardclock - real-time timer

SYNOPSIS

void
hardclock(*int cnt*, *int usermode*);

DESCRIPTION

The **hardclock**() function is called periodically based on pending work. The rate ranges from *hz* times per second on a very busy system, to twice a second on an idle system. The *cnt* argument reports an estimate of the number of ticks since the last call. Over long timescales, the average sum of *cnt* over one second is *hz*. See hz(9) for important details over shorter time scales. The *usermode* argument is non-zero when **hardclock**() is called from an context that interrupted usermode execution.

hardclock() may perform different tasks such as:

- Run the current process’s virtual and profile time (decrease the corresponding timers, if they are activated, and generate SIGVTALRM or SIGPROF, respectively).

- Increment the time-of-day, taking care of any ntpd(8) or adjtime(2) induced changes and leap seconds, as well as any necessary compensations to keep in sync with PPS signals or external clocks, if supported by the kernel.

- Schedule softclock interrupts (swi(9)) processing.

- Collect hwpmc(4) statistics.

- Do device polling, when enabled (see polling(4)).

- Implement software watchdog(9) processing.

- Enqueue epoch(9) processing.

SEE ALSO

adjtime(2), ntp_adjtime(2), signal(3), hwpmc(4), polling(4), ntpd(8), epoch(9), eventtimers(9), hz(9), swi(9), watchdog(9)