I/O Systems (3):
Clocks and Timers

CSE 2431: Introduction to Operating Systems
Instructor: Adam C. Champion, Ph.D.
Outline

• Clock Hardware
• Clock Software
• Soft Timers
Two Types of Clocks

- Simple clock: tied to 110- or 220-volt power line, causes an interrupt at every voltage cycle (50 or 60 Hz).

- Other type, used in modern computers, based on crystal oscillator.
  - High accuracy, in the range of several hundred megahertz
Clock Hardware

Crystal oscillator

Counter is decremented at each pulse

Holding register is used to load the counter

A programmable clock
Programmable Clocks

• Two modes: one-shot vs. square-wave
• Periodic interrupts are called “clock ticks”
• Interrupt frequency can be controlled by software
  – For example, with 500-MHz crystal and unsigned 32-bit registers, we can generate interrupts at intervals from 2 nsec to 8.6 sec.
Battery-Powered Backup Clock

• Prevents losing current time when power is off
• This clock is read at startup
• The time is translated into the number of clock ticks since a certain point of origin
  – Jan. 01, 1970 for UNIX
  – Jan. 01, 1980 for Windows
• At every clock tick, the real time is incremented by one count
Clock Software

- Common duties of clock software
  - Maintaining the time of day
  - Preventing processes from running longer than they are allowed to
  - Accounting for CPU usage
  - Handling the alarm system call made by user processes
  - Providing watchdog timers for parts of the system itself
  - Doing profiling, monitoring, and statistics gathering
Maintaining the Time of Day

- With a clock rate of 60 Hz, a 32-bit counter will overflow in just over 2 years.
- What should we do?
Preventing Processes from Running Too Long

• When a process is started, the scheduler initializes a counter to the value of that process’s quantum in clock ticks
• At every clock interrupt, the clock driver decrements the quantum counter by 1.
• When the counter gets to zero, the clock driver calls the scheduler to set up another process
CPU Accounting

• Most accurate way: use second timer, distinct from the main system timer.

• Less accurate, but simpler, way: charge the running process at tick time
  – What if many interrupts occur during a process’s run?
Handling Alarm System Calls

• A process requests OS to give it a warning after a certain interval (e.g., timeout for network packet transmission)

• OS simulates virtual clocks with single physical clock, each virtual clock for one process
  – Using a table to store signal times
  – Using a linked list to store signal times
Simulate Multiple Timers

• On each tick, the value in “Next signal” is decremented by one
Watchdog Timers

• Part of the operating system also need to set timers (called watchdog timers)
  – A process reads data from an idle floppy disk
• The mechanism for watchdog timers is the same as that for user signals
  – The only difference is the clock driver calls a procedure supplied by the caller, rather than causing a signal when a timer goes off
Profiling

• A mechanism to build up a histogram of its program counter so that a user program can see where it is spending its time

• At every tick, the driver checks to see if the current process is being profiled, if so, computes the bin# (a range of addresses) corresponding to the current program counter.

• It then increments that bin by one.

• This mechanism can also be used to profile the system itself
Motivation for Soft Timers

• A second clock available for timer interrupts
  – Specified by applications
  – No problems if interrupt frequency is low

• Two ways to manage I/O
  – Interrupt: low latency, but overhead due to context switching; strongly influence pipeline, TLB, and cache
  – Polling: high latency
Soft Timers

- Soft timers avoid interrupts
  - Kernel checks for soft timer expiration before it exits to user mode
  - If the timer expires, the scheduled event (e.g., packet transmission) is performed, with no context switch
  - Then the soft timer is reset to go off again
  - How well this works depends on rate of kernel entries
Reasons for Kernel Entries

• System calls
• TLB misses
• Page faults
• I/O interrupts
• CPU going idle
Example for Soft Timers

- Gigabit Ethernet accepting or delivering a full-size packet every 12 µsec
- An interrupt takes 4.45 µsec on a 300 MHz, 35% overhead [AD99]
- Intervals of kernel entry are 2–18 µsec with several CPU loads [AD99]
- It is doable with a soft timer go off every 12 µsec (with occasional missed deadlines)

Summary

• Clock Hardware
• Clock Software
• Soft Timers