What is “Real Time?”

• Definition: *The system must complete its computation by a certain time or it has failed.*
  – Just getting the right answer isn't good enough.
  – You must get the right answer on time.

• Examples:
  – Breaks must engage no more than 50 ms after the break pedal is pressed.
  – Core shutdown must begin no more than 1.5 s after reactor temperature goes critical.
Hard vs Soft

• Hard Real Time
  - No deadlines can be missed.
    • Not even one.
    • Not even a little bit.

• Soft Real Time
  - If a deadline is missed system performance is degraded, but the system still “works.”
    • Some deadlines can be missed.
    • Missing a little is better than missing a lot.
Providing Real Time is Hard

• Conventional OS preemptively switches tasks.
  – Waiting time is non-deterministic and unpredictable.
  – When an event occurs, the time before the waiting tasks restarts is indeterminate.

• Thread priorities help.
  – Thread boosts its priority so that it runs as soon as possible.
    • Depends on OS scheduling algorithm.
    • Does not address priority inversion.
    • Does not help if multiple threads have same priority.
QNX is a RTOS

- Implements needed scheduling algorithms.
  - Provides several choices for different needs.
  - All choices are accessible by user applications.
  - Supports many different priority levels.

- Solves priority inversion problem.
  - High priority threads never wait in an unbounded way for a lower priority thread.

- But...
Application Writer Must Help

• Real time programming is *hard*.  
  – Must carefully inventory time on critical paths.  
  – Must make sure thread waiting is bounded.  
  – Must make sure only one RT process running.  

• Multiple RT processes will interfere with each other. Probably neither will hit all deadlines.  
  – Not a problem in an embedded environment.  
    • One process is real time.  
    • Other processes are service processes (web server, terminal, logger, GUI tools, etc)
Soft Real Time is Easier

- Just boost thread priority and hope for the best.
  - May miss some deadlines. That's ok.
  - May occasionally miss a deadline by a lot.
    - Not great, but still ok.
  - Behavior depends on how heavily loaded the system is overall.
    - Unacceptable in a hard RT system.
- Windows and conventional Linux allow this.
- Special versions of Linux offer hard RT support.
Real Time Not Always Fast

• Many RT requirements involve time on the scale of milliseconds.
  – Modern machine can execute 1 million instructions per millisecond. This is quite a bit of work.
  – Small processors are much slower.

• Is there enough time to do the work required?
  – A slow CPU might be fine if the amount of work is low or the deadline is far out.
  – A fast CPU might not be good enough if the amount of work is large or the deadline is close in.

• RT is about *guaranteeing* the deadlines!
Real Time in CIS-4020

• We won't experiment with RT programming.
  – Not sufficient time.
  – Impossible under virtualization anyway.