Real Time Linux
Who Needs It?
(Not you!)

Presented by:
Steven Rostedt
srostedt@redhat.com

What is Real Time?

- Real Time Delivery?
- Real Time Video?
- Real Time Clock?
- Real Time Operating System?
- Real Time Presentation Language Translation?
- Real Time Presentation Stop Clock

What is Real Time?

- Real Time Delivery?
- Real Time Video?
- Real Time Clock?
- **Real Time Operating System?**
- Real Time Presentation Language Translation?
- Real Time Presentation Stop Clock

What is a Real Time Operating System?

- **Faster? NO!**
- **Real Time does not mean “fast”!**
- **What does an RTOS give us?**

  Determinism

  Determinism

  Determinism

What does being Deterministic give us?

- **Repeatability**
- **Reliable Results**
- **Known Worse Case Scenarios**
- **Known Reaction Times**

Real Time (Hard vs Soft)

- **Hard Real Time**
  - **Mathematically provable**
    - **More code, much harder to prove**
  - **Bounded Latency**
- **Soft Real Time**
  - **Can deal with outliers**
  - **Tries to be reliable**
  - **May have unbounded latency**

Hard Real Time Examples

- **Airplane engine controls**
- **Nuclear power plants**
- **Mars Lander**
- **Space Shuttle**

Soft Real Time Examples

- **Video systems**
- **Video games**
- **Some communication systems**

Vanilla Linux is a Soft Real Time System!

Real Time Linux (PREEMPT_RT)

- **Is it a Soft Real Time system?**

  Does not allow for outliers

  Does not allow for unbounded latency

- **Is it a “Hard Real Time” system?**

  Too big to be mathematically proven

- **What is PREEMPT_RT then?**

  Hard Real Time “Designed”!

Real Time Linux (PREEMPT_RT)

- Can not be mathematically proven
  - It is just too darn big
    - (people are trying though!)
- Tries to bound all latency
  - Unexpected latency are considered bugs
- The design follows that of any hard real time operating system.

Who uses PREEMPT_RT?

- Financial industries (NASDAQ)
- Audio recordings
  - A latency causes a “scratching” sound
- Navigational systems (TomTom / Garmin)
- Can not fail...
  - but nobody dies if it does (hopefully)

What PREEMPT_RT gave to current Linux

- High resolution timers
- Generic interrupt design
- Preemptible RCU locks
- Real Time scheduler
- EDF scheduler (SCHED_DEADLINE)
- Threaded interrupts
- Priority inheritance futexes
- Lockdep
- Ftrace - The Linux kernel tracer

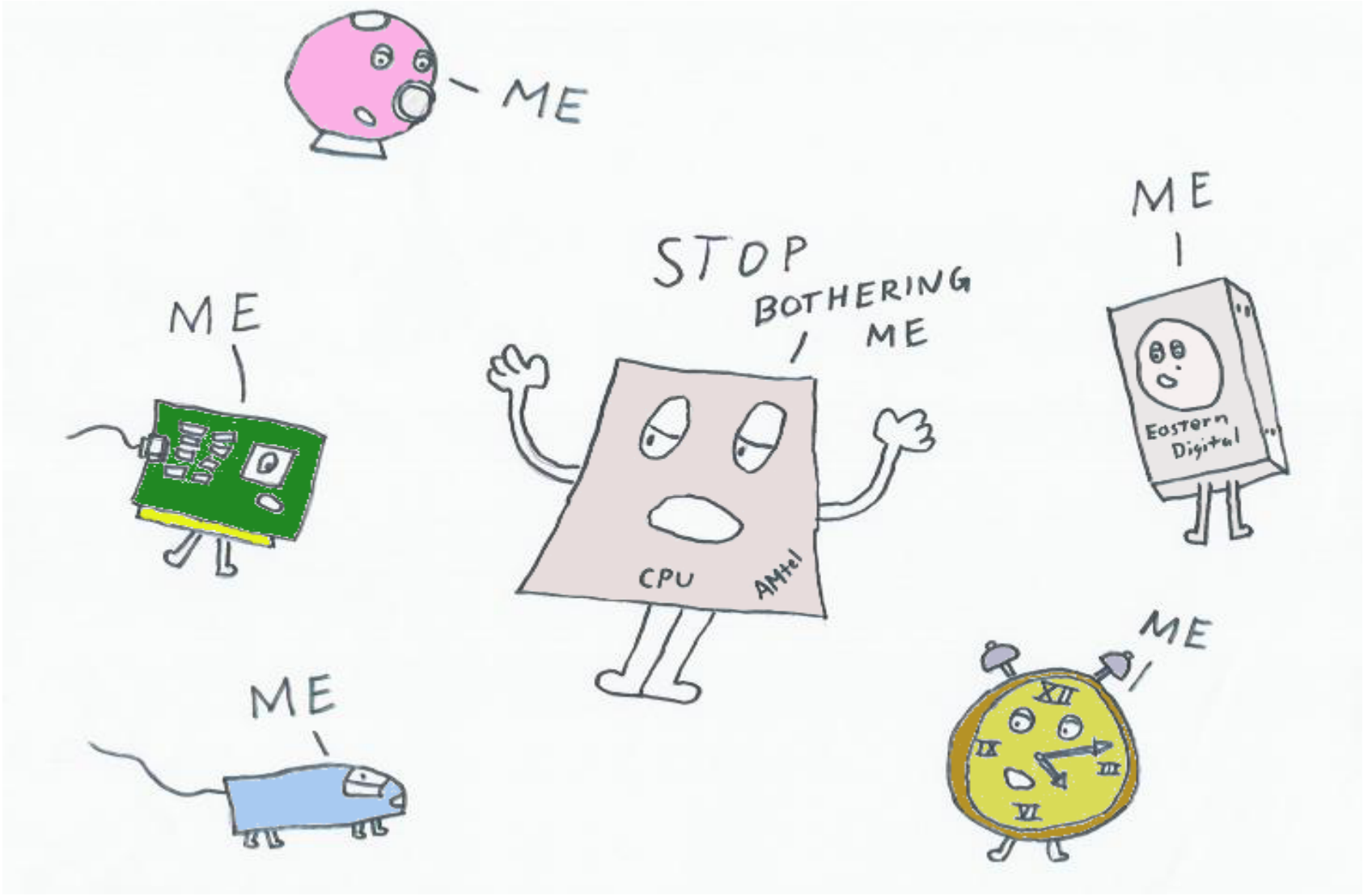
What is left?

- Spin locks to sleeping mutexes
  - Interrupts do not need to be disabled
  - Helps against reaction time latency
    - task wakes up, timer response, etc
- Priority inheritance on kernel locks
  - Helps against “unbounded priority inversion”

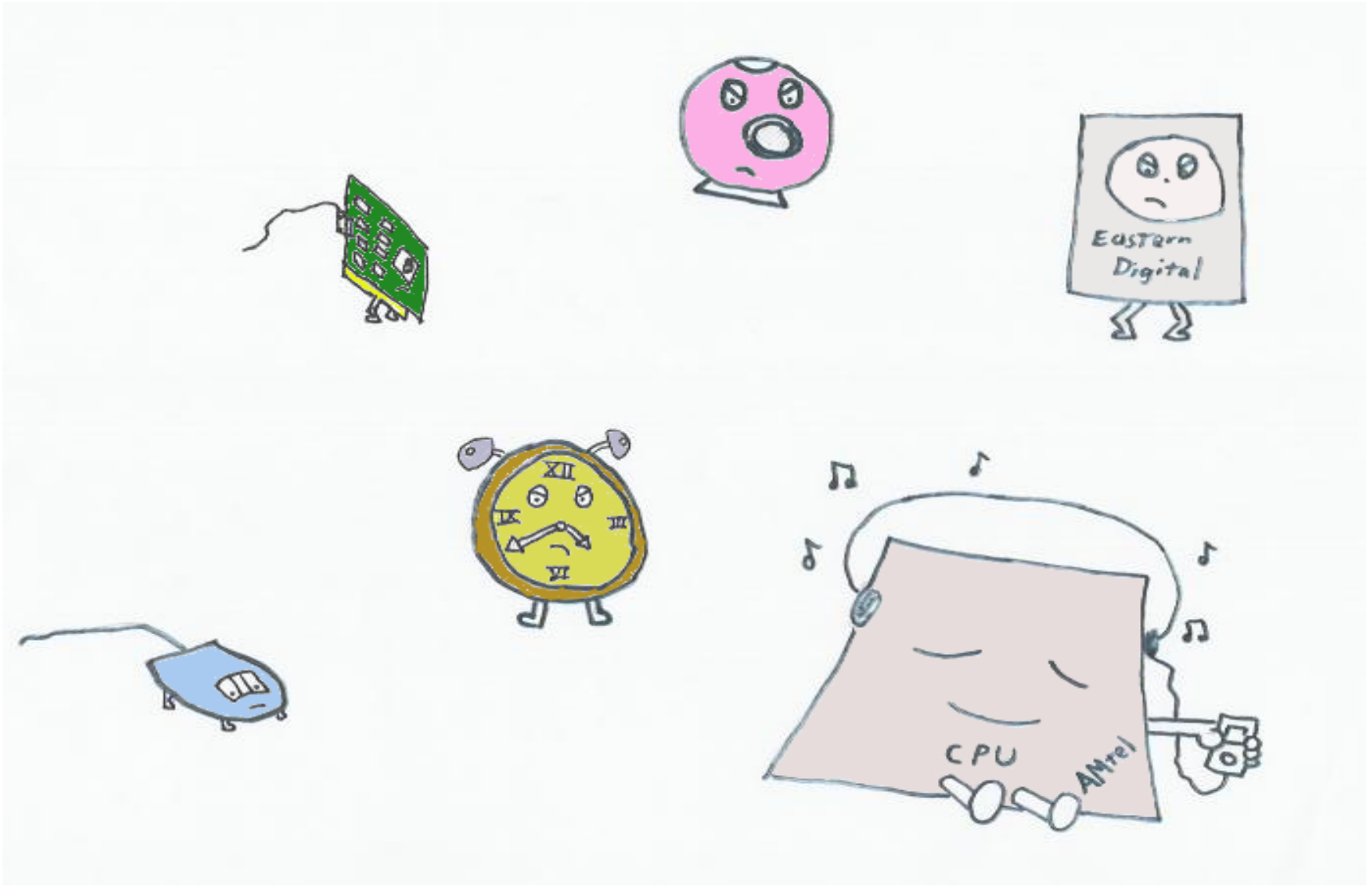
What is latency?

- The time between when an event is expected to happen, to the time it actually happens
- Causes of latency?
  - Interrupts being disabled
  - Current interrupt executing
  - Shared resources and locking

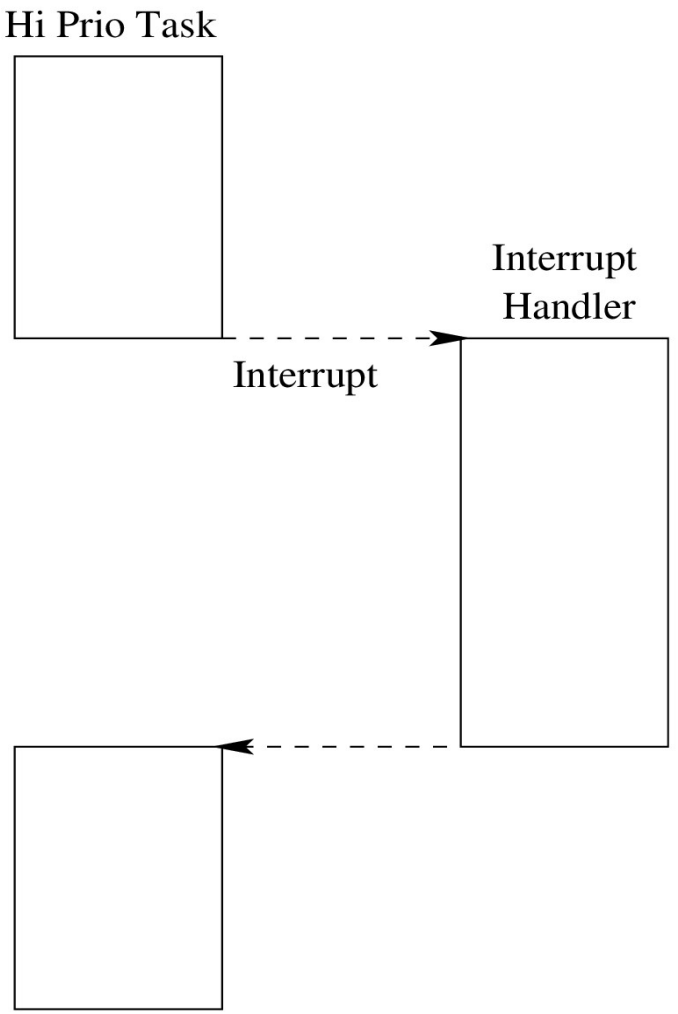
Interrupts



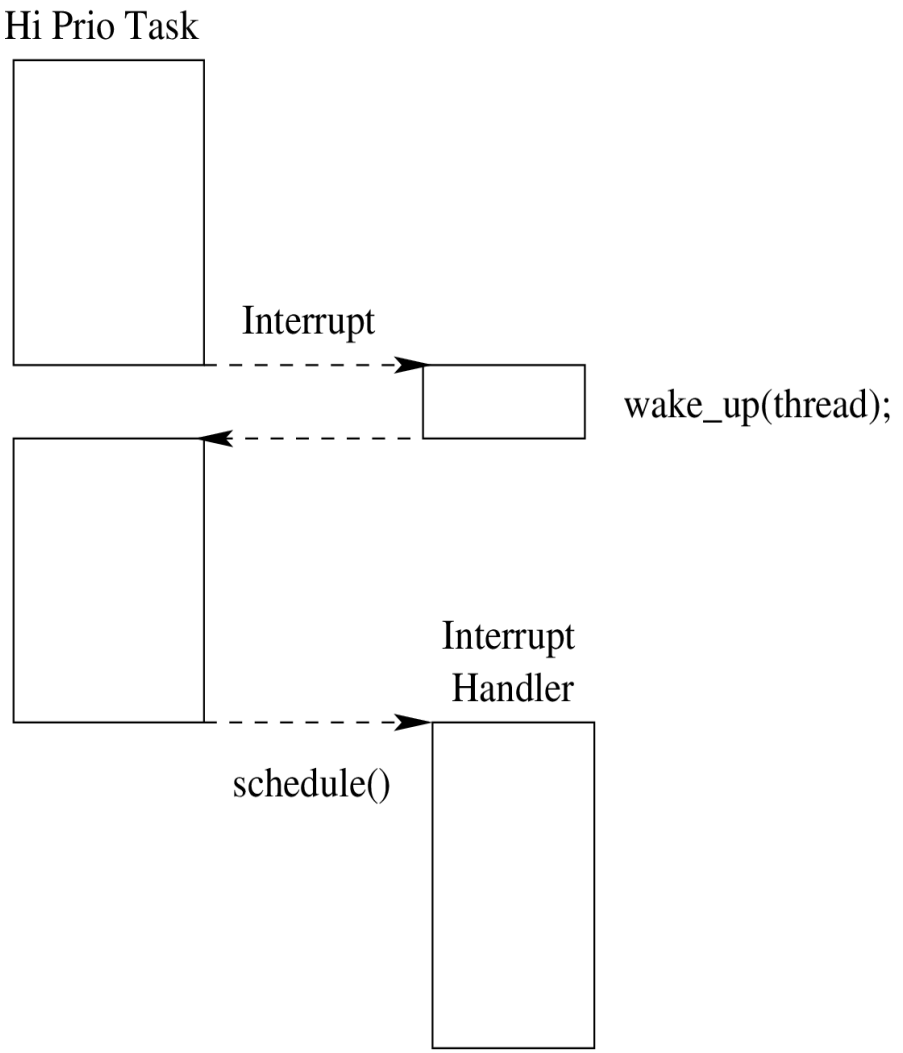
Interrupts disabled



Interrupt Inversion



Threaded Interrupts



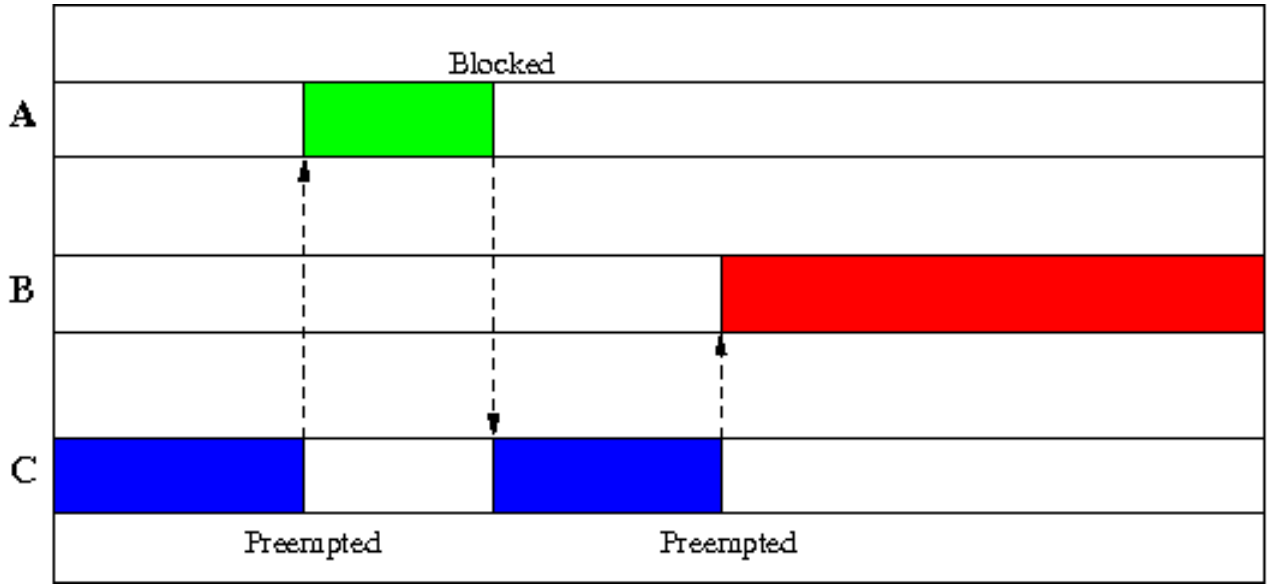
Interrupt Threads

```
# ps ax |grep irq
root         52  0.0  0.0      0     0 ?        S    19:15   0:00 [irq/9-acpi]
root         68  0.0  0.0      0     0 ?        S    19:15   0:00 [irq/24-pciehp]
root         69  0.0  0.0      0     0 ?        S    19:15   0:00 [irq/25-pciehp]
root         72  0.2  0.0      0     0 ?        S    19:15   0:26 [irq/14-ata_piix]
root         73  0.0  0.0      0     0 ?        S    19:15   0:00 [irq/15-ata_piix]
root         79  0.0  0.0      0     0 ?        S    19:15   0:00 [irq/19-ata_piix]
root         86  0.0  0.0      0     0 ?        S    19:15   0:00 [irq/18-ata_gene]
root         93  0.0  0.0      0     0 ?        S    19:15   0:00 [irq/19-ehci_hcd]
root         94  0.0  0.0      0     0 ?        S    19:15   0:00 [irq/23-ehci_hcd]
root         95  0.0  0.0      0     0 ?        S    19:15   0:00 [irq/16-uhci_hcd]
root         96  1.1  0.0      0     0 ?        S    19:15   2:13 [irq/21-uhci_hcd]
root         97  0.0  0.0      0     0 ?        S    19:15   0:00 [irq/18-uhci_hcd]
root         99  0.0  0.0      0     0 ?        S    19:15   0:00 [irq/23-uhci_hcd]
root        100  0.0  0.0      0     0 ?        S    19:15   0:00 [irq/19-uhci_hcd]
root        102  0.0  0.0      0     0 ?        S    19:15   0:00 [irq/16-uhci_hcd]
root        104  0.0  0.0      0     0 ?        S    19:15   0:00 [irq/12-i8042]
root        105  0.0  0.0      0     0 ?        S    19:15   0:00 [irq/1-i8042]
root        108  0.0  0.0      0     0 ?        S    19:15   0:00 [irq/8-rtc0]
root        114  0.0  0.0      0     0 ?        S    19:15   0:00 [irq/4-serial]
root        315  0.0  0.0      0     0 ?        S    19:15   0:00 [irq/27-i915]
root        698  0.0  0.0      0     0 ?        S    19:15   0:00 [irq/6-floppy]
root       1676  0.0  0.0      0     0 ?        S    19:15   0:00 [irq/18-i801_smb]
root       2012  0.0  0.0      0     0 ?        S    19:15   0:00 [irq/28-snd_hda_]
```

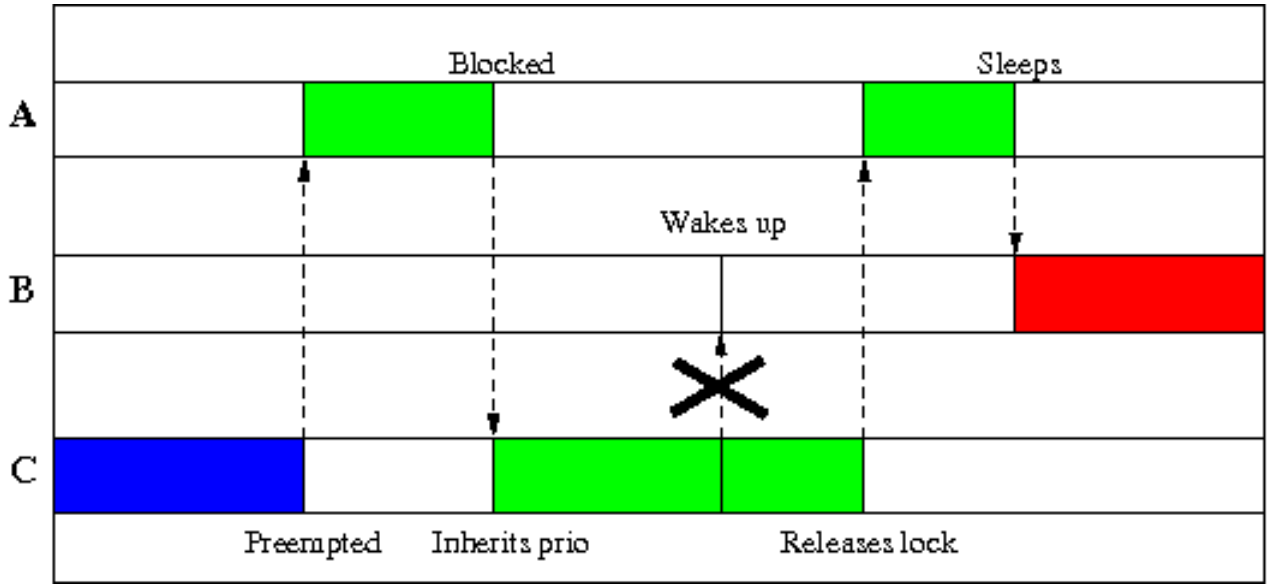
Latency

- **Latency always happens**
  - **Events are never instantaneous**
- **Priority Inversion**
  - **When something runs when something else should be**
  - **Always happens too**
- **Bounded Priority Inversion**
  - **We know the worse case**
- **Unbounded Priority Inversion**
  - **No idea when it will finish**

Unbounded Priority Inversion



Bounded Priority Inversion (using Priority Inheritance)



Hardware does matter!

- Cache and TLB misses
- SMI

  System Monster^WManagement Interrupt
- The hardware must also be deterministic

  (Stay tuned, same Bat Channel)

Questions?