

Meeting Real-Time Requirements With Linux

Jeffrey Fortin
Director, SE Field Engineering

Biographical Information



- **University of Rhode Island**
 - BS Computer Science 1985
- **Sperry Univac**
 - MK1 FCS (AN/UYK-7)
- **Raytheon**
 - MK2 FCS (AN/UYK-43)
 - JTCTS Infrastructure Lead
- **Wind River**
 - FAE Florida GEO
 - Director, SE Field Engineering
- **Marathon**
 - Best Time – Disney 3:49:36
 - Ran ~15 marathons since 2002

Abstract

Linux is being used more and more in embedded devices such as cell phones, networking gear, and even MP3 players. As software engineers work to integrate the Linux run-time into these devices, they often find that there can be several challenges to meeting real-time requirements. So what is real-time and how can we meet real-time requirements using Linux?

Outline

- **A matter of time**
- **What is Real-Time?**
- **Having a real hard time with hard real time?**
- **Additional topics**

What is time?



Computer Time

- **Clocks**
 - TOD Clock
 - System Clock
- **Timers**
 - Hardware Based
 - Software Based
- **Time Codes**
 - SMPTE
 - IRIG

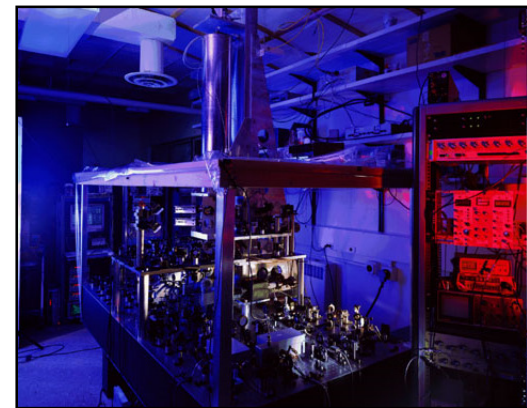
Problems with Time

- **Time is not universal**
 - There is no fixed universal time. Time is relative to the observer.
 - Time Dilation is explained in terms of both general and special relativity
- **What we mean by time is not universal**
 - Time can be relative to the rotation of the earth (TOD)
 - Time can mean a fixed repeated interval (Periodic)
 - Predictable Response Time
 - Bounded (50 ms +-5us)
 - Fixed (50 ms +- 0)
 - Real-Time

Problems with Computer Time

- **Drift**
 - When one clock falls out of sync with another clock
- **Synchronization**
 - Network
 - GPS
 - Clock to the earth's non-regular period
- **Jitter**

Atomic Cesium Fountain NIST-F1 (1999)



What is a Real Time System?

- **Time is essential for correctness**
- **What does this mean?**
 - “... performance deadlines on its computations and actions”¹
 - **Reactive or Event Driven**
 - **Time Driven**
- **My definition**
 - **A system whose behavior must be coordinated with the behavior of a real object.**

¹ Doing Hard Time, Douglass, Addison Wesley 199, p.58

Example Real Time Systems

- Traffic Control
- Locks in the Panama Canal
- Lunar Landing Control
- Airplane Flight Control
- Weapon Launch Control
- Radar Tracking
- Ride Control
- Antilock Breaks



Basic Real-Time Concepts

- **Timeliness**
- **Responsiveness**
- **Concurrency**
- **Predictability**
- **Correctness and Robustness**

Real-Time RTOS Requirements

- **Software Drivers**
- **Scalability**
- **Scheduling**
- **Automated Start (unassisted boot)**
- **Hardware and Processor flexibility (including multi-core)**
- **Memory Management**
- **Concurrency Management**
- **Data Access Control**
- **Time Management**
- **Networking**
- **I/O Services and File Systems**
- **Language Support**
- **Graphics Toolkits**
- **Development and Debug facilities**

Having a hard time with hard real time?

- **Challenges in using Linux for real-time systems**
 - **Build and intended as a GPOS**
 - **Responsiveness Oriented**
 - **Interrupt lock-out**
 - **Fair share scheduling (especially noticeable on SMP systems)**
 - **Unbounded memory**
 - **Strongest on x86 processors**
- **How can these challenges be overcome?**

Using Linux for Real-Time Systems

- **Native Approach**
 - Linux 2.6 kernel and PREEMPT_RT
- **Multi-kernel Approach**
- **Supervised Approach**

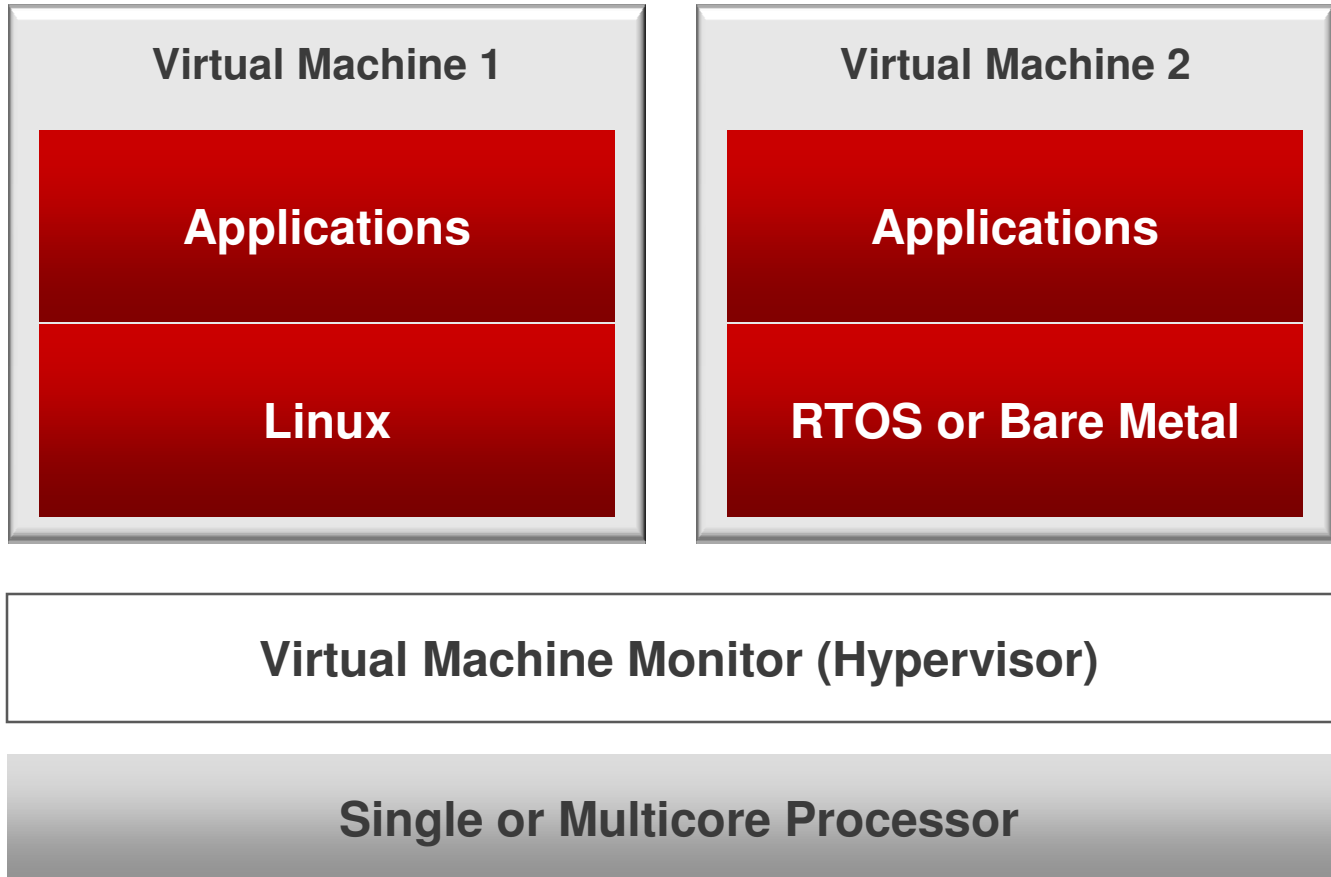
Native Approach

- **Many real time projects ongoing within the community**
- **Most significant project started by Ingo Molnár**
- **Linux 2.6 kernel and PREEMPT_RT**
 - **Removed pre-emption lockouts**
 - **Address the responsiveness of the operating system**
 - **Also improves timeliness**
- **Now considered mainline**
- **Requires cooperation with other contributed elements**
- **Drivers are being updated to cooperate within this scheme**

Multi-kernel Approach

- Takes a “best of both worlds” approach
- Use Linux for non-real-time requirements
- Use a small micro-kernel RTOS to enable real-time
- Micro-kernel takes over interrupts
- Micro-kernel can implemented as a kernel module
- Issues that can come up
 - Programming environment for the micro-kernel may be different than Linux
 - Development tools may not be the same as for Linux
 - Debug code in the micro-kernel

Supervised Approach



Supervised Approach

- **Use a hypervisor to run Linux and an RTOS/Bare Metal**
 - Xen is an example
 - Good for Best Effort requirements
 - Requires a DOM0 Linux for control
- **Use a “thin” hypervisor designed for real-time**
 - Used to manage memory, devices, and access to CPU(s)
 - Leverage hardware para-virtualization

References

- **Doing Hard Time, Douglass, Addison Wesley**
- **White Paper “Wind River Linux and VxWorks Real-Time Capabilities: A Comparison”**
(http://www.windriver.com/whitepapers/whitepaper.php?f=rt_linux_vxworks_comparison_whitepaper.pdf)
- **http://windriver.com/products/platforms/real-time_core/**

WIND RIVER