Real-Time challenges and concepts:

Predictable response concepts and strategies

Dr. Sam Siewert

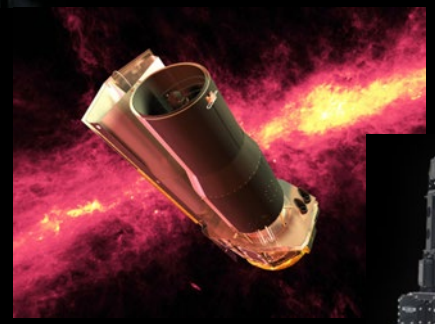
Electrical, Computer and Energy Engineering

Embedded Systems Engineering Program

Segment Outline

- Course Content and Background
- Minimal Home Lab Setup
- Optional Textbook (Recommended, but not required)
- Basic Definitions
- Real-Time Rates of Well-known Real-Time Services
- Design Challenge of Real-Time Applications and Services
- Why use Software?
- Making Software behave like Hardware
- Complex Real-Time Systems (Hardware, Firmware, Software)

- NASA Johnson Mission Control, SAIL (1990-1992)
  - Software Avionics – Ascent / Entry Guidance
  - RT Flight Software and Mission Control
- Ph.D. Work – RT and Interactive Systems (1994-2000)
  - Optical Navigation and Control Systems and Software
  - Extension to Rate Monotonic Theory (Statistical RMA)
- RT Instrumentation and Machine Vision (1997-2000)
  - Spitzer Space Telescope, MIPS Observing Modes
    - Mosaic Sky Scan and Super-resolution,
    - Total Power and Spectral Energy Distribution
    - Raw
- Network Processing – Not RT (2000 – 2012)
  - 10G Ethernet, Fiber Channel, SAS/SATA, RAID
- Cable Labs RT “Head End in a Box” (2006)
  - Broadcast UTC (GPS Time, NTP)
  - Soft Real-Time Encode, Transcode, Decode, Multiplex
- Teaching and Consulting (2012 – Present)
  - Software and Computer Engineering - Real-Time UAS/UAV
- Current RT Research (ICARUS Drone Net)
  - RT Multi-spectral Object Detection, Fusion, 3D Mapping
  - Ground and flight instruments
    - RADAR, LIDAR, Infrared, Visible, and Acoustic
    - Urban UAS Traffic Management



Sam Siewert

Associate Professor Adjunct
Univ. of Colorado Boulder
Computer Engineering
siewerts@colorado.edu

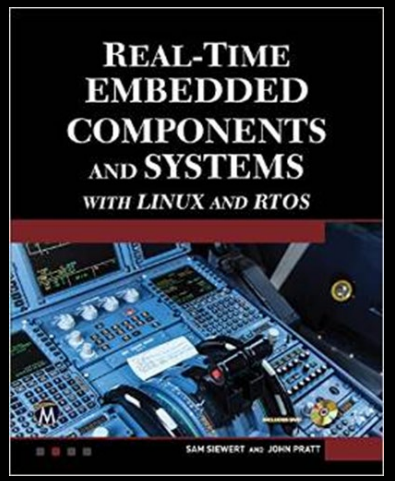
Associate Professor
Embry Riddle Aeronautical Univ.
Computer and Software Engineering
Samuel.Siewert@erau.edu

- **Use any Native Linux system**
  - **Laptop that boots Linux**
  - **Raspberry Pi 3b+ or newer (est. $80)**
  - **NVIDIA Jetson with Jetpack Linux**
  - **Any other Linux embedded system with USB-2 or USB-3 interfaces**

- **Real-Time Embedded Components and Systems with Linux and RTOS, 2nd Edition, Sam Siewert and John Pratt, October 2015, 978-1942270041, Mercury Learning, Amazon**

- 1/4 Concepts and Practices
- 1/4 Theory and Analysis
- 1/4 RT System Design (Mission Critical)
- and 1/4 Development [Project - Observer Synchronization of Clock Domains]

My R-Pi 3b+ Setup
- USB2 C270 camera
- Mouse, keyboard
- HDMI monitor
- Running RT Canny Transform

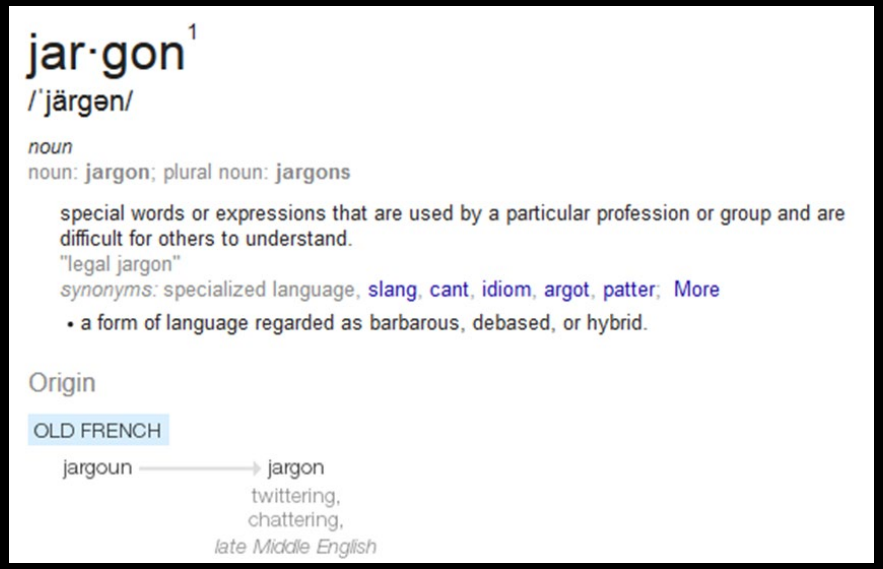


Recommended Text

Supporting Materials on Coursera Course Resources

- **Real-Time Embedded System Jargon**
  - Engineering Jargon
  - Computer Jargon
  - Embedded Jargon
  - Now, Real-Time Jargon

- **Jargon Glossary**
  - Download (back of Textbook, on Coursera)
  - Scan (you many know many terms already)
  - Question (note those you don't know)
  - Read (read definition)
  - Recite (those new to you before a quiz)
  - Review (make RT jargon your vocabulary)

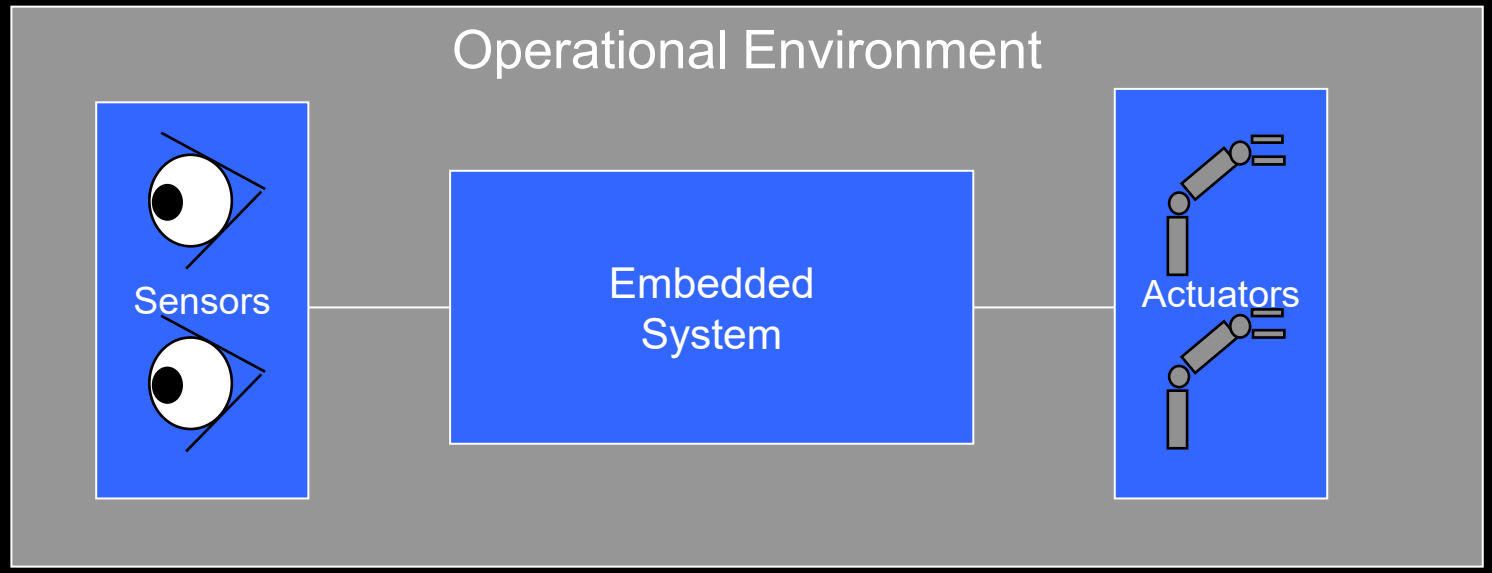


Learn terminology from:

Actuator to Zettabyte

Nobody fully agrees on some of these, but the most common definitions were collected over 30 years of practice.

- To be Embedded is …
- A Compute Node That Provides Specific Services by Processing Inputs and Producing Required Responses
  - Provides Specific Services Rather than General Purpose Computing
  - Often Limited or No Direct Connection to User Input/Output (e.g. Antilock brakes)
  - Contained within a Larger System as a Sub-system (flight control on aircraft)



- To be Real-Time is …
- Services that must be provide upon request prior to a deadline
  - A deadline is a constraint for processing
  - If no deadline constraint exists for a system, it is not real-time
  - Most real-time systems handle requests on a periodic basis
  - Some real-time systems require deterministic response before a deadline
  - Some required predictable response (some missed deadlines are allowed)
- Impact of a missed deadline as specified by design constraint
  - Loss of property, life, financial, or system itself - HRT (Hard Real-Time)
  - Loss of service, quality of that service, and annoyance - SRT (Soft Real-Time)
- No deadline constraint - BE (Best Effort)
  - Desktop systems are interactive, low latency response, but no response constraint
  - May be responsive, but in a best effort fashion (some waiting… hourglass)

- **Many Examples of Real-Time Embedded Systems**
  - Real Time – Must Respond to Requests for Service by a Deadline relative to request
  - Failure to Respond Prior to Deadline Results in a System Failure
  - Request Rate for Service Driven by Real-World Events
  - Controls Processes and Delivers Deadline Driven Services
  - Specific Examples (with deadline constraints)
    - Medical Devices (e.g. Pulse Oximeter)
    - Unmanned Aerial Systems
    - Anti-Lock Braking
    - Streaming Media (Video and Audio)
    - Process Control
    - Aircraft Flight Control
    - Robotic Systems

Example Real-Time Event Rates

Source	Rate	Period (msec)	Period (μsec)	RT Quality	Data Rate
Audio	8K Hz	0.125 msec	125 μsec	PSTN VOIP	**64 Kbps**, DS0
Audio	20K Hz	0.05 msec	50 μsec	Human limit	20 bits per ear x 2 x 20,000 = **800 Kbps**
Sound Card	48K Hz	0.02 msec	20.83 μsec	Stereo 2.0, Surround Sound 5.1, 7.1	2x8x48,000= **768 Kbps**
Analog Video	30 Hz (29.97 color)	33.33 msec	33,333.33 μsec	VGA NTSC (even/odd)	640x480x3x30= **26.37 MB/sec**
Projector	24 fps (48 or 72 Hz illumination)	41.67 msec	41,666.67 μsec	Cinema film (32-bit color)	2K equivalent of 2048x1080x24x4 = **202.5 MB/sec**
Graphics	60 Hz	16.67 msec	16,666.67 μsec	QXGA Graphics (24-bit color)	2048x1536x60x3= **540 MB/sec**
Machine Vision Camera	120 fps	8.33 msec	8333.33 μsec	Machine Vision	2048x1536x2x120= **720 MB/sec**

Notes On Relative and Absolute Time

- Sources of Global Time
  - GPS (GNSS)
  - NTP (Network Time Protocol) from NIST
  - UTC (NIST)

- POSIX / Linux Supports Both
  - Relative Time Based on Interval Timer Hardware
    - x86 PIT
  - Absolute Time Based on Battery-Backed Clock
    - RT Clock
      - Julian/Gregorian Date and TOD
      - Battery Backed or Network Time Protocol
      - Elapsed Time Since a Specific Epoch
  - Be Clear on Which is Specified (Relative or Absolute)

Why are RT Embedded Systems a Challenge?

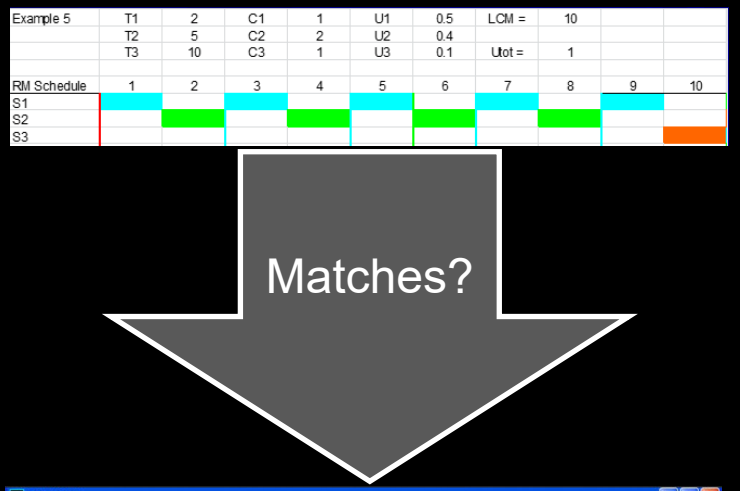
1. Correct results on time – Deadlines (constraint)
2. Multi-service Concurrency - Multi-threaded Software
3. Multiple interfaces to service concurrently
4. Function and Service Allocation – CPU Software, HW Co-processor (e.g. FGPA, MPEG decoder), or Software Co-processing (e.g. DSP, GP-GPU)
5. Management of CPU, IO, and Memory Resources
6. Modern architecture – high throughput, less deterministic
7. Sensors / Actuators (control Physical Process)
8. Networks (Latency and QoS)
9. Persistent Memory Devices (e.g. Flash, EEPROM)
10. Memory Hierarchy (Register, Cache, RAM, Flash)

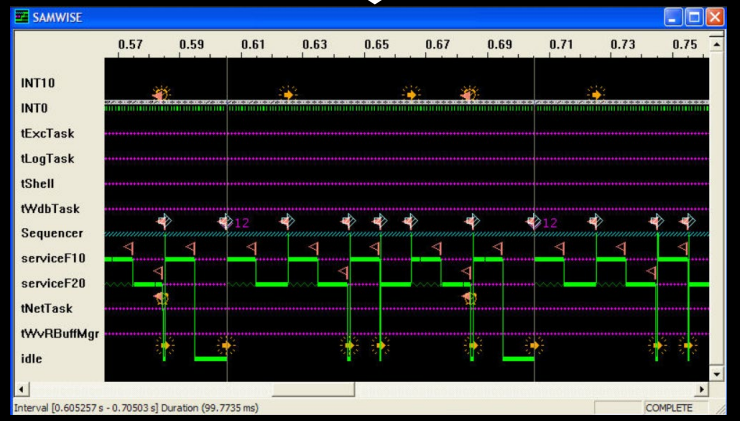
Simplifying Real-Time Systems and Ensuring Success

1. RT Service and CPU Resource Management Policies (e.g. Rate Monotonic, EDF)
2. RT Theory and Best Practices (Theory, Analysis, Models, System, Verification)
3. RMA Resource Theory (proof of feasibility of design)
4. Performance Prediction and Measurement (predicted to actual comparison)
5. When to Allocate Services/Functions to HW, FW, or SW
6. Multi-threaded RTOS and Real-Time Linux Systems
7. RTOS or POSIX RT Mechanisms (e.g. message queue, signal, semaphore)
8. Analysis and Debug Tools
9. I/O Device Interfaces and Driver
10. Abstracted Non-Volatile Memory File systems

- **Rate Monotonic Concepts and Principles**
  - Best Effort - Non-RT processing (Windows, desktop Linux, AOS, iOS, etc.)
  - Soft Real-Time - predictable response (Netflix, SmartTV, MPEG video, digital video games, etc.)
  - Hard Real-Time - Commercial aviation FCS (Flight Control System), Anti-lock braking, Heart-lung machine, Air Traffic Control, etc.
- **Scheduling of Services (Concurrency)**
  - Math Models - Rate Monotonic Analysis (RMA) of fixed priority, Adaptive Analysis of dynamic priority
  - Feasibility - determined by manual analysis or mathematical feasibility tests (algorithms) - can concurrent services meet deadlines (constraint)
  - Specification of RT constraints - S_i, T_i, C_i, D_i
  - Schedulers and Hardware vs. Software RT Services

RMA Theory and Analysis



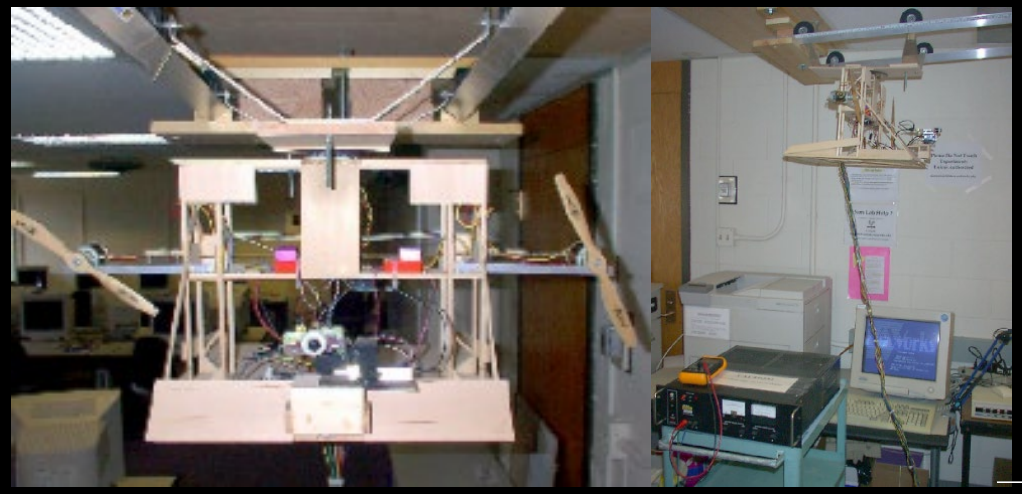


Actual Service Time Traces

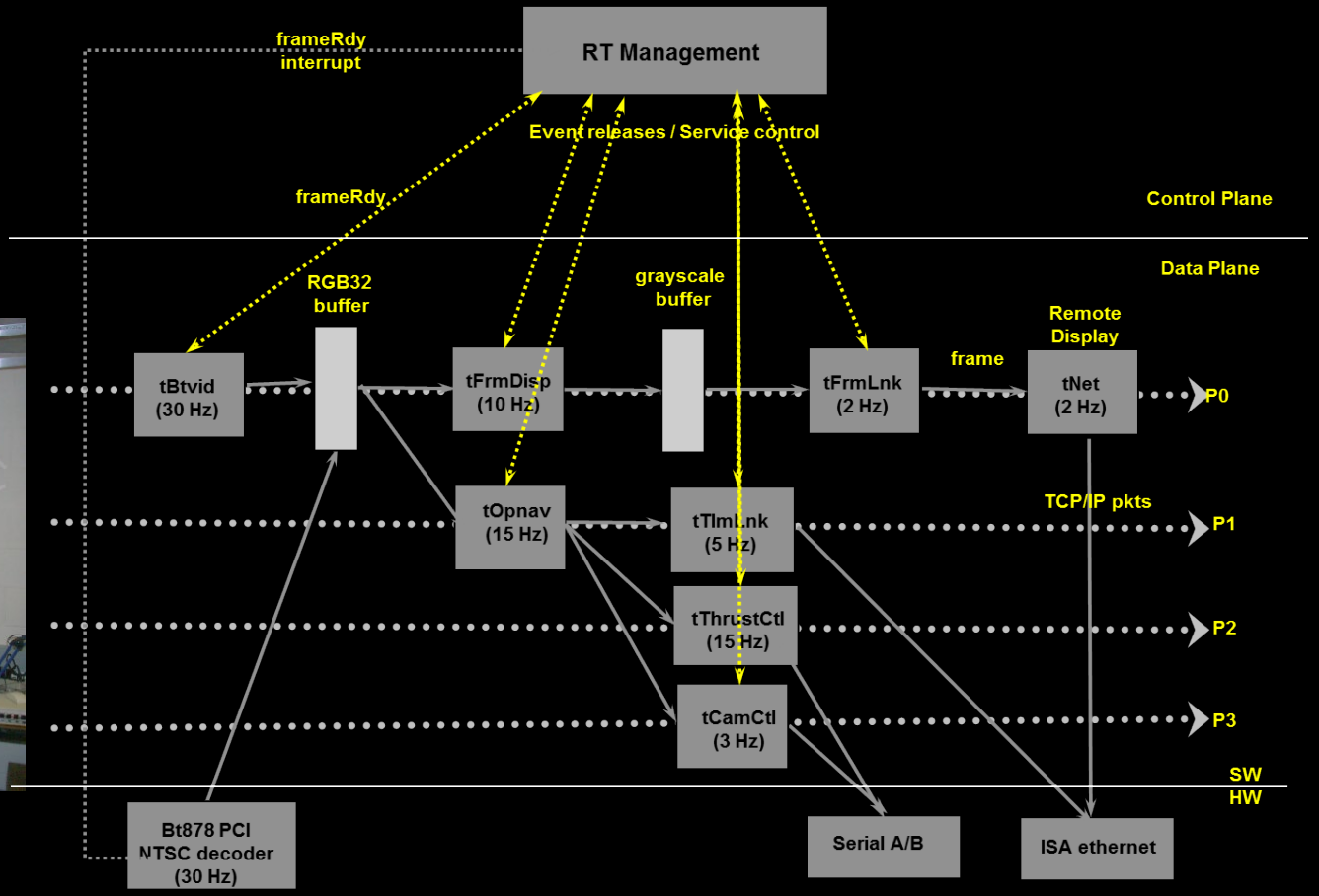
Complex Multi-Service Systems

- Multiple Software Services (Reconfigurable rates)
- Synchronization Between Services
- Communication Between Services
- Multiple Sensor Input and Actuator Output Interfaces
- Intermediate IO, Shared Memory, Messaging

Service	Name	T (msec)	C (msec)	Utility	WC (msec)	WC Utility		Service	Name	T (msec)	C (msec)	Utility	WC (msec)	WC Utility
1	tExc	1	0.05	5.0%	0.05	5.0%		6	tNet	100	8	8.0%	10	10.0%
2	tBtvid	33.33	0.06	0.2%	1.2	3.6%		7	tCamCtl	200	0.61	0.3%	1.53	0.8%
3	tOpnav	66.67	20.91	31.4%	23.07	34.6%		8	tTlmLnk	200	0.384	0.2%	1.39	0.7%
4	tRACECtl	66.67	0.19	0.3%	1.27	1.9%		9	tFrmLnk	500	55.36	11.1%	58.05	11.6%
5	tFrmDisp	100	38.77	38.8%	40.08	40.1%		**TOTAL**				**95.2%**		**108.3%**



Rail-Guided, Air-Powered Controls Experiment



- Summary
  - Basic Definitions
    - Embedded
    - Real-Time
    - Embedded Real-Time System
  - Well-know Real-Time Services
    - Digital video
    - Digital audio
    - Digital control
    - Monitoring (sensor networks)
  - Software complicates, but needed for field upgrade, flexibility, future proofing
  - Making Software behave like Hardware
    - goal for this course (predictable)
    - Ideal goal is deterministic
  - Complex Real-Time Systems involve Hardware, Firmware, and Software