Design of Mission-Critical Apps & Systems:

Expanding to RT Theory for RTES Design

Dr. Sam Siewert

Electrical, Computer and Energy Engineering

Embedded Systems Engineering Program

University of Colorado
Boulder

Pre-requisites and Required Skills

- Knowledge, Skills, Theory, Understanding and Analysis from Prior RTES Coursera Courses
  - RTES Concepts and Practices
  - RTES Theory and Analysis
- Intermediate to Advanced C Programmer
- Intermediate C++ Programming Skills
- Intermediate POSIX Threading – Concurrency
- Intermediate SCHED_FIFO and POSIX 1003.1b (d, j) RT Extensions Knowledge and Experience
- Linux Platform at home and experience with self-support for tools and configuration

Mission Critical Apps & Systems

Can be small scale, deeply embedded Cyclic Executives

- e.g. Anti-lock braking system in a car
- Cruise control in a car
- Simple auto-pilot for a UAV, e.g. Pixhawk PX4
- MPEG Decode in a smart pone

Can be larger scale, systems with an RTOS

- e.g. Aircraft avionics Flight Control System
- Flight Management System
- Earth orbiting spacecraft bus
- Instruments for science
- Transportation systems
- Medical systems (diagnostic, therapeutic, monitoring)

Can be scalable systems with an OS + RT Extensions

- Distributed RT System – e.g. intelligent transportation, shipping, and military
- Soft RT Digital Media head end (data center)
- Automated financial systems (stock market)

Course Flow

- Standard Path
- Jump in at "RTES Theory and Analysis"
- Exit after "RTES Theory and Analysis"
- Exit after "RTES Mission Critical Apps & Systems"
- Complete full series

RTES Concepts and Practices

Recommended start
(new to RTES)
Proficient C

RMA basics

RTES Theory and Analysis

Know some RMA & Linux
(Jump in)
More RMA depth

RMA practitioner

Mission Critical Design

Advanced Course

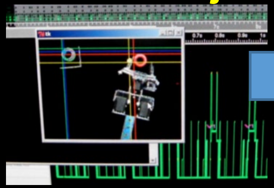
- Intermediate C/C++
- Know RMA well
- POSIX RT, RTOS, CE

RTES practitioner

RTES Project

Advanced Course

- Solve RT Problem
- Construct a System



RTES designer

Real-Time with OS + Extensions

First feasible in late 1990's … today

Key OS added POSIX RT Extensions

Portable Operating Systems Interface

Real-Time extensions focus

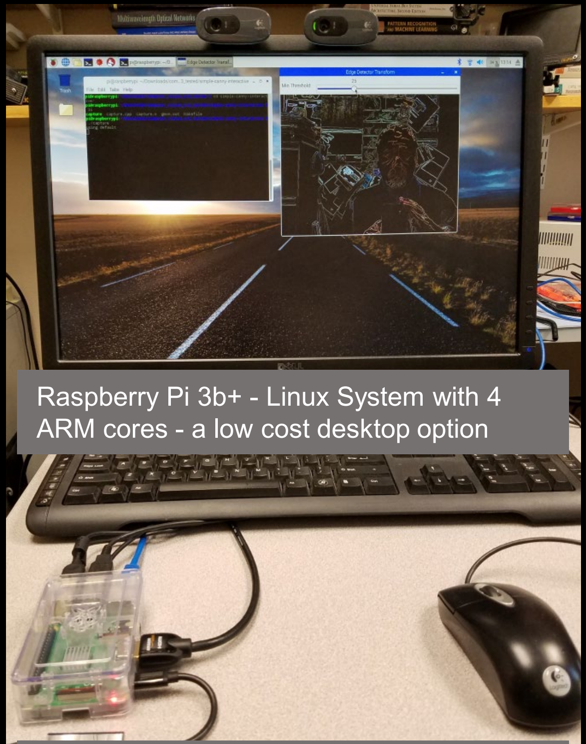
- Scheduling – RMA fixed and dynamic
  - Linux SCHED_FIFO
  - Linux SCHED_DEADLINE
- Synchronization
- Shared memory processing
- AMP and SMP (thread affinity)
- Advanced memory features (locking pages)
- Software signals (that queue)
- Message queues

Gallmeister, Bill O. "Programming for the real World, POSIX. 4." *OReilly & Associates, Inc* (1995).

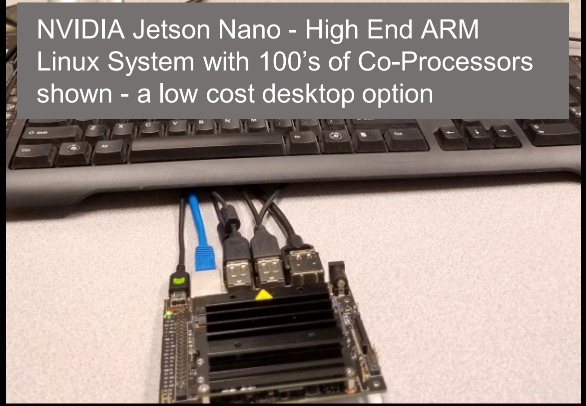
Obenland, Kevin M. "The use of posix in real-time systems, assessing its effectiveness and performance." *The MITRE Corporation* (2000).

Required Equipment for Course

- Option #1 - Raspberry Pi 3b+
  - $80 kit
  - CanaKit for Raspberry Pi 3 B+ on Amazon
  - CanaKit for Raspberry Pi 4 with 2,4,8 GB RAM
  - You own it, can be used for entire course
- Option #2 - NVIDIA Jetson Nano
  - $99 for students
  - JetPack Linux installation
- Option #3 – Virtual-Box with Ubuntu LTS
  - Use Windows or Mac PC
  - Software Development ONLY
  - No real-time execution or camera streaming
  - Learn Linux with minimal cost and risk
  - Real-time testing with native Linux (Option #1, 2)
- Option #4 - Linux Laptop (Native installation)
- Option #5 - Intel NUC with Ubuntu LTS



Raspberry Pi 3b+ - Linux System with 4 ARM cores - a low cost desktop option

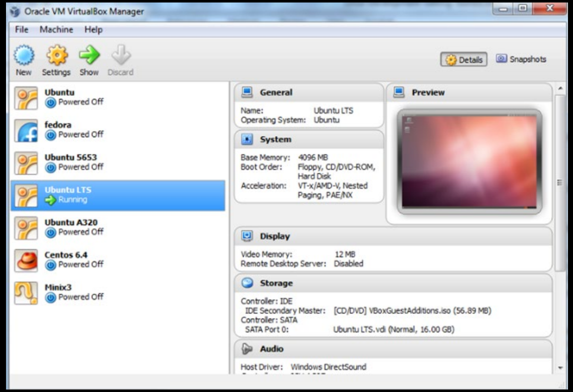


NVIDIA Jetson Nano - High End ARM Linux System with 100's of Co-Processors shown - a low cost desktop option

Home Lab Setup for Course (Resources Video)

- From RT Theory and Analysis
- Use for this Course as well
- Used for all Courses in Series
  - RT Concepts and Practices
  - RT Theory and Analysis
  - Design of Mission-Critical Applications and Systems
  - RT Embedded Systems Project

Virtual Box Linux
Functional Development



Course #3 – Mission Critical Apps & Systems

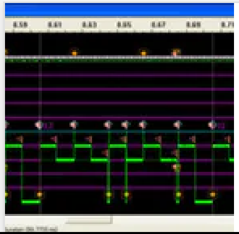
- Course #1 – RT Concepts and Practices
  - Real-Time Embedded Concepts (HW, FW, SW)
  - Concurrent programming
  - RT Scheduling – Rate Monotonic
  - Feasibility and Safety Margin
  - RM LUB
  - Cyclic Executive, RTOS and OS + RT Extensions
  - Best Effort, SRT, HRT
  - Static and Dynamic Priorities
  - AMP vs. SMP and use of Linux SCHED_FIFO
  - Basic tracing and comparison to expected timing

- Course #2 – RT Theory and Analysis
  - Full derivation of the RM LUB
  - Dynamic Priority timing analysis practice and theory
  - In-depth practice with Exact Worst-Case analysis
  - Issues with RMA, shared memory, priority inversion
  - In-depth look at dynamic priorities
  - Wrap-up of the state or practices, starting point for RT R&D

Course Focus

Designs with no SPOF (Single Point of Failure)

Full resource view of computer CPU + I/O + Memory (Power)



Course launched on Aug 20, 2020

Real-Time Embedded Systems Concepts and Practices

Instructor: Sam Siewert

Launched

Course launched on Aug 20, 2020

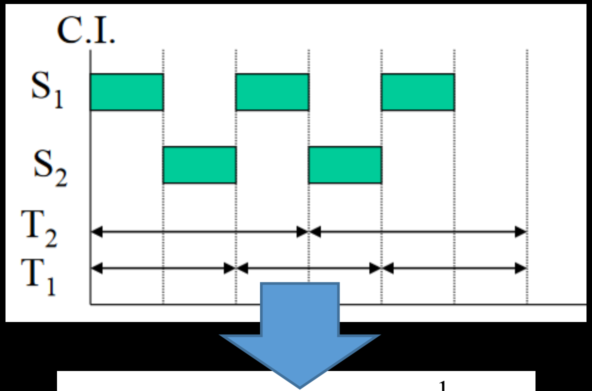
Real-Time Embedded Systems Theory and Analysis

Instructor: Sam Siewert

Launched

RTES Theory and Analysis (Review)

- Focus on CPU and Shared Memory RT Services (Beyond Concepts)
  - RM Least Upper Bound – Derivation and **what it means**
  - Exact Feasibility Analysis – **Lehoczky, Sha, Ding**
  - Comparison of fixed and **dynamic priority policies**
    - **EDF, LLF, Enhanced LLF**
    - **SCHED_DEADLINE for Linux (*new!*)**
  - Real-Time Service Design Patterns
  - Analysis and Verification of Real-Time
  - Secondary Resources and Synchronization
    - **Priority Inversion**
    - **Deadlock, Livelock**
  - Solutions to Design Challenges with RMA
    - Solved – PIP, PCP/PCEP, T < D, T > D, and Period Transform
    - Research Topics – RTES with VMs, RT SMP (vs. AMP or AMP emulation practice), RT with Co-processors
    - Linux – RT_PREEMPT patch, kernel/user space, RT Linux distributions



$$U = \sum_{i=1}^{m} (Ci/Ti) \le m(2^{\frac{1}{m}} - 1)$$

Discrete Math

Calculus Bound

Exact Analysis of Worst Case

Key Background Papers

Beyond RM LUB and Fixes

Lehoczky, John, Lui Sha, and Yuqin Ding. "The rate monotonic scheduling algorithm: Exact characterization and average case behavior." *RTSS*. Vol. 89. 1989.

Buttazzo, Giorgio C. "Rate monotonic vs. EDF: Judgment day." *International Workshop on Embedded Software*. Springer, Berlin, Heidelberg, 2003.

Design of Mission-Critical Apps & Systems (Goals)

- Focus on Memory, I/O, and Build of no SPOF Apps and Systems
  - Performance Tuning and Timing Analysis for RMA (deadline challenges)
  - I/O Device Interfaces and Drivers
  - Working memory, persistent memory, error detection and correction
  - Design concept for High Availability
  - Design concept for High Reliability
  - Integration of RMA for SRT and HRT with HA/HR Systems
  - Applications – Digital Media (Audio, Video, Computer Vision)
  - Systems – UAS/UAV, Small Satellites, Avionics, and Robotics