Introduction

- RTEMS is a tool designed specifically for real-time embedded systems
- The RTEMS product is an executive not a full operating system
- It provides a development environment:
  - Compilers
  - Debuggers
  - Support for target hardware
Ancient history

- Developed for U.S. Army missile command beginning in 1988
- Open-source license
- Maintained and commercially supported by OnLine Applications Research
- Written to Real-Time Executive Interface Definition
  - VMEBus Industry Trade Association (VITA)
  - Proposal originally sponsored by Motorola.
  - Matched pSOS+ and VMEEexec.
- C and ADA bindings
- “Super Core” with various APIs
  - RTEMS
  - POSIX
  - ITRON
- FreeBSD (4.3?) network stack ported to support EPICS
RTEMS Architecture

BSD TCP/IP Stack
- CORBA
- SNMP
- RPC/XDR
- TFTP
- HTTPD
- DHCP
- ICMP
- BOOTP
- PPP

Add-on Libraries
- zlib
- readline
- ncurses
- Tcl

Add-on Libraries
- libav

GUIs
- OpenPalm
- MicroWindows
- OpenGUI
- FLTK
- picoTk
- NanoX

Ada95
- GNAT
- Classic API Bindings

POSIX Compliant Filesystem
- IMFS
- TARFS
- FAT
- TFTP client
- NFS client

Remote Debugging

SAPI

Classic API

POSIX Threads

Performance Monitoring API

μ-ITRON API

SuperCore

Board Support Package

SuperCore CPU

LibCPU

LibCHIP

Hardware
**RTEMS Supported APIs**

- RTEID/ORKID
- POSIX 1003.1b
- ANSI C Standard Library
- ITRON

Note that some “standards” imply others
- POSIX 1003.1b references ANSI C Library
- EL/IX references POSIX and BSD
Communication and Synchronization Mechanisms

- Classic API Managers
  - Semaphores (include mutex functionality)
  - Events
  - Message Queues
  - Signals

- POSIX API Managers
  - Semaphores
  - Mutexes
  - Condition Variables
  - Message Queues
  - Signals
Memory Allocation

- Classic API Managers
  - Partitions
  - Regions
  - Dual-Ported Memory
- ANSI/ISO C Memory Allocation
  - malloc/free
- BSD Network Memory Allocation
  - mbufs
Interrupt Processing

- Quick response
- Simplifies user ISRs by allowing high-level language usage
- May alter task execution
- Interrupts disabled for minimal time in services
- Can incorporate board specific vectoring for chained interrupts or PICs
Portability

- Major design goal of RTEMS
- Isolate hardware dependencies
- Developed in a high level language
- Source code availability
- Available on multiple processor families
- Numerous board support packages included
- C, C++, and Ada applications are supported
- Applications port easily to other processors
Processors Supported by RTEMS

- ARM
- Motorola MC680x0
- Motorola MC683xx
- Motorola Coldfire
- PowerPC
- Intel i386 and above
- Intel i960
- MIPS

- OpenCores OR32
- SPARC
- AMD A29K
- Hewlett-Packard PA-RISC
- Hitachi H8
- Hitachi SH
- Texas Instruments C3x
- Texas Instruments C4x
User Extension Points

- Task creation
- Task initiation
- Task restart
- Task deletion
- Context switch
- Task exit
- Fatal errors

- Example – ‘spy’ task CPU usage monitoring
Development Environment

- Based on GNU Tools
  - GNU Compiler Collection (C, C++, Ada, Java, Fortran)
  - GNU Debugger
- GNU autoconf scripts for configuring RTEMS
- Makefile templates for application software and custom device drivers
  - Isolates processor and language dependencies
- OAR is an active member of the free software effort
  - Compiler/library changes get folded back into official distributions
- Newlib
RTEMS Task Scheduling

- Priority based (strict)
- Timeslicing
- Round-robin
- Preemption
- Rate Monotonic Scheduling
- Sporadic servers
What those familiar with vxWorks will find missing…..

- The range of hardware, particularly VME CPUs and PC network interfaces, for which support exists
- The nice, comfortable, integrated development environment
- The common boot procedure and configuration technique
- The vxWorks shell
  - EPICS provides the IOC shell
  - Till Straumann has written Cexp, a C expression parser
- The telnet access to the console
  - RTEMS version tested with the EPICS IOC shell – needs work
- Dynamic loading
  - The Cexp package provides this, too
- The license fee