Learning Real-Time Programming Concepts through VxWorks Lab Experiments

Andrew J. Kornecki
korn@db.erau.edu
Embry Riddle Aeronautical University, Daytona Beach, FL

Janusz Zalewski
jza@ece.engr.ucf.edu
University of Central Florida, Orlando, FL

Daniel Eyassu
daniel.eyassu@lmco.com
Lockheed-Martin Information Systems, Orlando, FL

CSEE@T2000 Paper Presentation

Agenda
- Background
- Laboratory Infrastructure
- Tornado
- Experiment Format
- Real Time Concepts
- Class Projects
- Closing Remarks

Background (1)
- Modern computer applications:
  - have predictable and guaranteed timing behavior thus failing if the timing constraints are not met (real-time)
  - interact with and control the environment (safety critical)
  - operate continuously as a part of a larger system (embedded)

Background (2)
- Therefore:
  - industry badly needs engineers familiar with concepts and practices applicable to development of time-critical reactive systems
  - concepts of timing, concurrency, synchronization and communication, resource sharing, and external device handling need to be addressed in computing programs

Real Time System Components
- Hardware - processor, bus, peripheral devices, memory, cache
- Operating System - scheduling, thread management, memory management, interrupt handling, input/output, application programming interface (API), priority inheritance avoidance
- Application Design - methodologies, CASE Tools, notations, process
- Application Implementation - synchronization, communication, threads, development support, run-time support

Nature of Real-Time Software:
- timely response to events
- dynamic tasks to handle for events, entities, and functions (process, thread)
- tasks activated periodically or sporadically
- completion of each task before the deadline
- task interaction through synchronization and communication
- system shall not hang/crash
Application Domains

- process control (chemical industry, food processing)
- robotics (manufacturing, automated control)
- avionics (flight management, GPS)
- aerospace (jet engine control, fly-by-wire)
- military (weapon management, encryption)
- data collection (acquisition, signal processing)
- communication (fax machines, digital phones)
- appliances (microwave, dishwasher, thermostats)
- automotive (engine/cruise control, anti-lock brakes)
- computer peripherals (printers, terminals, modems)

The Development Environment

- native environment: development on the same platform as the resulting executable
- host/target environment: development on the host with the executable downloaded to target

Laboratory Infrastructure

- Students must have access to a development environment supporting the entire software life-cycle
- Students must be able to develop code on the host and download, debug and test on the target
- Typical software components of the lab must include CASE tools, development tools, real-time kernels, performance analysis tools
- Selected environment have been Tornado, courtesy of a software grant from Wind River System (WRS) Academic Program

WRS Tornado - Components

- VxWorks Kernel on target
- Network and interfacing facilities:
  - Target Server on the host and Debug Agent on the target
  - Boot ROM on target
  - Module Loader and Symbol Table
- Development tools on the host:
  - WindSh - C and Tk/Tcl interface to target
  - CrossWind - graphic GNU gdb debugger
  - Browser - graphic presentation of target code
  - WindView - graphic logic analyzer
  - VxSim - target simulator

WRS Tornado - VxWorks

- VxWorks - a widely adopted industrial real-time operating system (RTOS)
- VxWorks is flexible, with powerful API’s, scaleable, reliable, used in mission-critical applications
- VxWorks micro-kernel supports real-time features (fast multitasking, hardware interrupts, both priority-preemptive and round-robin scheduling, efficient inter-task communication mechanisms, etc.)
Experiment Format - General
- Experiments are designed to be completed by a student independently, while learning the theory component in the classroom.
- Lab description is accessible from the ERAU Real-Time Lab Web server (http://rt.db.erau.edu) to provide an easy access to the experiments both from the laboratory and from home.
- Each lab experiment contains: introduction, objectives, description, example program, procedures, follow on experiment, and additional information.

Experiment Format - Topics
- Timing
- Multitasking and Concurrency
- Synchronization and Semaphores
- Communication and Message Queues
- Scheduling
  - round robin
  - priority pre-emptive
  - priority inversion
- Signals and Timers
- Interrupts

Class Projects - General
- Following the lab work, students are engaged in a small team project requiring the team to produce software life-cycle artifacts.
- The team uses elements of Personal/Team Software Process.
- The project deliverables include Internet accessible documentation and in-class presentation with the system demonstration.
- The projects are implemented on VME VxWorks target with user interface on a remote UNIX/WindowNT workstation (TCP/IP).

Class Projects - Examples
- Real-Time Data Acquisition and Control
- Real-Time GUI Implementation
- TCAS Simulator
- Autonomous Lunar Explorer
- Web Game
- Avionics Data Acquisition
- Security System
- Coffee Maker
- ...
  Check ERAU CS450 and MSE545 course pages

Conclusions
- Laboratory experiments facilitate learning real-time concepts.
- Lab settings encourage experimentation and analysis of results.
- Web interface provides easy access.
- Projects challenge students giving them opportunity to apply the concepts learned in classroom.