Deos SafeMC™


Presenter: Theresa Rickman
Military Aerospace Accounts

- Flight Software Workshop -
Thursday December 7th, 2017
Leading provider of mission/safety-critical software solutions for 30 years.

- Headquarters in Phoenix, AZ
  - World-wide presence
- Primary market: Certifiable avionics software
Core Competencies

- **Certifiable, safety-critical RTOS**
  - Deos (ARINC-653, RMA, or hybrid)
    - One product line across all processor families
    - *Always* built for certification
      - No Deos versus Deos 653
      - No Deos versus Deos 178
      - one division, one code base

- **Integrated Development Environment (IDE)**
  - Development, testing & analysis tools

- **DO-178/ED-12 certification expertise**
  - First DO-178 DAL-A (Ada) product released in 1992
  - We perform our own certification work
  - We defend our certification artifacts during all audits
  - We do not reverse engineer certification artifacts
Background

• Deos designed from the ground up for:
  • Mission-/safety-critical applications & DAL A
  • Real-time performance
  • Integrated Modular Avionics (IMA)

• 1998: Initial certification baseline (DAL A)
• 2017: Latest certification baseline (DAL A)

... a mature, trusted product, widely deployed for over 15 years.
Certification Baselines (DAL A)

A single line of on-going DAL-A development

Jun 98 v 5.0 TC
Jun 01 v 6.5.0 FAA
May 02 v 6.9.2 FAA/JAA
Jul 02 v 6.12.0 FAA/JAA
Mar 05 v7.1.0 EASA/FAA
Dec 09 v7.2.3 EASA/FAA
May 11 v7.6.2 EASA/FAA
Aug 13 v7.10.6 TC/DERs
Mar 15 v8.3.1 KCA/FAA
May 17 v8.4.2 TC

people

product

continuity

plans

processes, procedures & tools

… best-in-class certification record, well-known & respected in the industry.

© 2017

This is a non-ITAR presentation, for public release and reproduction from FSW website.
Deos Deployments

Aircraft
- Sixteen commercial airframes
- Fifteen Business Jet airframes
- Two helicopters
- Eight Military Airframes

Avionics Functions
- Air Data Computer
- Air Data Inertial Reference Unit
- Cockpit Video
- Communications & Radios
- Data Recorders
- De-Icing
- Digital Engine Controller
- Displays
- Electronic Flight Bag
- Engine Control
- Enhanced Ground Proximity Warning
- Flight Controls
- Instrumentation
- Flight Management
- Health Management
- Maintenance
- Power Distribution
- Traffic Collision Avoidance System
- Weather Radar

... best-in-class service history on a broad range of aircraft & functions.

© 2017

This is a non-ITAR presentation, for public release and reproduction from FSW website.
Supported CPUs

• **ARM A**
  - i.MX6 (Cortex A9), Altera (Cyclone V),
  - Xilinx (Cortex A9)
  - Xilinx Ultrascale+ (Cortex A53) Xilinx ZCU102 reference board

• **Intel** (and compatibles)
  - x86 (Pentium, Atom, Celeron, Core (duo, i7, …), AMD GSeries)

• **PowerPC**
  - e6500/e5500/e500/e500mc (QorIQ), e200 micro-controllers (51xx, 52xx, 56xx, 57xx), G2/603e (82xx), G3 (7xx), e300 (83xx), G4/e600 (74xx, 86xx), 405x, 440x, 465x

• **MIPS**
  - R3000

... support for more processor families & processor cores than any other partitioned certifiable COTS RTOS...
Deos High-Level Architecture

…loosely-coupled, modular application software partitions.
Deos Highlights

• Pedigree – Unmatched record of deployment, support & certification for a partitioned RTOS
  • >10,000 aircraft, >10 Million of flight hours, > 40 aircraft types, >100 certifications

• Features
  • Time, space & resource partitioning with ARINC 653 and/or RMA scheduling
  • DAL-A Linker/loader for binary modularity - Enables reuse of software & certification credits, and minimizes change impacts
  • Data Distribution Service (IOI)
  • Standard: TCP/IP Optional: File system, ARINC 664/AFDX, ARINC-615 TDL, USB

• Performance
  • Cache partitioning, low system tick overheads, linear API performance, etc.
  • Slack scheduling & time budget transfer
  • Multicore Deos SafeMC scheduling

• Tooling
  • Ethernet & FTP based development – with PC-based processor simulator
  • Compiler independent (i.e., current version)
  • All tooling applicable through V&V (and deployment in some cases)
  • Tooling to determine WCE for apps and target
  • Source/Object code coverage tool provided

© 2017

This is a non-ITAR presentation, for public release and reproduction from FSW website.
Deos/RTEMS architecture

POSIX coupled with ARINC 653 on a DAL A certifiable platform

RTEMS POSIX Partition
- RTEMS POSIX Thread Scheduler & POSIX API Library
- RTEMS paravirtualized Deos Adapter

Deos 653 Partition
- ARINC 653 User Executable
- ARINC653 Process Scheduler & 653 P1 API Library
- IOI Lib

Deos RMA Process
- Deos RMA User Executable
- Deos API Library
- TCP/IP (LWIP)
- IOI Lib

User Mode
- Shared Memory

Kernel Mode
- Deos Registry
- PAL
- Target System Hardware and CPU

This is a non-ITAR presentation, for public release and reproduction from FSW website.
Multicore Guidance CAST-32A

- **Software Planning**
  - How many processors, what OS architectures and how they manage the cores.

- **Planning and configuration of MCP**
  - Document MCP settings to satisfy requirements
  - Document MCP settings contingency plans
  - Document resource partitioning and how you plan to mitigate contention issues.

- **Interference Channels and Resource Usage**
  - Identified the interference channels and chosen means of mitigation of the interference.

- **Software Verification**
  - Verify all the hosted software components function correctly and have sufficient time to complete their execution in the final configuration.
  - Verify that the data and control couplings between all the individual software components hosted on the same core or on different cores.

- **Error Detection and Handling, and Safety Nets**

- **Reporting of Compliance with the Objectives of this Document**

https://www.faa.gov/aircraft/air_cert/design_approvals/air_software/cast/cast_papers/media/cast-32A.pdf

---

© 2017

This is a non-ITAR presentation, for public release and reproduction from FSW website.
Safety Critical Multi-Core

Safety Critical Multicore Concerns:

1. Bound & control interference patterns
   A. Minimize contention for shared resources (e.g., cache & memory)
   B. Coordinate behaviors amongst cores

2. Getting good value from adding secondary cores
   Example concern: WCE will increase due to multicore interference patterns

Deos Multicore Solutions: SafeMC

1. Reduce interference patterns and reduce WCETs
   A. Memory pooling & cache partitioning
   B. Safe-scheduling

2. Performance enhancing features
   A. Slack scheduling, including Window Activation for multicore
      Recovers and applies additional slack resulting from higher WCETs
   B. Enable deterministic interrupting devices

1. Patented
Cache In Deterministic Systems

• The greatest performance factor for modern processors
• Growing in size and number of levels (e.g., L1, L2, and L3)
• Left uncontrolled, cache will cause performance variability (e.g., cache thrashing which increases the gap between best and worst case execution time (WCET))

Studies show that cache variability must be resolved in deterministic multicore systems
Cache Performance Variability

Cache variability is a significant issue for deterministic systems, that must be solved. Fixes include:

1. **Cache flushing** (e.g., flush cache between applications)
   - Good: Reduces performance variability
   - Bad: Forces cache flush overheads at an application context switch

2. **Disabling of cache**
   - Good: Eliminates cache performance variability
   - Bad: Huge performance penalty – forces the processor to a low level of performance. Also impractical for multicore processors.

3. **Cache Partitioning** – Several option with various results
   - Deos cache partitioning (patented)
   - Cache locking
Cache Partitioning with Deos

- Partitions Cache per Application
  - Best performance (reduces WCE) by eliminating shared cache thrashing across applications
  - Applications don’t have to manage cache
    - No H/W cache locking instructions used.

- Portable
  - Processor Agnostic (does not require H/W “hooks”)
  - Memory pools and partitions defined in XML configuration file, NOT in source code
  - No re-verification required platform to platform

- Applicable to both Mono-core & Multicore (x86, PPC and ARM)

- Patented Mechanism

This is a non-ITAR presentation, for public release and reproduction from FSW website.
Memory Pools & Cache Partitioning

- Partitions memory segmented into “pools”
- Cache segmented into “partitions” per core
- RAM addresses Segmented per application

**Core 0**
- APP 0
- APP 1

**Core 1**
- APP 2
- APP 3
- APP 4

**Shared Cache**
- pool 0
- pool 1
- pool 2
- pool 3
- pool 4

**RAM**
- APP 0
- APP 1

**Off Chip RAM (example)**
- APP 2
- APP 3
- APP 4

- Reduces cache thrashing
- XML Configuration (portable)
- Cache contention is bounded

... optimizes application ACET/WCET behaviors & bounds WCET behavior.

- Partition per application per core.
- No application specific code
- No cache locking instructions used.

---

© 2017

Safety Critical Software Solutions for Mission Critical Systems
Cache Partitioning – Bounding WCETs

- Bounds & controls cache interference patterns
- Can dramatically improve WCET performance
Deos Safe Scheduling for Multicore

- Bounds, controls & minimizes cross-core contention
  - Major frame partitioned into “windows”
  - Window boundaries align across cores
- Multiple scheduler/API types available
- Fine Grain locking for resource protection
- Allows for a mix of safety apps, or safety & non-safety apps
  … optimizes application ACET/WCET behaviors and bounds WCET behavior.
Fine Grain Locking

• In the kernel all locking is done in a single core space only, therefore, no cross core blocking is possible.
  • No cross core locks (No resources used for all cores)
  • No single lock for scheduling (each core has a scheduler)
  • No single lock for all kernel interface objects (each object created has its own lock)

• Cross core blocking is only possible if a developer designs it to happen
  • Threads on different cores share a kernel interface object (semaphore, event, mailbox, etc.)
  • Thread creates another thread and schedules it on a different core
  • Threads of different cores share a memory pool
  • In these cases affects limited to the cores in question and not the others.
Simple Multicore Scheduling

The simplest extension of RTOS Scheduling to multicore is to extend the existing single RTOS Scheduler to schedule multiple cores.

Issue: Critical locks are now system wide and extend across all cores (aka Coarse locking), this exacerbates cross core resource contention.

Legend: = scheduler

This is a non-ITAR presentation, for public release and reproduction from FSW website.
Deos’s Safe Scheduling runs system wide time windows and runs “per core” schedulers

- Critical locks (and critical code sections) are per core
- Enables fine grained locks during scheduling
- Limits cross core contention, system wide blocking and Worst Case Execution Times (WCETs)

Likewise, Deos implements fine grained locking for all application resources
Memory Throttling

• Utilize Hardware Built-in Performance Counters
  • Certain processors have hardware capability to count processor level events.
  • Setup CPU to send interrupt when a particular performance counter threshold is reached.
  • Last level cache miss is a selectable event to be counted.
  • When threshold is hit for a particular partition, it will not be scheduled until the next major frame.

This is a non-ITAR presentation, for public release and reproduction from FSW website.
Summary

- RTEMS users can now leverage Deos for hard time and space partitioning with a DAL A certifiable platform
- RTEMS integration package and support available directly through DDC-I
- Deos/RTEMS supports the same code base across Intel, Arm & PPC with multicore and 653 on all processor families
- Deos has unique capabilities for safety critical multicore certification:
  - Safe Scheduling
  - Cache Partitioning
  - Fine Grain Locking
  - RMA/653 scheduling
- **Ultrascale+ with ARINC 653 libraries available today**
Thank YOU

• Contact information
  • Theresa Rickman trickman@ddci.com
  • Gary Gilliland gilliland@ddci.com
  • Joel Sherrill (OAR) joel.sherrill@oarcorp.com