CORBA for Power Management

Rockwell Collins Advanced Technology Center
Advanced Computing Systems
Presented By: David Haverkamp, Dr. David Jensen, Steven Koenck
E-Mail: dahaverk@rockwellcollins.com
dwjensen@rockwellcollins.com
sekoenck@rockwellcollins.com
Introduction

- CORBA is being introduced in embedded systems
  - Benefits proven in workstation and networked systems
- Many embedded systems have difficult design constraints
  - Power limitations because of heat and/or batteries
  - Size and weight because of environment
- DARPA’s Power Aware Computing / Communications (PAC/C) provides an excellent test case
  - Mission Aware Power Management (MAPM) program will introduce PAC/C technologies into the JTRS platform
  - PAC/C program involvement has provided exposure to many novel technologies
  - MAPM funding has produced some of these results
- Our involvement in the Joint Tactical Radio System (JTRS) program has allowed us to research the application of CORBA in this challenging environment.
- In this talk, we discuss some preliminary approaches used in our investigations
  - Power management interface using CORBA
  - An approach for CORBA support in low power embedded systems
Power Management Interface Background

• Like all system engineering decisions, the choice of an appropriate power management policy in the early stages of the design process drives cost, portability, and maintainability.

• CORBA power management interfaces have the potential to introduce power and energy management to distributed systems.

• The OMG offers a potential venue for the creation of a standard set of interfaces for power management of Distributed Embedded systems.
• Power Scheduler acts as a Client sending Control messages through Mission Services.

• Mission Service directs command messages to either:
  – “CORBA Devices” invokes device drivers specific to:
    – General Purpose uProcessor, DSP, Modem, etc.
  – Domain Services
    – Software Radio Specific components.
      – setup of waveforms channel application stream
CORBA Interfaces to system Devices

• Embedded systems can be composed of many system processors
  – Software Radio
    – General purpose uP
    – DSP’s
    – Modem
    – Power Supplies.
  – Device power saving capabilities not all equal

• Power Management Interfaces need generic power saving policies and modes
  – stand-by, idle, reduced power, lower quality, and complete shutdown/halt.

• Advanced Configuration and Power Interface Specification (APCI)
  – uP interfaces can use some APCI concepts for generic device interfaces
  – operating system (OS)-directed configuration for
    – motherboard device configuration and
    – power management of both devices and entire systems.
  – Not currently oriented toward Distributed systems.

• Power Management via CORBA allows for an Abstract view of the underlying distributed system
• Scheduler control via CORBA interfaces allow a system oriented power management policy

  – Eager
  
  – Power implications easy, processor can always be on.
  
  – Could manipulate the processor clock to provide “just enough” performance

  – Lazy

  – Efficient but more involved
  
  – “Suspend”, “standby” or “shutdown” processors not being used

  – Wake-up policy implications
Current State of Power Aware Computing

• Power interfaces are conceptually simple
  – definition and implementation becomes complex because of the number of subsystems and power modes, application and operating system demands, and provisions for future extensions.

• Underlying Device (uP,DSP,etc) effects System Power Save Capabilities
  – Some specific uP have Dynamic Voltage and Frequency Scaling
    – Transmeta - Crusoe
    – Intel - StrongARM and XScale
    – IBM - PowerPC 405LP
  – Some power management is more appropriate for automatic compiler insertion in low-level assembly or as part of the BSP.
    – floating point is not used in applications so don’t clock FP unit
    – Can be a hardware register setup configuration or uP specific assembly instruction

• CORBA API being applied to JTRS platform and evaluated under DARPA PAC/C program
CORBA for Low Power Embedded Devices

Rockwell Collins Advanced Technology Center
Advanced Computing Systems
Presented By: David Haverkamp, Dr. David Jensen, Steven Koenck
E-Mail: dahaverk@rockwellcollins.com
dwjensen@rockwellcollins.com
sekoenck@rockwellcollins.com
Low Power Embedded Devices

• Target Platforms
  – Handheld Devices - Battery operated
  – Unmanned Air Vehicles – Mission Constraints
  – Ground Sensors – Long Life

• Intimidating set of requirements (Environmental Constraints)
  – Size & weight, limited battery capacity and available power
  – Power limitations affect the microprocessor selection, memory size, and software implementation.

• In contrast, embedded designers want the advantages of portability, maintainability, and design cost savings that CORBA designs offer.

• However, it is recognized that the current generation of commercial CORBA software and their supporting commercial operating systems assert a significant (often unattainable) power demand on these low power embedded devices.

• Future JTRS Clusters will support handheld devices and Unmanned Air Vehicles (UAVs) and introduce even greater power constraints.
  – Types of clusters – airborne, handheld, UAV, weapon data-link, munitions
Research Approach to Reduce Power Overhead

• We have identified a research path to reduce the power overhead introduced today with CORBA.
  – Experience in microprocessors, microcode, adaptive computing, and hardware-software co-design.

• Basic premise is to move computationally intensive algorithms from software to hardware.
  – Known power savings by eliminating microprocessor overhead through direct implementation in gates.
  – Dramatic acceleration of function \( \geq 100x \)
  – Embedded systems typically have fixed computation requirements
  – Moving algorithms to hardware allows slowing clock of entire system!

• We are applying this concept to the Software Communication Architecture
  – Rockwell Collins IRAD project: “Software Communication Architecture Processing Environment” (SCAPE)
  – Concept genesis under the DARPA PAC/C MAPM project
SCAPE Project

• **Software Communication Architecture Processing Environment (SCAPE) Approach**
  – Gates are cheap and plentiful - move computationally intensive software to hardware
  – Rationale - Accelerate functions to slow down clock.

• **5 Methods for Acceleration**
  – Hardwired Logic - memory mapped - common API for supportability
  – FPGA Logic - memory mapped - common API for supportability
  – Microcode - how do we support? document?
  – Traps to Software
  – Software logic (algorithm improvements)

• **Constructive Plan**
  – Identify SCA computing demands
  – Profile software performance
  – Accelerate computationally intense functions
  – Document results
SCAPE Power Savings

- Notional Example of Power Savings through SCAPE
- Subroutines represent CORBA, POSIX, and Application routines
- Power being expended initially by microprocessor

![Diagram showing power savings through SCAPE](image)

- Hardware Subroutines
  - uProc, uCode, Logic, FPGA
  - Clock 500MHz

- Power per Hardware
  - uP, uC, L, F, total

- Active Subroutine
  - C, F, J
SCAPE Power Savings - Continued

- Top three computationally intensive tasks migrated to hardware
- Custom microcode, logic, and FPGA implementation accelerate functions
- Power now being expended by all hardware elements

![Diagram showing power distribution and active subroutine over time.](image-url)
SCAPE Power Savings – First Round

• Real time requirements haven’t changed!
• Clock speed can be dramatically reduced – along with power expended!
• New set of subroutines can now be identified and repeat process

**Diagram:**
- **SOFTWARE**
  - System
  - Waveform
  - MPEG4
  - GPS
- **SUBROUTINES**
  - A
  - B
  - C
  - D
  - E
  - F
  - G
  - H
  - I
  - J
  - K
  - L
- **HARDWARE**
  - uProc
  - uCode
  - Logic
  - FPGA
  - Clock 250MHz

**Charts:**
- Power per Hardware
- Active Subroutine
Will this work?

• Three hardware areas to target computationally intensive software
  – Custom logic – very fast - expensive ASIC
  – Custom microcode – 10x typical vs software
  – FPGA logic – fast – relatively inexpensive vs ASIC

• Benefits of CORBA enable this approach
  – Interfaces are Standardized
  – Software developed on workstations using CORBA will port easily to SCAPE.
  – Optimized SCA Environment

• Identify key OS and CORBA functions to optimize
  – High usage
  – Compute intensive
During previous PAC/C PI meeting MAPM Software Radio Workshop - JTRS Engineering identified POSIX, CORBA, and network protocol processing as more computationally intensive.

Improving POSIX & CORBA will be more beneficial for this environment than focusing on traditional DSP algorithms.

Based on experience, Marshalling & Demarshalling algorithms (CORBA) are being analyzed for implementation in hardware (FPGA)

Beginning to profile SCA (Software Communication Architecture) to identify additional algorithms

Some preliminary profiling results from a DII COE server suggest that the 80:20 rule will apply to the SCAPE effort.

Pareto Principle - the 80:20 rule - "A minority of input produces the majority of results."
Conclusions

• CORBA offers many proven benefits including cost, portability, and maintainability.

• CORBA has challenges in the embedded systems market because of power and complexity overhead.

• We have identified approaches to ease the application of CORBA.
  – Power management APIs
  – Power efficient implementation of CORBA & POSIX.

• The power and energy saving benefits can be dramatic.
  – Rockwell Collins Mission Aware Power Management (MAPM) program under the DARPA PAC/C program has a goal to reduce energy use by 50% through the use of power management technologies.