UBICOMP

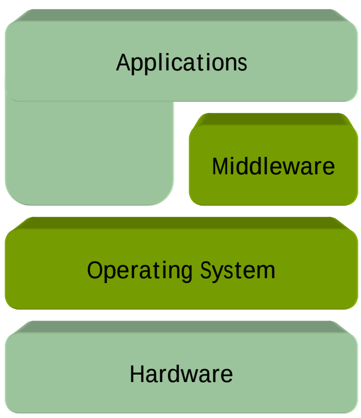
Episode 10: OS Issues

Hannes Frey and Peter Sturm
University of Trier

(C) 2004 AG SYSOFT – UNIVERSITY OF TRIER

Operating Systems …

- OS mediates between hardware and applications
- Goals
  - High-level abstractions
  - Efficiency
  - Utilization
- Sometimes performance penalties tolerated for good abstractions
- Different in embedded systems and ubiquitous computing?



UBICOMP (C) 2004 AG SYSOFT – UNIVERSITY OF TRIER

Requirements

- Memory
  - Minimal memory footprint at runtime
  - Virtual memory sometimes not possible
  - Configurability of OS
  - Portability
- Execution
  - Soft real-time and sometimes hard real-time capabilities
  - Easy to implement device drivers
  - Synchronization and communication primitives
- Licensing

UBICOMP (C) 2004 AG SYSOFT – UNIVERSITY OF TRIER

Candidate Systems

- Embedded Operating Systems
  - TinyOS
  - Symbian OS (mobile phones)
- True OS
  - Linux / Embedded Linux
  - Windows CE
  - Windows XP (?)
- Middleware
  - Various research systems
  - GecGo ☺ (see episode 8)
  - Jini

UBICOMP (C) 2004 AG SYSOFT – UNIVERSITY OF TRIER

10.1 TinyOS

Based on slides by
Arvind Easwaran

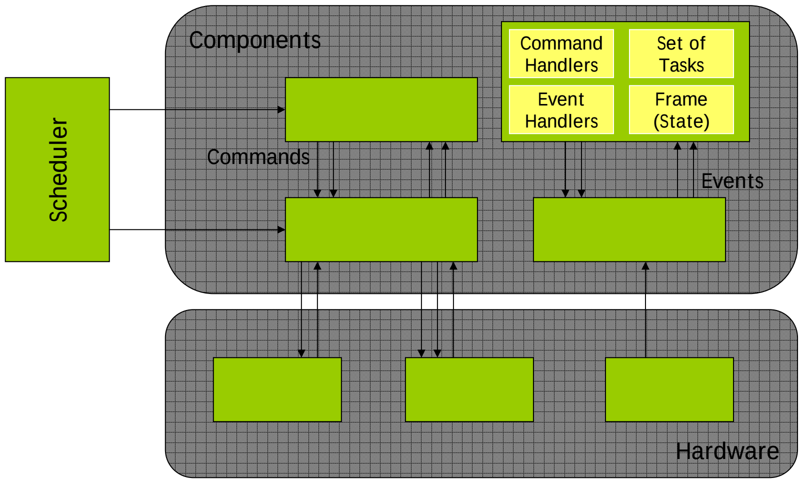
(C) 2004 AG SYSOFT – UNIVERSITY OF TRIER

Tiny OS

- Not really an operating system
  - No kernel
  - No process management
  - No virtual memory
  - Single shared stack
- Interrupt- and Event-driven architecture
  - Clock, Radio, and other hardware interrupts
  - Lower layer sends events to higher layer
- 2-level FIFO scheduler
  - Events and tasks
- Applications consist of interacting components
  - Software acting on and issuing of events

UBICOMP (C) 2004 AG SYSOFT – UNIVERSITY OF TRIER

TinyOS - Architecture



UBICOMP (C) 2004 AG SYSOFT - UNIVERSITY OF TRIER

Component Interface

- <CompName>.comp:

```
TOS_MODULE <CompName>;
ACCEPTS {
   // Command signatures
};
HANDLES {
   // Event signatures
};
USES … // Commands
SIGNALS … // Events
```

UBICOMP (C) 2004 AG SYSOFT - UNIVERSITY OF TRIER

Component Implementation

<CompName>**.c**

```
#define TOS_FRAME_TYPE
TOS_FRAME_BEGIN (<CompName>_frame) {
    // state declaration
}
TOS_FRAME_END (<CompName>_frame);
char TOS_COMMAND (<command_name>) () {
    // command implementation
}
char TOS_EVENT (<event_name>) () {
    // event implementation
}
```

UBICOMP (C) 2004 AG SYSOFT – UNIVERSITY OF TRIER

Component Description

- <CompName>**.desc**

```
INCLUDE {
    MAIN;
    <CompName>;
    <Comp_I>;
    <Comp_J>;
    …
};
// Wiring
<CompName>.<command>   <Comp_I>.<command>
…
<CompName>.<event> <Comp_J>.<event>
…
```

UBICOMP (C) 2004 AG SYSOFT – UNIVERSITY OF TRIER

Discussion

- No memory protection
  - Easy to corrupt and crash the system
- Heavy use of macros

UBICOMP (C) 2004 AG SYSOFT – UNIVERSITY OF TRIER

TinyOS - Scheduling

- 2-Level scheduler for events and tasks
- No real-time guarantess
- FIFO scheduling
  - Queue of size 7
- Tasks
  - Preemptable by events
  - Not preemted by other tasks
  - May call commands and events
  - Has assigned priority
- Lowest priority task dropped if queue full

UBICOMP (C) 2004 AG SYSOFT – UNIVERSITY OF TRIER

UBICOMP

10.2 Symbian OS

(C) 2004 AG SYSOFT – UNIVERSITY OF TRIER

UBICOMP

10.3 Linux

(C) 2004 AG SYSOFT – UNIVERSITY OF TRIER

UBICOMP

10.4 Windows XP

(C) 2004 AG SYSOFT – UNIVERSITY OF TRIER

Windows …

- What about Windows XP Home Edition or Professional Edition?

UBICOMP (C) 2004 AG SYSOFT – UNIVERSITY OF TRIER

Embedded XP

- Configurable Windows XP version
  - Separate GUI
  - Configuration support
- Tools
  - Target Analyzer (identify the required OS for a given hardware)
  - Target Designer (customize OS image)

UBICOMP (C) 2004 AG SYSOFT – UNIVERSITY OF TRIER

UBICOMP

10.5 Windows CE

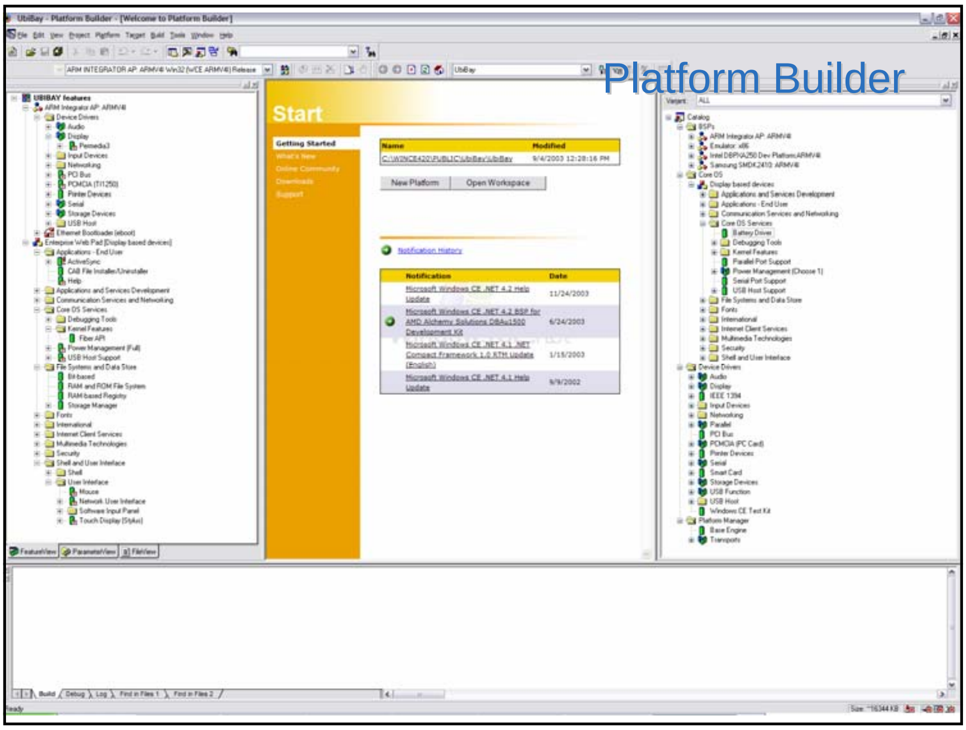
Slides by
John Eldrige and Mike Thomson (Microsoft)

(C) 2004 AG SYSOFT – UNIVERSITY OF TRIER

Windows CE

- Commercial operating system for embedded devices
- OS Customization
  - Platform Builder (OS customization)
- Application development
  - Embedded Visual C++
  - Embedded Visual Basic (not supported anymore)
  - .NET Compact Framework
  - All three variants available within MSDN AA

UBICOMP (C) 2004 AG SYSOFT – UNIVERSITY OF TRIER



Board Support Packages (BSP)

- Improve out-of-the-box experience
  - Evaluation & Learning
  - Prototype & Demonstration
- Shorten time to booting prototype
  - Sample drivers based on integrated peripherals
  - Many source examples available
- Decouple high-level app development from hardware and driver development
- At least one BSP per supported kernel included in PB, additional on the web
- Additional BSPs available on the web and included in reference hardware products

UBICOMP (C) 2004 AG SYSOFT – UNIVERSITY OF TRIER

Shipping BSPs in 4.2

Family	CPU	SDB Name	BSP	Kernel
ARM	Intel SA1110	Ship CSP for SA11x1	N/A	ARMV4
	Samsung S3C2410 ARM920	Samsung SMDK-2410 SDB ARM Integrator AP Dev Kit	Samsung_2410 ARMIntegrator	ARMV4I
	Intel XScale	Intel Lubbock Platform (DBPXA250)	XSC1BD	ARMV4 ARMV4I
MIPS	MIPS II flavors	Alchemy DBAu1500 SDB (web release) NEC DDB-Vr4122 (Eagle) SDB	DBAu1500 Eagle	MIPSII
	MIPS IV flavors (NEC Vr5432)	NEC Solution Gear Series	SG2_VR5500	MIPSII MIPSII_FP MIPSIV MIPSIV_FP
SHx	SH4	Hitachi US7750 HARP SDB ("Aspen")	Aspen	SH4
	SH3	Hitachi US7729 HARP SDB ("Keywest")	Keywest	SH3
x86	X86 (Intel, AMD, Via, SiS…)	Generic CE/PC machine	CEPC	x86
	Geode GX1 series	National Pompano Platform Any Geode based platform	Geode	x86

UBICOMP (C) 2004 AG SYSOFT – UNIVERSITY OF TRIER

Windows CE 4.2

- 32-Bit operating system
- Supports hard real-time applications
- Implements common Win32 API with some extensions

UBICOMP (C) 2004 AG SYSOFT – UNIVERSITY OF TRIER

Virtual Memory Basics

- Windows CE has a single flat 32-bit virtual address space that is shared by the entire system.
- “Virtual Addresses” refer to any address referenced by the CPU while the Memory Management Unit (MMU) is active.
- Every valid virtual address must map to some real physical address that can be used to identify a physical resource such as ROM, RAM, Flash, CPU registers, SoC components, bus-mapped components, etc.
- “Physical Addresses” are not directly addressable by the CPU except during initialization before the kernel has enabled the MMU.

UBICOMP (C) 2004 AG SYSOFT – UNIVERSITY OF TRIER

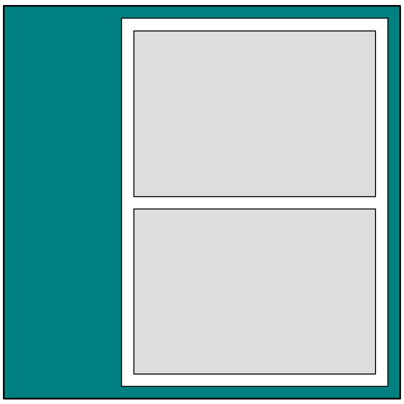
Virtual Memory Terminology

- Statically Mapped Virtual Address
  - A virtual address with a virtual-to-physical mapping that never changes
  - Can be used by kernel-mode code
- Dynamically Mapped Virtual Address
  - A virtual address with a virtual-to-physical mapping that can change (although it is not required to change)
  - Can be used by user-mode and kernel-mode code
  - Most addresses in lower 2 GB are dynamically mapped, A few addresses in upper 2 GB are dynamically mapped.

UBICOMP (C) 2004 AG SYSOFT – UNIVERSITY OF TRIER

32-bit Virtual Address Space

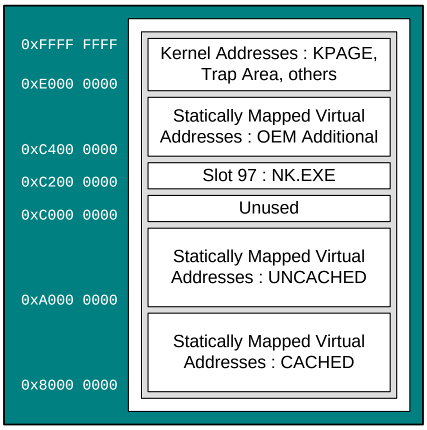
- The shared virtual address space keep be considered as two parts, kernel and user address spaces, each 2 GB.
- Kernel space is only accessible by threads with privileged access, called KMode.
- User space is accessible by all threads but is limited by process space protection.



UBICOMP (C) 2004 AG SYSOFT – UNIVERSITY OF TRIER

Kernel Address Space

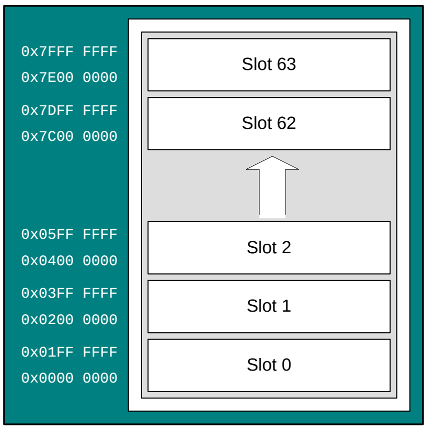
- Kernel Address Space contains statically mapped virtual addresses, NK.EXE pseudo-slot, and other kernel mappings.
- Up to 512 MB of physical resources can be mapped in the main statically mapped areas.
- Static Mapping under OEM control for ARM and x86 (OEMAddressTable).
- Static Mapping under CPU control for SHx and MIPS.



UBICOMP (C) 2004 AG SYSOFT – UNIVERSITY OF TRIER

User Address Space

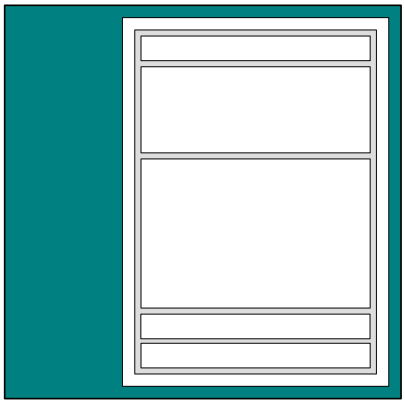
- User address space is divided into 64 “slots”
- Each slot is 32 MB
- A slot is a basic unit for virtual memory maintenance within the Windows CE kernel
- First 33 slots are used for processes
- Remaining slots for object store, memory mapped files and resource mappings



UBICOMP (C) 2004 AG SYSOFT – UNIVERSITY OF TRIER

Slot Usage Grouping

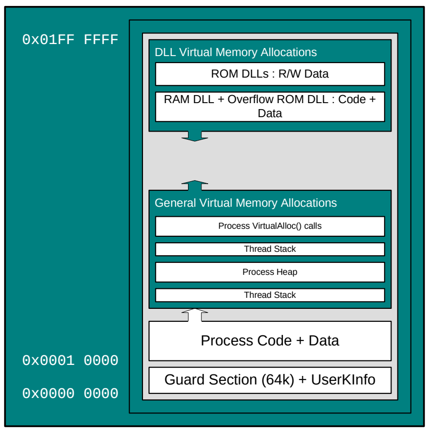
- Slot 0 is an alias to the currently running process's slot.
- A maximum of 32 processes can be running at one time.
- Threads may only access addresses within slots in which they have permissions.
- Object store is protected from all access outside of the filesystem.
- All memory mapped files are accessible by all threads.
- DLL Resources are accessible by all threads.



UBICOMP (C) 2004 AG SYSOFT – UNIVERSITY OF TRIER

Process Slot

- 32MB Process slot is shared by DLL, process, and all of its virtual allocations.
- All virtual allocations are 64kB-aligned.
- Pages may be committed within a virtual allocation on a page granularity (4kB).
- DLL allocations in a slot start at the high addresses and grow down.
- Process and general allocations start at the low addresses and grow up.



UBICOMP (C) 2004 AG SYSOFT – UNIVERSITY OF TRIER

Mapped Virtual Address (Dynamic)

- VirtualCopy
  - Maps process specific static virtual memory
  - Usable by the process that performed the mapping
  - Extends the OS memory support beyond 512MB
  - Specifies the physical to virtual mapping
  - Must first allocate virtual space with VirtualAlloc
  - Physical memory is mapped to the VirtualAlloc'd area
  - Used for mapping
    - Device I/O
    - Device specific memory

UBICOMP (C) 2004 AG SYSOFT – UNIVERSITY OF TRIER

Physically Contiguous Allocations

- Contiguous physical memory
  - AllocPhysMem
  - Guaranteed to be contiguous if successful
  - Succeeds only if the allocation size exists
  - No reshuffling of virtual to physical mappings
  - Useful for DMA, sharing to external devices

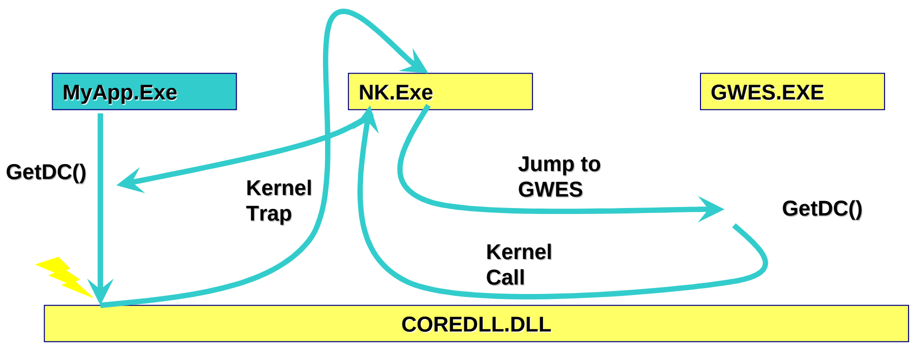
UBICOMP (C) 2004 AG SYSOFT – UNIVERSITY OF TRIER

System API Calls

- Many Win32 calls are calls to EXEs and not DLLs
- COREDLL provide a way to link a system API call to an system EXE
- Every system call causes an exception that is caught by the Kernel
  - Undefined address exception or CPU trap
- The Kernel then determines which EXE can fulfill the request
- During the whole process the user mode thread is the same thread that executes in the system EXE's process space
- As a thread migrates its access rights change to reflect what process it is operating in

UBICOMP (C) 2004 AG SYSOFT – UNIVERSITY OF TRIER

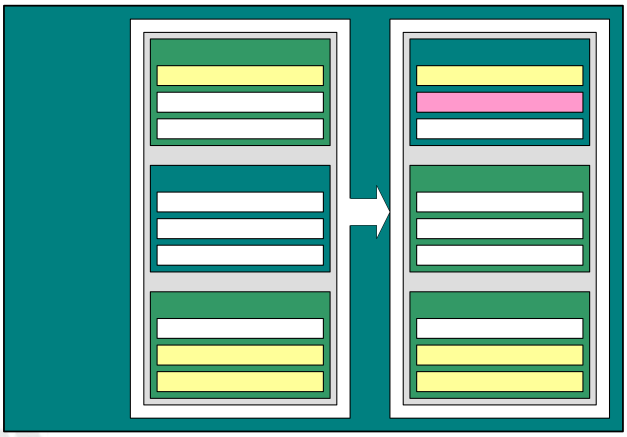
System API Calls



UBICOMP (C) 2004 AG SYSOFT – UNIVERSITY OF TRIER

Thread Migration

- The shared address space design partners tightly with the mechanism for system API calls (thread migration).



UBICOMP (C) 2004 AG SYSOFT – UNIVERSITY OF TRIER

APP.EXE Thread Stack

APP.EXE Process Heap

APP.EXE

DEVICE.EXE Thread Stack

DEVICE.EXE Process Heap

DEVICE.EXE

APP.EXE Thread Stack

APP.EXE Process Heap

APP.EXE

Process Pointer Mapping

- Kernel maps all parameters explicitly typed to be pointers between user process and system process
  - Does not map pointers embedded in structures
  - Mapping should only need to occur for drivers in Device Managers/Services space or in the OAL
- Memory buffer provided to a driver under Device Manager does not have direct access to the memory
  - The memory is owned by the calling process

UBICOMP (C) 2004 AG SYSOFT – UNIVERSITY OF TRIER

Process Pointer Mapping

- You must map the memory into the Device Manager Space
  - Use MapPtrToProcess
    - Need to know calling process
    - Maps a slot 0 based address to the caller address space
- To find out the calling process
  - Use GetCallerProcess

UBICOMP (C) 2004 AG SYSOFT – UNIVERSITY OF TRIER

ISR Versus IST

- ISR
  - Interrupt Service Routine
  - Kernel mode service
  - Two types
    - Static
    - Installable
- IST
  - Interrupt Service Thread
  - User mode thread

UBICOMP (C) 2004 AG SYSOFT – UNIVERSITY OF TRIER

Static ISR

- Built into the Kernel
  - X86 and ARM can use C code
  - SHx and MIPS must use ASM
    - Limited register availability
- Communication path is from ISR to IST only
  - Single return value that can trigger an event
  - A buffer can be shared between the ISR and IST but the location must be predefined
- Nested ISR support
  - Based on the CPU
  - Based on the OEM's initialization
- Stack is provided by the Kernel
  - Limited size based on CPU

UBICOMP (C) 2004 AG SYSOFT – UNIVERSITY OF TRIER

Installable ISR

- Can be loaded into the Kernel dynamically
  - LoadIntChainHandler
    - Called from the IST or application
  - Several ISRs can service the same IRQ
- Loads a C DLL
  - DLL must be self contained
    - Not reference any other DLLs
- IOControl path
  - Communication path from IST to ISR

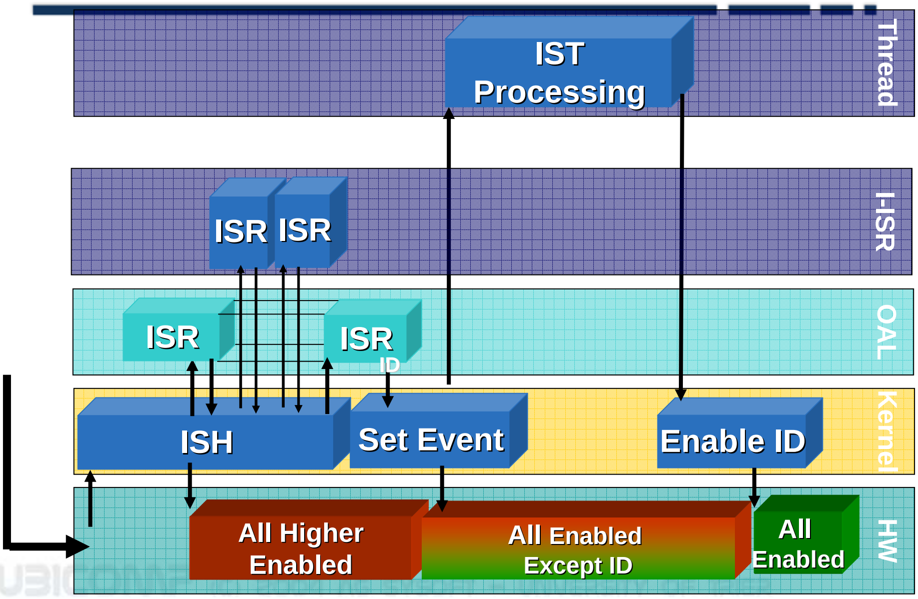
UBICOMP (C) 2004 AG SYSOFT – UNIVERSITY OF TRIER

Installable ISR

- Shared memory
  - Can be allocated dynamically when ISR is installed using an user defined kernel IOCTL
- OEM determining what IRQs can be serviced
  - Calls NKCallIntChain to pass the IRQ to installed ISRs
- ISRs are processed in order they were installed
  - First ISR to service the IRQ returns a SYSINTR_* value
    - Stops further processing
- Limited stack size base on CPU

UBICOMP (C) 2004 AG SYSOFT – UNIVERSITY OF TRIER

Windows CE 4.x Interrupts



Kernel Scheduler

- Preemptive multi-tasking
  - Thread runs until quantum completes
  - Thread runs until higher priority thread is ready to run
- Round-robin at a priority level
- Quantum is defined by the OEM and the Application
- Priority boosting only to correct priority inversion

UBICOMP (C) 2004 AG SYSOFT – UNIVERSITY OF TRIER

Where Latency Occurs

- For an ISR
  - The amount of time that interrupts are turned off
  - Time required for the kernel to vector to the ISR handler
    - Saving register, etc.
- For an IST
  - Time spent in an ISR and processing for an ISR
  - Time spent in a KCall
  - Time to schedule a thread

UBICOMP (C) 2004 AG SYSOFT – UNIVERSITY OF TRIER

Worst Case IST Latency

- General case
  - In the thread scheduler KCall and take an IRQ that will trigger a different IST
  - Software assisted TLB/cache miss on the IST thread

UBICOMP (C) 2004 AG SYSOFT – UNIVERSITY OF TRIER

Improvements To Latency

- Non-preemptable code reduced
  - Large Kcalls split apart and state saved to resume correctly
  - Reduces the latency for an IST
- Kernel data structures moved to statically mapped virtual address
  - This avoids any TLB misses associated with accessing its data
- Special-cased ISTs
  - An event registering for an IST can only be used in a WaitForSingleObject
- New priority inversion model reduces the upper bounds
  - Was a large KCall

UBICOMP (C) 2004 AG SYSOFT – UNIVERSITY OF TRIER

Nested Interrupts

- Based on support by the CPU and/or additional hardware
- X86
  - Single CPU interrupt with a PIC
  - PIC sets up priority nesting
  - PIC will mask appropriate IRQs
  - Interrupt is enabled before the ISR is entered
- ARM
  - Single CPU interrupt with an Interrupt register
    - No built in concept of priority IRQ
      - Except FIQ
  - Interrupts are not turned on before entering ISR
    - OEM can re-enable CPU interrupt

UBICOMP (C) 2004 AG SYSOFT – UNIVERSITY OF TRIER

Nested Interrupts

- SHx
  - IntrPrio array
    - Defines the priority of each IRQ
  - Kernel masks all lower and current IRQs
- MIPS
  - IntPriority array
    - Defines the priority of IRQ
  - IntrMask
    - Defines what IRQs are masked when an IRQ fires
    - 0x3f turns off nesting

UBICOMP (C) 2004 AG SYSOFT – UNIVERSITY OF TRIER

Thread Priorities

- 256 total priorities
  - Old 8 priorities are now the lowest priorities
  - Old priority APIs access the bottom 8 priorities
    - Set(Get)ThreadPriority
  - APIs to access full priority range
    - Ce(Set/Get)ThreadPriority
  - Top 248 can be protected by the OEM
- Priority THREAD_PRIORITY_TIME_CRITICAL and priority 0 are not run-to-completion by default

UBICOMP (C) 2004 AG SYSOFT – UNIVERSITY OF TRIER

Thread Scheduling

- Highest priority runnable thread is always scheduled
- Run for complete quantum or until blocked or preempted
- Preempted only by higher or same priority threads
- Run-to-completion threads are only preempted by higher priority threads
- Priorities are only boosted to correct priority inversion
- Nested inversion special cased
  - Single level support when all threads are unable to run

UBICOMP (C) 2004 AG SYSOFT – UNIVERSITY OF TRIER

Thread Quantum

- Per thread quantum
- Default set by the OEM in the OAL
  - dwDefaultThreadQuantum
- APIs to set Quantum
  - Ce(Set/Get)ThreadQuantum
- Quantum of 0 sets thread to run-to-completion
  - At any priority
  - Preempted only by higher priority threads

UBICOMP (C) 2004 AG SYSOFT – UNIVERSITY OF TRIER

System Tick

- 1 ms timer tick in normal mode
- Tick interrupt does not necessarily causes a reschedule
  - Check to determine if a reschedule is required
- Sleep(N) will generally wake up in N to N + 1 ms
- In Idle mode system tick is reset to next scheduled event
  - On system tick check for reschedule or nop

UBICOMP (C) 2004 AG SYSOFT – UNIVERSITY OF TRIER

Full Kernel Mode

- All threads are running in kernel mode
- No need to call SetKMode
- Entire system is open to all processes
  - All statically mapped virtual addresses
- Virtual protection is still in place
- Optimizations for high traffic functions

UBICOMP (C) 2004 AG SYSOFT – UNIVERSITY OF TRIER

Miscellaneous

- Periodic timers
  - Win32 multimedia timers
  - Max resolution is 1ms
  - Uses Sleep and SleepToTick

UBICOMP (C) 2004 AG SYSOFT – UNIVERSITY OF TRIER

Resources

- Windows Operating Systems
  - Embedded DevCon 2003 Resource CD

UBICOMP (C) 2004 AG SYSOFT – UNIVERSITY OF TRIER