Extended OS
Virtual Machines

References:
Abstraction & Virtualisation

(a) Abstraction

(b) Virtualization
Interface Levels

1. Application programs
2. Libraries
3. Operating system
4. Execution hardware

Software: API, ABI, ISA

Hardware: Memory translation, Main memory, System interconnect (bus), I/O devices and networking
Instruction Set Architecture

- Interface between software and hardware
- Divided between privileged and unprivileged parts
Application Binary Interface

- Interface between programs, hardware, and OS
- Consists of system call interface and unprivileged ISA
Application Programming Interface

- Interface between programs, hardware, and OS
- Consists of library calls + un-privileged ISA
  - Syscalls usually called through library.
**Process versus System Virtual Machine**

Diagram showing the comparison between a process and a system virtual machine. The diagram illustrates the architecture of a guest and host environment, with a focus on the virtualization layers and the processes involved.
OS is an extended virtual machine

- Multiplexes the “machine” between applications
  - Time sharing, multitasking, batching
- Provided a higher-level machine for
  - Ease of use
  - Portability
  - Efficiency
  - Security
  - Etc....
JAVA – Higher-level Virtual Machine

• write a program once, and run it anywhere
  – Architecture independent
  – Operating System independent

• Language itself was clean, robust, garbage collection

• Program compiled into bytecode
  – Interpreted or just-in-time compiled.
  – Lower than native performance
Conventional versus Emulation/Translation

(a) HLL program
Compiler front end
Intermediate code
Compiler back end
Object code
Loader
Memory image

(b) HLL program
Compiler
Portable code
Virtual memory image
VM loader
VM interpreter/compiler
Host instructions
Issues

• Legacy applications

• No isolation nor resource management between applets

• Security
  – Trust JVM implementation? Trust underlying OS?

• Performance compared to native
Is the OS the “right” level of extended machine?

• Security
  – Trust the underlying OS?

• Legacy application and OSs

• Resource management of existing systems suitable for all applications?

• What about activities requiring “root” privileges
Virtual Machine Monitors

- Provide scheduling and resource management
- Extended “machine” is the actual machine interface.
IBM VM/370

Virtual 370s

I/O instructions here

Trap here

CMS

CMS

CMS

System calls here

Trap here

VM/370

370 Bare hardware
Advantages

- Legacy OSes (and applications)
- Server consolidation
- Concurrent OSes
  - Linux – Windows
  - Primary – Backup
    - High availability
- Test and Development
- Security
  - VMM (hopefully) small and correct
- Performance near bare hardware
  - For some applications
**Figure 1-29.** (a) A type 1 hypervisor. (b) A type 2 hypervisor.
Virtual R3000???

- Interpret
  - System/161
    - slow
  - JIT dynamic compilation

- Run on the real hardware??
R3000 Virtual Memory Addressing

- MMU
  - address translation in hardware
  - management of translation is software

Figure 2.10 Virtual Memory Addressing
### R3000 Address Space Layout

- **kuseg:**
  - 2 gigabytes
  - MMU translated
  - Cacheable
  - user-mode and kernel mode accessible

<table>
<thead>
<tr>
<th>Segment</th>
<th>Address</th>
</tr>
</thead>
<tbody>
<tr>
<td>kseg0</td>
<td>0x00000000</td>
</tr>
<tr>
<td>kseg1</td>
<td>0xA0000000</td>
</tr>
<tr>
<td>kseg2</td>
<td>0x80000000</td>
</tr>
<tr>
<td>kuseg</td>
<td>0xC0000000</td>
</tr>
<tr>
<td>kuseg</td>
<td>0xFFFFFFFF</td>
</tr>
</tbody>
</table>
R3000 Address Space Layout

- **kseg0:**
  - 512 megabytes
  - Fixed translation window to physical memory
    - 0x80000000 - 0x9fffffff virtual = 0x00000000 - 0x1fffffff physical
    - MMU not used
  - Cacheable
  - Only kernel-mode accessible
  - Usually where the kernel code is placed
R3000 Address Space Layout

- **kseg1:**
  - 512 megabytes
  - Fixed translation window to physical memory
    - 0xa0000000 - 0xbfffffff virtual = 0x00000000 - 0x1fffffff physical
    - MMU not used
  - **NOT** cacheable
  - Only kernel-mode accessible
  - Where devices are accessed (and boot ROM)
R3000 Address Space Layout

- **kseg2:**
  - 1024 megabytes
  - MMU translated
  - Cacheable
  - Only kernel-mode accessible
Issues

- Privileged registers (CP0)
- Privileged instructions
- Address Spaces
- Exceptions (including syscalls, interrupts)
- Devices
mfc0
mfc0
mfc0
r1, c0, cp0

CPO Register

VM

Linux User Mode