Virtualization in System Architectures: A Perspective for Cloud Computing

Burkhart Wolff

Université Paris-Sud
Attempt of a Definition

- Primary goal of virtualization is to provide an abstraction layer for execution environments (operating system, processor, memory). Thus, it is possible to “make believe” to a process or a user (as best as possible) that
  a) he is the unique user of the resource and
  b) several heterogeneous resources were combined to a homogeneous environment.
Overview

• Motivation: What is Virtualization for? And when did it develop?

• Types of Virtualization:
  – Memory & Storage
  – Data
  – Networks
  – Desktop and User-Spaces
  – Processes & Processors (+ MMUs)

• Special Themes & Techniques

• Conclusion
History of Virtualization:
(src: en.wikipedia.org/wiki/Timeline_of_virtualization_development)

1965 IBM M44/44X, experimental paging system, in use at Thomas J. Watson Research Center. IBM announces the IBM System/360-67, a 32-bit CPU with virtual memory hardware (August 1965).
1967 CP-40 (January) and CP-67 (April) go into production time-sharing use.
1968 CP/CMS installed at eight initial customer sites. CP/CMS submitted to IBM Type-III Library by MIT's Cambridge Scientific CenterLincoln Laboratory, making system available to all IBM S/360 customers at no charge in source code form. Resale of CP/CMS access begins at time-sharing vendor National CSS (becoming a distinct version, eventually renamed VP/CSS).
1972 Announcement of virtual memory added to System/370 series. VM/370 announced – and running on announcement date. VM/370 includes the ability to run VM under VM (previously implemented both at IBM and at user sites under CP/CMS, but not made part of standard releases).
History of Virtualization:
(src: en.wikipedia.org/wiki/Timeline_of_virtualization_development)

1974-1998  [ongoing history of VM family and VP/CSS.]
1985  October 9, 1985: Announcement of the Intel 80286-based AT&T 6300+ with Simultask, a virtual machine monitor developed by Locus Computing Corporation in collaboration with AT&T, that enabled the direct execution of an Intel 8086 guest operating system under a host Unix System V Release 2 OS. Although the product was marketed with Microsoft MS-DOS as the guest OS, in fact the Virtual Machine could support any realmode operating system or standalone program (such as Microsoft Flight Simulator) that was written using only valid 8086 instructions (not instructions introduced with the 80286). Locus subsequently developed this technology into their "Merge" product line.

1987 January 1987: A "product evaluation" version of Merge/386 from Locus Computing Corporation was made available to OEMs. Merge/386 made use of the Virtual 8086 mode provided by the Intel 80386 processor, and supported multiple simultaneous virtual 8086 machines. The virtual machines supported unmodified guest operating systems and standalone programs such as Microsoft Flight Simulator; but in typical usage the guest was MS-DOS with a Locus proprietary redirector (also marketed for networked PCs as "PC-Interface") and a "network" driver that provided communication with a regular user-mode file server process running under the host operating system on the same machine. October 1987: Retail Version 1.0 of Merge/386 began shipping, offered with Microport Unix System V Release 3.
History of Virtualization:
(src: en.wikipedia.org/wiki/Timeline_of_virtualization_development)

1988 SoftPC 1.0 for Sun was introduced in 1988 by Insignia Solutions [1] SoftPC appears in its first version for Apple Macintosh. These versions (Sun and Macintosh) have only support for DOS.

1997 First version of Virtual PC for Macintosh platform was released in June 1997 by Connectix

1998 October 26, 1998, VMware filed for a patent on their techniques, which is granted as U.S. Patent 6,397,242 [2]

1999 February 8, 1999, VMware introduced VMware Virtual Platform for the Intel IA-32 architecture.

2000 IBM announces z/VM, new version of VM for IBM's 64-bit z/Architecture

2001 June, Connectix launches its first version of Virtual PC for Windows.[3] July, VMware created the first x86 server virtualization product.[4]


Late 2003, EMC acquired VMware for $635 million.

Late 2003, VERITAS acquired Ejascent for $59 million. November 10, 2003 Microsoft releases Microsoft Virtual PC, which is machine-level virtualization technology, to ease the transition to Windows XP.

2005 HP releases Integrity Virtual Machines 1.0 and 1.2 which ran only HP-UX October 24, 2005 VMware releases VMware Player, a free player for virtual machines, to the masses. Sun releases Solaris (operating system) 10, including Solaris Zones, for both x86/x64 and SPARC systems
History of Virtualization:
(src: en.wikipedia.org/wiki/Timeline_of_virtualization_development)

2005 HP releases Integrity Virtual Machines 1.0 and 1.2 which ran only HP-UX October 24, 2005
VMware releases VMware Player, a free player for virtual machines, to the masses. Sun releases Solaris (operating system) 10, including Solaris Zones, for both x86/x64 and SPARC systems.

2006 July 12, 2006 VMware releases VMware Server, a free machine-level virtualization product for the server market. Microsoft Virtual PC 2006 is released as a free program, also in July.
July 17, 2006 Microsoft bought Softricity.
August 16, 2006 VMware announces of the winners of the virtualization appliance contest.
September 26, 2006 moka5 delivers LivePC technology. HP releases Integrity Virtual Machines Version 2.0, which supports Windows Server 2003, CD and DVD burners, tape drives and VLAN.
December 11, 2006 Virtual Iron releases Virtual Iron 3.1, a free bare metal virtualization product for enterprise server virtualization market.

2007 January 15, 2007 innoTek released VirtualBox Open Source Edition (OSE), the first professional PC virtualization solution released as open source under the GNU General Public License (GPL). It includes some code from the Qemu project. Sun releases Solaris 8 Containers to enable migration of a Solaris 8 computer into a Solaris Container on a Solaris 10 system - for SPARC only.

2008 January 15, 2008 VMware, Inc. announced it has entered into a definitive agreement to acquire Thinstall, a privately-held application virtualization software company.
February 12, 2008 Sun Microsystems announced that it had entered into a stock purchase agreement to acquire innotek, makers of VirtualBox. In April, VMware releases VMware Workstation 6.5 beta, the first program for Windows and Linux to enable DirectX 9 accelerated graphics on Windows XP guests [7].
Types of Virtualization: – a first summary --

- Interest in Virtualization exploded in recent years:
Types of Virtualization: – a first summary --

- Interest in Virtualization exploded in recent years
- Virtualization Products became actually a Multi-Billion $ Market Industry
Types of Virtualization: – a first summary --

- Interest in Virtualization exploded in recent years
- Virtualization Products became actually a Multi-Billion $ Market Industry
- Virtualization Technology ranges from **Hardware** (Computer Architecture) to **Software** (Operating Systems)
Why Virtualization? – a first summary --

- Better usage of scarce resources (originally memory, later processor time)
- Protection of components (virtual machines against each other, networks against each other) → Security
- Sharing of Applications → “making one app amenable to several users”
- Making User Groups/Companies able to make a system transition (HW, SW)
- Dynamic “Creation” of Hardware, Load-Balancing
- New Optimization Schemes (e.g. code sharing between different VMs)
Types of Virtualization:
(a bit in historic order)

- Memory & Storage
- Data & Applications
- Desktop and User-Spaces
- Networks
- Operating System Environments for Processes
- Hardware (Processors, MMUs)
Overview

• An a bit more refined list of Types of Virtualization:
  - Virtual Memory
  - Memory Virtualization
  - Virtual Filesystems (Storage; Disks; etc)
  - Virtual Data-Bases
  - Virtual Networks
  - Virtual Desktops
  - Virtual Root Processes (Jails)
  - Full Application Virtualization
  - Full System Virtualization (OS-level)
  - Full System Virtualization (hardware-level; Hypervisors)
Types of Virtualization: Memory & Storage

• Classic: Virtual Memory
  – The OS virtualizes a “linear address space” to all its processes
    \[ \text{ld R1 15} \]
    \[ \text{sta R2 15} \]
    \[ \text{st R3 26} \] ...
  – all processes live in their own address space (in contrast to threads of a process, which live all in the same address space of the process)
Types of Virtualization: Memory & Storage

- Classic: Virtual Memory
  - Early on, Hardware Support was developed for Memory Virtualization, in form of Memory Management Units (MMU)
Types of Virtualization: Memory & Storage

• Classic: Virtual Memory
  - Early on, Hardware Support was developed for Memory Virtualization, in form of Memory Management Units (MMU)

  - Principle: buffered IO via TLB's, and per process management of a page table (representing the final map on the physical adresses). In case of a memory fault, the OS stops process execution and rearranges the pagetables and phys. memory.
Types of Virtualization: Memory & Storage

- Classic: Virtual Memory
  - Early on, Hardware Support was developed for Memory Virtualization, in form of Memory Management Units (MMU)
  - Optimizations:
    - TLB's with last access-bits and dirty-bits
  - Feature: Additional Security by Data Separation of Processes
Types of Virtualization: Memory & Storage

- Classic: FS Virtualization
  - Virtual Filesystems (Posix-Standard: Files are:
    - regular files
    - directories
    - threads
    - streams
    - devices
    ...

... and have a structured name-space ... /Users/bu/...
Types of Virtualization: Memory & Storage

- Classic: FS Virtualization
  - Virtual Filesystems
  - Hardware Abstraction Layer (HAL) for Devices
Types of Virtualization: Memory & Storage

- **Classic:** FS Virtualization
  - Virtual Filesystems
  - Hardware Abstraction Layer (HAL) for Devices
  + Mirroring

Diagram:
- `/`
- `Users`
- `etc`
- `tmp`
- `usr`
- `dev`
- `local`
- `usb A`
- `bu`
- `Desktop`
- `bu`
- `hard-link`
Types of Virtualization: Memory & Storage

• Classic:
  FS Virtualization
    - Virtual Filesystems
    Hardware Abstraction
      Layer (HAL) for Devices
        + network-file-system (NFS)

Disk on remote server

//

Desktop

“A”

usb A

local

usr

dev

etc

tmp

Users

wolff

bu

local

usb A
Types of Virtualization: Memory & Storage

- Not to confuse with:

Memory Virtualization.

- decouples volatile random access memory (RAM) resources from individual systems in the data center,
- aggregates those resources into a virtualized memory pool available to any computer in the cluster.
- The memory pool is accessed by the operating system or applications running on top of the operating system.
- The distributed memory pool can then be utilized as a high-speed cache, a messaging layer, or a large, shared memory resource for a CPU or a GPU application.
Types of Virtualization: Memory & Storage

- Not to confuse with:

Memory Virtualization.
Types of Virtualization: Memory & Storage

- Not to confuse with:

Memory Virtualization.

... the latter is typical for clusters and grids.

(special hardware for clusters)
Types of Virtualization: Memory & Storage

• Not to confuse with:

Memory Virtualization.

• Solutions:

  – RNA networks Memory Virtualization Platform - A low latency memory pool, implemented as a shared cache and a low latency messaging solution.

  – Gigaspaces - Application platforms for Java and .Net environments that offer an alternative to traditional application-servers. Includes in-memory data grid for grid computing.

  – ScaleMP - A platform to combine resources from multiple computers for the purpose of creating a single computing instance.

Types of Virtualization: Data Virtualization

- Layer: Data Models (beyond memory and file systems …)
  - Idea: Combining the Content of Data-Bases and File-Systems to one, unique (OO) Data-Base
  - Example: NPfIT Spine (British Telecom)
Types of Virtualization: Data Virtualization

- Layer: Data Models
  (beyond memory and file systems …)
Types of Virtualization: Data Virtualization

- Layer: Data Models (beyond memory and file systems ...)
  - Data Virtualization is based on the premise of the abstraction of data contained within a variety of data sources (databases, applications, file repositories, etc.)
  - Goal: presentation via a single-point access to the data.
  - During the abstraction or virtualization process the data sometimes is presented or described through the use of the metadata.
  - Such metadata is used to describe relational model elements with use of object oriented paradigm.
  - For example, common RDBMS elements like unique keys, table references, relational joints and views can be represented through virtual objects, object references.
  - Data virtualization also employs metadata objects and entities to create virtual globalized or composite entities from source entities.
Types of Virtualization: Networks

• Virtual Networks:
  - In Virtual Local Area Networks (VLAN) local devices were connected via firewalls and router (involving NAT's=) to local sub-networks. Connection is per default inhibited, but can be made possible in a controlled way similar to FTP protocol by stateful reconfiguration of firewalls.

    → Has in principle nothing to do with Virtualization ...

  - A Virtual Private Network (VPN) constructs a secured and concealed Network (usually built upon encryption techniques) via alien and not trustworthy networks ...
Types of Virtualization: Networks

• Virtual Networks:
  
  - In Virtual Local Area Networks (VLAN) local devices were connected via firewalls and router (involving NAT's=) to local sub-networks. Connection is per default inhibited, but can be made possible in a controlled way similar to FTP protocol by stateful reconfiguration of firewalls.

  $\Rightarrow$ Has in principle nothing to do with Virtualization ...

  - A Virtual Private Network (VPN) constructs a secured and concealed Network (usually built upon encryption techniques) via alien and not trustworthy networks ...
Types of Virtualization: Networks

• Virtual Networks:
  
  – In Virtual Local Area Networks (VLAN) local devices were connected via firewalls and router (involving NAT's=) to local sub-networks. Connection is per default inhibited, but can be made possible in a controlled way similar to FTP protocol by stateful reconfiguration of firewalls.

    → Has in principle nothing to do with Virtualization ...

  – A Virtual Private Network (VPN) constructs a secured and concealed Network (usually built upon encryption techniques) via alien and not trustworthy networks ...

![Diagram of network connections]
Types of Virtualization: Networks

- Virtual Networks:
  - In Virtual Local Area Networks (VLAN) local devices were connected via firewalls and router (involving NAT's=) to local sub-networks. Connection is per default inhibited, but can be made possible in a controlled way similar to FTP protocol by stateful reconfiguration of firewalls.
  
  → Has in principle nothing to do with Virtualization ... 

  - A Virtual Private Network (VPN) constructs a secured and concealed Network (usually built upon encryption techniques) via alien and not trustworthy networks ...
Types of Virtualization: Networks

• Virtual Networks:
  - In Virtual Local Area Networks (VLAN) local devices were connected via firewalls and router (involving NAT's=) to local sub-networks. Connection is per default inhibited, but can be made possible in a controlled way similar to FTP protocol by stateful reconfiguration of firewalls.

    → Has in principle nothing to do with Virtualization ...

  - A Virtual Private Network (VPN) constructs a secured and concealed Network (usually built upon encryption techniques) via alien and not trustworthy networks ...
Types of Virtualization: Networks

- Virtual Networks:
  - VPN technologies have a myriad of protocols, terminologies and marketing influences that define them. For example, VPN technologies can differ in:
    - The protocols they use to tunnel the traffic
    - The tunnel's termination point, i.e., customer edge or network provider edge
    - Whether they offer site-to-site or remote access connectivity
    - The levels of security provided
    - The OSI layer they present to the connecting network, such as Layer 2 circuits or Layer 3 network connectivity
  In Virtual Local Area Networks (VLAN) local devices were connected via firewalls and router (involving NAT's=) to local sub-networks. Connection is per default inhibited, but can be made possible in a controlled way similar to FTP protocol by stateful reconfiguration of firewalls.

- Solutions: OpenVPN (SSL/TLS-based), Cisco AnyConnect VPN, Microsofts Point-to-Point Encryption (MPPE), Ragula Systems Development Company “MPVPN”
Types of Virtualization: Virtual Desktops and User-Spaces

- **Desktop virtualization** (also client virtualization[1]),
  - separates a personal computer desktop environment from a physical machine using a client–server model of computing.
  - stores the resulting "virtualized" desktop on a remote central server, instead of on the local storage of a remote client;
  - thus, all of the programs, applications, processes, and data used are kept and run centrally.
  - users can access their desktops on any capable device, such as a traditional personal computer, notebook computer, smartphone, or thin client.
Types of Virtualization: Operating System-level Virtualization

- **“Virtual Root Processes in OS” (containers, jails, VPS)**
  - is a server virtualization method where the kernel of an operating system allows for multiple isolated user-space instances, instead of just one.
  - File-System: Shadowing Data from “real” System Data with Data from the user space.
  - On Unix systems, this technology can be thought of as an advanced implementation of the standard chroot mechanism.
  - In addition to isolation mechanisms, the kernel often provides resource management features to limit the impact of one container's activities on the other containers.

- **Features:**
  - separating several applications to separate containers for improved security, hardware independence, and added resource management features.
  - OS-level virtualization implementations that are capable of live migration can be used for dynamic load balancing of containers between nodes in a cluster.
  - Overhead is minimal.

- **Solutions:** chroot, VPS, iCore Virtual Accounts, Linux-VServer, Free BSD Jail ...
Types of Virtualization: Full Application Virtualization

- Full Application Virtualization
  - The Application virtualization layer
    - ... replaces a part of the runtime environment normally provided by the operating system.
    - ... intercepts all operation system calls as well as file and Registry operations of virtualized applications
    - ... transparently redirects them to a virtualized location, i.e. a system library and often a single file mimicking the file system.
    - ... An application (using only highlevel OS-libraries of A) never knows that it's accessing a virtual resource instead of a physical one.

- Solutions:
  - Microsoft Application Virtualization, Software Virtualization Solution, and VMware ThinApp. (B: Microsoft Windows XX)
  - Wine (B: Linux)
Types of Virtualization: Operating System-level Virtualization

- “Virtual Root Processes in OS” (containers, jails, VPS)
  - is a server virtualization method where the kernel of an operating system allows for multiple isolated user-space instances, instead of just one.
Types of Virtualization: Full Application Virtualization

- Full Application Virtualization refers to technologies that allow to run an application (a group of processes) for OS B to be run on OS A:
Types of Virtualization: Full Application Virtualization

• Full Application Virtualization
  - The Application virtualization layer
    - ... replaces a part of the runtime environment normally provided by the operating system.
    - ... intercepts all operation system calls as well as file and Registry operations of virtualized applications
    - ... transparently redirects them to a virtualized location, i.e. a system library and often a single file mimicking the file system.
    - ... An application (using only highlevel OS-libraries of A) never knows that it's accessing a virtual resource instead of a physical one.

• Solutions:
  - Microsoft Application Virtualization, Software Virtualization Solution, and VMware ThinApp. (B: Microsoft Windows XX)
  - Wine (B: Linux)
Types of Virtualization: Full System Virtualization

- Full System Virtualization (OS-Layer)
  - The boot-loader modifies guest operating system + desktop and overrides the native runtime environment with Application Virtualization Layers (see before)
    (You need a Windows-License !!!)
  - desktop is a special process in host OS, applications are just processes in host OS.
  - Desktop may have access to virtual devices (Printers, USB's, Virtual HardDisks), an own TCP/IP Stack with own Address, …

- Features:
  - sharing a computer system among multiple users
  - isolating users from each other (and from the control program)
  - emulating new hardware to achieve improved reliability, security and productivity.
Types of Virtualization: Full System Virtualization

- Full Application Virtualization refers to technologies that allow to run an application (a group of processes) for OS B to be run on OS A:

  - Desktop
  - Application Virtualization Layer for root process & Desktop OS B
  - Application Virtualization Layer for guest OS B
  - App 1
  - App 2
  - App 3

host operating system (OS) A

hardware
Types of Virtualization: Full System Virtualization

- **Solutions**
  - **Mac-on-Linux** is an open source virtual machine program for running Mac OS on PowerPC computers running Linux.
  - **Parallels Desktop for Mac** (actually also using hypervisor technology (Full System Virtualization Hardware Layer; see next section) mapping partially the host computer’s hardware resources directly to the virtual machine’s resources).

  - Each virtual machine thus operates identically to a standalone computer, with virtually all the resources of a physical computer.[3]
  - Because all guest virtual machines use the same hardware drivers irrespective of the actual hardware on the host computer, virtual machine instances are highly portable between computers.
  - For example, a running virtual machine can be stopped, copied to another physical computer, and restarted.
Types of Virtualization: Full System Virtualization

• Solutions
  – VM Ware
    • allowing to set up multiple x86 and x86-64 virtual computers and to use one or more of these virtual machines simultaneously with the hosting operating system.
    • For Windows, Linux, BSD variants, or others.
  – ...
Types of Virtualization:
Full System Virtualization (Hypervisor)

- Full Application Virtualization (Hardware-level). Refers to technologies that allow to run an entire operating system (with virtual MMU !!!) on a small operating system called “Hypervisor”.

![Diagram showing different virtualization layers and operating systems]
Types of Virtualization: Full System Virtualization (Hypervisor)

- Successful since existence of special Hardware (Chipsets Intel VT-x, AMD-V9)

- Requires Virtualization of an entire X86 Machine!

- Solutions
  - Parallels Workstation
  - VMware Workstation
  - VMware Server (formerly GSX Server),
  - VirtualBox
  - Xen
  - Microsoft's Hyper-V
Types of Virtualization: Full System Virtualization (Hypervisor)

- **Chipsets Intel VT-x:**
  - **Extended Page Table (EPT)** is an Intel second generation x86 virtualization technology for the MMU. When this feature is active, the ordinary IA-32 page tables (referenced by control register CR3) translate from linear addresses to guest-physical addresses. A separate set of page tables (the EPT tables) translate from guest-physical addresses to the host-physical addresses that are used to access memory. As a result, guest software can be allowed to modify its own IA-32 page tables and directly handle page faults. This allows a VMM to avoid the VM exits associated with page-table virtualization, which are a major source of virtualization overhead without EPT.

**Diagram:**
- **CPU**
- **MMU**
- **TLB**

**Legend:**
- **TLB**: Translation Look-aside Buffer
- **MMU**: Memory Management Unit
- **CPU**: Central Processing Unit
Types of Virtualization:
Full System Virtualization (Hypervisor)

- Chipsets Intel VT-x:
  - **Extended Page Table (EPT)** is an Intel second generation x86 virtualization technology for the MMU. When this feature is active, the ordinary IA-32 page tables (referenced by control register CR3) translate from linear addresses to guest-physical addresses. A separate set of page tables (the EPT tables) translate from guest-physical addresses to the host-physical addresses that are used to access memory. As a result, guest software can be allowed to modify its own IA-32 page tables and directly handle page faults. This allows a VMM to avoid the VM exits associated with page-table virtualization, which are a major source of virtualization overhead without EPT.
VMM Virtualisierung

- Hyper-V HyperVisor

Diagramm: 
- Root Partition
  - Hyper-V Aware Windows Server (Windows Server 2008)
  - Kernel
  - Device Drivers
    - Virtualization Service Provider (VSP)
  - Hypercall API

- Child Partition 1
  - Hyper-V Aware Windows Client (Windows Vista)
  - Kernel
  - Device Drivers
    - Virtualization Service Client (VSC)
  - Hypercall API

- Child Partition n
  - Hyper-V Aware Non-Windows OS
    - Linux with MS/XenSource
  - Kernel
  - Device Drivers
  - Linux VSC
  - Hypercall Adapter
  - Hypercall API
Hyper-V

- Features:
  - Complete Isolation of entire (Host) - Systems
  - Security mechanisms on hardware-level can be used, eg. for Data Execution Prevention (DEP)
  - Support for Network Address Translation (NAT) and Network Access Protection (NAP)
  - Control via Microsoft Management Console (MMC)
  - In Clusters: seamless Live-Migrationen (if physical host breaks, transfer of the VM to another Host)
  - Dynamic Loadbalancing
Hyper-V

• Features:
  • Host can be a Windows 2008 Core Server
  • Hyper-V can assign up to 4 processors to a guest OS (both 64-Bit as well as 32-Bit-Systems)
  • Virtual Hardware management can assign resources to Guests dynamically (if they are “Hype-V aware”)
  • Loadbalancing of network transfer.
  • Snapshots of Systems (for Backup !!!).
Comparison

• Overview
Conclusion

- Virtualization is (nearly) as old as Operating System Development
- Appeals to some Fundamental Aspect of Computer Science: Implementing some Data-Type (or : Interface) by something completely different
- Is based on the Core-Ability in Computer-Science: Abstraction, and finding GOOD Abstractions
Conclusion

• We have seen the following forms of Virtualization:
  - Virtual Memory
  - Memory Virtualization
  - Virtual Filesystems (Storage; Disks; etc)
  - Virtual Data-Bases
  - Virtual Networks
  - Virtual Desktops
  - Virtual Root Processes (Jails)
  - Full Application Virtualization
  - Full System Virtualization (OS-level)
  - Full System Virtualization (hardware-level; Hypervisors)
Special Themes & Techniques

- Coherent Memory Problems
- Software-based Virtualization (OS-level Virtualization)
- Memory Management Strategies
- MMU-Virtualization
Incoherent Memory Problems

- Total Store Order & Sequential Consistency in Multi-Core Architectures:

Example:

Let two threads T0 and T1 be two threads, where x and y are shared memory variables and r0 and r1 are registers:

\[
\text{assume}\{\text{volatile } x; \text{volatile } y; x=0, y=0 \}\]

T0: \( x = 1; \)   T1: \( y = 1; \)
\[
\text{r0} = y; \quad \text{r1} = x;
\]

\[
\text{assert}\{\text{r0}<>0 \text{ or } \text{r1}<>0}\}
\]

In a sequentially consistent execution, it is impossible for both r0 and r1 to be assigned 0.

Norbert Schirmer:
From Total Store Order to Sequential Consistency: A Practical Reduction Theorem
Incoherent Memory Problems

- Total Store Order & Sequential Consistency in Multi-Core Architectures:

Norbert Schirmer:
From Total Store Order to Sequential Consistency: A Practical Reduction Theorem

When verifying a concurrent program, it is usual to assume sequentially consistent memory. However, most modern multiprocessors buffer their stores, providing native sequential consistency only at a substantial performance penalty. To regain sequential consistency, a programmer has to follow an appropriate programming discipline. However, existing na"ive disciplines, such as protecting all shared accesses with locks to avoid data races, or flushing store buffers according to a protocol that allows arbitrary data races, are not flexible enough for building high-performance multiprocessor software.

We present a new discipline for concurrent programming under TSO (total store order, with store buffer forwarding). Instead of using concurrency primitives, such as locks, it is based on maintaining ownership information in ghost state, allowing the discipline to be expressed as a state invariant and verified through conventional program reasoning. If every execution of a program in a system without store buffers follows the discipline, then every execution of the program in a system with store buffers is sequentially consistent.
Incoherent Memory Problems

- Consequence: OS-level Programming even more complex!
  - Solution: Defensive Programming, Flushing TLB's at any time. ( = too expensive)
  - Solution: Fencing any global volatile variable

- Solution: Verification of ownership of volatile variables ...
Software-based Virtualization
(OS-level Virtualization)

• Key-Technique: Binary Translation
(Patents hold initially by VMWare Founders at Stanford)

  - Binary translation is used to rewrite certain instructions, like POPF, that would otherwise fail silently or behave differently when executed above ring 0.[2][3] making the classic trap-and-emulate virtualization impossible.[3] In order to obtain good performance, the translated basic blocks need to be cached in a coherent way that detects code patching (used in VxDs for instance), the reuse of pages by the guest OS, or even self-modifying code.

  - The POPF (pop flags) and POPFD (pop flags double) mnemonics reference the same opcode. The POPF instruction is intended for use when the operand-size attribute is 16 and the POPFD instruction for when the operand-size attribute is 32. Some assemblers may force the operand size to 16 when POPF is used and to 32 when POPFD is used. Others may treat these mnemonics as synonyms (POPF/POPFD) and use the current setting of the operand-size attribute to determine the size of values to be popped from the stack, regardless of the mnemonic used.

  - The effect of the POPF/POPFD instructions on the EFLAGS register changes slightly, depending on the mode of operation of the processor.
MMU-Virtualization

- MMU: Usually, a page-table is too large to be kept in an MMU.
  - Consequence: Must be kept in the heap, and searched and updated only in case of a trap.
  - In a Hypervisor-Virtualization, this leads to the necessity of Shadow Page Tables ...
  - … and multi-level Page-Tables ...
MMU-Virtualization

- MMU: Usually, a page-table is too large to be kept in an MMU.
  - Standard Procedure

![Diagram of TLB and page table interactions]

- virtual address → TLB
  - TLB hit → physical address
  - TLB miss → page table
    - page table hit → physical address
    - page not present → disk
    - disk → page table write
    - page table write → TLB

MMU-Virtualization

- MMU: Usually, a page-table is too large to be kept in an MMU.

*) 32 bits aligned to a 4-KByte boundary
MMU-Virtualization

- MMU: Usually, a page-table is too large to be kept in an MMU.
MMU-Virtualization

- MMU: Usually, a page-table is too large to be kept in an MMU.
  - Consequence: Must be kept in the heap, and updated only in case of a trap.
  - In a Hypervisor-Virtualization, this leads to the necessity of Shadow Page Tables ...
Memory Management Strategies

- Belady's Algorithm
- Least Recently Used (LRU)
- Most Recently Used (MRU)
- Pseudo-LRU
- Random Replacement
- Segmented LRU

![Direct Mapped Cache Fill Diagram](image1)

![2-Way Associative Cache Fill Diagram](image2)

Each location in main memory can be cached by just one cache location.

Each location in main memory can be cached by one of two cache locations.