Imagination

MIPS Virtualization & Security

David Lau
April 2015

Imagination

Very Brief Overview of Virtualization on MIPS

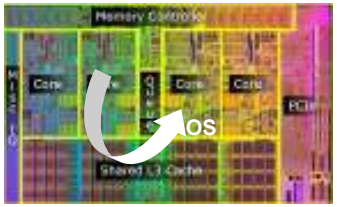
What is Virtualization?

- Operating Systems which use Virtual Memory (address mapping, demand paging) to allow multiple applications/processes to share HW in a time-multiplexed fashion

- Virtualization allows multiple Operating systems to share HW in a time-multiplexed fashion
  - Each Operating System is known as a “Guest”
  - Software that controls the whole system – called “Hypervisor”.
    - Sometimes referred to as Virtual-Machine-Manager (VMM).

Why Virtualization?

Virtualization Use-cases



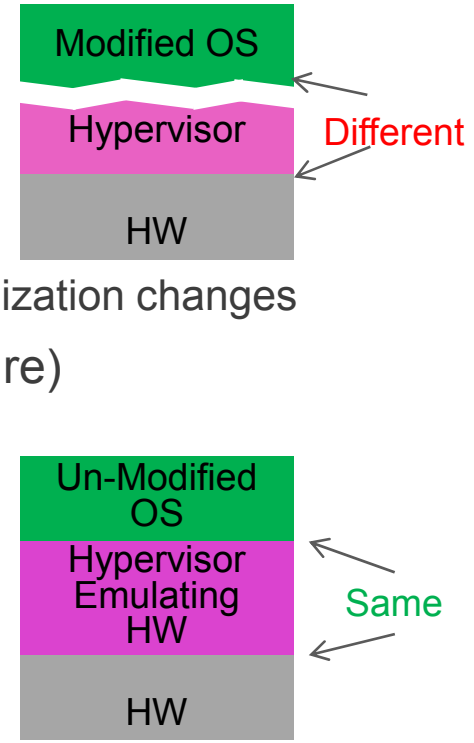
- Intelligent resource allocation
  - Turn off cores in SMP SOC if load low – move running OS from one core to another
  - Turn off machines in server room if load low – move OS from one blade to another

- Isolation and Security
  - A corrupted/hacked OS1 can not affect OS2
  - Run different OS for different purpose – eg personal account & corporate account
  - Application OS; Base-band RTOS; Media-playing RTOS all running on same CPU but can all be isolated from each other

- Reliability
  - If Media OS crashes, the Base-band RTOS can keep running
- Run different OS versions on same HW at the same time
  - Eg old application requiring older OS, but utilize newest OS for other apps
- More sophisticated HW management
  - Build more complicated systems using multiple simple RTOSes

Flavors of Virtualization (1)

- Para-Virtualization
  - Guest Operating System and/or Driver is modified for virtualization
  - Means the Guest can tell it has been virtualized
  - Pros - Potentially faster for I/O virtualization
  - Cons - When new SW versions show up, have to keep porting Virtualization changes
- Full Virtualization (main target for MIPS Virtualization architecture)
  - Guest Operating System and Drivers are un-touched, not modified
  - Means the Guest can NOT tell it has been virtualized
  - Pros – usually much faster for CPU virtualization; no SW porting
  - Cons – needs additional HW Same
- Main difference – implement 2nd MMU (Full) or not (Para)

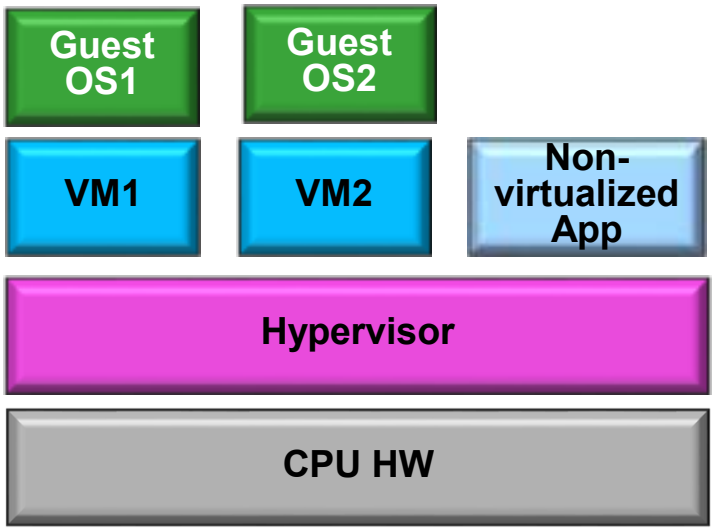


Flavors of Virtualization (2)

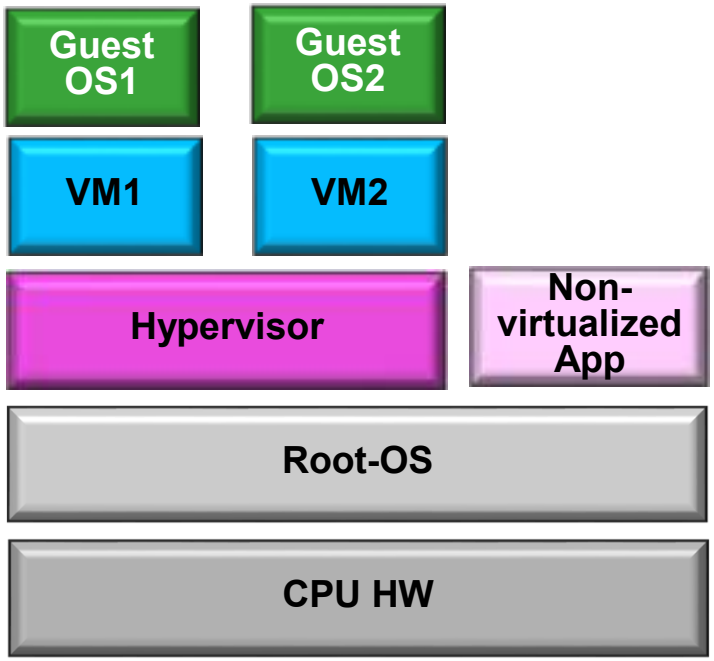
- Trap-and-emulate (e.g. Classical Virtualization)
  - De-privilege the OS by running it in user-mode
  - Entire OS is run in user address space
  - Trap (exit guest to hypervisor) for all privileged instructions & operations
- HW Assisted Virtualization
  - What IMG/MIPS is introducing in our next generation cores (P5600, I6400, M5150)
  - VZ Module of the MIPS Architecture
  - Equivalent to Intel Vt-x, AMD-V technologies

Flavors of Virtualization (3)

Type-I Hypervisor



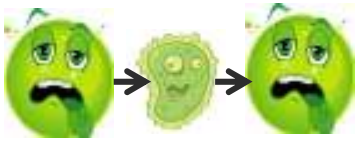
Type-II Hypervisor

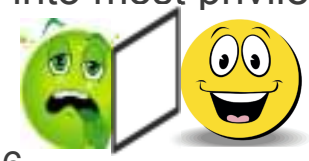


Requirements for Virtualization

What needed to be changed in the Base Architectures for Full Virtualization

- Guest Software can not change state which affects entire machine; Guest can only affect itself
  - All instructions which can change shared machine state must trap into most privileged mode
    - MIPS already does this.
    - MIPS does not have unprivileged “sensitive” instructions like Intel/AMD x86





  - The Guest can not modify memory-mapped resources which are shared among all Guests.
    - MIPS has unmapped memory regions that allow a Guest to potentially touch such resources – must be prevented by Virtualization architecture
- The system should be able to relocate the OS so multiple copies of the same OS can run in a time-multiplexed fashion (OS un-modified for used addresses)
  - MIPS has portions of the kernel code in unmapped memory regions which can’t be relocated.

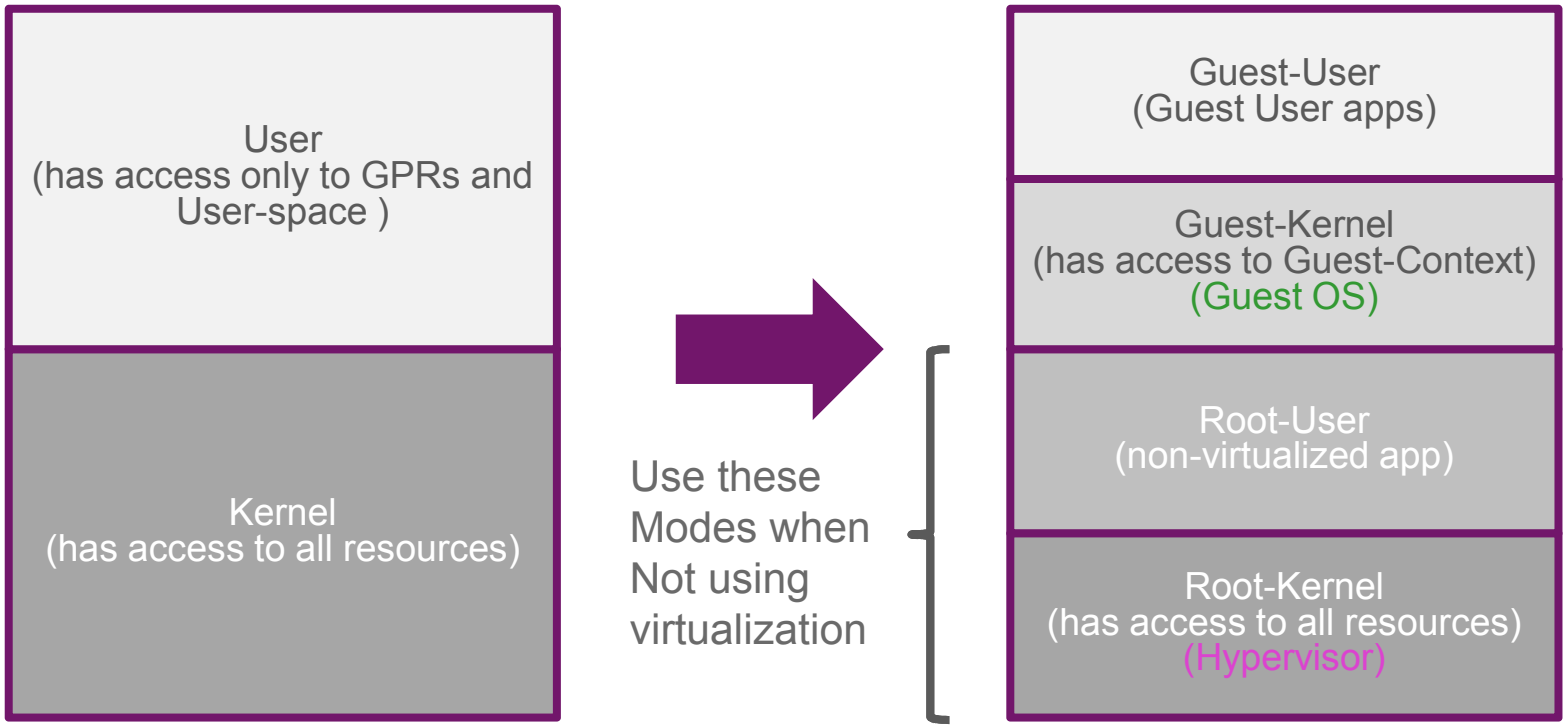
How to make Virtualization effective

MIPS VZ solutions for Virtualization Requirements

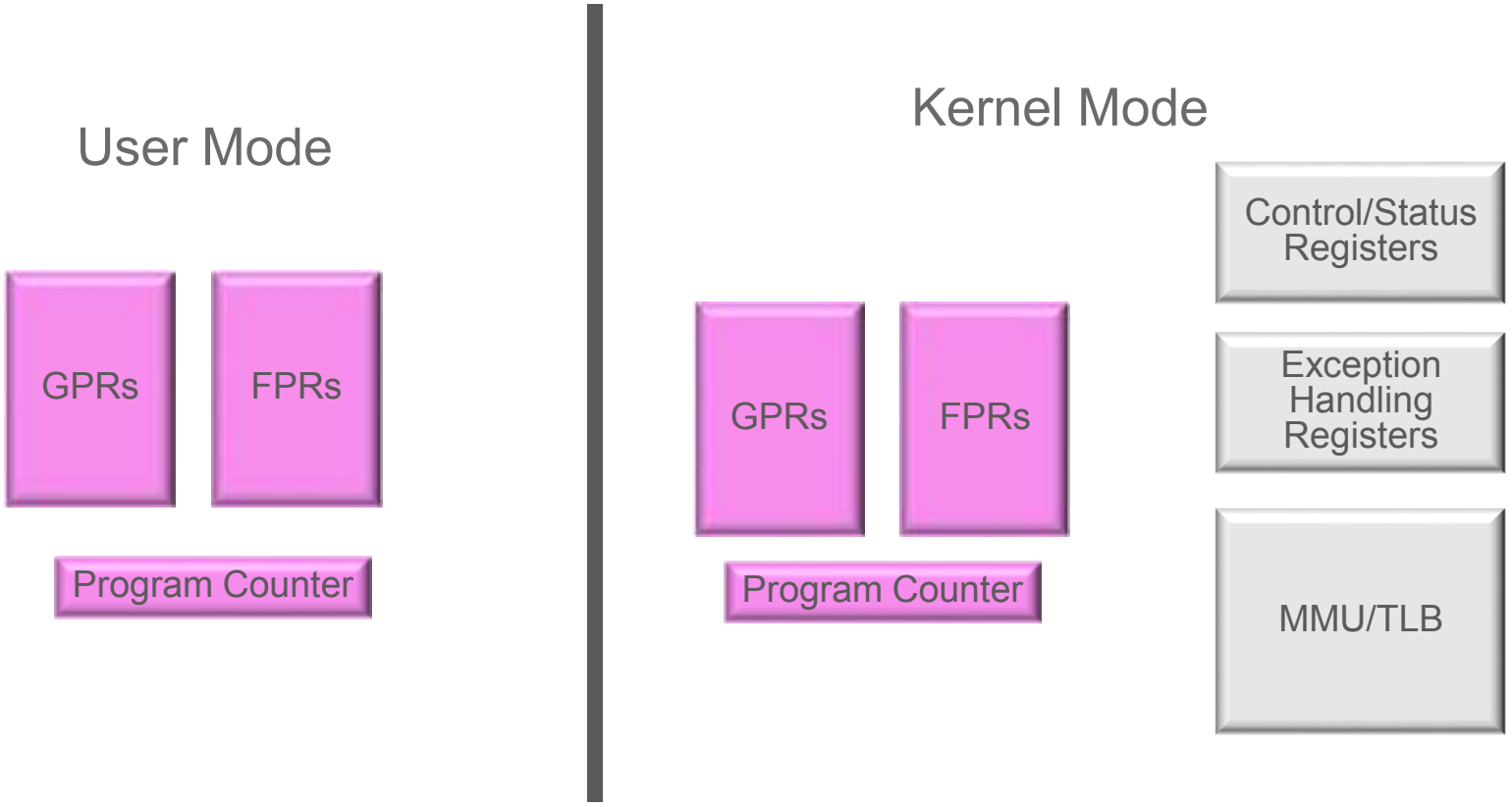
- Requirement 1 – Guest can not modify shared resources even in memory map
  - Solution – Do two levels of mapping – one for Guest and another for Hypervisor. What Guest believes is physical address is actually re-mapped (invisible to Guest).
- Requirement 2 - Need to run multiple copies of the same OS if using unmapped regions
  - Solution – same as above, do a second mapping
- Requirement 3 – Avoid performance loss due to excessive traps to hypervisor (Guest Exits).
  - Solution – make a copy of the control registers. The Guest modifies this “guest” copy, which does not affect whole machine state. Hypervisor will decide later if the state change is allowed later. This de-privileges the Guest OS.

New VZ Execution Modes

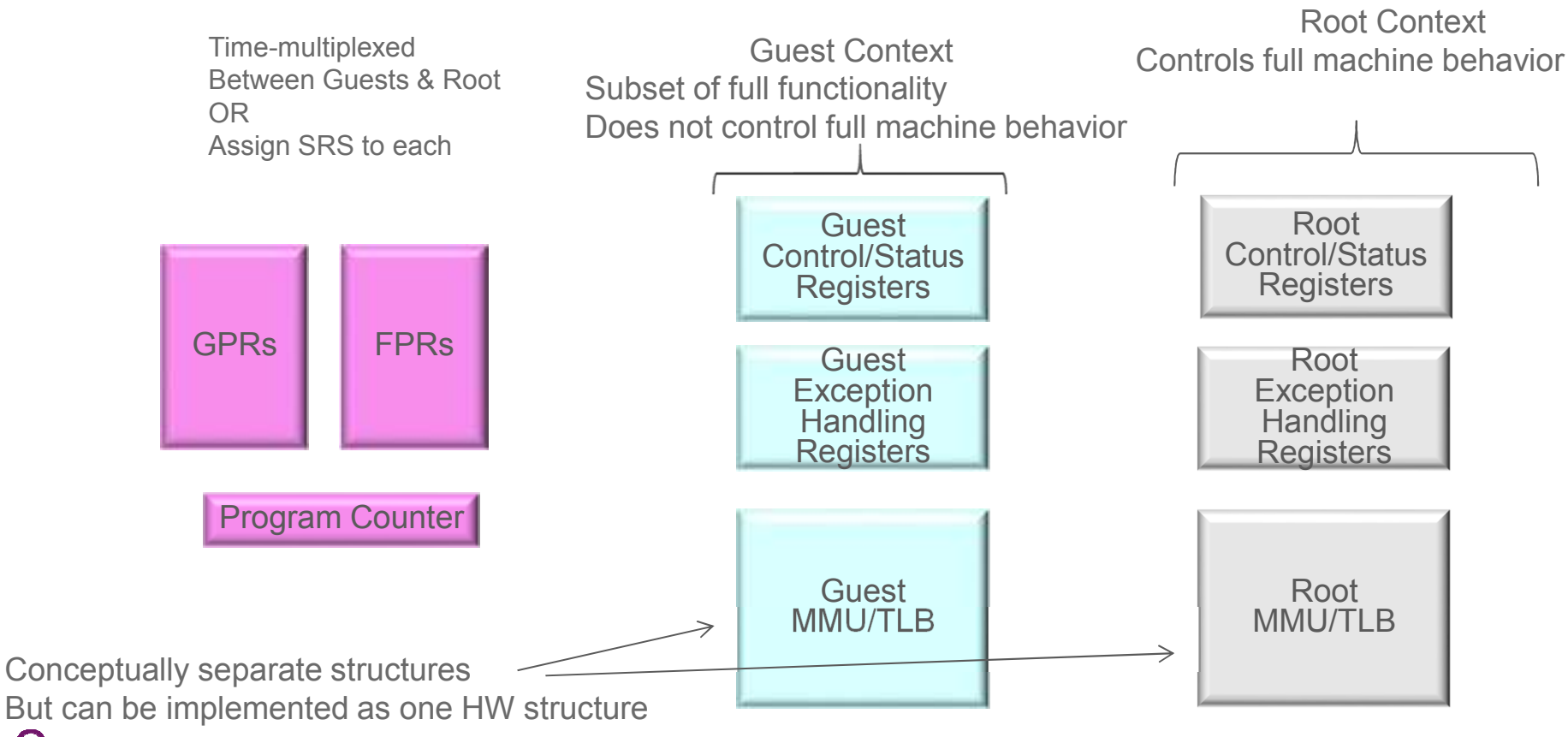
- **How to secure the hypervisor – by deprivileging the guest OS**



Accessible State for non-Virtualized CPU

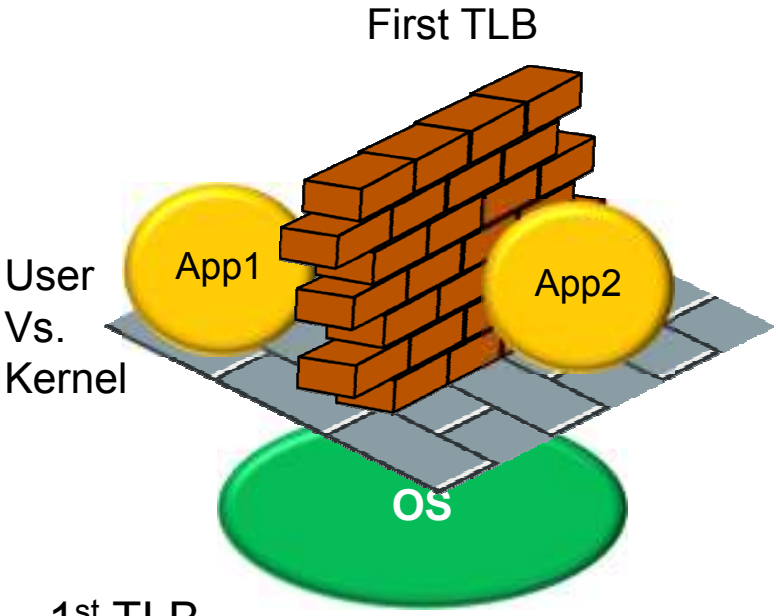


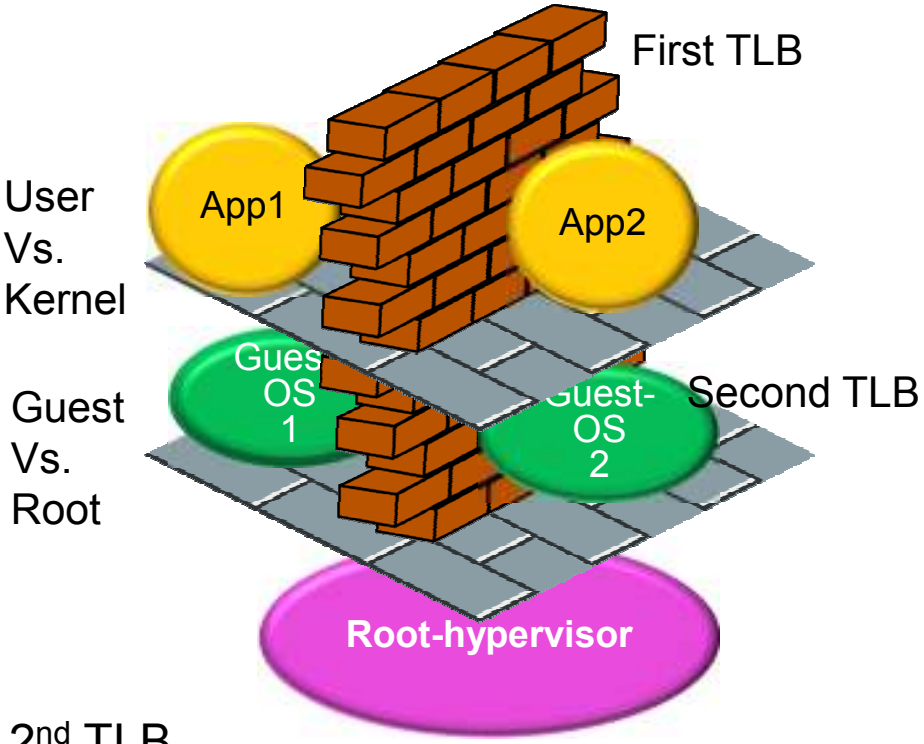
Accessible State for MIPS CPU with VZ Module



Multiple MMUs

Protection Provided





1st TLB

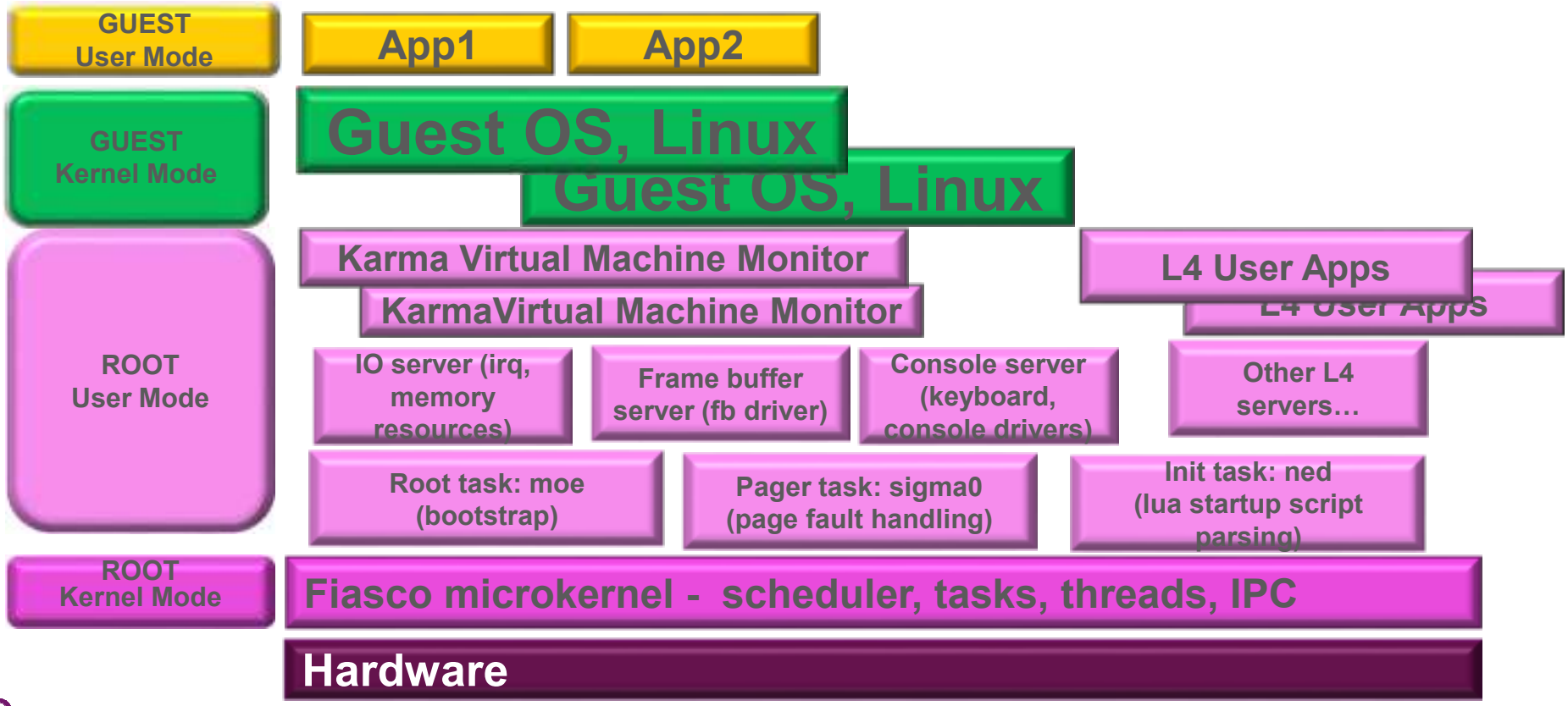
Root-Kernel vs.Root-User protected
UserApp1 protected vs. UserApp2 (like any OS uses Virtual Memory)

2nd TLB

Guest-Kernel vs. Guest-User Protected
Guest1(OS1) protected vs. Guest2(OS2

Fiasco-OS environment

Type-I Hypervisor, port done by IMG



For Distribution

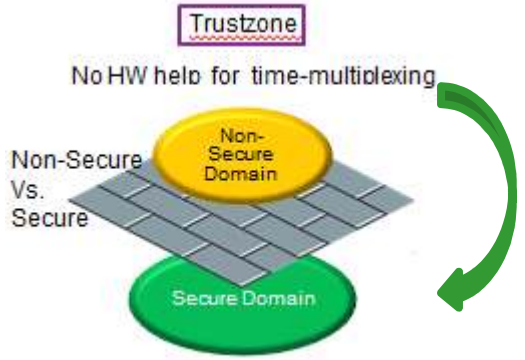
Imagination

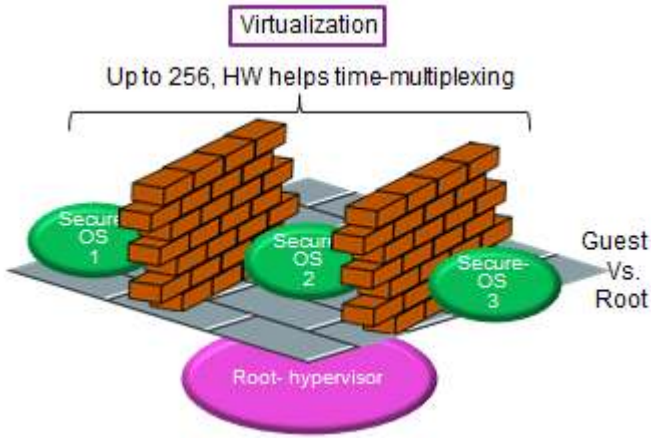
Virtualization & Security

MIPS Virtualization vs. TrustZone

- Trustzone limited to only two domains
  - Modern systems require multiple security domains
  - Trust zone forces time-multiplexing for multiple uses

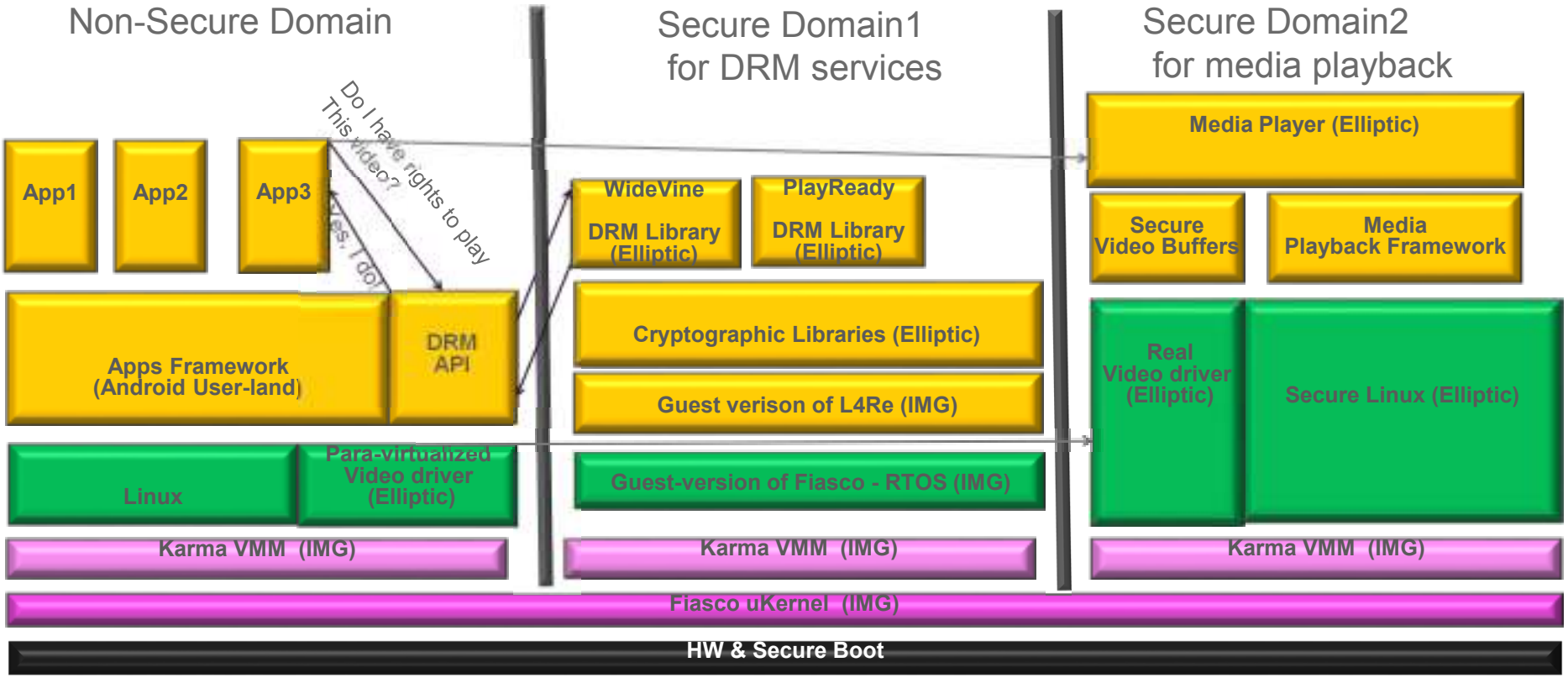
- MIPS-VZ can support up to 256 separate Guests





Final DRM system

Security Project – working with Elliptic Technologies



For Distribution

Imagination

Thank you

www.imgtec.com