

Linux Memory Management

What to Expect?

- ★ W's of Memory Management?
- ★ Memory Partitioning
- ★ Memory Organization
- ★ Memory Management in Linux
- ★ Swapping

Why Memory Management?

- ★ Transition from Uni-process → Multi-process
- ★ Every process needs its own memory
 - Memory Division
- ★ Program loaded to “any” available memory address
 - Relocation
- ★ Independence / Non-interference between Processes
 - Protection
- ★ Communication between Processes
 - Sharing
- ★ Process Memory with Attributes (like read only for code)
 - Logical Organization
- ★ Executing Programs bigger than available Memory
 - Physical Organization (Overlaying & Reusing)

What is Memory Management?

- ★ Key task: Division of Memory
- ★ Starting from the Division between
 - OS (Kernel)
 - Application (User)
- ★ Others follow
 - Relocation, Protection, Sharing
 - Logical & Physical Organization
- ★ May involve movement between primary & secondary

Memory Partitioning

- ★ Two traditional division mechanisms
 - Segmentation: Unequal sized
 - Examples: Linear Allocation, Buddy System
 - Paging: Equal sized
 - Examples: 4K-Paged Allocation
- ★ Examples in Linux
 - User-space malloc(): Segmentation
 - Slab sub-system: Paging

Pros & Cons

★ Memory Fragmentation

- External in Segmentation (gets worser)
- Internal in Paging

★ Space Efficiency

- Segmentation: More meta but lesser allocation wastage
- Paging: Negligible meta but significant allocation wastage

★ Time Efficiency

- Segmentation: Complex search & Storage Algorithm
- Paging: Simple Algorithms

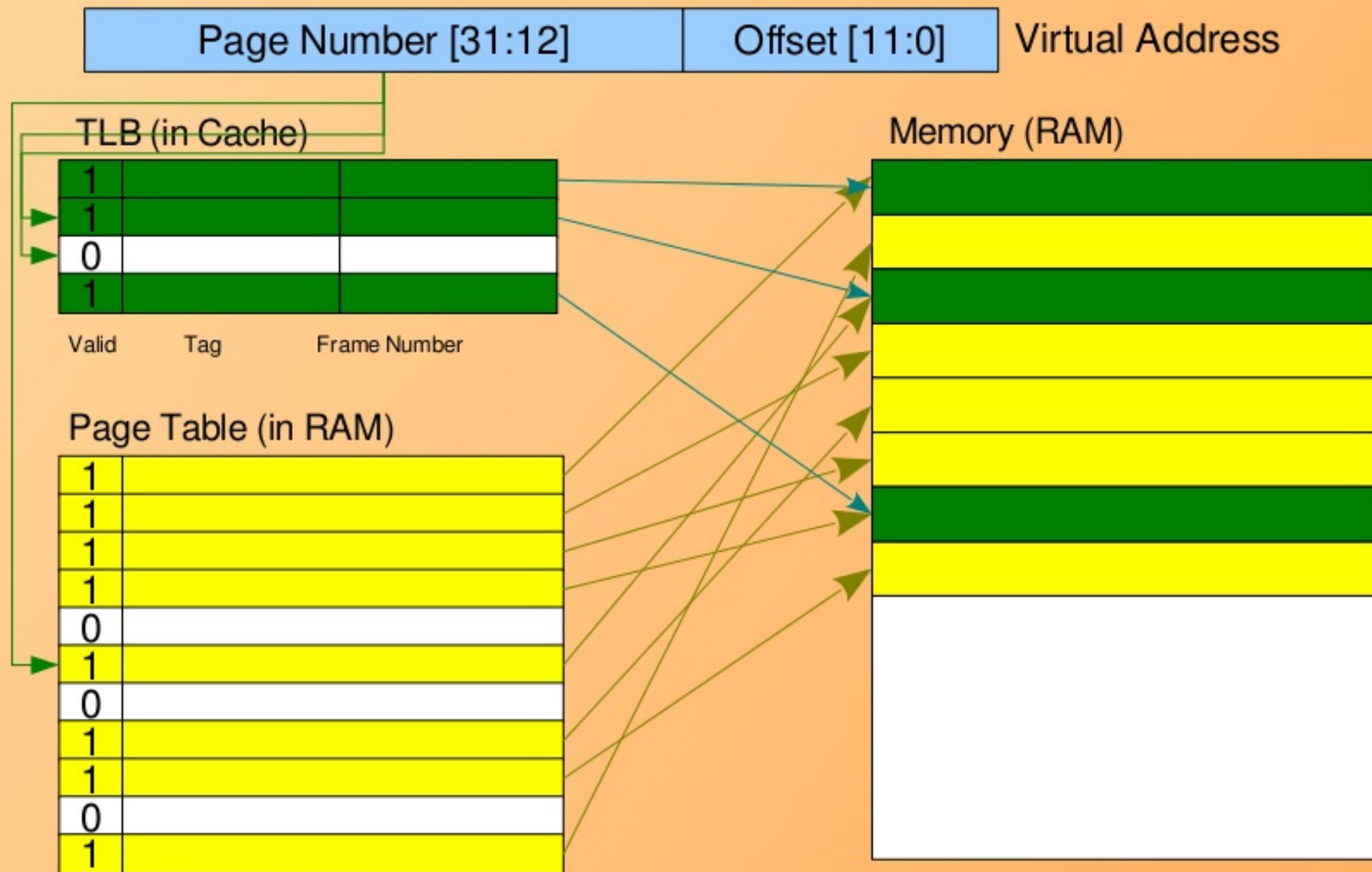
Memory Organization

- ★ All other needs (Relocation, Protection, Sharing, ...)
 - Beautifully achieved by the concept of virtualization
- ★ That is organizing the memory into
 - Physical & Virtual Address Spaces
 - And then using the virtual addressing for all purposes
- ★ Both may use either Paging or Segmentation
- ★ Best to use similar mechanisms
- ★ Linux uses Paging for both with 4KiB/8KiB page size
- ★ Virtualization can be achieved
 - Using special hardware called MMU
 - Using software MMU

Memory Organization ...

- ★ Linux Kernels ($\geq 2.5.46$) support both
 - Earlier kernels worked only with hardware MMU
 - uClinux used to be the solution for non-MMU processors & controllers
- ★ Latest Intel, ARM, and most other architectures have in-built MMU
- ★ Older x86, ARM7, and few such doesn't have a hardware MMU

Working of an MMU

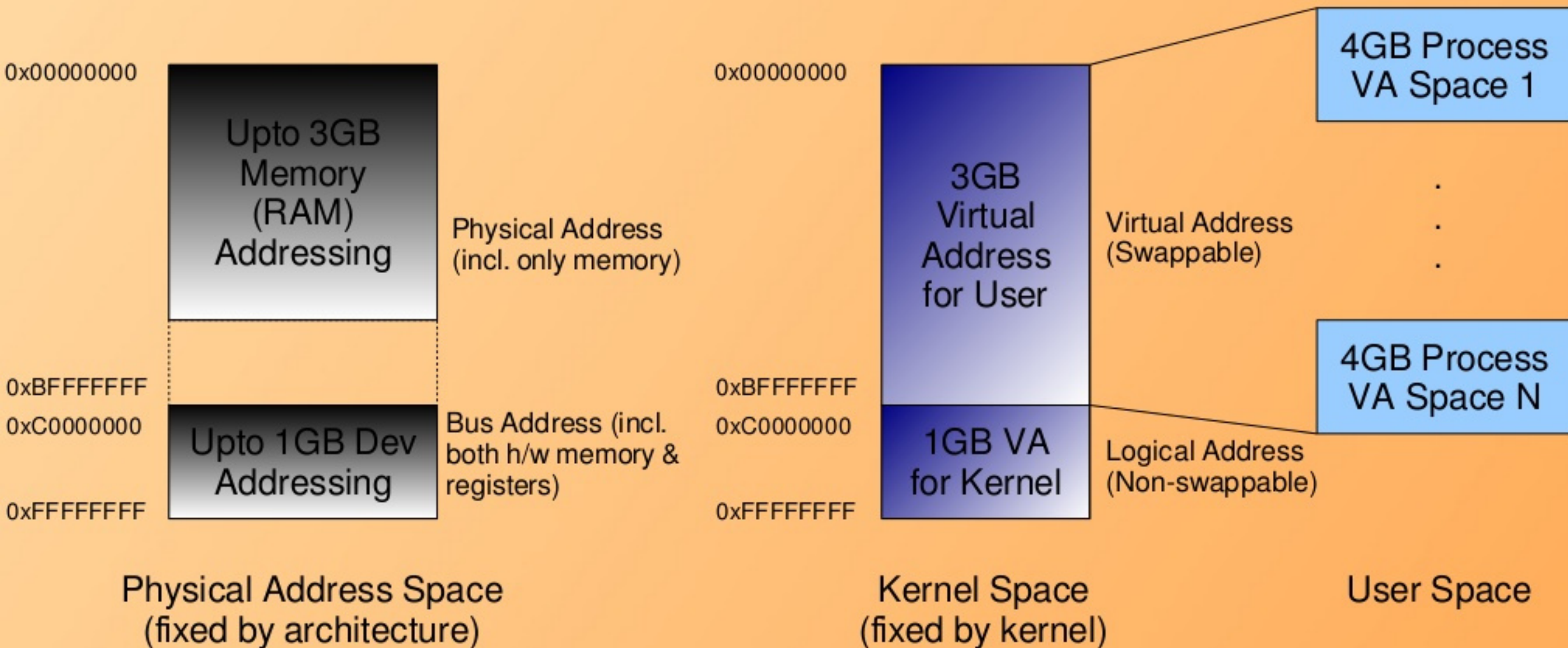


Memory Management in Linux

- ★ In Linux, all these combined
 - User Space & Kernel Space
 - Memory Page Partitioning
 - Physical & Virtual Address Spaces
 - Device Addresses, Mapping & Access
- ★ Takes the following form ...

Memory Organization in Linux

★ An Example assuming 32-bit architecture



Linux Memory Manager

- ★ Provides Access Control to h/w & memory resources
- ★ Provides Dynamic Memory to kernel sub-system
 - Drivers
 - File Systems
 - Stacks
- ★ Provides Virtual Memory to Kernel & User space
 - Kernel & User Processes run in their own virtual address spaces
 - Providing the various features of a Linux system
 - System reliability, Security
 - Communication
 - Program Execution Support

Swapping

- ★ Optimizes primary & secondary memory usage
- ★ Primary memory pages swapped out
 - When running out of memory
- ★ Secondary memory pages swapped in
 - When needed back
- ★ Linux maintains Page Cache in Primary Memory for
 - Code: Reloadable
 - Data: Unchanged vs Changed
 - Changed Data: Temporary (Dynamic) vs Persistent (Files)
- ★ Linux uses Swap (raw) partition on Secondary Memory
- ★ Related commands: mkswap, swapon/off

What all have we learnt?

- ★ W's of Memory Management?
- ★ Memory Partitioning
 - Segmentation & Paging
- ★ Memory Organization
 - Physical & Virtual Addressing
 - Working of an MMU
- ★ Memory Management in Linux
- ★ Swapping & Swap Partition

Any Queries?