

Jin-Soo Kim
(jinsoo.kim@snu.ac.kr)

Systems Software &
Architecture Lab.

Seoul National University

Spring 2020

Virtual Memory Implementations



Linux Virtual Memory System

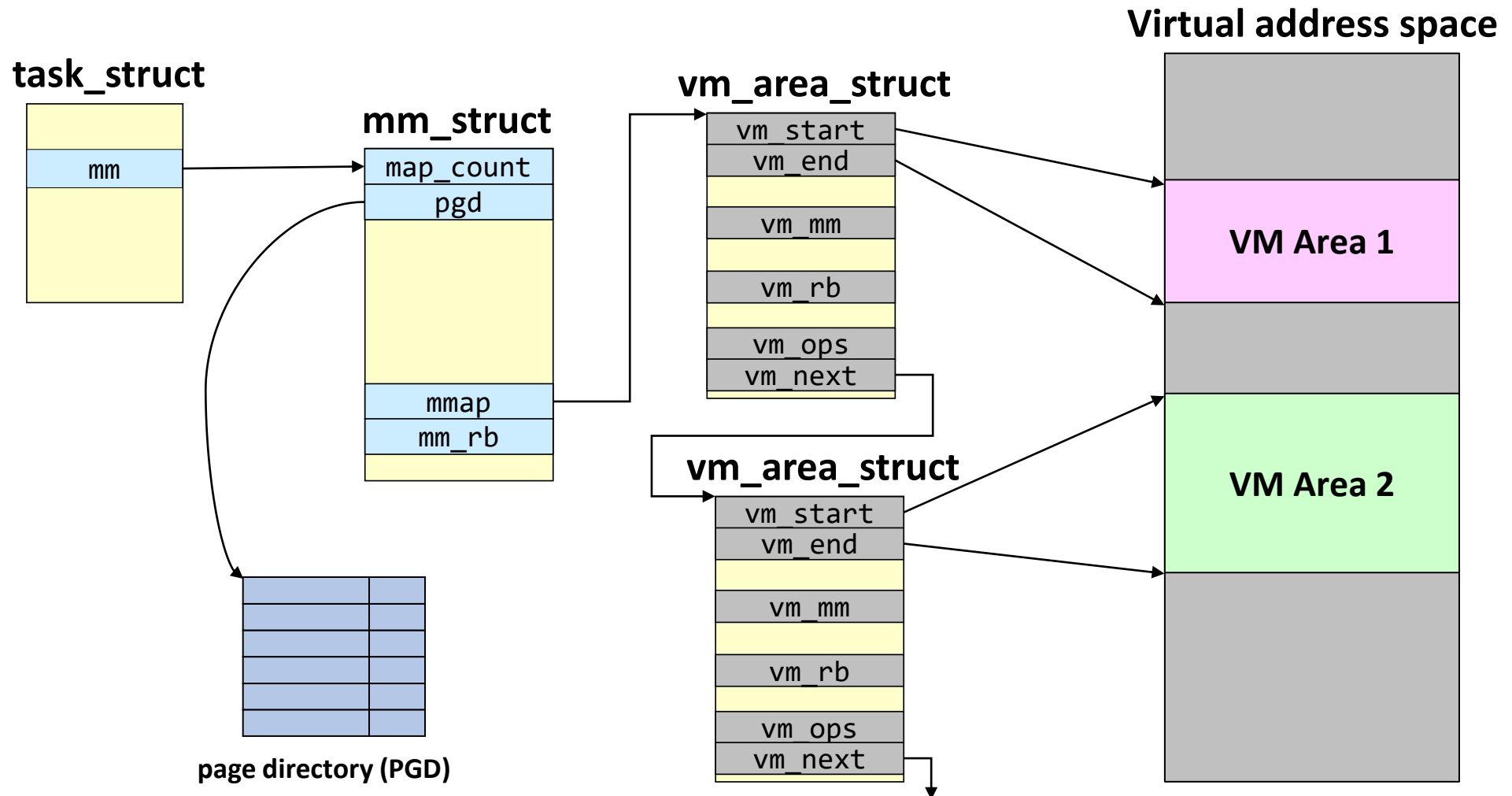
VMA

■ Virtual Memory Area

- A contiguous, page-aligned subset of the virtual address space
- VMAs are linked with a red-black tree for fast lookup of the region corresponding to any virtual address
- Described by a `vm_area_struct`

```
[sys:~-2028] cat /proc/self/maps
562a517de000-562a517e6000 r-xp 00000000 08:01 2228226 /bin/cat
562a519e5000-562a519e6000 r--p 00007000 08:01 2228226 /bin/cat
562a519e6000-562a519e7000 rw-p 00008000 08:01 2228226 /bin/cat
562a52e6a000-562a52e8b000 rw-p 00000000 00:00 0 [heap]
7f2244bd0000-7f2244db7000 r-xp 00000000 08:01 14286948 /lib/x86_64-linux-gnu/libc-2.27.so
7f2244db7000-7f2244fb7000 ---p 001e7000 08:01 14286948 /lib/x86_64-linux-gnu/libc-2.27.so
7f2244fb7000-7f2244fbb000 r--p 001e7000 08:01 14286948 /lib/x86_64-linux-gnu/libc-2.27.so
7f2244fbb000-7f2244fbd000 rw-p 001eb000 08:01 14286948 /lib/x86_64-linux-gnu/libc-2.27.so
7f2244fbd000-7f2244fc1000 rw-p 00000000 00:00 0
7f2244fc1000-7f2244fe8000 r-xp 00000000 08:01 14286873 /lib/x86_64-linux-gnu/ld-2.27.so
7f224500e000-7f2245030000 rw-p 00000000 00:00 0
7f2245030000-7f22451cb000 r--p 00000000 08:01 262169 /usr/lib/locale/locale-archive
7f22451cb000-7f22451cd000 rw-p 00000000 00:00 0
7f22451e8000-7f22451e9000 r--p 00027000 08:01 14286873 /lib/x86_64-linux-gnu/ld-2.27.so
7f22451e9000-7f22451ea000 rw-p 00028000 08:01 14286873 /lib/x86_64-linux-gnu/ld-2.27.so
7f22451ea000-7f22451eb000 rw-p 00000000 00:00 0
7ffffdbb92000-7ffffdbbb3000 rw-p 00000000 00:00 0 [stack]
7ffffdbbea000-7ffffdbbed000 r--p 00000000 00:00 0 [vvar]
7ffffdbbed000-7ffffdbbef000 r-xp 00000000 00:00 0 [vdso]
fffffffff600000-fffffffff601000 r-xp 00000000 00:00 0 [vsyscall]
```

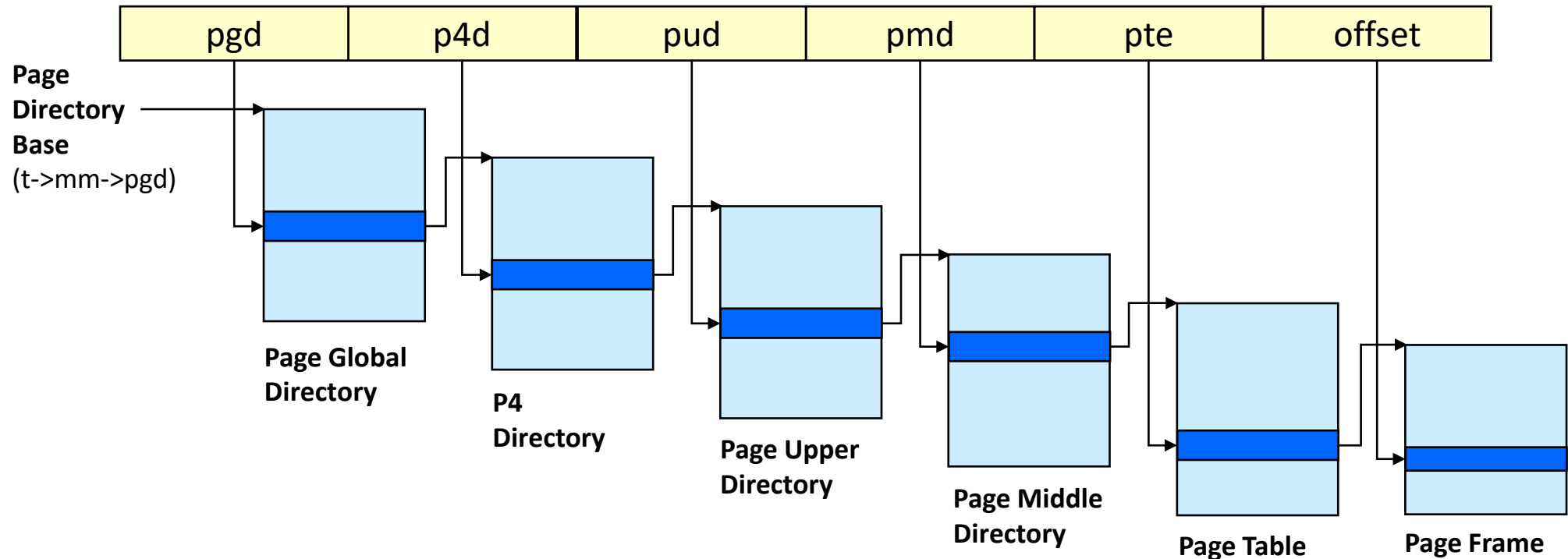
VMA Structure



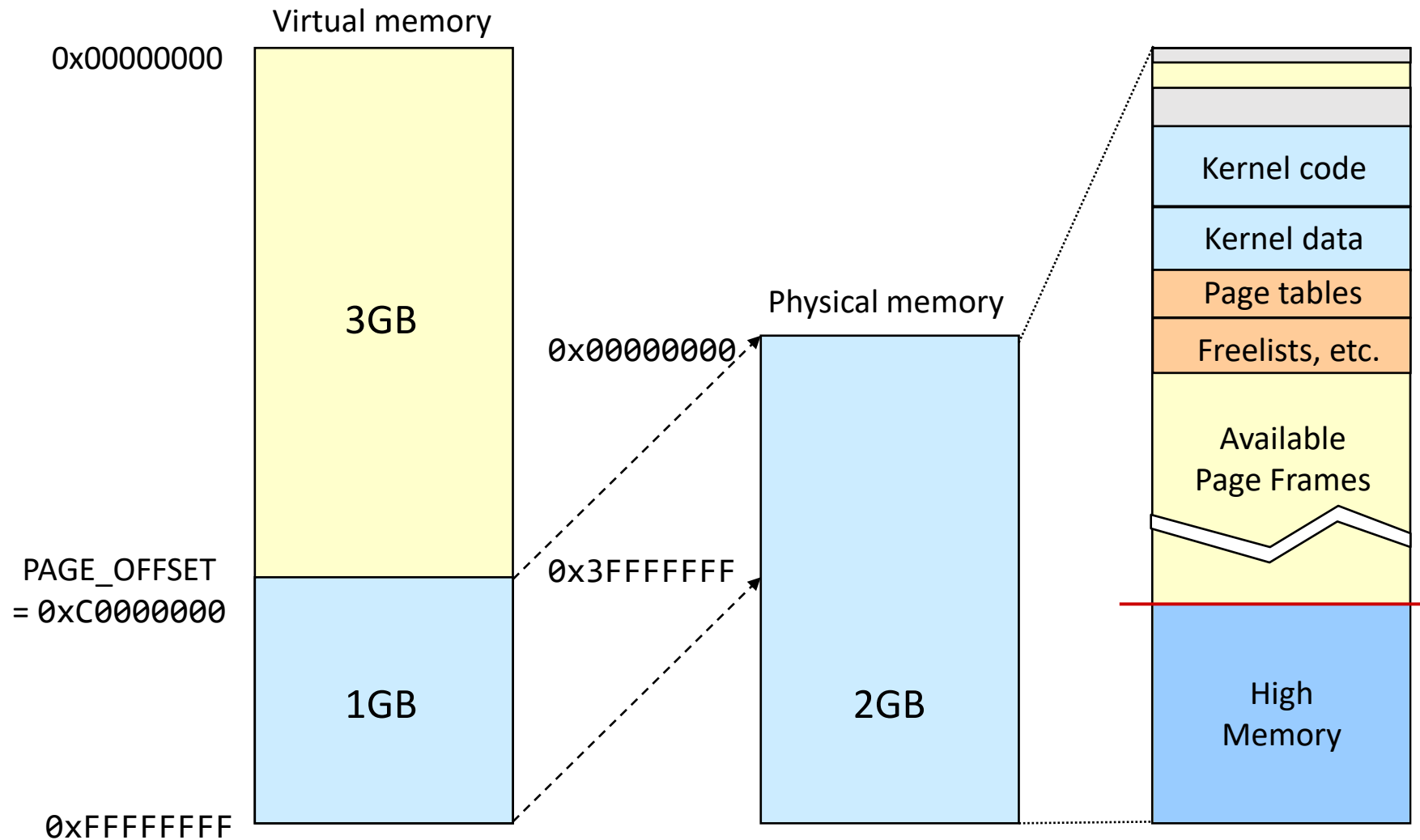
Paging in Linux

■ Five-level address translation

- 5-level paging for Intel “Ice Lake” processors and beyond (57-bit virtual address)
- For 48-bit virtual address, the size of P4D is set to 1

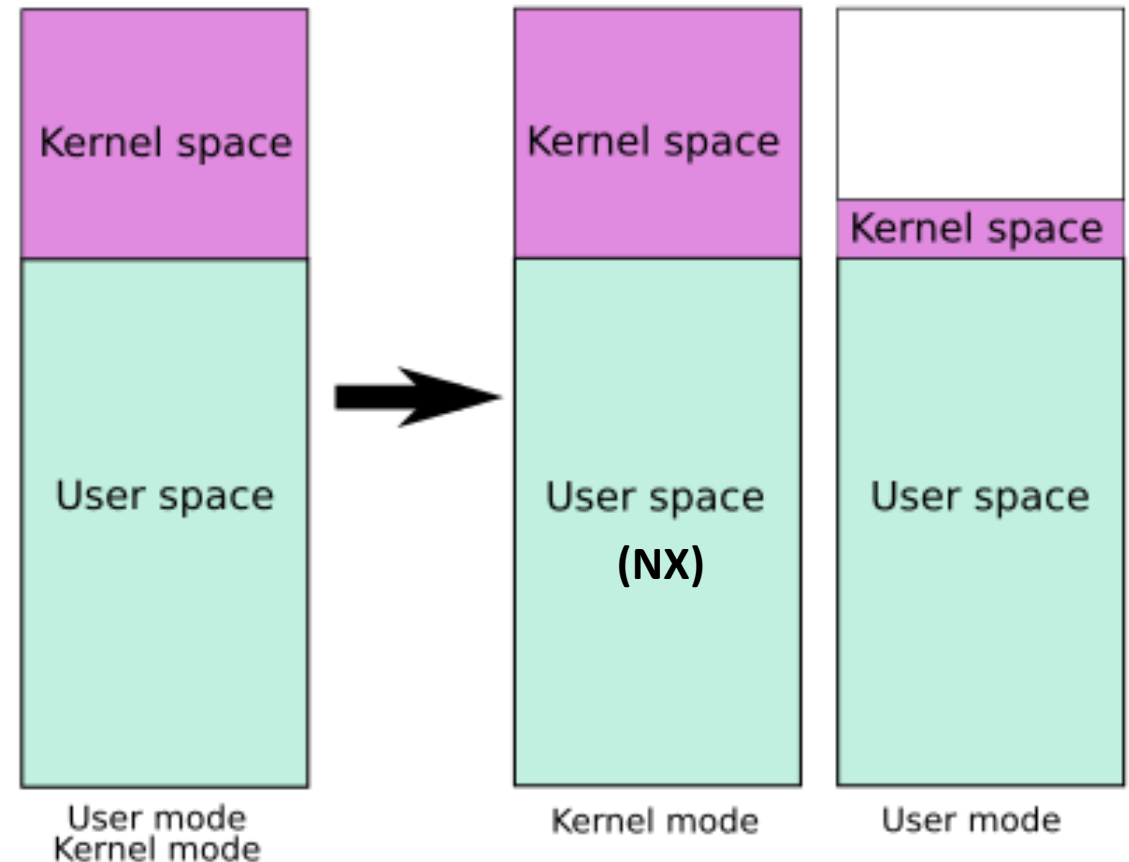


Managing Physical Memory (32-bit)



Kernel Page-Table Isolation (KPTI)

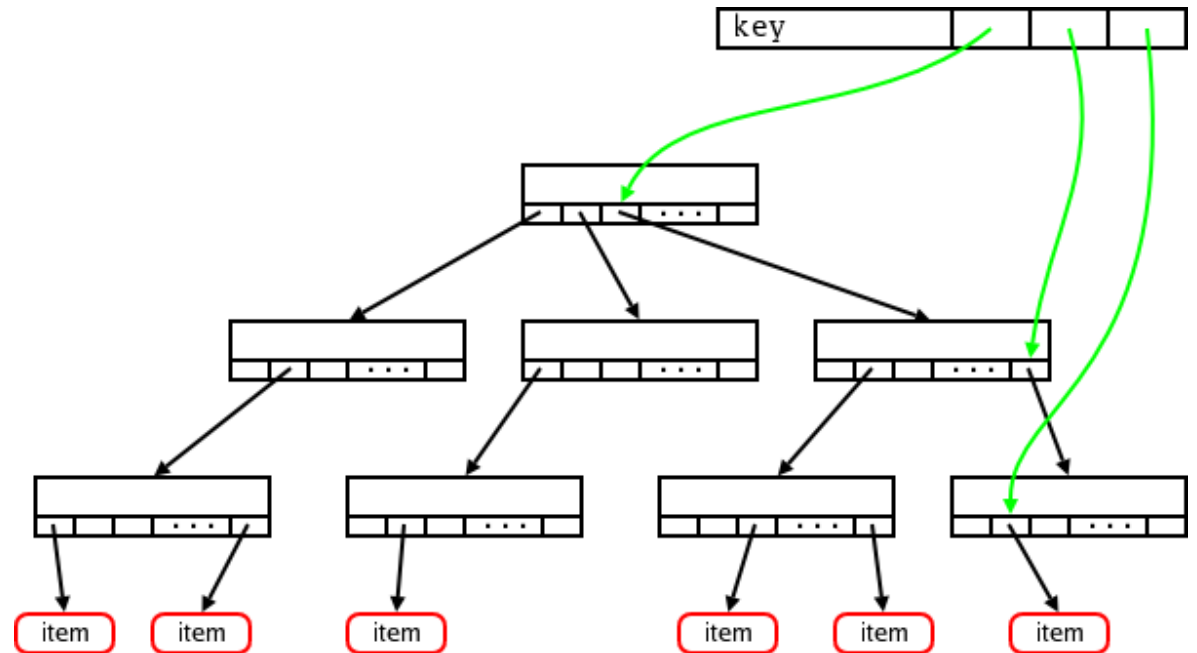
- To mitigate Meltdown vulnerability
- Separate page table for kernel
- Minimal kernel space for syscall, page fault & interrupt handling
- Merged in 4.15
- `CONFIG_PAGE_TABLE_ISOLATION=y`
- Disabled by 'nopti' at boot time
- ASID becomes critical to the performance



Page Cache

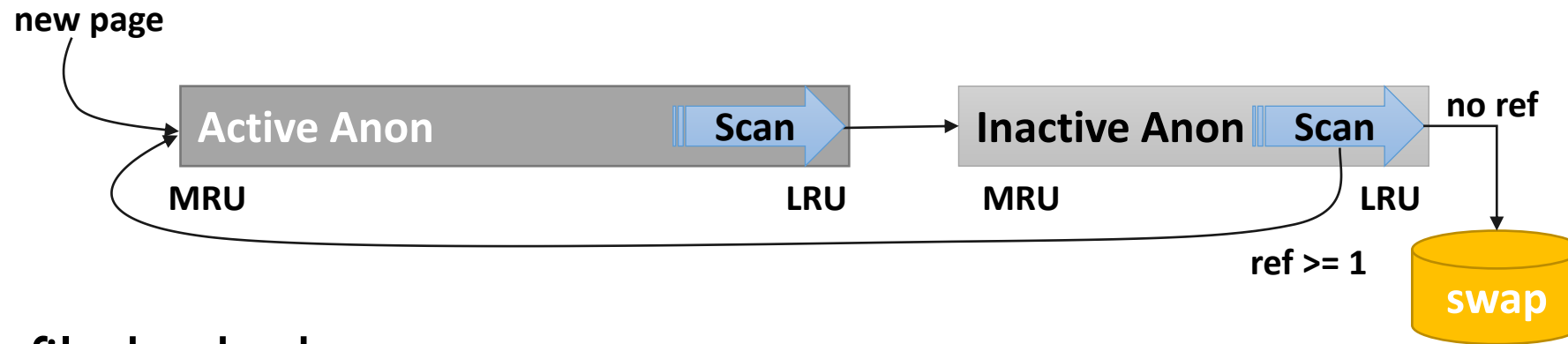
- A cache of pages in RAM
 - From reads and writes of regular filesystem files, block device files, mmap'ed files, ...
 - Group cached pages belonging to the same inode

- Page cache lookup
 - Each inode has a unique radix tree
 - Key: <inode, page offset>
 - The radix tree points to the cached page
 - Fanout: 64
(16 for small system)

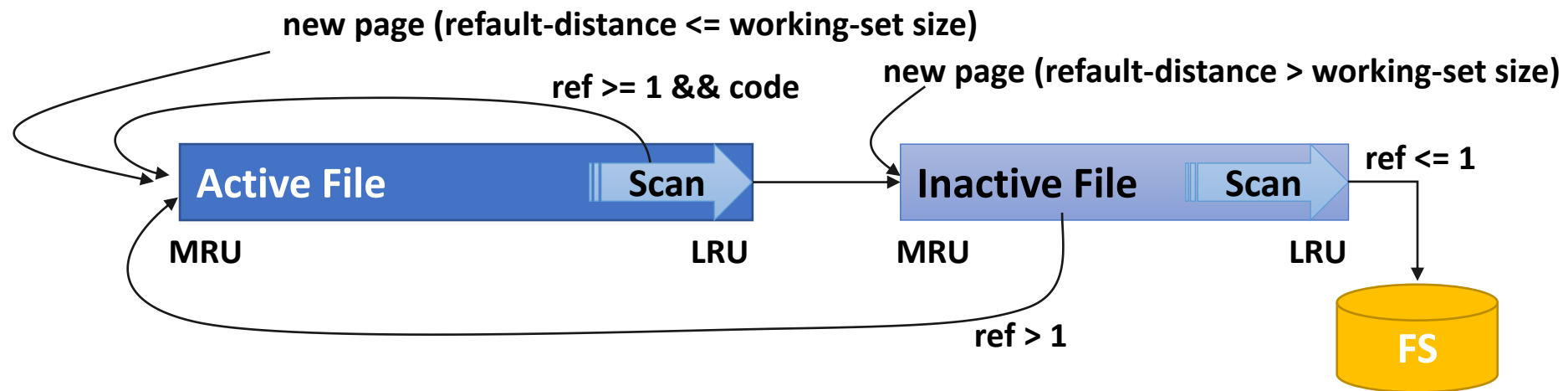


Linux Page Replacement (v5.x)

- For anonymous pages



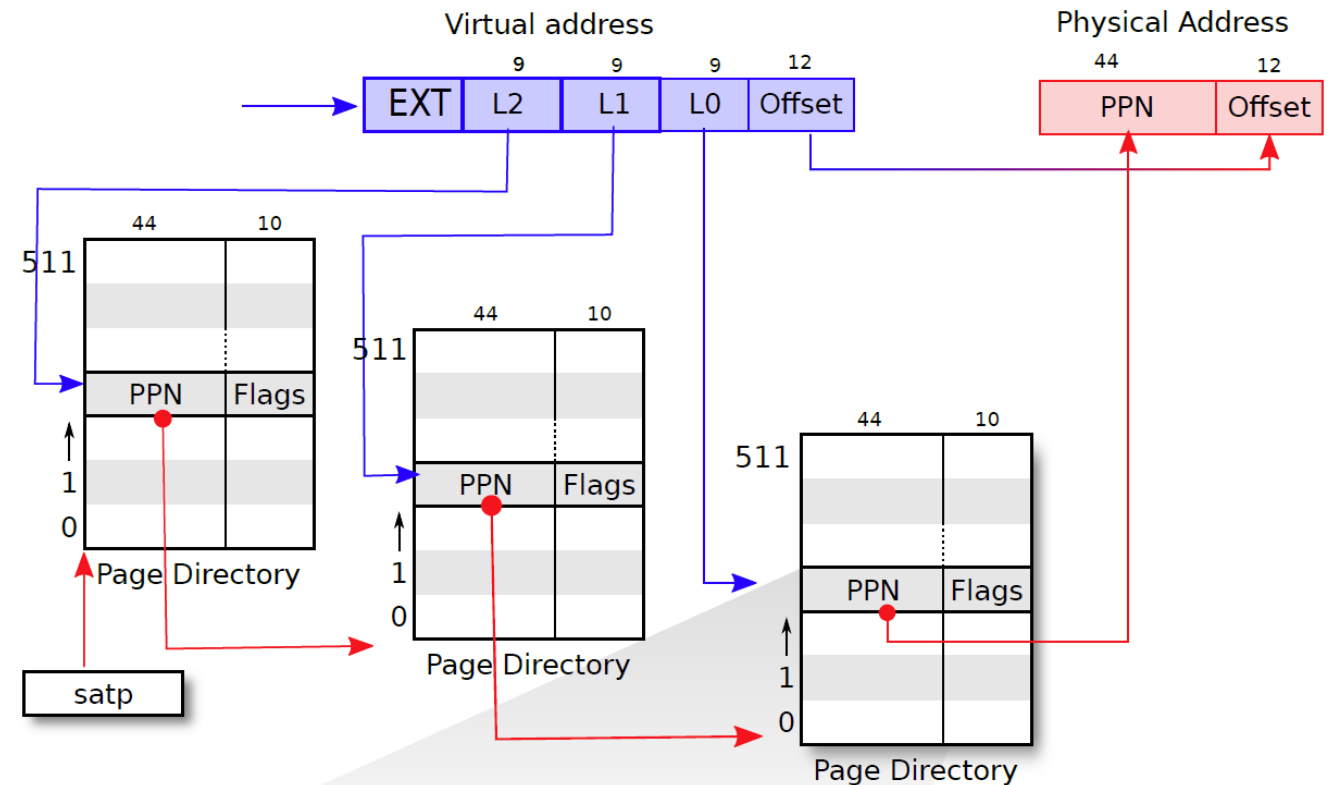
- For file-backed pages



Xv6 Virtual Memory System

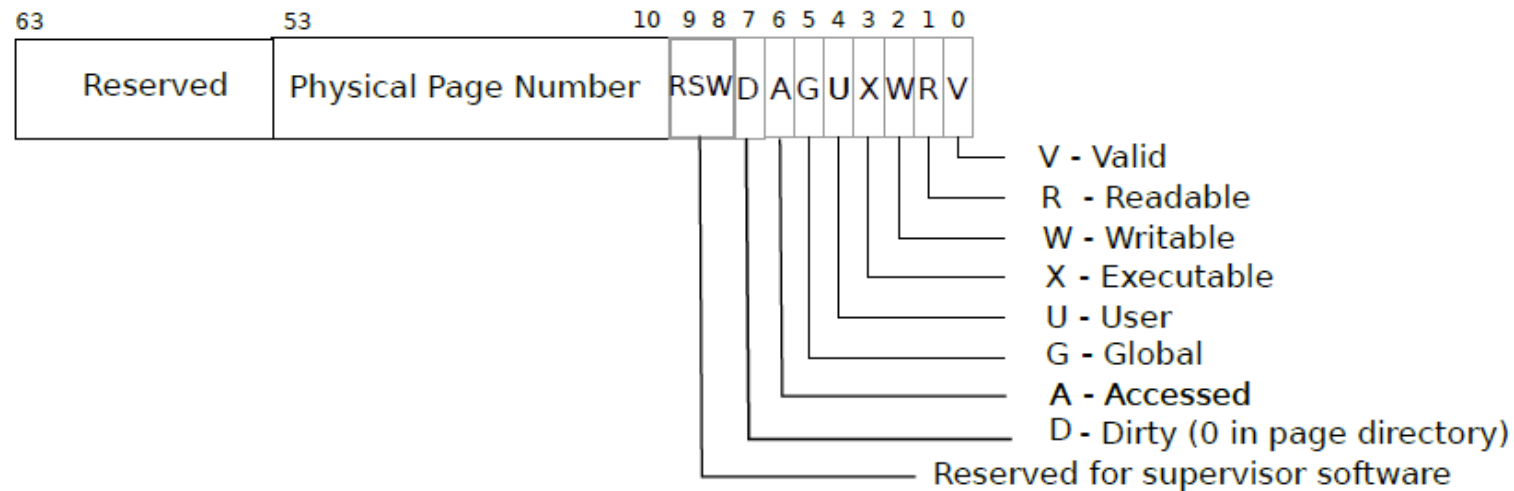
RISC-V Paging Hardware

- Sv39: page-based 39-bit virtual addressing
 - 39-bit virtual address → 56-bit physical address
 - 4KB page size
 - Three-level page table
 - satp register has the physical address of the root page table
 - Each CPU has its own satp register



RISC-V PTE

- Sv39 page table entry

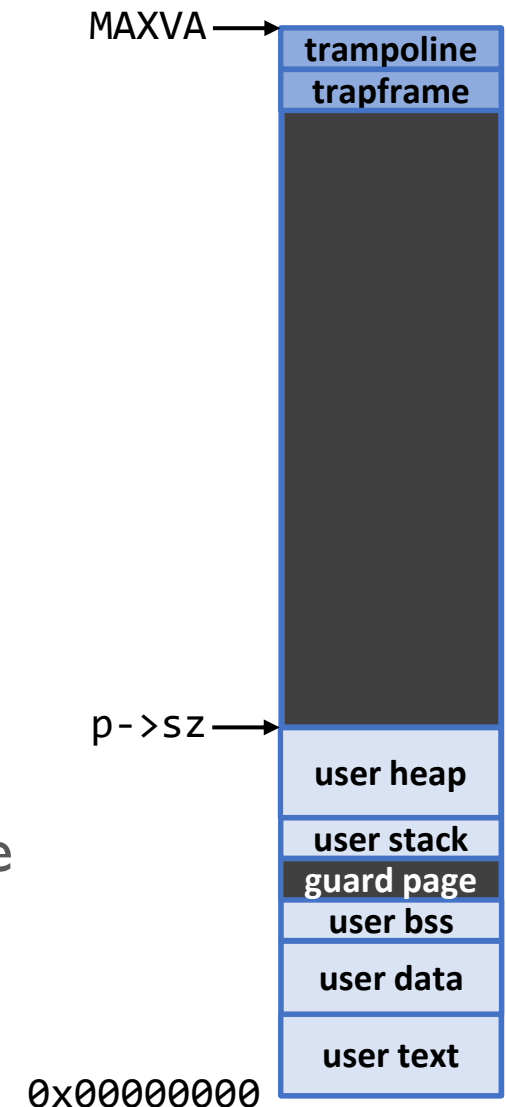


- Superpage support

- Any level of PTE may be a leaf PTE
- 2MB "megapages"
- 1GB "gigapages"

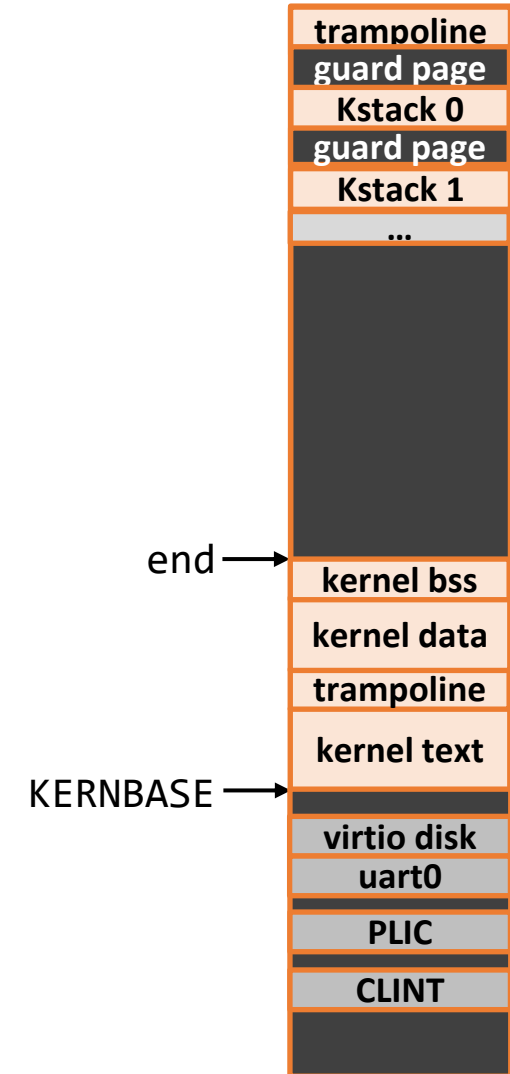
Process Address Space

- Xv6 uses only 38 bits for virtual addresses
 - $\text{MAXVA} = 2^{38} - 1 = 0x3ffffffffff$
 - `p->pagetable` points to the address of the root page table
- Guard page: used to detect stack overflow
- Trampoline page: contains the trap handler code
 - Should be mapped at the same virtual address both in kernel page table and in every user page table
- Trapframe page: used to save user registers
 - Needs to be accessed before switching to the kernel page table
 - Mapped to a per-process trapframe (`p->tf`) page in the kernel



Kernel Address Space

- Xv6 uses KPTI: the kernel has its own page table
- Identity mapping: most of the kernel's address is “direct-mapped” to physical memory
 - The kernel is located at KERNBASE (0x80000000) in both the virtual address space and in physical memory
 - I/O devices are mapped below KERNBASE
- Trampoline page is also mapped at the top of the kernel virtual address space
- Kernel stack pages: preallocated and separated with a guard page.



Physical Address Space

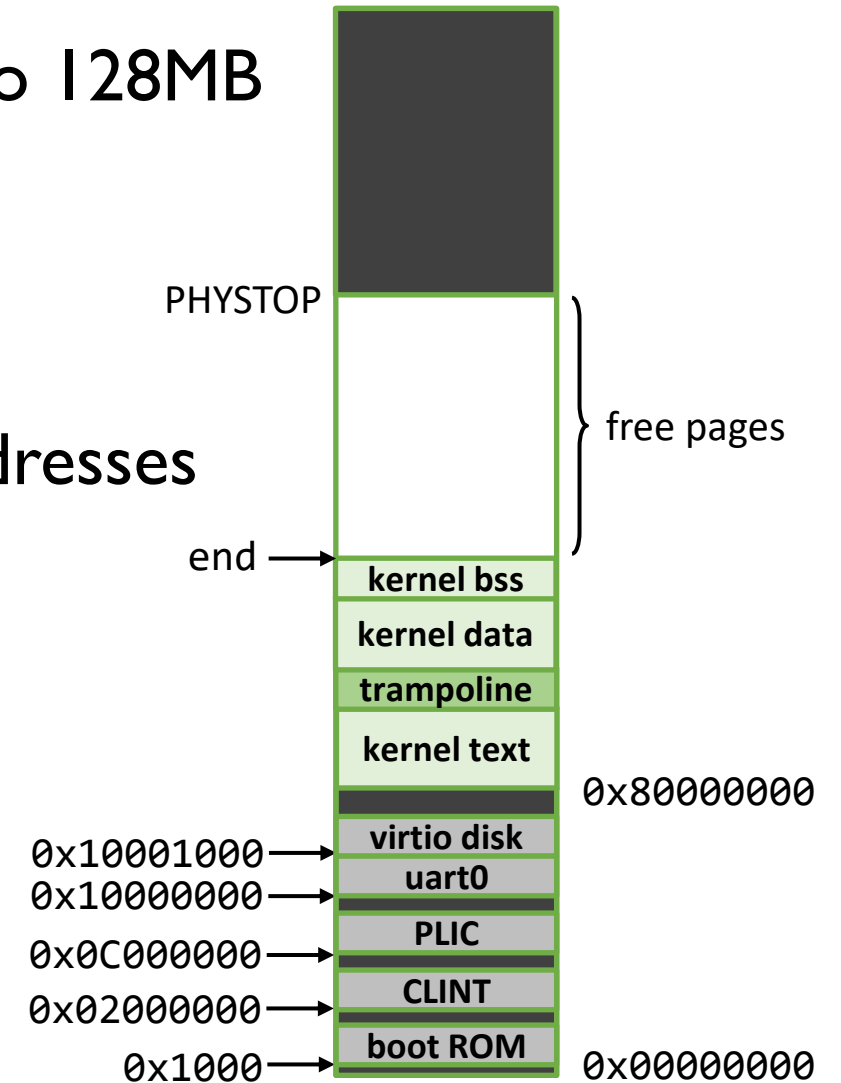
- Total amount of the physical memory is fixed to 128MB

- $\text{PHYSTOP} = \text{KERNBASE} + 128 * 1024 * 1024$
(@ kernel/memlayout.h)

- Physical memory starts at 0x80000000

- Various I/O devices are mapped to physical addresses below 0x80000000

- Boot ROM
- Core-Local Interruptor (CLINT)
- Platform-Level Interrupt Controller (PLIC)
- Console (uart0)
- Disk (virtio)



Summary

