

# Memory



ECE 373

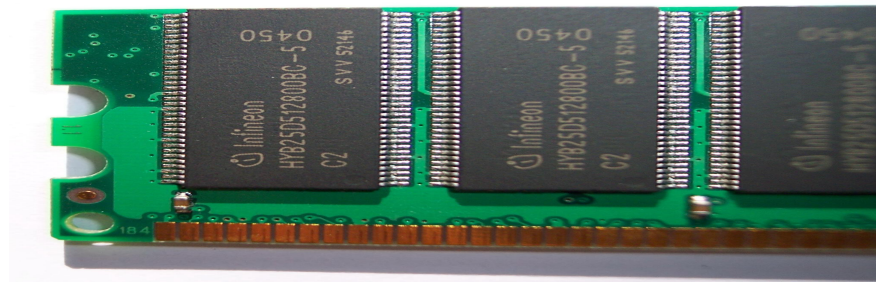
# Prelims

- Questions on lab or reading assignments, class?



# Linux memory basics

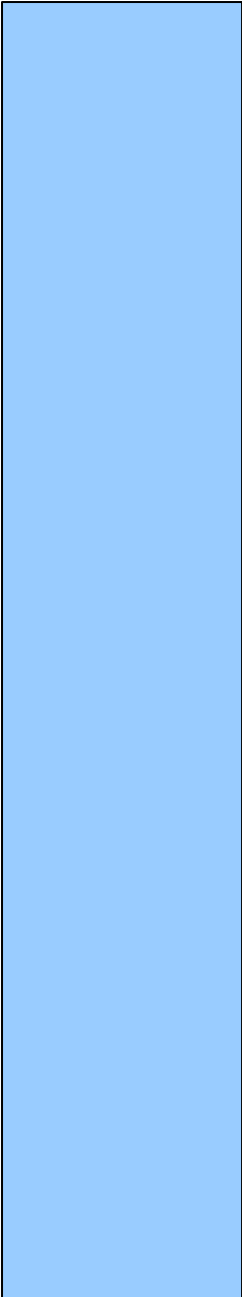
- Linux partitions memory into various chunks
- Physical, virtual, kernel logical, bus, kernel virtual
- How memory is partitioned dependent on:
  - Host architecture
  - Kernel configuration
  - Physical layout of chips



# Physical addresses

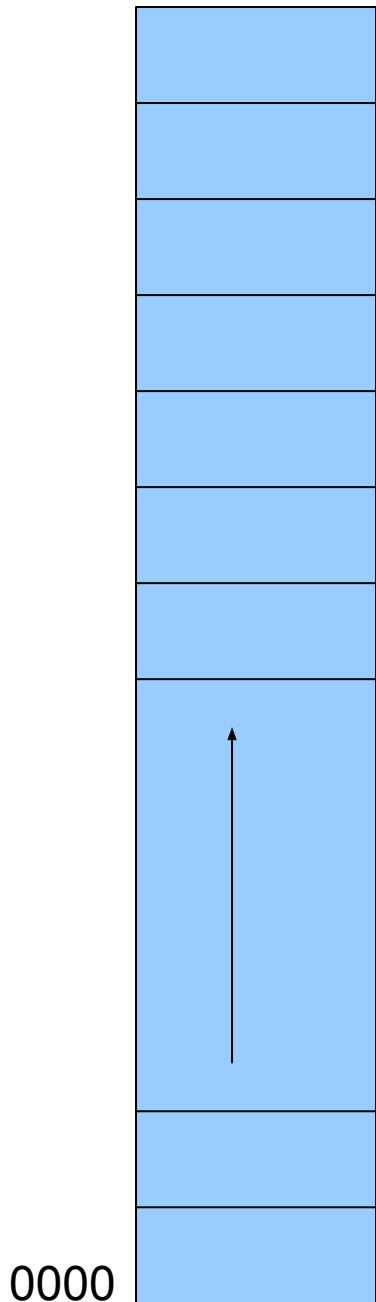
- Addresses the CPU uses between itself and RAM
- Systems we use (x86-64 / AMD64) use 64-bit addresses
- This space is all of internal RAM (the big pool)

0000



# Pages

- Physical memory cut up into pages
- Sizes vary between architectures
  - 4KB is common, some use 1MB
  - Linux defines `PAGE_SIZE` in `asm/page.h`
- Address = Page offset and page number
  - $(\text{page\_num} * \text{PAGE\_SIZE}) + \text{offset}$
- Tracked in kernel page tables
- TLB lookups



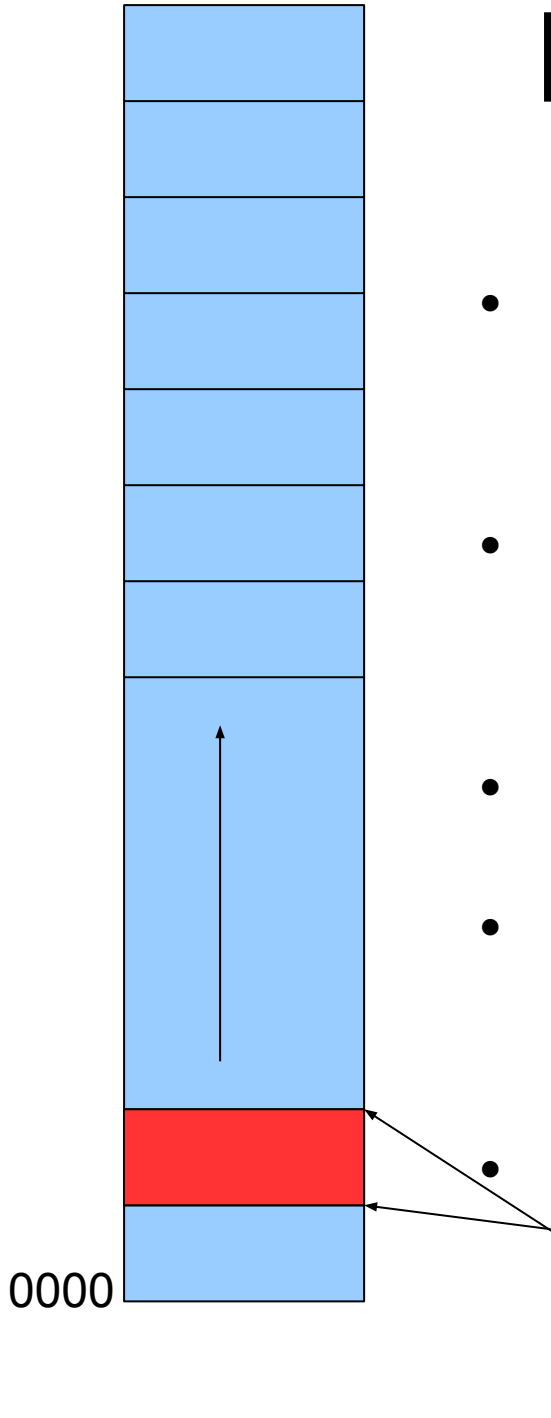
# Pages in Linux

- Pages maintain reference count
  - `alloc_page()` gets a page, increments `refcnt`
  - `free_page()` returns a page, decrements `refcnt`
  - page returned to free list when `refcnt == 0`
  - Most drivers use higher level `kmalloc/kfree`
- Free lists maintained by memory manager in the kernel

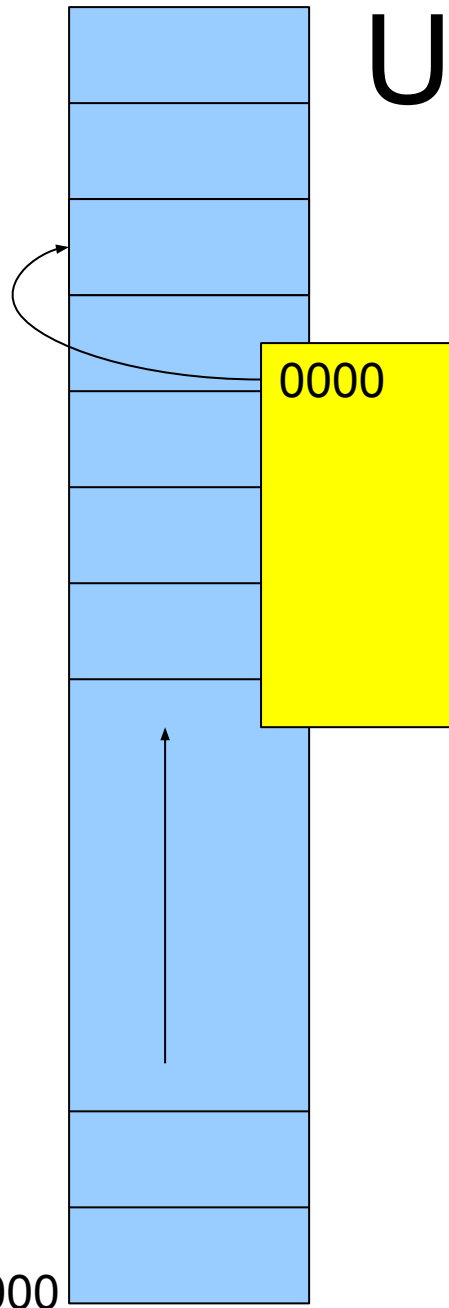
# Bus addresses



- How peripherals communicate with memory and CPU
- Peripherals include PCI bus, serial devices, etc.
- Typically same as physical address
- Times when they differ is bus isolation, or an IOMMU is used
- For this class, bus = physical



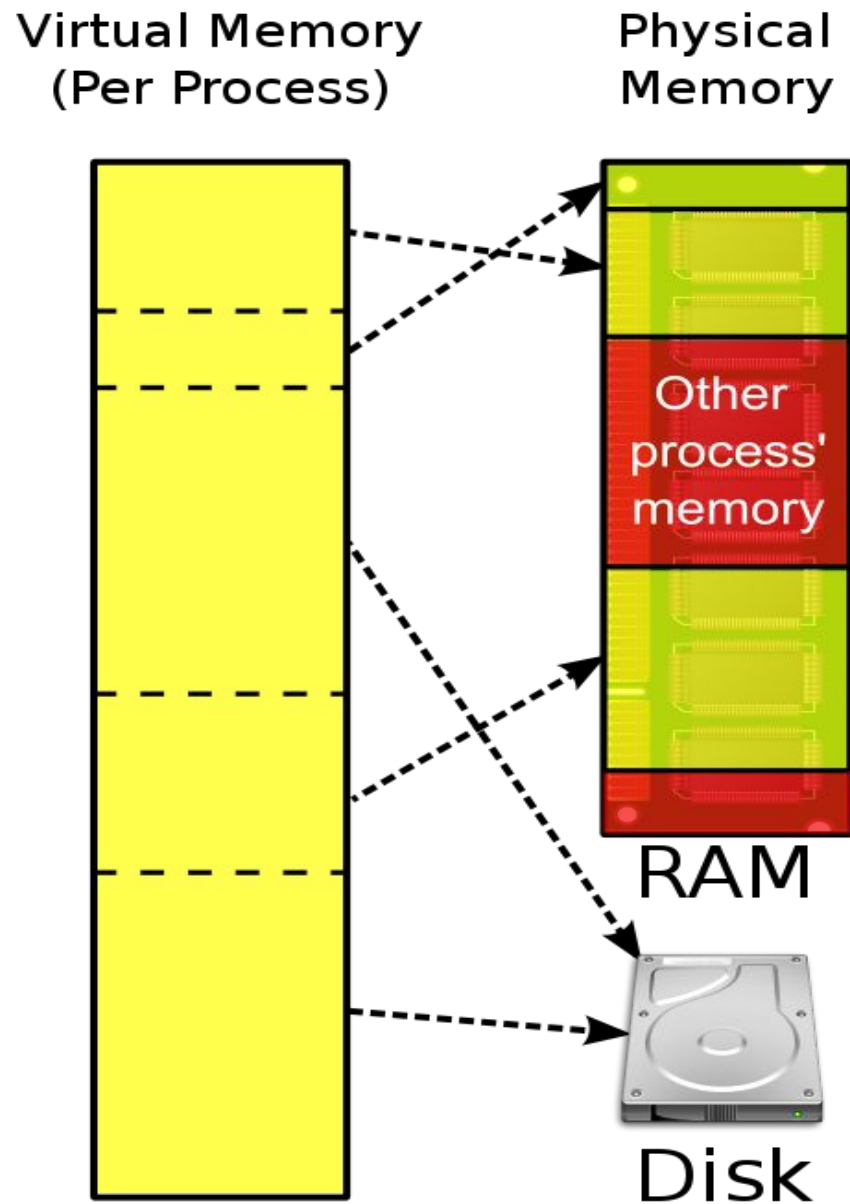
# User virtual addresses



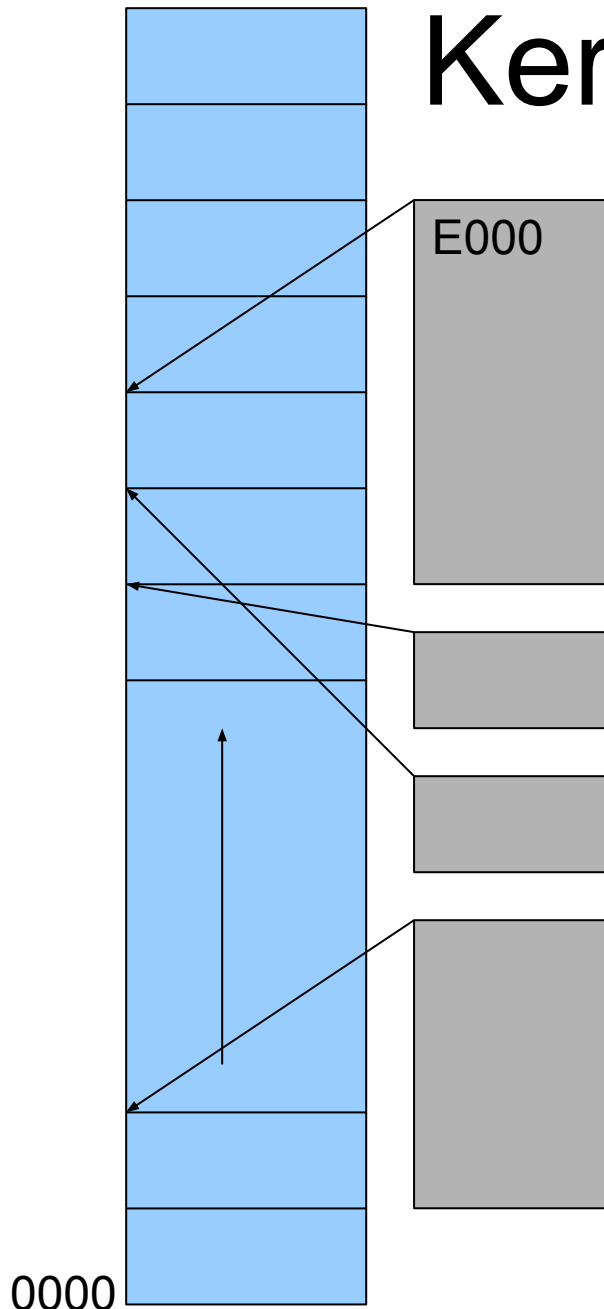
- Userspace applications always use virtual addresses
- Returned by things like `malloc()` and `new()`
- Virtual address space doesn't need to fit into actual amount of memory
- Processes get their own address space
- `SIGSEGV` when out-of-bounds



# User virtual address layout



# Kernel logical addresses



- All memory that kernel maps for kernel use
- Kernel memory that is returned by `kmalloc()`
- Not always physical address
- Mapped directly to physical address
- Typically at some constant offset from physical address
- Stored in things like `void *`

# Kernel virtual addresses

- Still mapped to physical addresses (isn't everything?)
- Not guaranteed to be one-to-one mapping to physical address
- Memory returned from `vmalloc()`
- Cannot be pinned

# Kernel virtual addresses

- Still mapped to physical addresses (isn't everything?)
- Not guaranteed to be one-to-one mapping to physical address
- Memory returned from `vmalloc()`
- Cannot be pinned
- Cannot be used for DMA or MMIO

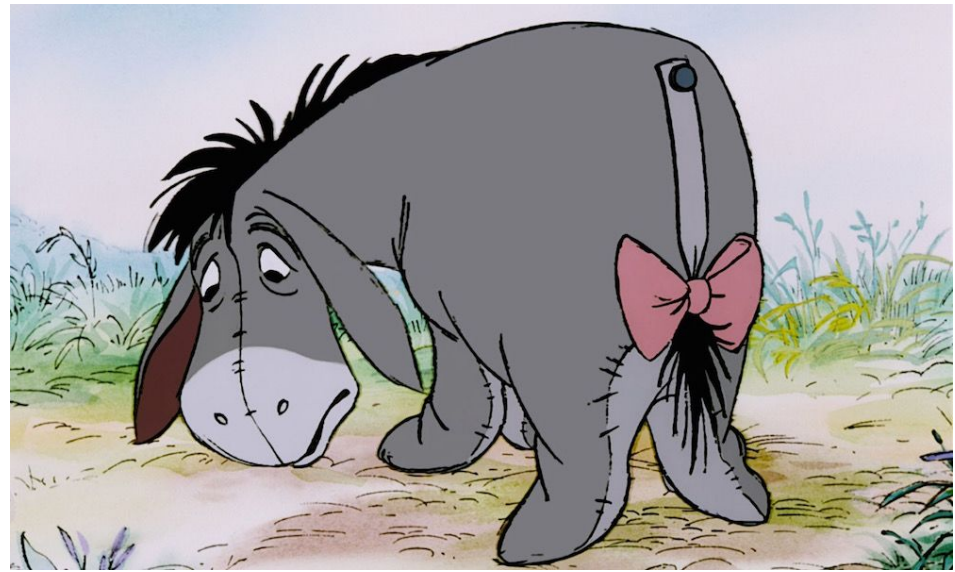
# The act of pinning

- Memory that cannot be swapped out
- Necessary for memory peripheral device or CPU needs



# The act of pinning

- Memory that cannot be swapped out
- Necessary for memory peripheral device or CPU needs
- Page faults can be fatal...
- Only physical, bus, or kernel logical addresses can be pinned



# virtual to physical (and back)

- Can derive physical address from virtual address
- Macros defined in `asm/page.h` to help
- `__pa()` - takes virtual, returns physical
- `__va()` - takes physical, returns virtual

# So what?

- CPU uses physical
- Peripherals use bus
- Users use virtual
- Kernel and drivers use logical, virtual, and bus
  - When and why?