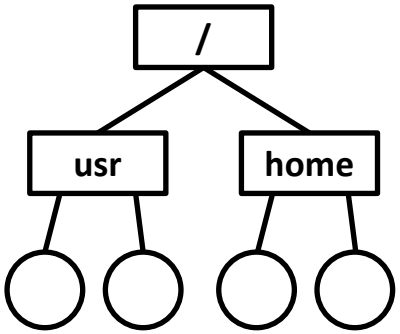
BIO

Block I/O disparity

Access unit: sector (512 bytes)
Max. request: 2^N sectors

Access unit: word (4~8 bytes)
Management unit: page (4KB)

Split large requests



Support sub-page cache

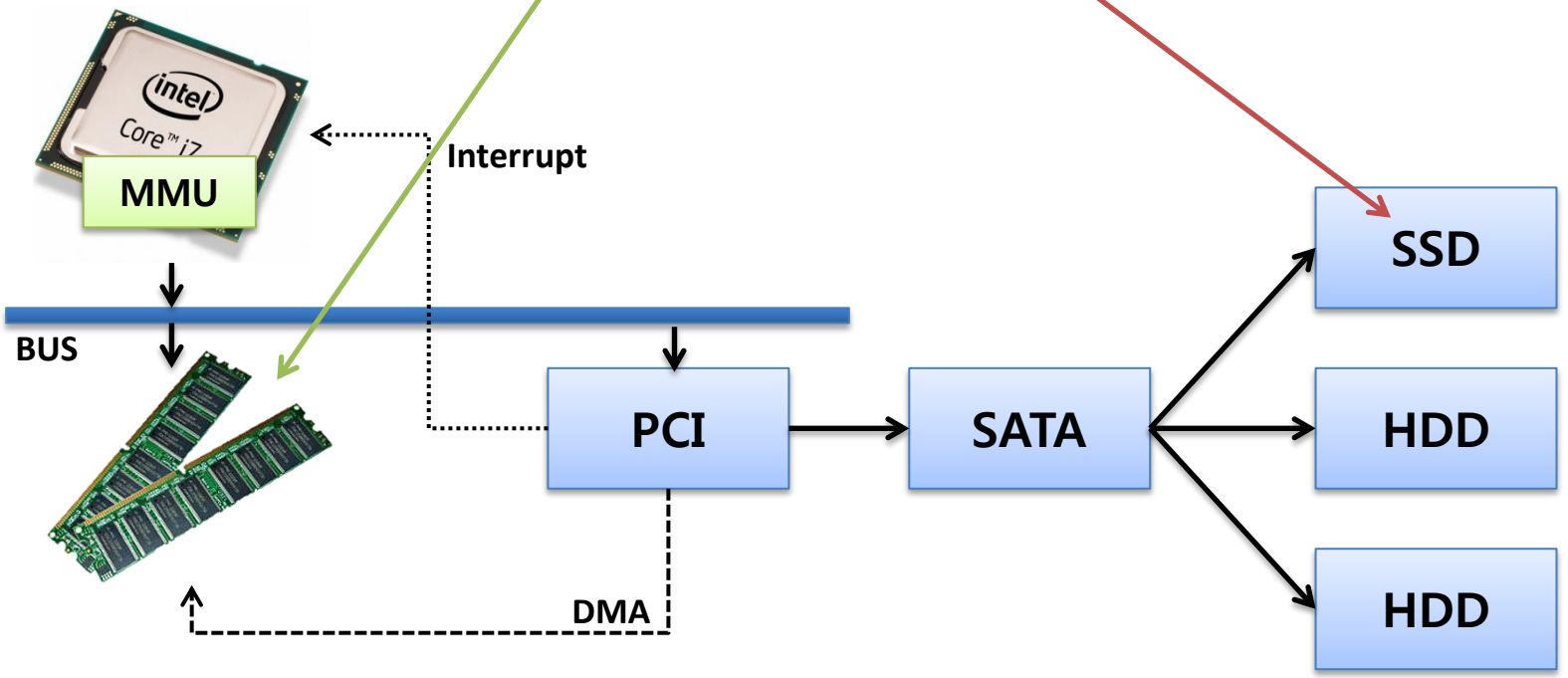
Access unit: block (2^n sectors)
Max. request: ∞

Cache management

- Unit of caching?
  - Page cache: efficiency of memory management
  - Buffer cache: support for block unit I/Os
- Linux implementation
  - Earlier linux: buffer cache only
  - Recent linux: buffer cache over page cache

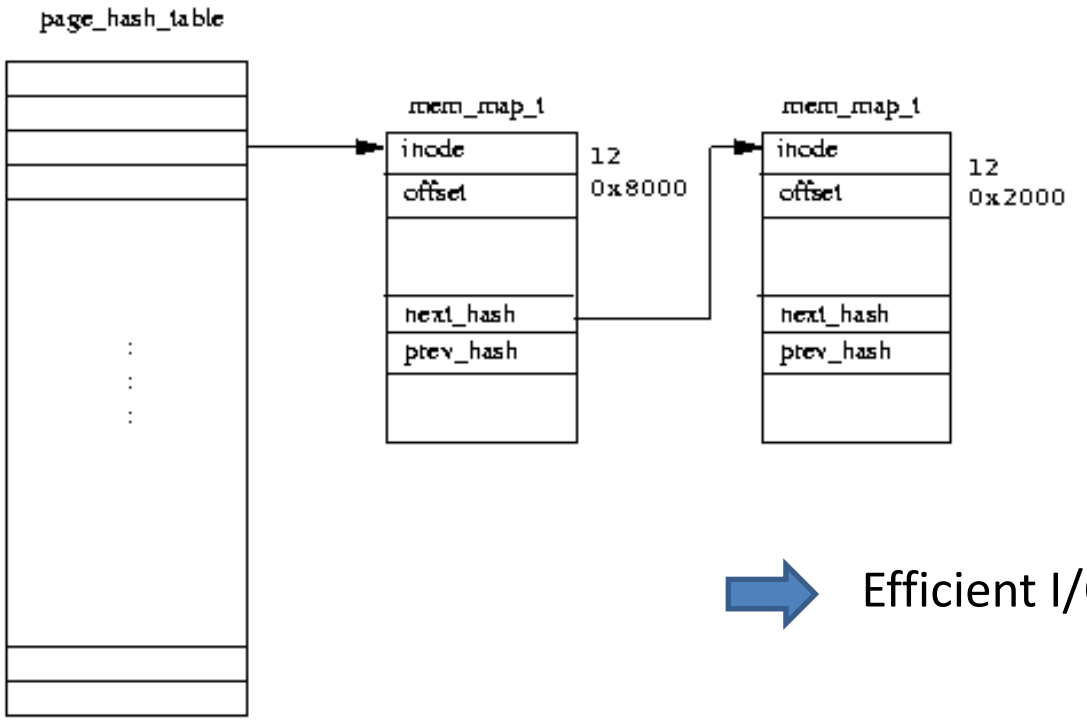
I/O subsystem architecture

- Read/write (device, addr, offset, nr sectors)
- Hard to manipulate



Page cache

- In-memory cache of disk blocks
  - Page unit
  - Retrieval: (inode, offset) ➔ `struct page *`

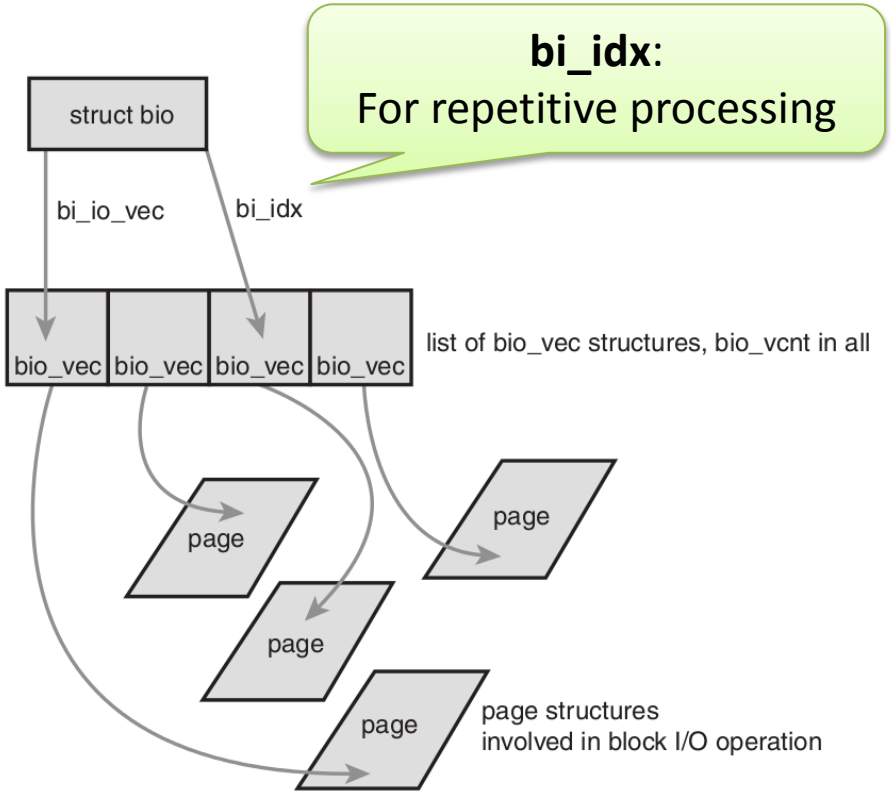


➡ Efficient I/Os in a page unit

struct bio

- Uses memory pages in storage I/O operations
  - No address translation: can use unmapped pages

Type	Name
block_device	**bi_bdev**
sector_t	**bi_sector**
struct bio *	next
long	bi_rw
int	bi_size
short	**bi_vcnt**
struct bio_vec *	**bi_io_vec**
bio_end_io_t	bi_end_io
void *	bi_private



New struct bio

- From v3.14, struct bio changed
  - For easier repetitive operations

Type	Name
block_device	**bi_bdev**
struct bio *	next
long	bi_rw
struct bvec_iter	**bi_iter**
short	**bi_vcnt**
struct bio_vec *	**bi_io_vec**
bio_end_io_t	bi_end_io
void *	bi_private

Type	Name
sector_t	bi_sector
int	bi_size
int	bi_idx
int	bi_bvec_done

Buffer cache

- A block-unit cache: struct buffer_head
  - For sub-page I/O requests