super_block 구조체 include/linux/fs.h dentry 구조체 include/linux/dcache.h → struct super_block { struct dentry { struct dentry *s_root; -→ struct dentry { struct hlist_bl_heads_anon; → struct hlist_bl_node d_hash; struct hlist bl node d hash; struct inode *d_inode; struct super_block *d_sb; struct dentry *d_parent; union { — struct inode *d_inode; path 구조체 struct lockref d_lockref; struct hlist_node d_alias; include/linux/path.h struct workqueue_struct *s_dio_done_wq; fs/dcache.c } d_u; struct path { static struct hlist_bl_head *dentry_hashtable; struct vfsmount *mnt; struct list_head d_lru; struct user_namespace *s_user_ns; struct dentry *dentry; ____ wait_queue_head_t *d_wait; Head → struct fdtable fdtab; struct list_head d_child; file 구조체 struct rcu_head rcu; struct list_head d_subdirs; include/linux/fs.h struct work_struct destroy_work; ___ struct path _____ struct hlist_node d_alias; __ f_path; struct list_head s_inodes_wb; ← *f_inode; — struct inode struct hlist_bl_node d_in_lookup_hash; __ struct address_space *f_mapping; struct rcu_head d_rcu; *private_data; struct hlist_bl_node d_hash; union { → struct list_head d_lru; ◆ struct list_head d_lru; struct dentry { →struct hlist_bl_node d_hash; struct list_lru s_dentry_lru ___cacheline_aligned_in_smp; 🔔 struct list_head d_lru; ← inode 구조체 (bdev file) inode 구조체 (Regular file) struct inode { struct inode { struct super_block *i_sb; struct super_block *i_sb; struct address_space *i_mapping; ----struct address_space *i_mapping; _ struct list_head i_io_list; struct list_head i_io_list; struct list_head i_lru; struct list_head i_lru; struct list_head i_sb_list;

struct list_head i_wb_list; struct list_head i_sb_list; struct list_head i_wb_list; struct rcu_head i_rcu; struct rcu_head i_rcu; fs/inode.c struct address_space { ← struct address_space { __ struct inode *host; static struct hlist_head *inode_hashtable; *private_data; }i_data; *i_private; *i_private; void void struct inode {

→ struct hlist_node i_hash;

← struct list_head i_lru; → struct hlist_node i_hash;

→ → struct list_head i_lru;
← struct list_lru s_inode_lru ___cacheline_aligned_in_smp; struct mount { struct list_head s_mounts; struct list_head mnt_instance; struct mount *mnt_parent; struct vfsmount { struct dentry *mnt_root; ——— — struct super_block *mnt_sb; struct list_head mnt_mounts; — -----. struct list_head mnt_child; struct task_struct { struct mnt_namespace { struct nsproxy { — struct mtd_info *s_mtd; struct dentry *mnt_mountpoint; struct nsproxy *nsproxy; — struct mnt_namespace *mnt_ns; — struct list_head list; struct list_head mnt_list; — — struct mnt_namespace *mnt_ns; — struct backing_dev_info *s_bdi; — struct block_device*s_bdev; void *s_fs_info; → struct xxx_sb_info { // File system private info. r ← struct mount { fs/super.c ____ struct mount *mnt_parent; static struct list_head super_blocks; struct vfsmount { struct super_block { struct super_block { struct dentry *mnt_root; struct list_head s_list; struct list_head s_list; struct list_head s_list; — struct super_block *mnt_sb; struct hlist_node s_instances; struct hlist_node s_instances; <</p> struct hlist_node s_instances; ◄— struct list_head mnt_child; -----. struct list_head mnt_mounts; struct list_head mnt_instance; fs/filesystems.c struct list_head mnt_list; static struct file_system_type *file_systems ___ struct mnt_namespace *mnt_ns; file_system_type 구조체 include/linux/fs.h struct file_system_type { struct file_system_type { ► NULL const char *name; const char *name; int fs_flags; int fs_flags; struct dentry *(*mount) (struct struct dentry *(*mount) (struct file_system_type *, int, const char *, void *); file system type *, int, const char *, void *) struct mount *mnt_parent; void (*kill_sb) (struct super_block *); void (*kill_sb) (struct super_block *); struct dentry *mnt_mountpoint; struct module *owner; struct module *owner; struct file_system_type * next; ----struct file_system_type * next; ——— struct vfsmount { struct dentry *mnt_root; struct super_block *mnt_sb; → struct hlist_head fs_supers; — struct list_head mnt_child; struct list_head mnt_mounts; struct list_head mnt_instance; struct block_device { → struct mtd_info { — struct list_head mnt_list; ◄ ____ struct mnt_namespace *mnt_ns; — struct super_block *bd_super; * A Memory Technology Device * is a type of device file in Linux bdev_inode 구조체 struct block_device * bd_contains; * for interacting with flash memory. fs/block_dev.c fs/block_dev.c struct backing_dev_info *bd_bdi; static struct list_head all_bdevs; struct bdev_inode {
 struct block_device { — struct request_queue * bd_queue; ■ struct list_head bd_list;
■ struct list_head bd_list_list_list_list_l _ struct gendisk * bd_disk; struct hlist_node i_hash; ◀ ___ struct inode *host; struct address_space *i_mapping; — struct request_queue *queue; struct request_queue { — struct block_device *i_bdev; struct backing_dev_info *backing_dev_info; }vfs_inode;

zone_mem_map은 2.6 버전 page_to_pfn에서 사용되었으나, page_to_pfn은 zone 대신 pgdat을 사용해서 구현가능하다. page_to_pfn은 zone_mem_map 변수를 사용하는 유일한 함수이다. 따라서 zone_mem_map은 제거되었다.

include/linux/mmzone.h	arch/x86/include/asm/mmzone_64.h	
edef struct pglist_data {	extern struct pglist_data *node_data[];	
struct zone node_zones[MAX_NR_ZONES]; struct zonelist node_zonelists[MAX_ZONELISTS];		
eans !SPARSEMEM */ ef CONFIG_FLAT_NODE_MEM_MAP	E cf ·	Discolar all address a
struct page *node_mem_map;	Physical address	Physical address
dif j_data_t;		
		Node #0
	Single node	
		Node #n
	FLATMEM	DISCONTIGMEM

task_struct 구조체

include/linux/sched.h

struct files_struct *files;

struct task_struct {

files_struct 구조체

include/linux/fdtable.h

struct files_struct {