```
  1: #ifndef _PROC_IPC_SHM_H    /* wrapper symbol for kernel use */
  2: #define _PROC_IPC_SHM_H    /* subject to change without notice */
 ...
 63:
 64: /*
 65:  * Message Operation Flags.
 66:  */
 67:
 68: #define    SHM_RDONLY    010000    /* attach read-only (else read-write) */
 69: #define    SHM_RND        020000    /* round attach address to SHMLBA */
 70:
 71:
 72: typedef ulong_t shmatt_t;
 73:
 74:
 75: /*
 76:  * Structure Definitions.
 77:  */
 78:
 79: #if !defined(_STYPES)
 80:
 81: /*
 82:  * There is a shared mem id data structure (kshmid_ds and shmid_ds) for each
 83:  * segment in the system.
 84:  */
 85: struct shmid_ds {
 86:     struct ipc_perm    shm_perm;    /* operation permission struct */
 87:     int        shm_segsz;    /* size of segment in bytes */
 88:     _VOID        *shm_pad0;    /* placeholder for historical shm_amp */
 89:     ushort_t    shm_lkcnt;    /* number of times it is being locked */
 90:     pid_t        shm_lpid;    /* pid of last shmop */
 91:     pid_t        shm_cpid;    /* pid of creator */
 92:     shmatt_t    shm_nattch;    /* used only for shminfo */
 93:     ulong_t        shm_cnattch;    /* used only for shminfo */
 94:     time_t        shm_atime;    /* last shmat time */
 95:     long        shm_pad1;    /* reserved for time_t expansion */
 96:     time_t        shm_dtime;    /* last shmdt time */
 97:     long        shm_pad2;    /* reserved for time_t expansion */
 98:     time_t        shm_ctime;    /* last change time */
 99:     long        shm_pad3;    /* reserved for time_t expansion */
100:     long        shm_pad4[SHM_PAD1];    /* reserve area  */
101: };
102:
103: #else    /* _STYPES */
104:
105: /* SVR3 binary compatibility shmid_ds */
106: struct shmid_ds {
107:     struct ipc_perm    shm_perm;    /* operation permission struct */
108:     int        shm_segsz;    /* size of segment in bytes */
109:     _VOID        *shm_pad0;    /* placeholder for historical shm_reg */
110:     ushort_t    shm_lkcnt;    /* number of times it is being locked */
111:     char         pad[SHM_PAD];
112:     o_pid_t        shm_lpid;    /* pid of last shmop */
113:     o_pid_t        shm_cpid;    /* pid of creator */
114:     ushort_t    shm_nattch;    /* used only for shminfo */
115:     ushort_t    shm_cnattch;    /* used only for shminfo */
116:     time_t        shm_atime;    /* last shmat time */
117:     time_t        shm_dtime;    /* last shmdt time */
118:     time_t        shm_ctime;    /* last change time */
119: };
120:
121: #endif    /* _STYPES */
 ...
170:
171:
172: struct    shminfo {
173:     int    shmmax,        /* max shared memory segment size */
174:         shmmin,        /* min shared memory segment size */
175:         shmmni,        /* # of shared memory identifiers */
176:         shmseg;        /* max attached shared memory      */
177:                 /* segments per process          */
178: };
179:
 ...
202:
203: #ifdef    _KERNEL
204:
205: /*
```



```
  1: #ifndef _LINUX_SHM_H_
  2: #define _LINUX_SHM_H_
  3:
  4: #include <linux/ipc.h>
  5: #include <asm/page.h>
  6:
  7: /*
  8:  * SHMMAX, SHMMNI and SHMALL are upper limits are defaults which can
  9:  * be increased by sysctl
 10:  */
 11:
 12: #define SHMMAX 0x2000000         /* max shared seg size (bytes) */
 13: #define SHMMIN 1             /* min shared seg size (bytes) */
 14: #define SHMMNI 4096             /* max num of segs system wide */
 15: #define SHMALL (SHMMAX/PAGE_SIZE*(SHMMNI/16)) /* max shm system wide (pages) */
 16: #define SHMSEG SHMMNI             /* max shared segs per process */
 17:
 18: #include <asm/shmparam.h>
 19:
 20: /* Obsolete, used only for backwards compatibility and libc5 compiles */
 21: struct shmid_ds {
 22:     struct ipc_perm        shm_perm;    /* operation perms */
 23:     int            shm_segsz;    /* size of segment (bytes) */
 24:     __kernel_time_t        shm_atime;    /* last attach time */
 25:     __kernel_time_t        shm_dtime;    /* last detach time */
 26:     __kernel_time_t        shm_ctime;    /* last change time */
 27:     __kernel_ipc_pid_t    shm_cpid;    /* pid of creator */
 28:     __kernel_ipc_pid_t    shm_lpid;    /* pid of last operator */
 29:     unsigned short        shm_nattch;    /* no. of current attaches */
 30:     unsigned short         shm_unused;    /* compatibility */
 31:     void             *shm_unused2;    /* ditto - used by DIPC */
 32:     void            *shm_unused3;    /* unused */
 33: };
 34:
 35: /* Include the definition of shmid64_ds and shminfo64 */
 36: #include <asm/shmbuf.h>
 37:
 38: /* permission flag for shmget */
 39: #define SHM_R        0400    /* or S_IRUGO from <linux/stat.h> */
 40: #define SHM_W        0200    /* or S_IWUGO from <linux/stat.h> */
 41:
 42: /* mode for attach */
 43: #define    SHM_RDONLY    010000    /* read-only access */
 44: #define    SHM_RND        020000    /* round attach address to SHMLBA boundary */
 45: #define    SHM_REMAP    040000    /* take-over region on attach */
 46:
 47: /* super user shmctl commands */
 48: #define SHM_LOCK     11
 49: #define SHM_UNLOCK     12
 50:
 51: /* ipcs ctl commands */
 52: #define SHM_STAT     13
 53: #define SHM_INFO     14
 54:
 55: /* Obsolete, used only for backwards compatibility */
 56: struct    shminfo {
 57:     int shmmax;
 58:     int shmmin;
 59:     int shmmni;
 60:     int shmseg;
 61:     int shmall;
 62: };
 63:
 64: struct shm_info {
 65:     int used_ids;
 66:     unsigned long shm_tot;    /* total allocated shm */
 67:     unsigned long shm_rss;    /* total resident shm */
 68:     unsigned long shm_swp;    /* total swapped shm */
 69:     unsigned long swap_attempts;
 70:     unsigned long swap_successes;
 71: };
 72:
 73: #ifdef __KERNEL__
 74:
 75: /* shm_mode upper byte flags */
 76: #define    SHM_DEST    01000    /* segment will be destroyed on last detach */
 77: #define SHM_LOCKED      02000   /* segment will not be swapped */
 78:
```


(For readability, not all matches are indicated.)

```
206:  * Macros to lock and unlock a kshmid_ds cell.
207:  * idp is a pointer to the kshmid_ds cell to be locked/unlocked.
208:  */
209: #define SHMID_LOCK(kshmp)    LOCK_PLMIN(&(kshmp)->kshm_lck)
210: #define SHMID_UNLOCK(kshmp, pl)    UNLOCK_PLMIN(&(kshmp)->kshm_lck, (pl))
211:
212: struct proc;
213: extern int    shmconv(int , struct kshmid_ds **);
214: extern void    shminit(void);
215: extern void    shmfork(struct proc *, struct proc *);
216: extern void    shmexit(struct proc *pp);
217: extern void    shmexec(struct proc *pp);
218:
219: #else
220:
221: #ifdef __STDC__
222:
223: void    *shmat(int, const void *, int);
224: int    shmctl(int, int, struct shmid_ds *);
225: int    shmdt(const void *);
226: int    shmget(key_t, size_t, int);
227: #else
228:
229: int shmctl();
230: int shmget();
231: void *shmat();
232: int shmdt();
233:
234: #endif /* __STDC__ */
235:
236: #endif /* _KERNEL */
237:
238: #if defined(__cplusplus)
239:     }
240: #endif
241:
242: #endif /* _PROC_IPC_SHM_H */
```



```
 79: asmlinkage long sys_shmget (key_t key, size_t size, int flag);
 80: asmlinkage long sys_shmat (int shmid, char *shmaddr, int shmflg, unsigned long *addr);
 81: asmlinkage long sys_shmdt (char *shmaddr);
 82: asmlinkage long sys_shmctl (int shmid, int cmd, struct shmid_ds *buf);
 83: extern void shm_unuse(swp_entry_t entry, struct page *page);
 84:
 85: #endif /* __KERNEL__ */
 86:
 87: #endif /* _LINUX_SHM_H_ */
```


(For readability, not all matches are indicated.)