```
  1: #ifndef _PROC_IPC_MSG_H    /* wrapper symbol for kernel use */
  2: #define _PROC_IPC_MSG_H    /* subject to change without notice */
 ...
 88:
 89: /*
 90:  * Message Operation Flags.
 91:  */
 92:
 93: #define    MSG_NOERROR    010000    /* no error if message to big on rcv */
 94:
 95:
 96: #if !defined(_STYPES)
 97:
 98: /*
 99:  * There is one msg queue id data structure for each msg queue in the system.
100:  */
101: struct msqid_ds {
102:     struct ipc_perm    msg_perm;    /* operation permission struct */
103:     struct msg    *msg_first;    /* ptr to first message on q */
104:     struct msg    *msg_last;    /* ptr to last message on q */
105:     ulong_t        msg_cbytes;    /* current # bytes on q */
106:     msgqnum_t    msg_qnum;    /* # of messages on q */
107:     msglen_t    msg_qbytes;    /* max # of bytes on q */
108:     pid_t        msg_lspid;    /* pid of last msgsnd */
109:     pid_t        msg_lrpid;    /* pid of last msgrcv */
110:     time_t        msg_stime;    /* last msgsnd time */
111:     long        msg_pad1;    /* reserved for time_t expansion */
112:     time_t        msg_rtime;    /* last msgrcv time */
113:     long        msg_pad2;    /* time_t expansion */
114:     time_t        msg_ctime;    /* last change time */
115:     long        msg_pad3;    /* time_t expansion */
116:     long        msg_pad4[MSG_PAD];    /* reserve area */
117: };
118:
119: #else    /* _STYPES */
120:
 ...
192: /*
193:  * There is one msg structure for each message that may be in the system.
194:  */
195: struct msg {
196:     struct msg    *msg_next;    /* ptr to next message on queue */
197:     long        msg_type;    /* message type */
198:     void        *msg_vaddr;    /* ptr to msg text (and header) */
199:     ulong_t        msg_ts;        /* msg text size in bytes */
200: };
201:
202: /*
203:  * Message information structure.
204:  */
205: struct msginfo {
206:     int    msgmap;    /* # of entries in msg map */
207:     int    msgmax;    /* max message size */
208:     int    msgmnb;    /* max # bytes on queue */
209:     int    msgmni;    /* # of message queue identifiers */
210:     int    msgssz;    /* msg segment size (should be word size multiple) */
211:     int    msgtql;    /* # of system message headers */
212:     ushort_t msgseg;    /* # of msg segments (MUST BE < 32768) */
213: };
214:
215: struct msgstat {
216:     long    msgtxt;        /* bytes of message text configured */
217:     long    msgtxtresv;    /* bytes of message text currently reserved */
218:     long    msghdresv;    /* # of message headers currently reserved */
219:     long    msgtxtfail;    /* # of times failed message text reserve */
220:     long    msghdfail;    /* # of times failed message header reserve */
221: };
222:
223: #endif  /* _KERNEL || _KMEMUSER */
224:
225:
226: /*
227:  * User message buffer template for msgsnd() and msgrcv() system calls.
228:  */
229: #if !defined(_XOPEN_SOURCE)
230: struct msgbuf {
231:     long    mtype;        /* message type */
232:     char    mtext[1];    /* message text */
```

```
  1: #ifndef _LINUX_MSG_H
  2: #define _LINUX_MSG_H
  3: dlfcn
  4: #include <linux/ipc.h>
  5:
  6: /* ipcs ctl commands */
  7: #define MSG_STAT 11
  8: #define MSG_INFO 12
  9:
 10: /* msgrcv options */
 11: #define MSG_NOERROR     010000  /* no error if message is too big */
 12: #define MSG_EXCEPT      020000  /* recv any msg except of specified type.*/
 13:
 14: /* Obsolete, used only for backwards compatibility and libc5 compiles */
 15: struct msqid_ds {
 16:     struct ipc_perm msg_perm;
 17:     struct msg *msg_first;        /* first message on queue,unused  */
 18:     struct msg *msg_last;        /* last message in queue,unused */
 19:     __kernel_time_t msg_stime;    /* last msgsnd time */
 20:     __kernel_time_t msg_rtime;    /* last msgrcv time */
 21:     __kernel_time_t msg_ctime;    /* last change time */
 22:     unsigned long  msg_lcbytes;    /* Reuse junk fields for 32 bit */
 23:     unsigned long  msg_lqbytes;    /* ditto */
 24:     unsigned short msg_cbytes;    /* current number of bytes on queue */
 25:     unsigned short msg_qnum;    /* number of messages in queue */
 26:     unsigned short msg_qbytes;    /* max number of bytes on queue */
 27:     __kernel_ipc_pid_t msg_lspid;    /* pid of last msgsnd */
 28:     __kernel_ipc_pid_t msg_lrpid;    /* last receive pid */
 29: };
 30:
 31: /* Include the definition of msqid64_ds */
 32: #include <asm/msgbuf.h>
 33:
 34: /* message buffer for msgsnd and msgrcv calls */
 35: struct msgbuf {
 36:     long mtype;         /* type of message */
 37:     char mtext[1];      /* message text */
 38: };
 39:
 40: /* buffer for msgctl calls IPC_INFO, MSG_INFO */
 41: struct msginfo {
 42:     int msgpool;
 43:     int msgmap;
 44:     int msgmax;
 45:     int msgmnb;
 46:     int msgmni;
 47:     int msgssz;
 48:     int msgtql;
 49:     unsigned short  msgseg;
 50: };
 51:
 52: #define MSGMNI    16   /* <= IPCMNI */     /* max # of msg queue identifiers */
 53: #define MSGMAX  8192   /* <= INT_MAX */   /* max size of message (bytes) */
 54: #define MSGMNB 16384   /* <= INT_MAX */   /* default max size of a message queue */
 55:
 56: /* unused */
 57: #define MSGPOOL (MSGMNI*MSGMNB/1024)  /* size in kilobytes of message pool */
 58: #define MSGTQL  MSGMNB            /* number of system message headers */
 59: #define MSGMAP  MSGMNB            /* number of entries in message map */
 60: #define MSGSSZ  16                /* message segment size */
 61: #define __MSGSEG ((MSGPOOL*1024)/ MSGSSZ) /* max no. of segments */
 62: #define MSGSEG (__MSGSEG <= 0xffff ? __MSGSEG : 0xffff)
 63:
 64: #ifdef __KERNEL__
 65:
 66: asmlinkage long sys_msgget (key_t key, int msgflg);
 67: asmlinkage long sys_msgsnd (int msqid, struct msgbuf *msgp, size_t msgsz, int msgflg);
 68: asmlinkage long sys_msgrcv (int msqid, struct msgbuf *msgp, size_t msgsz, long msgtyp, int msgf
 69: asmlinkage long sys_msgctl (int msqid, int cmd, struct msqid_ds *buf);
 70:
 71: #endif /* __KERNEL__ */
 72:
 73: #endif /* _LINUX_MSG_H */
```


(For readability, not all matches are indicated.)

```
233: };
234: #endif /* !defined(_XOPEN_SOURCE) */
235:
236:
237: #ifdef _KERNEL
238:
239: extern int    msgconv(const int, struct kmsqid_ds **, const int);
240: extern void    msginit(void);
241:
242: #else /* !_KERNEL */
243:
244: /* Prototypes for application system call interfaces. */
245:
246: #ifdef __STDC__
247:
248: extern int    msgctl(int, int, struct msqid_ds *);
249: extern int    msgget(key_t, int);
250: extern int    msgrcv(int, void *, size_t, long, int);
251: extern int    msgsnd(int, const void *, size_t, int);
252:
253: #else
254:
255: extern int    msgctl();
256: extern int    msgget();
257: extern int    msgrcv();
258: extern int    msgsnd();
259:
260: #endif /* __STDC__ */
261:
262: #endif /* !_KERNEL */
263:
264: #if defined(__cplusplus)
265:     }
266: #endif
267:
268: #endif /* _PROC_IPC_MSG_H */
```


(For readability, not all matches are indicated.)