```
====================== lib/sysutil/getuptime.c ==================
#include "sysutil.h"

/*===========================================================================*
 *                                getuptime                                  *
 *===========================================================================*/
PUBLIC int getuptime(ticks)
clock_t *ticks;                         /* uptime in ticks */
{
    message m;
    int s;

    m.m_type = SYS_TIMES;               /* request time information */
    m.T_PROC_NR = NONE;                 /* ignore process times */
    s = _taskcall(SYSTASK, SYS_TIMES, &m);
    *ticks = m.T_BOOT_TICKS;
    return(s);
}



============= lib/syslib/sys_times.c =============
#include "syslib.h"
#include "syslib.h"

PUBLIC int sys_times(proc, ptr)
int proc;                       /* proc whose times are needed */
clock_t ptr[5];                 /* pointer to time buffer */
{
/* Fetch the accounting info for a proc. */
  message m;
  int r;

  m.T_PROC_NR = proc;
  r = _taskcall(SYSTASK, SYS_TIMES, &m);
  ptr[0] = m.T_USER_TIME;
  ptr[1] = m.T_SYSTEM_TIME;
  ptr[2] = m.T_CHILD_UTIME;
  ptr[3] = m.T_CHILD_STIME;
  ptr[4] = m.T_BOOT_TICKS;
  return(r);
}
```



```
====================== kernel/system/do_times.c =============
/* The kernel call implemented in this file:
 *   m_type:    SYS_TIMES
 *
 * The parameters for this kernel call are:
 *    m4_l1:    T_PROC_NR              (get info for this process)
 *    m4_l1:    T_USER_TIME            (return values ...)
 *    m4_l2:    T_SYSTEM_TIME
 *    m4_l5:    T_BOOT_TICKS
 */

#include "../system.h"

#if USE_TIMES

/*===========================================================================*
 *                                do_times                                   *
 *===========================================================================*/
PUBLIC int do_times(m_ptr)
register message *m_ptr;        /* pointer to request message */
{
/* Handle sys_times().  Retrieve the accounting information. */
  register struct proc *rp;
  int proc_nr;

  /* Insert the times needed by the SYS_TIMES kernel call in the message.
   * The clock's interrupt handler may run to update the user or system time
   * while in this code, but that cannot do any harm.
   */
  proc_nr = (m_ptr->T_PROC_NR == SELF) ? m_ptr->m_source : m_ptr->T_PROC_NR;
  if (isokprocn(proc_nr)) {
      rp = proc_addr(m_ptr->T_PROC_NR);
      m_ptr->T_USER_TIME   = rp->p_user_time;
      m_ptr->T_SYSTEM_TIME = rp->p_sys_time;
  }
  m_ptr->T_BOOT_TICKS = get_uptime();
  return(OK);
}

#endif /* USE_TIMES */
```