```
            if (i % 8 == 7) ipc_to[++j] = ' ';
        }
        ipc_to[j] = '\0';
      printf("%8s %4d   %s   %s  %3d %7lu %7lu   %s\n",
          ip->proc_name, ip->proc_nr,
               s_flags_str(ip->flags), s_traps_str(ip->trap_mask),
        ip->priority, (long)ip->initial_pc, ip->stksize, ipc_to);
  }
  printf("\n");
}

/*===========================================================================*
 *                              recent_dmp                                   *
 *===========================================================================*/
PUBLIC void recent_dmp()
{
  register struct proc *rp;
  static struct proc *oldrp = BEG_PROC_ADDR;
  register struct priv *sp;
  int r, i, n = 0;

  /* First obtain a fresh copy of the current process and system table. */
  if ((r = sys_getprivtab(priv)) != OK) {
        report("IS","warning: couldn't get copy of system privileges table", r);
        return;
  }
  if ((r = sys_getproctab(proc)) != OK) {
        report("IS","warning: couldn't get copy of process table", r);
        return;
  }

  printf("\n-nr---id--name----flags-pri-privaddr-recent\n");

  for (rp = oldrp; rp < END_PROC_ADDR; rp++) {
        if (isemptyp(rp)) continue;
        if (++n > 23) break;
        if (proc_nr(rp) == IDLE)        printf("(%2d) ", proc_nr(rp));
        else if (proc_nr(rp) < 0)       printf("[%2d] ", proc_nr(rp));
        else                            printf(" %2d  ", proc_nr(rp));
        r = -1;
        for (sp = &priv[0]; sp < &priv[NR_SYS_PROCS]; sp++)
                if (sp->s_proc_nr == rp->p_nr) { r ++; break; }
        if (r == -1 && ! (rp->p_rts_flags & SLOT_FREE)) {
                sp = &priv[USER_PRIV_ID];
        }
        printf("(%02u) %-7.7s %s  %2d %4x     %6d",
                sp->s_id, rp->p_name,
                s_flags_str(sp->s_flags),
                rp->p_priority,
                rp->p_priv,
                rp->p_recent_time
                );
        printf("\n");
  }
  if (rp == END_PROC_ADDR) rp = BEG_PROC_ADDR; else printf("--more--\r");
  oldrp = rp;
}

/*===========================================================================*
 *                              sched_dmp                                    *
 *===========================================================================*/
PUBLIC void sched_dmp()
{
  struct proc *rdy_head[NR_SCHED_QUEUES];
  struct kinfo kinfo;
```