```
VAX/VMS PL/M-96 V1.0 COMPILATION OF MODULE PIN
OBJECT MODULE PLACED IN PIN.OBJ
COMPILER INVOKED BY:  PLM86 PIN.PLM OPTIMIZE(3) DEBUG XREF


                $title ('PIN RSP - reads characters from keyboard')
                $set (debug=0)
                $compact
  1             pin:
                do;

                /* PIN performs physical input from for each physical console
                   and places the characters into the input queue associated
                   with each virtual console.  Switch screen commands are
                   received from the XIOS and acted on by PIN.

                   Control S/Q, Control O and Control C are intercepted and
                   processed by PIN.  The input queues are created by VOUT.
                */

                $include (:f1:comlit.lit)

  2   1   =     declare
          =             lit                 literally           'literally',
          =             dcl                 lit                 'declare',
          =             true                lit                 '0ffh',
          =             false               lit                 '0',
          =             no                  lit                 'not',
          =             boolean             lit                 'byte',
          =             forever             lit                 'while true',
          =             cr                  lit                 '13',
          =             lf                  lit                 '10',
          =             tab                 lit                 '9';

  3   1         dcl rsp$link word external;       /* segment of SYSDAT */

                $include (:f1:mfunc.lit)

          =     /* Concurrent CP/M function numbers */

  4   1   =     dcl             m$prtbuf                lit         '9',
          =                     m$select                lit         '14',
          =                     m$openf                 lit         '15',
          =                     m$closef                lit         '16',
          =                     m$deletef               lit         '19',
          =                     m$readf                 lit         '20',
          =                     m$writef                lit         '21',
          =                     m$makef                 lit         '22',
          =                     m$getlogin              lit         '24',
          =                     m$curdsk                lit         '25',
          =                     m$setdma                lit         '26',
          =                     m$setatt                lit         '30',
          =                     m$setusr                lit         '32',
          =                     m$readrf                lit         '33',
          =                     m$writerf               lit         '34',
```

```
        =                   m$resetdrv          lit     '37',
        =                   m$errmode           lit     '45',
        =                   m$dirbios           lit     '50',
        =                   m$makeq             lit     '134',
        =                   m$openq             lit     '135',
        =                   m$deleteq           lit     '136',
        =                   m$readq             lit     '137',
        =                   m$creadq            lit     '138',
        =                   m$writeq            lit     '139',
        =                   m$cwriteq           lit     '140',
        =                   m$delay             lit     '141',
        =                   m$dispatch          lit     '142',
        =                   m$setprior          lit     '145',
        =                   m$detach            lit     '147',
        =                   m$setcns            lit     '148',
        =                   m$parse             lit     '152',
        =                   m$getcns            lit     '153',
        =                   m$sysdat            lit     '154',
        =                   m$getpd             lit     '156',
        =                   m$abort             lit     '157';

        =   /* Internal calls */

5   1   =   dcl             mi$sleep            lit     '0212H',
        =                   mi$wakeup           lit     '0213H';
        =
            $include (:f1:mxfunc.lit)

        =   /* MP/M-86 XIOS function numbers */

6   1   =   dcl     mx$conin            lit '1',
        =           mx$conout           lit '2',
        =           mx$lstout           lit '4',
        =           mx$switch           lit '7',
        =           mx$upstatus         lit '8';

            $include (:f1:sd.lit)

        =   /* System Data Page */

7   1   =     dcl sysdat$pointer pointer;
8   1   =     dcl sysdat$ptr structure(
        =       offset word,
        =       segment word) at (@sysdat$pointer);
9   1   =     declare sd based sysdat$pointer structure (
        =         supmod (4) word,
        =       /*  rtmmod (4) word,
        =         memmod (4) word,
        =         ciomod (4) word,
        =         bdosmod (4) word,
        =         xiosmod (4) word,
        =         netmod (4) word,
        =         reservd (4) word */
        =         spacel(28) word,
        =         mpmseg word,
        =         rspseg word,
        =         endseg word,
```

```
          =         module$map byte,
          =         ncns byte,
          =         nlst byte,
          =         nccb byte,
          =         nflags byte,
          =         srchdisk byte,
          =         mmp word,
          =         nslaves byte,
          =         dayfile byte,
          =         tempdisk byte,
          =         tickspersec byte,
          =         lul word,
          =         ccb word,
          =         flags word,
          =         mdul word,
          =         mfl word,
          =         pul word,
          =         qul word,
          =         qmau (4) word,
          =         rlr word,
          =         dlr word,
          =         drl word,
          =         plr word,
          =         slr word,
          =         thrdrt word,
          =         qlr word,
          =         mal word,
          =         version word,
          =         vernum word,
          =         mpmvernum word,
          =         tod_day word,
          =         tod (3) byte,
          =         ncondev byte,
          =         nlstdev byte,
          =         nciodev byte,
          =         lcb word,
          =         openvec word,
          =         lockmax byte,
          =         openmax byte,
          =         space2 (2) word,
          =         cmod byte );
          =
          =
10   1    =    declare sd$byte based sysdat$pointer (1) byte;
11   1    =    dcl    ncondev lit '83h',
          =           nlstdev lit '84h',
          =           nciodev lit '85h';

               $include (:f1:proces.lit)

          =    /*
          =        Proces Literals MP/M-8086 II
          =    */

12   1    =    declare pnamsiz literally '8';

13   1    =    declare pd$hdr literally 'structure
```

```
        =     (link word,thread word,stat byte,prior byte,flag word,
        =     name (8) byte,uda word,dsk byte,user byte,ldsk byte,luser byte,
        =     mem word";

14   1  =  declare pd$structure literally "pd$hdr,
        =    dvract word,wait word,org byte,net byte,parent word,
        =    cns byte,abort byte,conmode word,lst byte,sf3 byte,sf4 byte,sf5 byte,
        =    reservd (4) byte,pret word,scratch word)";

15   1  =     declare psrun                lit "00",
        =             pspoll               lit "01",
        =             psdelay              lit "02",
        =             psswap               lit "03",
        =             psterm               lit "04",
        =             pssleep              lit "05",
        =             psdq                 lit "06",
        =             psnq                 lit "07",
        =             psflagwait           lit "08",
        =             psciowait            lit "09";

16   1  =     declare pf$sys               lit "00001h",
        =             pf$keep              lit "00002h",
        =             pf$kernal            lit "00004h",
        =             pf$pure              lit "00008h",
        =             pf$table             lit "00010h",
        =             pf$resource          lit "00020h",
        =             pf$raw               lit "00040h",
        =             pf$ctlc              lit "00080h",
        =             pf$active            lit "00100h",
        =             pf$tempkeep          lit "00200h",
        =             pf$ctld              lit "00400h",
        =             pf$childabort        lit "00800h",
        =             pf$noctls            lit "01000h";

17   1  =     declare pcm$ll               lit "00001h",
        =             pcm$ctls             lit "00002h",
        =             pcm$rout             lit "00004h",
        =             pcm$ctlc             lit "00008h",
        =             pcm$ctlo             lit "00080h",
        =             pcm$rsx              lit "00300h";
18   1       declare pd$pointer pointer;
19   1       declare pd$ptr structure (offset word, segment word) at (@pd$pointer);
20   1       declare pd based pd$pointer pd$structure;


21   1       declare ncopies byte external,   /* copy number of this process, corresponds */
                     cnsnum byte at(@ncopies);/* with physical console number */

22   1       declare ctrlC    lit "3";        /* some ASCII codes */
23   1       declare ctrlD    lit "4";
24   1       declare bell     lit "7";
25   1       declare ctrlO    lit "15";
26   1       declare ctrlP    lit "16";
27   1       declare ctrlQ    lit "17";
28   1       declare ctrlS    lit "19";
29   1       declare esc      lit "27";
```

```
        /*      -       global variables        -       */

        $include (:f1:vccb.lit)
=           /* Concurrent CP/M Character Control Block Structure */

=       /*
=              00       |       attach        |        queue        |
=                       +---------+---------+---------+---------+
=              04       |  flag   | startcol| column  |  nchar  |
=                       +---------+---------+---------+---------+
=              08       |  mimic  | msource |   pc    |    vc   |
=                       +---------+---------+---------+---------+
=              0C       |  btmp   | resrvd  |        state      |
=                       +---------+---------+---------+---------+
=              10       |     maxbufsiz     |        vinq       |
=                       +---------+---------+---------+---------+
=              14       |       voutq       |        vcmxq      |
=                       +---------+---------+---------+---------+
=              18       | qpbflgs | qpbfill |      qpbqaddr     |
=                       +---------+---------+---------+---------+
=              1C       |      qpbnmsgs     |     qpbbuffptr    |
=                       +---------+---------+---------+---------+
=              20       |       qbuff       |      cosleep      |
=                       +---------+---------+---------+---------+
=              24       |      usleep       |       vsleep      |
=                       +---------+---------+---------+---------+
=              28       |           ... reserved ...            |
=                       +---------+---------+---------+---------+

=       */

30  1   =   dcl ccb$structure lit 'structure (attach address, queue address,
=           flag byte, startcol byte, column byte, nchar byte, mimic byte, msource byte,
=           ccb$tail1';
31  1   =   dcl ccb$tail1 lit
=           'pc byte, vc byte, btmp byte, reserved byte, state word, maxbufsiz word,
=           ccb$tail2';
32  1   =   dcl ccb$tail2 lit
=           'vinq address, voutq address, vcmxq address,
=           qpbflags byte, qpbresrvd byte, qpbqaddr address,
=           qpbnmsgs address, qpbbuffptr address, qbuff address, cosleep word,
=           usleep word, vsleep word, r1 word, r2 word)';

33  1   =     declare                                 /* flag values                  */
=           cf$listcp       lit     '001h',     /* control P toggle             */
=           cf$compc        lit     '002h',     /* suppress output              */
=           cf$switchs      lit     '004h',     /* XIOS supports switch screening */
=           cf$conout       lit     '008h',     /* XIOS console output ownership */
=           cf$vout         lit     '010h',     /* process writing to VDUTW     */
=           cf$bufp         lit     '020h';     /* toggle to control printer echo */
=                                               /* on control P when background */
=                                               /* and buffered                 */
=       /* values of state byte */
```

```
        =                                   /* conout goes to XIOS */

        =   /* state word flags */

34   1  =   dcl
        =   csm$buffered            lit     '0001h',
        =   csm$background          lit     '0002h',
        =   csm$purging             lit     '0004h',
        =   csm$noswitch            lit     '0008h',
        =   csm$suspend             lit     '0010h',
        =   csm$abort               lit     '0020h',
        =   csm$filefull            lit     '0040h',
        =   csm$ctrlS               lit     '0080h',
        =   csm$ctrlO               lit     '0100h',
        =   csm$ctrlP               lit     '0200h';

35   1  =   dcl x$init$offset lit '0Ch',
        =       x$init$pointer pointer,
        =       x$init$ptr structure (offset word, segment word) at (@x$init$pointer),
        =       x$init based x$init$pointer structure
        =         (tick byte, ticks$sec byte, door byte, resrvd (2) byte,
        =         nvcns byte, nccb byte, nlst byte, ccb word, lcb word);


36   1  =   dcl lcb$structure lit 'structure (attach address, queue address,
        =    flag byte, startcol byte, column byte, nchar byte,
        =    mimic byte, msource byte)';

37   1      declare ccb$pointer pointer;
38   1      declare ccb$ptr structure(offset word,segment word) at (@ccb$pointer);
39   1      declare ccb based ccb$pointer ccb$structure;

40   1      declare old$ccb$pointer pointer;
41   1      declare old$ccb based old$ccb$pointer ccb$structure;

42   1      declare lcb$pointer pointer;
43   1      declare lcb$ptr structure(offset word,segment word) at (@lcb$pointer);
44   1      declare lcb based lcb$pointer lcb$structure;

45   1      declare screen byte;          /* current foreground screen number */

            $include (:f1:qd.lit)

        =   /* Queue Descriptor */

46   1  =   dcl qnamsiz lit '8';

47   1  =   dcl qd$structure lit 'structure(
        =     link  word,
        =     net byte,
        =     org byte,
        =     flags word,
        =     name(qnamsiz) byte,
        =     msglen word,
        =     nmsgs word,
        =     dq word,
        =     nq word,
```

```
        =     msgcnt word,
        =     msgout word,
        =     buffer word)';

        =   /* queue flag values */

48  1   =   dcl qf$mx       lit '001h'; /* Mutual Exclusion */
49  1   =   dcl qf$keep     lit '002h'; /* NO DELETE        */
50  1   =   dcl qf$hide     lit '004h'; /* Not User writable    */
51  1   =   dcl qf$rsp      lit '008h'; /* rsp queue        */
52  1   =   dcl qf$table    lit '010h'; /* from qd table    */
53  1   =   dcl qf$rpl      lit '020h'; /* rpl queue        */
54  1   =   dcl qf$dev      lit '040h'; /* device queue     */

        =   /* Queue Parameter Block */

55  1   =   dcl qpb$structure lit 'structure(
        =     flgs    byte,
        =     net     byte,
        =     qaddr   word,
        =     nmsgs   word,
        =     buffptr word,
        =     name (qnamsiz) byte )';

56  1       declare qpb qpb$structure;

57  1       dcl null word data (0ffffh);          /* sent to VOUTQ to wake up VOUT */
            /* Note: this forces a 2 byte constant section and thus hex generation */

58  1       dcl apb structure (                   /* abort parameter block */
              pd word, term word, cns byte, rsrvd byte) initial (0,0,0,0);

59  1       dcl cword word,                       /* format of console input from XIOS */
                chars (2) byte at (@cword),
                char byte at (@chars(0)),
                char$type byte at(@chars(1)),
                ct$switch lit '0ffh',
                ct$data   lit  '0';

60  1       mon1: procedure(func,a) external;
61  2         dcl func byte, a address;
62  2       end mon1;

63  1       mon2: procedure(func,a) byte external;
64  2         dcl func byte, a address;
65  2       end mon2;

66  1       mon3: procedure(func,a) address external;
67  2         dcl func byte, a address;
68  2       end mon3;

69  1       mon4: procedure(func,a) pointer external;
70  2         dcl func byte, a address;
71  2       end mon4;

72  1       intsys: procedure (cx, dx, bx) external;   /* internal O.S. functions */
73  2         dcl (cx, dx, bx) word;
```

```
 74   2      end intsys;

             /* the following 4 procedures call the XIOS directly, the PXIOS.A86   */
             /* sets the registers to make this legal.  DS = system data segment,  */
             /* ES = UDA.  The parameters are passed AX=FUNC, CX=P1, DX=P2          */

 75   1      pxios1: procedure(func,p1,p2) external;
 76   2      dcl (func,p1,p2) address;
 77   2      end pxios1;

             /*pxios2: procedure(func,p1,p2) byte external;
             dcl (func,p1,p2) address;
             end pxios2;*/

 78   1      pxios3: procedure(func,p1,p2) word external;
 79   2      dcl (func,p1,p2) address;
 80   2      end pxios3;

             /*pxios4: procedure(func,p1,p2) pointer external;
             dcl (func,p1,p2) address;
             end pxios4;*/

 81   1      conin: procedure;
 82   2        cword = pxios3(mx$conin, 0, 0);     /* get console input from XIOS     */
 83   2      end;                                  /* AX=func, CX=0, DX(device#)=0    */

 84   1      print$msg: procedure(len, endchar, sptr);    /* print string to delimiter */
 85   2        dcl (len, i, endchar) byte, sptr pointer,  /* or len number of chars    */
                string based sptr (1) byte;
 86   2        i = 0;
 87   2        do while string(i) <> endchar and i < len;
 88   3          call pxios1(mx$conout, string(i), screen);
 89   3          i = i + 1;
 90   3        end;
 91   2      end print$msg;

             $if debug=1
             error: procedure (msg$ptr);
               dcl msg$ptr pointer;
               call print$msg(0ffh, 0, .(cr, lf, '**** PIN ERROR ****', cr, lf, 0));
               call print$msg(0ffh, '$', msg$ptr);
               halt;
             end error;

             $endif

 92   1      read$change$mxq: procedure (qaddr);
 93   2      dcl qaddr address;
 94   2        qpb.qaddr = qaddr;
 95   2        call mon1 (m$readq, .qpb);
 96   2      end read$change$mxq;

 97   1      write$change$mxq: procedure (qaddr);
 98   2        dcl qaddr address;
 99   2        qpb.qaddr = qaddr;
100   2        call mon1 (m$writeq, .qpb);
101   2      end write$change$mxq;
```

```
102   1     sleep: procedure (list$root);
103   2       dcl list$root word;
104   2       call intsys(mi$sleep, list$root,  ps$ciowait);
105   2     end sleep;

106   1     wake$up: procedure (list$root);
107   2       dcl list$root word;
108   2       call intsys(mi$wakeup, list$root, 0);
109   2     end wake$up;

            /* The conout flag is set and "owned" before any process calls the XIOS
               console output routine for a particular screen.  PIN sets this flag
               to insure there is no process in the XIOS console output code.  The
               ccb.cosleep is a temporary location for processes waiting to own the
               the XIOS conout bit.                                                  */

110   1     set$conout$flag: procedure(ccb$ptr);
111   2       dcl ccb$ptr pointer;
112   2       dcl ccb based ccb$ptr ccb$structure;
113   2       disable;
114   2       do while (ccb.flag and cf$conout) <> 0;     /* Another process is in XIOS */
115   3         call sleep (.ccb.cosleep);                /* PIN gets awakened 1st:     */
116   3       end;                                        /* better priority            */
117   2       ccb.flag = ccb.flag or cf$conout;
118   2       enable;
119   2     end set$conout$flag;

120   1     reset$conout$flag: procedure (ccb$ptr);
121   2       dcl ccb$ptr pointer;
122   2       dcl ccb based ccb$ptr ccb$structure;
123   2       ccb.flag = ccb.flag and not cf$conout;      /* wake sleeping process      */
124   2       call wakeup(.ccb.cosleep);
125   2     end reset$conout$flag;

126   1     wake$vout: procedure(ccb$ptr);
127   2       dcl ccb$ptr pointer;
128   2       dcl ccb based ccb$ptr ccb$structure;
129   2       if (ccb.state and csm$buffered) = 0 then
130   2         return;                         /* dynamic mode */
131   2       qpb.qaddr = ccb.voutq;
132   2       qpb.buffptr = .null;              /* VOUT message is 2 byte format      */
133   2       call mon1(m$cwriteq, .qpb);       /* null message if first byte = 0fffh */
134   2       qpb.qaddr = ccb.voutq;
135   2       qpb.buffptr = .null;              /* VOUT needs two wake-ups in some    */
136   2       call mon1(m$cwriteq, .qpb);       /* situations                         */
137   2       call wake$up(.ccb.vsleep);
138   2     end wake$vout;

139   1     write$vinq: procedure(c);
140   2       dcl c byte;
141   2       qpb.qaddr = ccb.vinq;
142   2       qpb.buffptr = .c;
143   2       if mon1(m$cwriteq, .qpb) = 0ffffh then   /* ring console bell if type */
144   2       do;
145   3         call set$conout$flag(@ccb);  /* XIOS is not reentrant on same console */
146   3         call pxios1(mx$conout, bell, screen);  /* ahead buffer if full      */
```

```
147   3          call reset$conout$flag(@ccb);
148   3        end;
149   2      end write$vinq;

150   1      set$ccb: procedure (vc);        /* base VCCB structure */
151   2        dcl vc byte;
152   2        old$ccb$pointer = ccb$pointer;
153   2        ccb$ptr.offset = sd.ccb + size(ccb) * vc;
154   2      end set$ccb;


               /* the functions below act on special keys received from the keyboard */

155   1      dcl control$$has$been$pressed boolean initial (false);

156   1      switch: procedure;            /* switch virtual consoles */
157   2        if char >= sd.ncns then
158   2          return;                   /* check for legal range */
159   2        if (ccb.state and csm$noswitch) <> 0 then  /* no switch state */
160   2          return;
161   2        if char = ccb.vc then   /* request is for currently selected screen */
162   2          return;

163   2        call set$ccb(screen := char);   /* switch old$ccb and cco structures */

                 /*        -  Switch Out Action  -          */

164   2        call read$change$mxq(oldccb.vcmxq);          /* read the MX 1st THEN */
165   2        call set$conout$flag(@oldccb);               /* the conout flag      */

166   2        oldccb.state = oldccb.state or csm$background;
167   2        if (oldccb.state and csm$purging) <> 0 then
168   2          oldccb.state = oldccb.state and not double(csm$purging);
                 /* turn off purge */

                 /* Ensure the two affected screens are not currently being updated  */
                 /* by the XIOS console output routines.                             */

169   2        call read$change$mxq(ccb.vcmxq);
170   2        call set$conout$flag$(@ccb);

171   2        call pxios1(mx$switch, 0, screen);

172   2        call reset$conout$flag(oldccb$pointer);
173   2        call write$change$mxq(oldccb.vcmxq);  /* allow VOUT to change state     */
174   2        if (oldccb.state and csm$buffered) <> 0 then  /* background buffered */
175   2          call wake$vout(@oldccb);            /* send chars to VOUT if buffer, */
                                                   /* else user process hangs on JSLEEP  */

                 /*        -  Switch In Action  -                 */

176   2        ccb.state = ccb.state and not double(csm$background);
177   2        if (ccb.state and csm$buffered) <> 0 then
178   2        do;                                  /* buffer or buffer error states */
                                                   /* turn on purge */
179   3          ccb.state = (ccb.state or csm$purging) and not double(csm$filefull);
                                                   /* turn off error could print msg */
```

```
180   3          call wake$vout(@ccb);                    /* here eventually */
181   3        end;
182   2        call reset$conout$flag(ccb$pointer);
183   2        call write$change$mxq(ccb.vcmxq);
184   2        if (ccb.state and csm$ctrlS) <> 0 then   /* we "own" the XIOS console     */
185   2          controlS$has$been$pressed = true;      /* output flag                  */
                 else
186   2          controlS$has$been$pressed = false;
187   2        call pxios1(mx$upstatus, 0, 0);
188   2      end switch;

189   1      dcl drive$letters (17) byte initial ('ABCDEFGHIJKLMNOP ');

190   1      controlC: procedure;
191   2        dcl (junk,cur$drive,logged$in$drives) word;
192   2        dcl letter$index byte;
193   2        if (pd.conmode and pcm$ctlc) <> 0 then
194   2        do;
195   3          call write$vinq(char);
196   3          return;
197   3        end;
198   2        call read$change$mxq(ccb.vcmxq); /* keep CCB state from changing        */
                                              /* while we test and change it         */
199   2        ccb.state = ccb.state and not double(csm$ctrlS or csm$ctrlO);
200   2        controlS$has$been$pressed = false;
                                              /* control C turns off control S and   */
                                              /* control O, doesn't change control P */
201   2        if (ccb.state and csm$purging) <> 0 then /* stop purge                 */
202   2          ccb.state = ccb.state and not double (csm$purging) or csm$abort;
203   2        qpb.qaddr = ccb.vinq;                 /* drain input queue, we have better */
204   2        qpb.buffptr = .junk;                  /* priority than user process or TMP */
205   2        do while mon2(m$creadq, .qpb) <> 0ffh;  /* drain type-ahead q */
206   3        end;
207   2        ccb.nchar = 0;                        /* zero console status look ahead    */
208   2        call write$change$mxq(ccb.vcmxq);
209   2        call wake$vout(@ccb);              /* let VOUT clean up if buffering       */
210   2        call wake$up(.ccb.usleep);         /* user process could have gone to sleep */
                                                 /* during this rigamarole               */

211   2        apb.pd = ccb.attach;               /* CTD keeps aborts from happening      */
212   2        call mon1(m$abort, .apb);          /* while in the XIOS, do abort after    */
                                                 /* VOUT has cleaned up, otherwise the TMP*/
                                                 /* with a better priority can print its  */
                                                 /* prompt, and then VOUT prints one last */
                                                 /* purge character                      */

                       /* reset drives and print which fail */
213   2        logged$in$drives = mon3(m$getlogin,0);
214   2        cur$drive = 1;                     /* drive to reset                       */
215   2        letter$index = 0;
216   2        do junk = 0 to 15;
217   3          if (logged$in$drives and cur$drive) <> 0 then
218   3            if mon2(m$resetdrv, cur$drive) <> 0 then
219   3            do;
220   4              drive$letters(letter$index) = 'A' + junk;
221   4              letter$index = letter$index + 1;
222   4            end;
```

```
223   3          cur$drive = shl(cur$drive,1);
224   3        end;
225   2        if letter$index > 0 then
226   2        do;
227   3          call set$conout$flag(@ccb);
228   3          call print$msg(0ffh,0,@(cr,lf,'Open file on drive(s) ',0));
229   3          call print$msg(1,0,@drive$letters(0));
230   3          do junk = 1 to letter$index - 1;
231   4            call print$msg(1,0,@(',',));
232   4            call print$msg(1,0,@drive$letters(junk));
233   4          end;
234   3          call print$msg(2,0,@(cr,lf));
235   3          ccb.startcol, ccb.column = 0;        /* for function 10 - line redraw */
236   3          call reset$conout$flag(@ccb);
237   3        end;
238   2      end control C;

239   1      control0: procedure;              /* toggle console output byte bucket     */
240   2        if (pd.conmode and pcm$ctl0) <> 0 then  /* ignore if control P or if func */
241   2        do;
242   3          call write$vinq(char);
243   3          return;
244   3        end;
245   2        call read$change$mxq(ccb.vcmxq);
246   2        ccb.state = ccb.state xor csm$ctrl0;
247   2        call write$change$mxq(ccb.vcmxq);
248   2        call wake$vout(@ccb);
249   2        call pxios1(mx$upstatus, 0, 0);
250   2      end control0;

251   1      turn$off$ctrl0: procedure;
252   2        call read$change$mxq(ccb.vcmxq);
253   2        ccb.state = ccb.state and not double(csm$ctrl0);
254   2        call write$change$mxq(ccb.vcmxq);
255   2        call wake$vout(@ccb);
256   2        call pxios1(mx$upstatus, 0, 0);
257   2      end turn$off$ctrl0;

258   1      controlS: procedure;
259   2        if (pd.flag and pf$noctls) <> 0 then  /* special condition for CLI day */
260   2          return;                             /* file logging */
261   2        if (pd.conmode and pcm$ctlS) <> 0 then
262   2        do;
263   3          call write$vinq(char);
264   3          return;
265   3        end;
266   2        call read$change$mxq(ccb.vcmxq);
267   2        ccb.state = ccb.state and not double(csm$ctrl0) or csm$ctrlS;
                                                    /* control S turns off control 0 */
268   2        call pxios1(mx$upstatus, 0, 0);
269   2        call write$change$mxq(ccb.vcmxq);
270   2        controlS$has$been$pressed = true;
271   2      end controlS;

272   1      controlQ: procedure;
273   2        call read$change$mxq(ccb.vcmxq);
274   2        ccb.state = ccb.state and not double(csm$ctrlS);
```

```
275   2       call write$change$mxq(ccb.vcmxq);
276   2       call wake$vout(@ccb);
277   2       call wakeup(.ccb.usleep);
278   2       call pxios1(mx$upstatus, 0, 0);
279   2       control$has$been$pressed = false;
280   2     end controlQ;

281   1     controlP: procedure;
282   2       if (pd.conmode and pcm$rout) <> 0 then /* control P is ignored if console */
283   2       do;
284   3         call write$vinq(char);
285   3         return;                          /* mode is raw output */
286   3       end;
287   2       call turn$off$ctrlO;
288   2       call read$change$mxq(ccb.vcmxq);
289   2       if (ccb.state and csm$ctrlP) = 0 then  /* turn control P on */
290   2       do;
291   3         lcb$ptr.offset = sd.lcb + pd.lst * size(lcb);
292   3         disable;
293   3         if lcb.attach = 0 then
294   3         do;
295   4           lcb.attach = 0ffffh;
296   4           lcb.msource = screen;
297   4           ccb.mimic = pd.lst;
298   4           ccb.state = ccb.state or csm$ctrlP;
299   4           enable;
300   4         end;
                  else
301   3         do;
302   4           enable;
303   4           call set$conout$flag(@ccb);
304   4           call print$msg(0ffh,0,a(cr,lf,'Printer Busy',cr,lf,0));
305   4           ccb.column,ccb.startcol = 0;          /* for function 10 */
306   4           call reset$conout$flag(@ccb);
307   4         end;
308   3       end;
              else                                  /* turn off control P */
309   2       do;
310   3         disable;
311   3         lcb$ptr.offset = sd.lcb + ccb.mimic * size(lcb);
312   3         lcb.attach = 0;
313   3         lcb.msource,ccb.mimic = 0ffh;
314   3         ccb.state = ccb.state and not double(csm$ctrlP);
315   3         ccb.flag = ccb.flag and not cf$bufp;
316   3         call wakeup(.lcb.queue);
317   3         enable;
318   3       end;
319   2       call write$change$mxq(ccb.vcmxq);
320   2       call pxios1(mx$upstatus, 0, 0);
321   2     end controlP;

322   1     raw: procedure boolean;
323   2       if (pd$ptr.offset := ccb.attach) = 0 then  /* 0 during initialization */
324   2          return(true);
325   2       if (pd.flag and pf$raw) = 0 then          /* set by function 6 only  */
326   2         return(false);
327   2       call controlQ;            /* avoid deadlock if user is mixing func 6  */
```

```
328   2       return(true);             /* other console I/O calls                */
329   2     end raw;

330   1     plmstart: procedure public;

331   2       sysdat$ptr.segment, lcb$ptr.segment, ccb$ptr.segment = rsp$link;
                                        /* init pointers */
332   2       ccb$ptr.offset = sd.ccb;  /* CCB 0 is first in the table */
333   2       pd$pointer = mon4(m$getpd, 0);
334   2       screen = 0;               /* initial foreground console is 0 */

335   2       do forever;
336   3         call conin;
337   3         if char$type = ct$switch then
338   3           call switch;
339   3         else if char$type = ct$data then
340   3         do;
341   4           if raw then
342   4             call write$ving(char);
                  else
343   4           do;
344   5             if control$has$been$pressed then
345   5             do;
346   6               if char = ctrlC then
347   6                 call controlC;
348   6               else if char = ctrlQ then
349   6                 call controlQ;
350   6               else if char = ctrlP then
351   6                 call controlP;
                        else
352   6                 do;
353   7                   call set$conout$flag(ccb$pointer);  /* guard against unlikely */
354   7                   call pxios1(mx$conout,bell,screen); /* race condition  */
355   7                   call reset$conout$flag(ccb$pointer);
356   7                 end;
357   6             end;
                    else
358   5             do; /* control$ has not been pressed */
359   6               if char = ctrlC then
360   6                 call controlC;
361   6               else if char = ctrlS then
362   6                 call controlS;
363   6               else if char = ctrlO then
364   6                 call controlO;
365   6               else if char = ctrlP then
366   6                 call controlP;
                        else
367   6                 do;
368   7                   if (ccb.state and csm$ctrlO) <> 0 then
369   7                     call turn$off$ctrlO;
370   7                   call write$ving(char);
371   7                 end;
372   6             end;
373   5           end;/* else (if not raw) */
374   4         end;  /* if char$type <> ct$data and char$type <> ct$switch then */
                    /* XIOS console input is ignored                            */
              end;    /* do forever */
```

```
376   2       end plmstart;
377   1       end pin;
```

CROSS-REFERENCE LISTING

DEFN	ADDR	SIZE	NAME, ATTRIBUTES, AND REFERENCES											
63	0000H	2	A.	WORD PARAMETER	64									
60	0000H	2	A.	WORD PARAMETER	61									
69	0000H	2	A.	WORD PARAMETER	70									
66	0000H	2	A.	WORD PARAMETER	67									
20	0021H	1	ABORT.	BYTE MEMBER(PD)										
58	0028H	6	APB.	STRUCTURE INITIAL		211	212							
112	0000H	2	ATTACH	WORD MEMBER(CCB)										
122	0000H	2	ATTACH	WORD MEMBER(CCB)										
128	0000H	2	ATTACH	WORD MEMBER(CCB)										
44	0000H	2	ATTACH	WORD MEMBER(LCB)		293	295	312						
39	0000H	2	ATTACH	WORD MEMBER(CCB)		211	323							
41	0000H	2	ATTACH	WORD MEMBER(OLDCCB)										
24			BELL	LITERALLY	146	354								
2			BOOLEAN.	LITERALLY	155	322								
122	000CH	1	BTMP	BYTE MEMBER(CCB)										
39	000CH	1	BTMP	BYTE MEMBER(CCB)										
112	000CH	1	BTMP	BYTE MEMBER(CCB)										
128	000CH	1	BTMP	BYTE MEMBER(CCB)										
41	000CH	1	BTMP	BYTE MEMBER(OLDCCB)										
56	0006H	2	BUFFPTR.	WORD MEMBER(UPB)		132	135	142	204					
72	0000H	2	BX	WORD PARAMETER	73									
139	0004H	1	C.	BYTE PARAMETER AUTOMATIC			140	142						
122	0000H	44	CCB.	STRUCTURE BASED(CCBPTR)			123	124						
39	0000H	44	CCB.	STRUCTURE BASED(CCBPOINTER)				141	145	147	153	159	161	
				169	170	176	177	179	180	183	184	198	199	201	202	203	
				207	208	209	210	211	227	235	236	245	246	247	248	252	
				253	254	255	266	267	269	273	274	275	276	277	288	289	
				297	298	303	305	306	311	313	314	315	319	323	368		
9	0004H	2	CCB.	WORD MEMBER(SD)		153	332							
128	0000H	44	CCB.	STRUCTURE BASED(CCBPTR)			129	131	134	137				
112	0000H	44	CCB.	STRUCTURE BASED(CCBPTR)			114	115	117					
35	0008H	2	CCB.	WORD MEMBER(XINIT)										
37	000CH	4	CCBPOINTER	POINTER	38	39	141	145	147	152	159	161	169	170
				176	177	179	180	182	183	184	198	199	201	202	203	207	
				208	209	210	211	227	235	236	245	246	247	248	252	253	
				254	255	266	267	269	273	274	275	276	277	288	289	297	
				298	303	305	306	311	313	314	315	319	323	353	355	368	
126	0004H	4	CCBPTR	POINTER PARAMETER AUTOMATIC				127	128	129	131	134	137	
110	0004H	4	CCBPTR	POINTER PARAMETER AUTOMATIC				111	112	114	115	117		
38	000CH	4	CCBPTR	STRUCTURE AT	153	331	332							
120	0004H	4	CCBPTR	POINTER PARAMETER AUTOMATIC				121	122	123	124			
30			CCBSTRUCTURE	LITERALLY	39	41	112	122	128					
31			CCBTAIL1	LITERALLY	39	41	112	122	128					
32			CCBTAIL2	LITERALLY	39	41	112	122	128					
33			CFBUFP	LITERALLY	315									
33			CFCOMPC.	LITERALLY										
33			CFCONOUT	LITERALLY	114	117	123							
33			CFLISTCP	LITERALLY										
33			CFSWITCHS.	LITERALLY										

```
  33                  CPVOUT . . . . . .   LITERALLY
  59  002EH      1    CHAR . . . . . .     BYTE AT          157  161  163  195  242  263  284  342  346  348
                                             350  359  361  363  365  370
  59  002EH      2    CHARS. . . . . .     BYTE ARRAY(2) AT              59
  59  002FH      1    CHARTYPE . . . . .   BYTE AT          337  339
   9  0090H      1    CMOD . . . . . .     BYTE MEMBER(SD)
  20  0020H      1    CNS. . . . . .       BYTE MEMBER(PD)
  58  0004H      1    CNS. . . . . .       BYTE MEMBER(APB)
  21  0000H      1    CNSNUM . . . . . .   BYTE EXTERNAL(1) AT
  39  0006H      1    COLUMN . . . . . .   BYTE MEMBER(CCB)          235  305
 112  0006H      1    COLUMN . . . . . .   BYTE MEMBER(CCB)
 122  0006H      1    COLUMN . . . . . .   BYTE MEMBER(CCB)
  44  0006H      1    COLUMN . . . . . .   BYTE MEMBER(LCB)
  41  0006H      1    COLUMN . . . . . .   BYTE MEMBER(OLDCCB)
 128  0006H      1    COLUMN . . . . . .   BYTE MEMBER(CCB)
  81  0000H     20    CONIN. . . . . .     PROCEDURE STACK=000AH          336
  20  0022H      2    CONMODE. . . . . .   WORD MEMBER(PD)       193  240  251  282
 190  02E1H    407    CONTROLC . . . . .   PROCEDURE STACK=001EH          347  360
 239  0478H     79    CONTROLO . . . . .   PROCEDURE STACK=001EH          364
 281  05ACH    277    CONTROLP . . . . .   PROCEDURE STACK=001EH          351  366
 272  0566H     70    CONTROLQ . . . . .   PROCEDURE STACK=0018H          327  349
 258  04FDH    105    CONTROLS . . . . .   PROCEDURE STACK=001EH          362
 155  0038H      1    CONTROLSHASBEENPRESSED  BYTE INITIAL       185  186  200  270  279  344
 128  0022H      2    COSLEEP. . . . . .   WORD MEMBER(CCB)
  39  0022H      2    COSLEEP. . . . . .   WORD MEMBER(CCB)
  41  0022H      2    COSLEEP. . . . . .   WORD MEMBER(OLDCCB)
 112  0022H      2    COSLEEP. . . . . .   WORD MEMBER(CCB)          115
 122  0022H      2    COSLEEP. . . . . .   WORD MEMBER(CCB)          124
   2                  CR . . . . . .       LITERALLY        228  234  304
  34                  CSMABORT . . . . .   LITERALLY        202
  34                  CSMBACKGROUND. . . . LITERALLY        166  176
  34                  CSMBUFFERED. . . . . LITERALLY        129  174  177
  34                  CSMCTRLO . . . . .   LITERALLY        199  246  253  257  368
  34                  CSMCTRLP . . . . .   LITERALLY        289  293  314
  34                  CSMCTRLS . . . . .   LITERALLY        184  199  267  274
  34                  CSMFILEFULL. . . . . LITERALLY        179
  34                  CSMNOSWITCH. . . . . LITERALLY        159
  34                  CSMPURGING . . . . . LITERALLY        167  168  179  201  202
  34                  CSMSUSPEND . . . . . LITERALLY
  59                  CTDATA . . . . . .   LITERALLY        339
  22                  CTRLC. . . . . .     LITERALLY        346  359
  23                  CTRLO. . . . . .     LITERALLY
  25                  CTRLO. . . . . .     LITERALLY        363
  26                  CTRLP. . . . . .     LITERALLY        350  365
  27                  CTRLQ. . . . . .     LITERALLY        348
  28                  CTRLS. . . . . .     LITERALLY        361
  59                  CTSWITCH . . . . .   LITERALLY        337
 191  0032H      2    CURDRIVE . . . . .   WORD         214  217  218  223
  59  002EH      2    CWORD. . . . . .     WORD          59   82
  72  0000H      2    CX . . . . . .       WORD PARAMETER          73
   9  004FH      1    DAYFILE. . . . . .   BYTE MEMBER(SD)
   2                  DCL. . . . . .       LITERALLY
   9  006AH      2    DLR. . . . . .       WORD MEMBER(SD)
  35  0002H      1    DOOR . . . . . .     BYTE MEMBER(XINIT)
                      DOUBLE . . . . . .   BUILTIN          168  176  179  199  202  253  267  274  314
 189  0039H     17    DRIVELETTERS . . . . BYTE ARRAY(17) INITIAL          220  229  232
   9  006CH      2    DRL. . . . . .       WORD MEMBER(SD)
```

```
  20  0012H     1  DSK. . . . . .   BYTE MEMBER(PD)
  20  0018H     2  DVRACT . . . . .  WORD MEMBER(PD)
  72  0000H     2  DX . . . . . .   WORD PARAMETER          73
  84  0008H     1  ENDCHAR. . . . .  BYTE PARAMETER AUTOMATIC          85   87
   9  0044H     2  ENDSEG . . . . .  WORD MEMBER(SD)
  29            ESC. . . . . .   LITERALLY
   2            FALSE. . . . . .  LITERALLY        155  186  209  279  326
  44  0004H     1  FLAG . . . . . .  BYTE MEMBER(LCB)
  39  0004H     1  FLAG . . . . . .  BYTE MEMBER(CCB)          315
 112  0004H     1  FLAG . . . . . .  BYTE MEMBER(CCB)          114  117
 122  0004H     1  FLAG . . . . . .  BYTE MEMBER(CCB)          123
  41  0004H     1  FLAG . . . . . .  BYTE MEMBER(OLDCCB)
  20  0006H     2  FLAG . . . . . .  WORD MEMBER(PD)       253  325
 128  0004H     1  FLAG . . . . . .  BYTE MEMBER(CCB)
   9  0056H     2  FLAGS. . . . . .  WORD MEMBER(SD)
  56  0000H     1  FLGS . . . . . .  BYTE MEMBER(QPB)
   2            FOREVER. . . . . .  LITERALLY        335
  78  0000H     2  FUNC . . . . . .  WORD PARAMETER          79
  66  0000H     1  FUNC . . . . . .  BYTE PARAMETER          67
  63  0000H     1  FUNC . . . . . .  BYTE PARAMETER          64
  60  0000H     1  FUNC . . . . . .  BYTE PARAMETER          61
  75  0000H     2  FUNC . . . . . .  WORD PARAMETER          75
  69  0000H     1  FUNC . . . . . .  BYTE PARAMETER          70
  85  0037H     1  I. . . . . .     BYTE         86   87   88   89
  72  0000H        INTSYS . . . . . .  PROCEDURE EXTERNAL(6) STACK=0000H          104  108
 191  0030H     2  JUNK . . . . . .  WORD         204  216  220  230  232
  44  0000H    10  LCB. . . . . .   STRUCTURE BASED(LCBPOINTER)          291  293  295  296  311  312
                                      313  316
   9  0086H     2  LCB. . . . . .   WORD MEMBER(SD)       291  311
  35  000AH     2  LCB. . . . . .   WORD MEMBER(XINIT)
  42  0014H     4  LCBPOINTER . . . . .  POINTER            43   44  293  295  296  312  313  316
  43  0014H     4  LCBPTR . . . . . .  STRUCTURE AT         291  311  331
  36            LCBSTRUCTURE . . . . .  LITERALLY          44
  20  0014H     1  LDSK . . . . . .  BYTE MEMBER(PD)
  84  000AH     1  LEN. . . . . .   BYTE PARAMETER AUTOMATIC          85   87
 192  004AH     1  LETTERINDEX. . . . . .  BYTE         215  220  221  225  230
   2            LF . . . . . .   LITERALLY        228  234  304
  20  0000H     2  LINK . . . . . .  WORD MEMBER(PD)
 106  0004H     2  LISTROOT . . . . . .  WORD PARAMETER AUTOMATIC          107  108
 102  0004H     2  LISTROOT . . . . . .  WORD PARAMETER AUTOMATIC          103  104
   2            LIT. . . . . .   LITERALLY          2    4    5    6   11   15   16   17   22   23
                                      24   25   26   27   28   29   30   31   32   33   34   35   36
                                      46   47   48   49   50   51   52   53   54   55   59
   9  008AH     1  LOCKMAX. . . . . .  BYTE MEMBER(SD)
 191  0034H     2  LOGGEDINDRIVES . . . .  WORD         213  217
  20  0024H     1  LST. . . . . .   BYTE MEMBER(PD)       291  297
   9  0052H     2  LUL. . . . . .   WORD MEMBER(SD)
  20  0015H     1  LUSER. . . . . .  BYTE MEMBER(PD)
   4            MABORT . . . . . .  LITERALLY        212
   9  0076H     2  MAL. . . . . .   WORD MEMBER(SD)
 128  0010H     2  MAXBUFSIZ. . . . . .  WORD MEMBER(CCB)
 112  0010H     2  MAXBUFSIZ. . . . . .  WORD MEMBER(CCB)
  39  0010H     2  MAXBUFSIZ. . . . . .  WORD MEMBER(CCB)
 122  0010H     2  MAXBUFSIZ. . . . . .  WORD MEMBER(CCB)
  41  0010H     2  MAXBUFSIZ. . . . . .  WORD MEMBER(OLDCCB)
   4            MCLOSEF. . . . . .  LITERALLY
   4            MCREAD4. . . . . .  LITERALLY        205
```

```
     4                  MCURDSK. . . . . .   LITERALLY
     4                  MCWRITEQ . . . . . .   LITERALLY        133  136  143
     4                  MDELAY . . . . . .   LITERALLY
     4                  MDELETEF . . . . . .   LITERALLY
     4                  MDELETEQ . . . . . .   LITERALLY
     4                  MDETACH. . . . . .   LITERALLY
     4                  MDIRBIOS . . . . . .   LITERALLY
     4                  MDISPATCH. . . . . .   LITERALLY
     9  0058H      2    MDUL . . . . . .   WORD MEMBER(SD)
    20  0016H      2    MEM. . . . . .   WORD MEMBER(PD)
     4                  MERRMODE . . . . . .   LITERALLY
     9  005AH      2    MFL. . . . . .   WORD MEMBER(SD)
     4                  MGETCNS. . . . . .   LITERALLY
     4                  MGETLOGIN. . . . . .   LITERALLY        213
     4                  MGETPD . . . . . .   LITERALLY        333
   122  0008H      1    MIMIC. . . . . .   BYTE MEMBER(CCB)
    41  0008H      1    MIMIC. . . . . .   BYTE MEMBER(OLDCCB)
    44  0008H      1    MIMIC. . . . . .   BYTE MEMBER(LCB)
    39  0008H      1    MIMIC. . . . . .   BYTE MEMBER(CCB)          297  311  313
   112  0008H      1    MIMIC. . . . . .   BYTE MEMBER(CCB)
   128  0008H      1    MIMIC. . . . . .   BYTE MEMBER(CCB)
     5                  MISLEEP. . . . . .   LITERALLY        104
     5                  MIWAKEUP . . . . . .   LITERALLY        108
     4                  MMAKEF . . . . . .   LITERALLY
     4                  MMAKEQ . . . . . .   LITERALLY
     9  004CH      2    MMP. . . . . .   WORD MEMBER(SD)
     9  0046H      1    MODULEMAP. . . . . .   BYTE MEMBER(SD)
    60  0000H           MON1 . . . . . .   PROCEDURE EXTERNAL(2) STACK=0000H          95  100  133  136  212
    63  0000H           MON2 . . . . . .   PROCEDURE BYTE EXTERNAL(3) STACK=0000H        205  218
    66  0000H           MON3 . . . . . .   PROCEDURE WORD EXTERNAL(4) STACK=0000H        143  213
    69  0000H           MON4 . . . . . .   PROCEDURE POINTER EXTERNAL(5) STACK=0000H          333
     4                  MOPENF . . . . . .   LITERALLY
     4                  MOPENQ . . . . . .   LITERALLY
     4                  MPARSE . . . . . .   LITERALLY
     9  0040H      2    MPMSEG . . . . . .   WORD MEMBER(SD)
     9  007CH      2    MPMVERNUM. . . . . .   WORD MEMBER(SD)
     4                  MPRTBUF. . . . . .   LITERALLY
     4                  MREADF . . . . . .   LITERALLY
     4                  MREADQ . . . . . .   LITERALLY         95
     4                  MREADRF. . . . . .   LITERALLY
     4                  MRESETDRV. . . . . .   LITERALLY        218
     4                  MSELECT. . . . . .   LITERALLY
     4                  MSETATT. . . . . .   LITERALLY
     4                  MSETCNS. . . . . .   LITERALLY
     4                  MSETDMA. . . . . .   LITERALLY
     4                  MSETPRIOR. . . . . .   LITERALLY
     4                  MSETUSR. . . . . .   LITERALLY
   128  0009H      1    MSOURCE. . . . . .   BYTE MEMBER(CCB)
   112  0009H      1    MSOURCE. . . . . .   BYTE MEMBER(CCB)
    41  0009H      1    MSOURCE. . . . . .   BYTE MEMBER(OLDCCB)
    44  0009H      1    MSOURCE. . . . . .   BYTE MEMBER(LCB)          296  313
    39  0009H      1    MSOURCE. . . . . .   BYTE MEMBER(CCB)
   122  0009H      1    MSOURCE. . . . . .   BYTE MEMBER(CCB)
     4                  MSYSDAT. . . . . .   LITERALLY
     4                  MWRITEF. . . . . .   LITERALLY
     4                  MWRITEQ. . . . . .   LITERALLY        100
     4                  MWRTTERF . . . . . .   LITERALLY
```

```
     6                  MXCONIN. . . . . .    LITERALLY         32
     6                  MXCONOUT . . . . . .  LITERALLY         88  146  354
     6                  MXLSTOUT . . . . . .  LITERALLY
     6                  MXSWITCH . . . . . .  LITERALLY        171
     6                  MXUPSTATUS . . . . .  LITERALLY        187  249  256  268  278  320
    56  0008H      8    NAME . . . . . .      BYTE ARRAY(8) MEMBER(QPB)
    20  0008H      8    NAME . . . . . .      BYTE ARRAY(8) MEMBER(PD)
    35  0006H      1    NCCB . . . . . .      BYTE MEMBER(XINIT)
     9  0049H      1    NCCB . . . . . .      BYTE MEMBER(SD)
   128  0007H      1    NCHAR. . . . . .      BYTE MEMBER(CCB)
   122  0007H      1    NCHAR. . . . . .      BYTE MEMBER(CCB)
   112  0007H      1    NCHAR. . . . . .      BYTE MEMBER(CCB)
    44  0007H      1    NCHAR. . . . . .      BYTE MEMBER(LCB)
    41  0007H      1    NCHAR. . . . . .      BYTE MEMBER(OLDCCB)
    39  0007H      1    NCHAR. . . . . .      BYTE MEMBER(CCB)          207
    11                  NCIODEV. . . . . .    LITERALLY
     9  0085H      1    NCIODEV. . . . . .    BYTE MEMBER(SD)
     9  0047H      1    NCNS . . . . . .      BYTE MEMBER(SD)       157
    11                  NCONDEV. . . . . .    LITERALLY
     9  0083H      1    NCONDEV. . . . . .    BYTE MEMBER(SD)
    21  0000H      1    NCOPIES. . . . . .    BYTE EXTERNAL(1)           21
    20  001DH      1    NET. . . . . .        BYTE MEMBER(PD)
    56  0001H      1    NET. . . . . .        BYTE MEMBER(QPB)
     9  004AH      1    NFLAGS . . . . . .    BYTE MEMBER(SD)
    35  0007H      1    NLST . . . . . .      BYTE MEMBER(XINIT)
     9  0048H      1    NLST . . . . . .      BYTE MEMBER(SD)
    11                  NLSTDEV. . . . . .    LITERALLY
     9  0084H      1    NLSTDEV. . . . . .    BYTE MEMBER(SD)
    56  0004H      2    NMSGS. . . . . .      WORD MEMBER(QPB)
     2                  NO . . . . . .        LITERALLY
     9  004EH      1    NSLAVES. . . . . .    BYTE MEMBER(SD)
    57  0000H      2    NULL . . . . . .      WORD DATA        132  135
    35  0005H      1    NVCNS. . . . . .      BYTE MEMBER(XINIT)
     8  0000H      2    OFFSET . . . . . .    WORD MEMBER(SYSDATPTR)
    19  0000H      2    OFFSET . . . . . .    WORD MEMBER(PDPTR)        323
    43  0000H      2    OFFSET . . . . . .    WORD MEMBER(LCBPTR)       291  311
    38  0000H      2    OFFSET . . . . . .    WORD MEMBER(CCBPTR)       153  332
    35  0000H      2    OFFSET . . . . . .    WORD MEMBER(XINITPTR)
    41  0000H     44    OLDCCB . . . . . .    STRUCTURE BASED(OLDCCBPOINTER)     164  165  166  167  168  173
                                               174  175
    40  0010H      4    OLDCCBPOINTER. . . .  POINTER          41  152  164  165  166  167  168  172  173  174
                                               175
     9  008BH      1    OPENMAX. . . . . .    BYTE MEMBER(SD)
     9  0088H      2    OPENVEC. . . . . .    WORD MEMBER(SD)
    20  001CH      1    ORG. . . . . .        BYTE MEMBER(PD)
    78  0000H      2    P1 . . . . . .        WORD PARAMETER        79
    75  0000H      2    P1 . . . . . .        WORD PARAMETER        76
    78  0000H      2    P2 . . . . . .        WORD PARAMETER        79
    75  0000H      2    P2 . . . . . .        WORD PARAMETER        76
    20  001EH      2    PARENT . . . . . .    WORD MEMBER(PD)
   128  000AH      1    PC . . . . . .        BYTE MEMBER(CCB)
   112  000AH      1    PC . . . . . .        BYTE MEMBER(CCB)
   122  000AH      1    PC . . . . . .        BYTE MEMBER(CCB)
    41  000AH      1    PC . . . . . .        BYTE MEMBER(OLDCCB)
    39  000AH      1    PC . . . . . .        BYTE MEMBER(CCB)
    17                  PCM11. . . . . .      LITERALLY
    17                  PCMCTLC. . . . . .    LITERALLY        193
```

```
 17                    PCMCTLO. . . . . .  LITERALLY       240
 17                    PCMCTLS. . . . . .  LITERALLY       241
 17                    PCMROUT. . . . . .  LITERALLY       282
 17                    PCMRSX . . . . . .  LITERALLY
 20  0000H     48      PD . . . . . .  STRUCTURE BASED(PDPOINTER)          193  240  259  261  282  291
                                        297  325
 58  0000H      2      PD . . . . . .  WORD MEMBER(APS)          211
 13                    PDHOR. . . . . .  LITERALLY       20
 18  0004H      4      PDPOINTER. . . . . .  POINTER          19   20  193  240  259  261  282  291  297  325
                                        333
 19  0004H      4      PDPTR. . . . . .  STRUCTURE AT          323
 14                    PDSTRUCTURE. . . . . .  LITERALLY       20
 16                    PFACTIVE . . . . . .  LITERALLY
 16                    PFCHILDABORT . . . . . .  LITERALLY
 16                    PFCTLC . . . . . .  LITERALLY
 16                    PFCTLD . . . . . .  LITERALLY
 16                    PFKEEP . . . . . .  LITERALLY
 16                    PFKERNAL . . . . . .  LITERALLY
 16                    PFNOCTLS . . . . . .  LITERALLY       259
 16                    PFPURE . . . . . .  LITERALLY
 16                    PFRAW. . . . . .  LITERALLY       325
 16                    PFRESOURCE . . . . . .  LITERALLY
 16                    PFSYS. . . . . .  LITERALLY
 16                    PFTABLE. . . . . .  LITERALLY
 16                    PFTEMPKEEP . . . . . .  LITERALLY
  1  0000H              PIN. . . . . .  PROCEDURE STACK=0000H
330  06EEH    229      PLMSTART . . . . . .  PROCEDURE PUBLIC STACK=0022H
  9  006EH      2      PLR. . . . . .  WORD MEMBER(SD)
 12                    PNAMSIZ. . . . . .  LITERALLY
 20  002CH      2      PRET . . . . . .  WORD MEMBER(PD)
 84  0014H     76      PRINTMSG . . . . . .  PROCEDURE STACK=0012H          228  229  231  232  234  304
 20  0005H      1      PRIOR. . . . . .  BYTE MEMBER(PD)
 15                    PSCIOWAIT. . . . . .  LITERALLY       104
 15                    PSDELAY. . . . . .  LITERALLY
 15                    PSDQ . . . . . .  LITERALLY
 15                    PSFLAGWAIT . . . . . .  LITERALLY
 15                    PSNQ . . . . . .  LITERALLY
 15                    PSPOLL . . . . . .  LITERALLY
 15                    PSRUN. . . . . .  LITERALLY
 15                    PSSLEEP. . . . . .  LITERALLY
 15                    PSSWAP . . . . . .  LITERALLY
 15                    PSTERM . . . . . .  LITERALLY
  9  005CH      2      PUL. . . . . .  WORD MEMBER(SD)
 75  0000H             PXIOS1 . . . . . .  PROCEDURE EXTERNAL(7) STACK=0006H          88  146  171  187  249
                                        256  268  278  320  354
 78  0000H             PXIOS3 . . . . . .  PROCEDURE WORD EXTERNAL(8) STACK=0000H          82
 97  0004H      2      QADDR. . . . . .  WORD PARAMETER AUTOMATIC          98   99
 92  0004H      2      QADDR. . . . . .  WORD PARAMETER AUTOMATIC          93   94
 56  0002H      2      QADDR. . . . . .  WORD MEMBER(QPB)          94   99  131  134  141  203
122  0020H      2      QBUFF. . . . . .  WORD MEMBER(CCB)
112  0020H      2      QBUFF. . . . . .  WORD MEMBER(CCB)
 41  0020H      2      QBUFF. . . . . .  WORD MEMBER(OLDCCB)
 39  0020H      2      QBUFF. . . . . .  WORD MEMBER(CCB)
128  0020H      2      QBUFF. . . . . .  WORD MEMBER(CCB)
 47                    QDSTRUCTURE. . . . . .  LITERALLY
 54                    QFDEV. . . . . .  LITERALLY
 50                    QFHIDE . . . . . .  LITERALLY
```

```
 49                 QFKEEP . . . . . .   LITERALLY
 48                 QFMX . . . . . .     LITERALLY
 53                 QFRPL. . . . . .     LITERALLY
 51                 QFRSP. . . . . .     LITERALLY
 52                 QFTABLE. . . . . .   LITERALLY
  9  0074H      2  QLR. . . . . .       WORD MEMBER(SD)
  9  0060H      8  QMAU . . . . . .     WORD ARRAY(4) MEMBER(SD)
 46                 QNAMSIZ. . . . . .   LITERALLY         56
 56  0018H     16  QPD. . . . . .       STRUCTURE         94   95   99  100  131  132  133  134  135  136
                                          141  142  143  203  204  205
122  001EH      2  QPDBUFFPTR . . . . .  WORD MEMBER(CCB)
112  001EH      2  QPDBUFFPTR . . . . .  WORD MEMBER(CCB)
 41  001EH      2  QPDBUFFPTR . . . . .  WORD MEMBER(OLDCCB)
 39  001EH      2  QPDBUFFPTR . . . . .  WORD MEMBER(CCB)
128  001EH      2  QPDBUFFPTR . . . . .  WORD MEMBER(CCB)
128  0018H      1  QPDFLAGS . . . . . .  BYTE MEMBER(CCB)
122  0018H      1  QPDFLAGS . . . . . .  BYTE MEMBER(CCB)
112  0018H      1  QPDFLAGS . . . . . .  BYTE MEMBER(CCB)
 41  0018H      1  QPDFLAGS . . . . . .  BYTE MEMBER(OLDCCB)
 39  0018H      1  QPDFLAGS . . . . . .  BYTE MEMBER(CCB)
128  001CH      2  QPDNMSGS . . . . . .  WORD MEMBER(CCB)
122  001CH      2  QPDNMSGS . . . . . .  WORD MEMBER(CCB)
112  001CH      2  QPDNMSGS . . . . . .  WORD MEMBER(CCB)
 41  001CH      2  QPDNMSGS . . . . . .  WORD MEMBER(OLDCCB)
 39  001CH      2  QPDNMSGS . . . . . .  WORD MEMBER(CCB)
128  001AH      2  QPDQADDR . . . . . .  WORD MEMBER(CCB)
122  001AH      2  QPDQADDR . . . . . .  WORD MEMBER(CCB)
112  001AH      2  QPDQADDR . . . . . .  WORD MEMBER(CCB)
 41  001AH      2  QPDQADDR . . . . . .  WORD MEMBER(OLDCCB)
 39  001AH      2  QPDQADDR . . . . . .  WORD MEMBER(CCB)
128  0019H      1  QPDRESRVD. . . . . .  BYTE MEMBER(CCB)
122  0019H      1  QPDRESRVD. . . . . .  BYTE MEMBER(CCB)
112  0019H      1  QPDRESRVD. . . . . .  BYTE MEMBER(CCB)
 41  0019H      1  QPDRESRVD. . . . . .  BYTE MEMBER(OLDCCB)
 39  0019H      1  QPDRESRVD. . . . . .  BYTE MEMBER(CCB)
 55                 QPDSTRUCTURE . . . .  LITERALLY         56
128  0002H      2  QUEUE. . . . . .     WORD MEMBER(CCB)
122  0002H      2  QUEUE. . . . . .     WORD MEMBER(CCB)
112  0002H      2  QUEUE. . . . . .     WORD MEMBER(CCB)
 44  0002H      2  QUEUE. . . . . .     WORD MEMBER(LCB)          316
 41  0002H      2  QUEUE. . . . . .     WORD MEMBER(OLDCCB)
 39  0002H      2  QUEUE. . . . . .     WORD MEMBER(CCB)
  9  005EH      2  QUL. . . . . .       WORD MEMBER(SD)
122  0028H      2  R1 . . . . . .       WORD MEMBER(CCB)
112  0028H      2  R1 . . . . . .       WORD MEMBER(CCB)
 41  0028H      2  R1 . . . . . .       WORD MEMBER(OLDCCB)
 39  0028H      2  R1 . . . . . .       WORD MEMBER(CCB)
128  0028H      2  R1 . . . . . .       WORD MEMBER(CCB)
122  002AH      2  R2 . . . . . .       WORD MEMBER(CCB)
112  002AH      2  R2 . . . . . .       WORD MEMBER(CCB)
 41  002AH      2  R2 . . . . . .       WORD MEMBER(OLDCCB)
 39  002AH      2  R2 . . . . . .       WORD MEMBER(CCB)
128  002AH      2  R2 . . . . . .       WORD MEMBER(CCB)
322  06C1H     45  RAW. . . . . .       PROCEDURE BYTE STACK=001CH          341
 92  0060H     23  READCHANGEMXQ. . . .  PROCEDURE STACK=000AH          164  169  198  245  252  266  273
                                          288
128  000DH      1  RESERVD. . . . . .   BYTE MEMBER(CCB)
```

```
122  000DH    1  RESERVD. . . . . .  BYTE MEMBER(CCB)
 20  0028H    4  RESERVD. . . . . .  BYTE ARRAY(4) MEMBER(PD)
112  000DH    1  RESERVD. . . . . .  BYTE MEMBER(CCB)
 41  000DH    1  RESERVD. . . . . .  BYTE MEMBER(OLDCCB)
 39  000DH    1  RESERVD. . . . . .  BYTE MEMBER(CCB)
120  00DFH   22  RESETCONOUTFLAG. . .  PROCEDURE STACK=0014H          147  172  182  266  306  355
 35  0003H    2  RESRVD1. . . . . .  BYTE ARRAY(2) MEMBER(XINIT)
  9  0066H    2  RLR. . . . . .  WORD MEMBER(SD)
  3  0000H    2  RSPLINK. . . . . .  WORD EXTERNAL(0)          331
  9  0042H    2  RSPSEG . . . . . .  WORD MEMBER(SD)
 58  0005H    1  RSRVD. . . . . .  BYTE MEMBER(APB)
 20  002EH    2  SCRATCH. . . . . .  WORD MEMBER(PD)
 45  0036H    1  SCREEN . . . . . .  BYTE        88  146  163  171  296  334  354
  9  0000H  145  SD . . . . . .  STRUCTURE BASED(SYSDATPOINTER)      153  157  291  311  332
 10  0000H    1  SDBYTE . . . . . .  BYTE BASED(SYSDATPOINTER) ARRAY(1)
  8  0002H    2  SEGMENT. . . . . .  WORD MEMBER(SYSDATPTR)          331
 19  0002H    2  SEGMENT. . . . . .  WORD MEMBER(PDPTR)
 43  0002H    2  SEGMENT. . . . . .  WORD MEMBER(LCBPTR)        351
 38  0002H    2  SEGMENT. . . . . .  WORD MEMBER(CCBPTR)        331
 35  0002H    2  SEGMENT. . . . . .  WORD MEMBER(XINITPTR)
150  0193H   39  SETCCB . . . . . .  PROCEDURE STACK=0004H          163
110  0088H   39  SETCONOUTFLAG. . . .  PROCEDURE STACK=0014H          145  165  179  227  303  353
 20  0025H    1  SF3. . . . . .  BYTE MEMBER(PD)
 20  0026H    1  SF4. . . . . .  BYTE MEMBER(PD)
 20  0027H    1  SF5. . . . . .  BYTE MEMBER(PD)
             SHL. . . . . .  BUILTIN         223
             SIZE . . . . . .  BUILTIN         153  291  311
102  008EH   21  SLEEP. . . . . .  PROCEDURE STACK=000CH          115
  9  0070H    2  SLR. . . . . .  WORD MEMBER(SD)
  9  0008H   56  SPACE1 . . . . . .  WORD ARRAY(28) MEMBER(SD)
  9  008CH    4  SPACE2 . . . . . .  WORD ARRAY(2) MEMBER(SD)
 84  0004H    4  SPTR . . . . . .  POINTER PARAMETER AUTOMATIC          65   87   88
  9  0046H    1  SRCHDISK . . . . . .  BYTE MEMBER(SD)
128  0005H    1  STARTCOL . . . . . .  BYTE MEMBER(CCB)
122  0005H    1  STARTCOL . . . . . .  BYTE MEMBER(CCB)
112  0005H    1  STARTCOL . . . . . .  BYTE MEMBER(CCB)
 44  0005H    1  STARTCOL . . . . . .  BYTE MEMBER(LCB)
 41  0005H    1  STARTCOL . . . . . .  BYTE MEMBER(OLDCCB)
 39  0005H    1  STARTCOL . . . . . .  BYTE MEMBER(CCB)          235  305
 20  0004H    1  STAT . . . . . .  BYTE MEMBER(PD)
128  000EH    2  STATE. . . . . .  WORD MEMBER(CCB)          129
122  000EH    2  STATE. . . . . .  WORD MEMBER(CCB)
112  000EH    2  STATE. . . . . .  WORD MEMBER(CCB)
 41  000EH    2  STATE. . . . . .  WORD MEMBER(OLDCCB)        166  167  168  174
 39  000EH    2  STATE. . . . . .  WORD MEMBER(CCB)          159  176  177  179  184  199  201  202
                                     246  253  267  274  289  298  314  368
 85  0000H    1  STRING . . . . . .  BYTE BASED(SPTR) ARRAY(1)          87   88
  9  0000H    8  SUPMOD . . . . . .  WORD ARRAY(4) MEMBER(SD)
156  018AH  295  SWITCH . . . . . .  PROCEDURE STACK=0018H          338
  7  0000H    4  SYSDATPOINTER. . . .  POINTER            8    9   10  153  157  291  311  332
  8  0000H    4  SYSDATPTR. . . . . .  STRUCTURE AT          331
  2             TAB. . . . . .  LITERALLY
  9  0050H    1  TEMPDISK . . . . . .  BYTE MEMBER(SD)
 58  0002H    2  TERM . . . . . .  WORD MEMBER(APB)
  9  0072H    2  THRDRT . . . . . .  WORD MEMBER(SD)
 20  0002H    2  THREAD . . . . . .  WORD MEMBER(PD)
 35  0000H    1  TICK . . . . . .  BYTE MEMBER(XINIT)
```

```
     9  0051H     1  TICKSPERSEC. . . . . BYTE MEMBER(SD)
    35  0001H     1  TICKSSEC . . . . . . BYTE MEMBER(XINIT)
     9  0080H     3  TOD. . . . . . . . . BYTE ARRAY(3) MEMBER(SD)
     9  007EH     2  TOD_DAY. . . . . . . WORD MEMBER(SD)
     2             TRUE . . . . . . . . . LITERALLY        185  270  324  328  335
   251  04C7H    54  TURNOFFCTRLO . . . . . PROCEDURE STACK=0018H        287  369
    20  0010H     2  UDA. . . . . . . . . WORD MEMBER(PD)
    20  0013H     1  USER . . . . . . . . BYTE MEMBER(PD)
   122  0024H     2  USLEEP . . . . . . . WORD MEMBER(CCB)
   112  0024H     2  USLEEP . . . . . . . WORD MEMBER(CCB)
    41  0024H     2  USLEEP . . . . . . . WORD MEMBER(OLDCCB)
    39  0024H     2  USLEEP . . . . . . . WORD MEMBER(CCB)          210  277
   128  0024H     2  USLEEP . . . . . . . WORD MEMBER(CCB)
   128  0008H     1  VC . . . . . . . . . BYTE MEMBER(CCB)

   122  0008H     1  VC . . . . . . . . . BYTE MEMBER(CCB)
   112  0008H     1  VC . . . . . . . . . BYTE MEMBER(CCB)
    41  0008H     1  VC . . . . . . . . . BYTE MEMBER(OLDCCB)
    39  0008H     1  VC . . . . . . . . . BYTE MEMBER(CCB)          161
   150  0004H     1  VC . . . . . . . . . BYTE PARAMETER AUTOMATIC       151  153
   128  0016H     2  VCMXQ. . . . . . . . WORD MEMBER(CCB)
   122  0016H     2  VCMXQ. . . . . . . . WORD MEMBER(CCB)

   112  0016H     2  VCMXQ. . . . . . . . WORD MEMBER(CCB)
    41  0016H     2  VCMXQ. . . . . . . . WORD MEMBER(OLDCCB)        164  173
    39  0016H     2  VCMXQ. . . . . . . . WORD MEMBER(CCB)          169  183  198  208  245  247  252  254
                                           266  269  273  275  288  319
     9  007AH     2  VERNUM . . . . . . . WORD MEMBER(SD)
     9  0078H     2  VERSION. . . . . . . WORD MEMBER(SD)
   128  0012H     2  VINQ . . . . . . . . WORD MEMBER(CCB)
   122  0012H     2  VINQ . . . . . . . . WORD MEMBER(CCB)
   112  0012H     2  VINQ . . . . . . . . WORD MEMBER(CCB)
    41  0012H     2  VINQ . . . . . . . . WORD MEMBER(OLDCCB)
    39  0012H     2  VINQ . . . . . . . . WORD MEMBER(CCB)          141  203
   128  0014H     2  VOUTQ. . . . . . . . WORD MEMBER(CCB)          131  134
   122  0014H     2  VOUTQ. . . . . . . . WORD MEMBER(CCB)
   112  0014H     2  VOUTQ. . . . . . . . WORD MEMBER(CCB)
    41  0014H     2  VOUTQ. . . . . . . . WORD MEMBER(OLDCCB)
    39  0014H     2  VOUTQ. . . . . . . . WORD MEMBER(CCB)
   122  0026H     2  VSLEEP . . . . . . . WORD MEMBER(CCB)
   112  0026H     2  VSLEEP . . . . . . . WORD MEMBER(CCB)
    41  0026H     2  VSLEEP . . . . . . . WORD MEMBER(OLDCCB)
    39  0026H     2  VSLEEP . . . . . . . WORD MEMBER(CCB)
   128  0026H     2  VSLEEP . . . . . . . WORD MEMBER(CCB)          137
    20  001AH     2  WAIT . . . . . . . . WORD MEMBER(PD)
   106  00A3H    21  WAKEUP . . . . . . . PROCEDURE STACK=000CH        124  137  210  277  316
   126  00F5H    84  WAKEVOUT . . . . . . PROCEDURE STACK=0014H        175  180  209  248  255  276
    97  0077H    23  WRITECHANGEMXQ . . . . PROCEDURE STACK=000AH        173  183  208  247  254  269  275
                                           319
   139  0149H    74  WRITEVINQ. . . . . . PROCEDURE STACK=001AH        195  242  263  284  342  370
    35  0000H    12  XINIT. . . . . . . . STRUCTURE BASED(XINITPOINTER)
    35             XINITOFFSET. . . . . LITERALLY
    35  0008H     4  XINITPOINTER . . . . . POINTER          35
    35  0008H     4  XINITPTR . . . . . . STRUCTURE AT
```


MODULE INFORMATION:

```
     CODE AREA SIZE     = 0703H    20030
```

```
VARIABLE AREA SIZE = 0048H     72D
MAXIMUM STACK SIZE = 0022H     34D
795 LINES READ
0 PROGRAM WARNINGS
0 PROGRAM ERRORS

END OF PL/M-86 COMPILATION
```

```
VAX/VMS 8086 LOCATER, V1.0VX

INPUT FILE: PIN.LNK
OUTPUT FILE: PIN.DAT
CONTROLS SPECIFIED IN INVOCATION COMMAND:
  ORDER(SM(CODE,DATA,DATA,CONST,STACK))AD(SM(CODE(0),DATA(10000H)))SS(
  STACK(0))TO PIN.DAT
WARNING 26:  DECREASING SIZE OF SEGMENT
  SEGMENT:  STACK
WARNING 36:  SEGMENTS OVERLAP
  SEGMENT:  CODE
  SEGMENT:  (NO NAME)
  LOW OVERLAP ADDRESS :  00200H
  HIGH OVERLAP ADDRESS:  00216H


SYMBOL TABLE OF MODULE RPTN

BASE    OFFSET TYPE SYMBOL              BASE    OFFSET TYPE SYMBOL

1000H   0000H  PUB  RSPLINK             1000H   0010H  PUB  PD
1000H   0004H  PUB  NCOPIES             1000H   0020H  PUB  UDASEG
1000H   0088H  PUB  U_RETSEG            1000H   0086H  PUB  U_WRKSEG
1000H   00A0H  PUB  U_INSYS             0000H   0750H  PUB  PLMSTART
0000H   0040H  PUB  PXIOS1              0000H   0040H  PUB  PXIOS2
0000H   0040H  PUB  PXIOS3              0000H   0040H  PUB  PXIOS4
0000H   0000H  PUB  XDOS                0000H   0000H  PUB  MON1
0000H   0000H  PUB  MON2                0000H   0000H  PUB  MON3
0000H   0000H  PUB  MON4                0000H   000FH  PUB  INTSYS
```

MODULE = PIN

BASE	OFFSET	TYPE	SYMBOL	BASE	OFFSET	TYPE	SYMBOL
1000H	01C0H	SYM	MEMORY	1000H	0140H	SYM	SYSDATPOINTER
1000H	0140H	SYM	SYSDATPTR	1000H	0140H	BAS	SD
1000H	0140H	BAS	SDBYTE	1000H	0144H	SYM	PDPOINTER
1000H	0144H	SYM	PDPTR	1000H	0144H	BAS	PD
1000H	0004H	SYM	CNSNUM	1000H	0148H	SYM	XINITPOINTER
1000H	0148H	SYM	XINITPTR	1000H	0148H	BAS	XINIT
1000H	014CH	SYM	CCBPOINTER	1000H	014CH	SYM	CCBPTR
1000H	014CH	BAS	CCB	1000H	0150H	SYM	OLDCCBPOINTER
1000H	0150H	BAS	OLDCCB	1000H	0154H	SYM	LCBPOINTER
1000H	0154H	SYM	LCBPTR	1000H	0154H	BAS	LCB
1000H	0176H	SYM	SCREEN	1000H	0158H	SYM	QPB
1000H	018CH	SYM	NULL	1000H	0168H	SYM	APB
1000H	016EH	SYM	CWORD	1000H	016EH	SYM	CHARS
1000H	016EH	SYM	CHAR	1000H	016FH	SYM	CHARTYPE
0000H	0062H	SYM	CONIN	0000H	0076H	SYM	PRINTMSG
STACK	000AH	SYM	LEN	STACK	0008H	SYM	ENDCHAR
STACK	0004H	SYM	SPTR	1000H	0177H	SYM	I
STACK	0004H	BAS	STRING	0000H	00C2H	SYM	READCHANGEMXQ
STACK	0004H	SYM	QADDR	0000H	00D9H	SYM	WRITECHANGEMXQ
STACK	0004H	SYM	QADDR	0000H	00F0H	SYM	SLEEP
STACK	0004H	SYM	LISTROOT	0000H	0105H	SYM	WAKEUP
STACK	0004H	SYM	LISTROOT	0000H	011AH	SYM	SETCONOUTFLAG
STACK	0004H	SYM	CCBPTR	STACK	0004H	BAS	CCB
0000H	0141H	SYM	RESETCONOUTFLAG	STACK	0004H	SYM	CCBPTR
STACK	0004H	BAS	CCB	0000H	0157H	SYM	WAKEVOUT
STACK	0004H	SYM	CCBPTR	STACK	0004H	BAS	CCB
0000H	01ABH	SYM	WRITEVINQ	STACK	0004H	SYM	C
0000H	01F5H	SYM	SETCCB	STACK	0004H	SYM	VC
1000H	0178H	SYM	CONTROLSHASBEENP-RESSED	0000H	021CH	SYM	SWITCH
1000H	0179H	SYM	DRIVELETTERS	0000H	0343H	SYM	CONTROLC
1000H	0170H	SYM	JUNK	1000H	0172H	SYM	CURDRIVE
1000H	0174H	SYM	LOGGEDINDRIVES	1000H	018AH	SYM	LETTERINDEX
0000H	04DAH	SYM	CONTROLO	0000H	0529H	SYM	TURNOFFCTRLO
0000H	055FH	SYM	CONTROLS	0000H	05C8H	SYM	CONTROLQ
0000H	060EH	SYM	CONTROLP	0000H	0723H	SYM	RAW
0000H	0750H	SYM	PLMSTART	0000H	0062H	LIN	81
0000H	0065H	LIN	82	0000H	0074H	LIN	83
0000H	0076H	LIN	84	0000H	0079H	LIN	86
0000H	007EH	LIN	87	0000H	00A5H	LIN	88
0000H	00B8H	LIN	89	0000H	00BCH	LIN	90
0000H	00BEH	LIN	91	0000H	00C2H	LIN	92
0000H	00C5H	LIN	94	0000H	00CBH	LIN	95
0000H	00D5H	LIN	96	0000H	00D9H	LIN	97
0000H	00DCH	LIN	99	0000H	00E2H	LIN	100
0000H	00ECH	LIN	101	0000H	00F0H	LIN	102
0000H	00F3H	LIN	104	0000H	0101H	LIN	105
0000H	0105H	LIN	106	0000H	0108H	LIN	108
0000H	0116H	LIN	109	0000H	011AH	LIN	110
0000H	011DH	LIN	113	0000H	011EH	LIN	114
0000H	012BH	LIN	115	0000H	0132H	LIN	116
0000H	0134H	LIN	117	0000H	013CH	LIN	118
0000H	013DH	LIN	119	0000H	0141H	LIN	120
0000H	0144H	LIN	125	0000H	014CH	LIN	124

```
0000H  0153H  LIN  125          0000H  0157H  LIN  126
0000H  015AH  LIN  129          0000H  0169H  LIN  131
0000H  0173H  LIN  132          0000H  0179H  LIN  133
0000H  0183H  LIN  134          0000H  018DH  LIN  135
0000H  0193H  LIN  136          0000H  019DH  LIN  137
0000H  01A7H  LIN  138          0000H  01ABH  LIN  139
0000H  01AEH  LIN  141          0000H  0169H  LIN  142
0000H  01BFH  LIN  143          0000H  01CEH  LIN  145
0000H  01D7H  LIN  146          0000H  01E8H  LIN  147
0000H  01F1H  LIN  149          0000H  01F5H  LIN  150
0000H  01F8H  LIN  152          0000H  0203H  LIN  153
0000H  0218H  LIN  154          0000H  021CH  LIN  156
0000H  021FH  LIN  157          0000H  022CH  LIN  158
0000H  022EH  LIN  159          0000H  023EH  LIN  160
0000H  0240H  LIN  161          0000H  024DH  LIN  162
0000H  024FH  LIN  163          0000H  025CH  LIN  164
0000H  0267H  LIN  165          0000H  0270H  LIN  166
0000H  027FH  LIN  167          0000H  0287H  LIN  168
0000H  028DH  LIN  169          0000H  0298H  LIN  170
0000H  02A1H  LIN  171          0000H  02B2H  LIN  172
0000H  02B8H  LIN  173          0000H  02C6H  LIN  174
0000H  02D6H  LIN  175          0000H  02DBH  LIN  176
0000H  02EAH  LIN  177          0000H  02F2H  LIN  179
0000H  0300H  LIN  180          0000H  0305H  LIN  182
0000H  030EH  LIN  183          0000H  0319H  LIN  184
0000H  0329H  LIN  185          0000H  0330H  LIN  186
0000H  0335H  LIN  187          0000H  0341H  LIN  188
0000H  0343H  LIN  190          0000H  0346H  LIN  193
0000H  0356H  LIN  195          0000H  035DH  LIN  196
0000H  035FH  LIN  198          0000H  036AH  LIN  199
0000H  0379H  LIN  200          0000H  037EH  LIN  201
0000H  0386H  LIN  202          0000H  0394H  LIN  203
0000H  039FH  LIN  204          0000H  03A5H  LIN  205
0000H  03B3H  LIN  207          0000H  03BCH  LIN  208
0000H  03C3H  LIN  209          0000H  03CCH  LIN  210
0000H  03D7H  LIN  211          0000H  03E1H  LIN  212
0000H  03E8H  LIN  213          0000H  03F8H  LIN  214
0000H  03FEH  LIN  215          0000H  0403H  LIN  216
0000H  0410H  LIN  217          0000H  041CH  LIN  218
0000H  042AH  LIN  220          0000H  043AH  LIN  221
0000H  043EH  LIN  223          0000H  0442H  LIN  224
0000H  0449H  LIN  225          0000H  0453H  LIN  227
0000H  045CH  LIN  228          0000H  046BH  LIN  229
0000H  047AH  LIN  230          0000H  048DH  LIN  231
0000H  049CH  LIN  232          0000H  04AFH  LIN  233
0000H  0486H  LIN  234          0000H  04C5H  LIN  235
0000H  04D3H  LIN  236          0000H  04D8H  LIN  238
0000H  04DAH  LIN  239          0000H  04DDH  LIN  240
0000H  04EDH  LIN  242          0000H  04F4H  LIN  243
0000H  04F6H  LIN  245          0000H  0501H  LIN  246
0000H  0508H  LIN  247          0000H  0512H  LIN  248
0000H  0518H  LIN  249          0000H  0527H  LIN  250
0000H  0529H  LIN  251          0000H  052CH  LIN  252
0000H  0537H  LIN  253          0000H  0541H  LIN  254
0000H  0548H  LIN  255          0000H  0551H  LIN  256
0000H  055DH  LIN  257          0000H  055FH  LIN  258
0000H  0562H  LIN  259          0000H  0572H  LIN  260
0000H  0574H  LIN  261          0000H  0584H  LIN  263
```

```
0000H  0588H  LIN   264        0000H  058DH  LIN   266
0000H  0598H  LIN   267        0000H  05AAH  LIN   268
0000H  05B6H  LIN   269        0000H  05C1H  LIN   270
0000H  05C6H  LIN   271        0000H  05C8H  LIN   272
0000H  05CBH  LIN   273        0000H  05D6H  LIN   274
0000H  05E0H  LIN   275        0000H  05E7H  LIN   276
0000H  05F0H  LIN   277        0000H  05F8H  LIN   278
0000H  0607H  LIN   279        0000H  060CH  LIN   280
0000H  060FH  LIN   281        0000H  0611H  LIN   282
0000H  0621H  LIN   284        0000H  0628H  LIN   285
0000H  062AH  LIN   287        0000H  062DH  LIN   288
0000H  0638H  LIN   289        0000H  0648H  LIN   291
0000H  0663H  LIN   292        0000H  0664H  LIN   293
0000H  066EH  LIN   295        0000H  0673H  LIN   296
0000H  067AH  LIN   297        0000H  068AH  LIN   298
0000H  0690H  LIN   299        0000H  0692H  LIN   300
0000H  0692H  LIN   302        0000H  0693H  LIN   303
0000H  069CH  LIN   304        0000H  06ABH  LIN   305
0000H  06B9H  LIN   306        0000H  06BEH  LIN   308
0000H  06C0H  LIN   310        0000H  06C1H  LIN   311
0000H  06DCH  LIN   312        0000H  06E5H  LIN   313
0000H  06F3H  LIN   314        0000H  06F9H  LIN   315
0000H  06FEH  LIN   316        0000H  0709H  LIN   317
0000H  070AH  LIN   319        0000H  0715H  LIN   320
0000H  0721H  LIN   321        0000H  0723H  LIN   322
0000H  0726H  LIN   323        0000H  0735H  LIN   325
0000H  0745H  LIN   326        0000H  0749H  LIN   327
0000H  074CH  LIN   328        0000H  0750H  LIN   329
0000H  0750H  LIN   330        0000H  0753H  LIN   331
0000H  075FH  LIN   332        0000H  076AH  LIN   333
0000H  077CH  LIN   334        0000H  0781H  LIN   335
0000H  0781H  LIN   336        0000H  0784H  LIN   337
0000H  0788H  LIN   338        0000H  0790H  LIN   339
0000H  0797H  LIN   341        0000H  079EH  LIN   342
0000H  07A7H  LIN   344        0000H  07AEH  LIN   346
0000H  07B5H  LIN   348        0000H  07BCH  LIN   349
0000H  07C1H  LIN   350        0000H  07C8H  LIN   353
0000H  07D1H  LIN   354        0000H  07E2H  LIN   355
0000H  07EBH  LIN   357        0000H  07EDH  LIN   359
0000H  07F4H  LIN   360        0000H  07F9H  LIN   361
0000H  0800H  LIN   362        0000H  0805H  LIN   363
0000H  080CH  LIN   364        0000H  0811H  LIN   365
0000H  0818H  LIN   366        0000H  081DH  LIN   368
0000H  082DH  LIN   369        0000H  0830H  LIN   370
0000H  0833H  LIN   375        0000H  0833H  LIN   376
0000H  0062H  LIN   377
```


MEMORY MAP OF MODULE RPIN

SEGMENT MAP

```
START     STOP      LENGTH ALIGN NAME            CLASS          OVERLAY

00000H    00834H  C 0835H  G    CODE            CODE
00200H    00216H    0017H  A    (ABSOLUTE)
10000H    1013FH    0140H  G    DATS            DATA
10140H    1018AH    004BH  W    DATA            DATA
```

```
1018CH    1018AH    002FH   W   CONST           CONST
1018CH    1018CH    0000H   W   STACK           STACK
101C0H    101C0H    0000H   G   ??SEG
101C0H    101C0H    0000H   W   MEMORY          MEMORY


GROUP MAP

ADDRESS   GROUP OR SEGMENT NAME
10000H    DGROUP
            DATS
            CONST
            DATA
00000H    CGROUP
            CODE
```

VAX/VMS 8086/8087/8088 MACRO ASSEMBLER V1.0 ASSEMBLY OF MODULE RPIN
OBJECT MODULE PLACED IN RHPIN.OBJ
NO INVOCATION LINE CONTROLS


```
LOC  OBJ                  LINE     SOURCE

                             1     ;       Code and Data Interface for PIN RSP
                             2     ;       Virtual console support for Concurrent CP/M
                             3
                             4     ;       March 30, 1982
                             5
                             6     name    rpin
                             7
                             8     cgroup  group   code
                             9     dgroup  group   dats
                            10
                            11     assume   cs:cgroup,ds:dgroup
                            12
                            13             public  xdos,mon1,mon2,mon3,mon4,intsys
                            14             public  rsplink, pd, ncopies, udaseg
                            15             public  u_retseg, u_wrkseg, u_insys
                            16             extrn  plmstart:near
                            17
----                        18     code    segment public  'CODE'
                            19
0000                        20     xdos    proc
0000 55                     21             push    bp
0001 8BEC                   22             mov     bp,sp
0003 8B5604                 23             mov     dx,[bp+4]
0006 8B4E06                 24             mov     cx,[bp+6]
0009 CDE0                   25             int     224
000B 5D                     26             pop     bp
000C C20400                 27             ret     4
                            28     xdos    endp
                            29
000F                        30     intsys  proc                            ;call O.S. as if we are
000F 55                     31             push bp                         ;already in it
0010 8BEC                   32             mov bp,sp
0012 8B5E04                 33             mov bx,[bp+4]
0015 8B5606                 34             mov dx,[bp+6]
0018 8B4E08                 35             mov cx,[bp+8]
                            36
0018 1E                     37             push ds
001C 8E1E0000       R       38             mov ds,rsplink                  ;DS = Sysdat Segment
0020 8B366800               39             mov si,ds:word ptr rlr
0024 8E441090               40             mov es,[si+p_uda]
0028 FF1E000090             41             call ds:dword ptr [supervisor]
002D 1F                     42             pop ds
                            43
002E 5D                     44             pop bp
002F C20600                 45             ret 6
                            46     intsys  endp
                            47
  0000                      48     mon1    equ     xdos
  0000                      49     mon2    equ     xdos
  0000                      50     mon3    equ     xdos
```

```
LOC  OBJ                  LINE     SOURCE

  0000                      51     mon4    equ     xdos
----                        52     code    ends
                            53
----                        54     dats    segment  public 'DATA'
                            55
  0068                      56     rlr             equ     68h     ;ready list root
  0010                      57     p_uda           equ     10h     ;UDA in process descriptor
  0000                      58     supervisor      equ     0       ;supervisor entry point for internal
  0047                      59     nvcns           equ     47h
  0010                      60     rsphdr_len      equ     16
  0030                      61     pd_len          equ     30H
  0100                      62     uda_len         equ     100H
  0060                      63     insysoff        equ     60H
  0000                      64     rsp_top         equ     0
  0010                      65     rsp_pd          equ     rsp_top + rsphdr_len
  0040                      66     rsp_uda         equ     rsp_pd + pd_len
  0140                      67     rsp_bottom      equ     rsp_uda + uda_len
                            68
0000                        69             org     rsp_top
                            70                             ;RSP header
0000 0000                   71     rsplink dw      0       ;becomes system data page paragraph
0002 0000                   72     sdatvar dw      0       ;tell gensys to one
0004 00                     73     ncopies db      0
0005 0000                   74             dw      0,0,0,0, 0
0007 0000
0009 0000
000B 0000
000D 0000
000F 00                     75             db      0
0010                        76             org     rsp_pd
0010 0000                   77     pd      dw      0,0             ;link fields
0012 0000
0014 00                     78             db      0               ;status
0015 B9                     79             db      185             ;initial priority better than TMP - worse than
                            80                                     ;VOUT
0016 0300                   81             dw      3               ;flags - system and keep

0018 50494E20202020         82             db      'PIN     '      ;name
     20
001E 0400                   83     udaseg  dw      rsp_uda/10h     ;uda paragraph
0022 00                     84             db      0,0             ;disk,user
0023 00
0024 00                     85             db      0,0             ;ldisk,luser
0025 00
0026 FF00                   86             dw      0ffh            ;puremem - re-entrant
                            87                                     ;rest of pd
                            88
0040                        89     org     rsp_uda                 ;start of uda
0040 0000                   90     uda             dw      0
0042 0000                   91                     dw      0       ;no default DMA
0044 0000                   92                     dw      0,0,0,0, 0,0,0,0, 0,0,0,0
0046 0000
0048 0000
004A 0000
004C 0000
004E 0000
```

```
LOC  OBJ                  LINE     SOURCE

0050 0000
0052 0000
0054 0000
0056 0000
0058 0000
005A 0000
005C 0000                   93                    dw      0,0,0,0,  0,0,0,0,  0,0,0,0
005E 0000
0060 0000
0062 0000
0064 0000
0066 0000
0068 0000
006A 0000
006C 0000
006E 0000
0070 0000
0072 0000
0074 3A01              R    94                    dw      offset stk_top
0076 0000                   95                    dw      0,0,0,0,  0,0,0,0
0078 0000
007A 0000
007C 0000
007E 0000
0080 0000
0082 0000
0084 0000
0086 0000                   96      u_wrkseg      dw      0
0088 0000                   97      u_retseg      dw      0
00A0                        98      org     rsp_uda + insysoff
00A0 01                     99      u_insys       db      1
00A1 00                    100                    db      0       ;u_stat_save
00A2 0000                  101                    dw      0       ;ccb
00A4 0000                  102                    dw      0       ;lcb
00A6 00                    103                    db      0       ;print string delimiter
                           104
00A7 (147                  105                    db      93h dup (0cch)  ;fill rest of UDA stack with INT3s
     CC
     )
                           106
013A 0000              E   107      stk_top       dw      plmstart
013C 0000                  108                    dw      0,0             ;segment, flags: unknown
013E 0000
                           109
----                       110      dats ends
                           111      end

ASSEMBLY COMPLETE, NO ERRORS FOUND
```

```
VAX/VMS 8086/8087/8088 MACRO ASSEMBLER V1.0 ASSEMBLY OF MODULE PXIOS
OBJECT MODULE PLACED IN PXIOS.OBJ
NO INVOCATION LINE CONTROLS

LOC  OBJ                  LINE     SOURCE

                             1     ;       Interface to call Physical XIOS
                             2     ;       From a process not in the O.S.
                             3     ;       code reentrant, separate data areas per process
                             4     ;       used by VOUT and PIN RSPs
                             5
                             6     name    pxios
                             7
                             8     cgroup  group   code
                             9     dgroup  group   dats
                            10
                            11     assume  cs:cgroup
                            12     assume  ds:dgroup
                            13
----                        14     dats    segment public 'DATA'
                            15             extrn   rsplink:word    ;segment of SYSDAT
                            16             extrn   udaseg:word     ;UDA must be in ES for XIOS call
                            17             extrn   u_retseg:word, u_wrkseg:word, u_insys:byte
----                        18     dats    ends
                            19
----                        20     code    segment public 'CODE'
                            21             public  pxios1, pxios2, pxios3, pxios4
  0028                      22             xiosmod equ     28h
  0010                      23             p_uda   equ     10h
  0068                      24             rlr     equ     68h
                            25
  0000                      26     pxios2  equ     pxios1
  0000                      27     pxios3  equ     pxios1
  0000                      28     pxios4  equ     pxios1
                            29
0000                        30     pxios1  proc
0000 55                     31             push    bp
0001 8BEC                   32             mov     bp,sp
                            33
0003 8B4608                 34             mov     ax,[bp+8]               ;set up registers
0006 8B4E06                 35             mov     cx,[bp+6]
0009 8B5604                 36             mov     dx,[bp+4]
                            37
000C 1E                     38             push    ds
000D 8E1E0000         E     39             mov     ds,rsplink              ;SYSDAT
0011 8B366800               40             mov     si,ds:word ptr rlr      ;ready list root
0015 8E4410                 41             mov     es,[p_uda+si]           ;UDA
0018 FF1E2800               42             call    ds:dword ptr [xiosmod]
001C 1F                     43             pop     ds
                            44
001D 5D                     45             pop     bp
001E C20600                 46             ret     6
                            47     pxios1  endp
                            48
----                        49     code    ends
                            50     end
```

LOC OBJ LINE SOURCE

ASSEMBLY COMPLETE, NO ERRORS FOUND