```
SERIES-III PL/M-86 X193 COMPILATION OF MODULE R3CNFG
OBJECT MODULE PLACED IN :F1:MMX88.03J
COMPILER INVOKED BY:  PLM86.86 :F1:MMX88.P86 COMPACT ROM OPTIMIZE(3)


  1          R3CNFG:
               DO;
               /*****************************************************************
               *
               * TITLE:          MMX CONFIGURATION TABLE PROTOTYPE: MMX 88
               *
               * DATE:           04-01-82
               *
               * REVISION:       X005
               *
               * LANGUAGE:       PLM/86
               *
               * ABSTRACT:       THIS FILE CONTAINS THE CONFIGURATION TABLES
               *                 FOR THE MMX RUNTIME TASKS.  THESE TABLES MUST
               *                 BE FILLED WITH DATA CORRESPONDING TO THE MMX
               *                 APPLICATION.  ONE SUCH FILE MUST BE BUILT AND
               *                 COMPILED FOR EACH DEVICE IN THE SYSTEM.
               *
               *****************************************************************/

               /*
                * INCLUDE THE CONFIGURATION TABLE DEFINITIONS.
               */
               $INCLUDE(:f1:mmx88.lit)
                 =
                 =             /***********************************************\
                 =             |                                               |
                 =             |            R 3 C N F G . L I T                |
                 =             |                                               |
                 =             |  iMMX 800 CONFIGURATION LITERAL DECLARATIONS  |
                 =             |                                               |
                 =             \***********************************************/
                 =
                 =
  2   1          =     DECLARE DSDT$TYPE LITERALLY' STRUCTURE
                 =        (function$name WORD,
                 =         sckt$dev$id BYTE,
                 =         sckt$port$id BYTE,
                 =         src$dev$id BYTE,
                 =         src$port$id BYTE,
                 =         local$pool$id BYTE,
                 =         partition$id BYTE)';
                 =
  3   1          =     DECLARE LPT$ROM$TYPE LITERALLY 'STRUCTURE
                 =        (function$name WORD,
                 =         buffer$pool$id BYTE)';
                 =
  4   1          =     DECLARE LPT$RAM$TYPE LITERALLY 'STRUCTURE
                 =        (port$state     BYTE,
```

```
        =           port$exch (20) BYTE)';
        =

   5  1 =       DECLARE DCM$cntrl$elmnts  LITERALLY
        =          'RQD$out POINTER,
        =           RQ$out$size BYTE,
        =           RQE$out$size BYTE,
        =           RQD$in POINTER,
        =           RQ$in$size BYTE,
        =           RQE$in$size BYTE';
        =
        =
   6  1 =       DECLARE DCM$ROM$TYPE LITERALLY 'STRUCTURE
        =          (DCM$cntrl$elmnts )';
        =
        =
   7  1 =       DECLARE DCM$q$elmnts LITERALLY
        =          'channel$state BYTE,
        =           crq$head POINTER,
        =           cmd$rdy$state BYTE,
        =           cwq$head POINTER,
        =           cmd$wait$state BYTE';
        =
   8  1 =       DECLARE RQE$elmnts LITERALLY
        =          'request BYTE,
        =           src$port$id BYTE,
        =           dest$dev$id BYTE,
        =           dest$port$id BYTE,
        =           src$dev$id BYTE,
        =           data$ptr (2) ADDRESS,
        =           data$length WORD,
        =           partition$id BYTE,
        =           owner$dev$id BYTE';
        =
   9  1 =       DECLARE DCM$RAM$TYPE LITERALLY 'STRUCTURE
        =          (DCM$q$elmnts,
        =           RQE$elmnts )';
        =


  10  1 =       DECLARE IDS$TYPE LITERALLY 'STRUCTURE
        =          (offset  ADDRESS,
        =           page    ADDRESS)';
        =

  11  1 =       DECLARE BLOCK$TABLE$TYPE LITERALLY 'STRUCTURE
        =          (pool$id    BYTE,
        =           start$adr  WORD,
        =           length     WORD)';
        =
  12  1 =       DECLARE POOL$TABLE$TYPE   LITERALLY  'STRUCTURE
        =                     (free$list$token SELECTOR,
        =                      length          WORD,
        =                      pool$id         BYTE,
```

```
=                cr$permit(2)    POINTER,
=                cr$exch (20)    BYTE,
=                uc$get$exch(20) BYTE)';

13  1  =     DECLARE  SFT$TYPE  LITERALLY  'STRUCTURE
=                 (device$mode    BYTE,
=                  intr$type      BYTE,
=                  intr$location  WORD,
=                  intr$value     WORD,
=                  clr$intr$type  BYTE,
=                  clr$location   WORD,
=                  clr$value      WORD)';

       $INCLUDE(:f1:mmxp88.lit)
=      /*****************************************************************
=      *                                                                *
=      *                    INCLUDE FILE MMXPRT.LIT                     *
=      *                    =======================                     *
=      *                                                                *
=      *   THIS INCLUDE FILE DEFINES THE MMX, MIP AND GMF SYSTEM PORT NAMES *
=      *                                                                *
=      *   USED IN THE MSP OPERATING SYSTEM. INCLUDE THIS FILE IN YOUR CODE *
=      *                                                                *
=      *   TO MAKE IT INDEPENDENT OF CHANGES IN PORT NAMES OR DEVICES.  *
=      *                                                                *
=      *****************************************************************/
=               /* SPU-0 IS DEVICE 0 IT CURRENTLY HAS 13 SYSTEM      */
=               /* PORTS USED AS SHOWN BELOW                         */

14  1  =     DECLARE

=          spu                  literally  '00'  , /* device                */
=          XTH$RETURN$PORT      LITERALLY  '000H', /* XTH BIOS RECEIVE PORT      */
=          CXU$RETURN$PORT1     LITERALLY  '001H', /* CXU RECEIVE PORT1   */
=          CXU$RETURN$PORT2     LITERALLY  '002H', /* CXU RECEIVE PORT2    */
=          TMC                  LITERALLY  '003H', /* TAPS MASTER CONTROLLER     */
=          CM                   LITERALLY  '004H', /* TAPS CM                    */
=          TAC                  LITERALLY  '005H', /* TAPS APPLICATION CONTROLLER */
=          AM1                  LITERALLY  '006H', /* APPLICATION MANAGER #1     */
=          AM2                  LITERALLY  '007H', /* APPLICATION MANAGER #2     */
=          AM3                  LITERALLY  '008H', /* APPLICATION MANAGER #3     */
=          IO1                  LITERALLY  '009H', /* I/O MANAGER #1             */
=          IO2                  LITERALLY  '00AH', /* I/O MANAGER #2             */
=          TS1                  LITERALLY  '00BH', /* TAPS SPARE PORT #1         */
=          TS2                  LITERALLY  '00CH', /* TAPS SPARE PORT #2         */

=                /* CXU IS DEVICE 1 IT CURRENTLY HAS 2 SYSTEM       */
=                /* PORTS USED AS SHOWN                              */

=          cxu                  literally  '01'  , /* device           */
=          CXU$PORT1            LITERALLY  '100H', /* CXU  PORT1       */
=          CXU$PORT2            LITERALLY  '101H', /* CXU  PORT2       */
=          cxu$wraparound       literally  '102h', /* cxu  internal    */

=                /* 544 IS DEVICE 2 IT HAS 1 SYSTEM PORT             */
```

```
        =       i544                    literally  '02'  , /* device              */
        =       XTH$i544$PORT1          LITERALLY  '200H'; /* XTH BIOS XMIT PORT */
        =     /* XTH$i544$PORT1 SHOULD BE SET TO 200H IF 544 BOARD*/
              $INCLUDE(:f1:mmx188.lit)
        =     /*
        =       LITERALS FOR SPU,CXU AND 544 MMX SYSTEM CONFIGURATION
        =     */
 15   1 =     DECLARE
        =         SPU$POOL$ADDRESS    LITERALLY '0EEA30H',
        =         SPU$POOL$TOKEN      LITERALLY  '0EEA3H',
        =         SPU$POOL$LENGTH     LITERALLY   '15D0H',
        =         CXU$POOL$ADDRESS   LITERALLY '030000H',
        =         CXU$POOL$TOKEN     LITERALLY  '03000H',
        =         CXU$POOL$LENGTH    LITERALLY    '100H',
        =         i544$BASE           LITERALLY  '0EA00H',
        =         i544$BASE$ADDR      LITERALLY '0EA000H',
        =         i544$POOL$OFFSET    LITERALLY   '1000h',
        =         i544$POOL$LENGTH    LITERALLY   '0A40H',
        =         ONBOARD$RQD$SIZE    LITERALLY     '144', /* NO. OF RQE's = 8 */
        =         SPU$CXU$RQD$SIZE   LITERALLY      '80', /* NO. OF RQE's = 4 */
        =         rqe$size           literally       '4',  /* no. of bytes/rqe */

        =         i544$POOL$ADDR      LITERALLY '0EB000H', /*i544$BASE$ADDR+i544$POOL$OFFSET*/
        =         i544$POOLSTOKEN     LITERALLY  '0EB00H',
        =         i544$RQD$IN         LITERALLY '0EA060H', /* LEAVE ROOM FOR 544 ONBOARD */
        =         i544$RQD$OUT        LITERALLY '0EA010H', /* RQD WHICH IS 40 BYTES LONG */
        =         ONBOARD$RQD         LITERALLY '0EE8A0H', /*SPU$POOL$ADDRESS - ONBOARD$RQD$SIZE*/
        =         spu$to$cxu$rqd     LITERALLY '04E850H', /*CNBOARD$RQD - SPU$CXU$RQD$SIZE*/
        =         cxu$to$spu$rqd     LITERALLY '04E800H', /*SPU$CXU$RQD$OUT - SPU$CXU$RQD$SIZE*/
        =         cxu$to$cxu         literally '001000h'  /* cxu wrap rqd */
        =         ;

 16   1         declare reserved                literally '0';
 17   1         declare free$space$pool         literally '0';
 18   1         declare ids0                    literally '0';
 19   1         declare port01                  literally '01';
 20   1         declare port02                  literally '02';
 21   1         declare port03                  literally '03';
 22   1         declare zilch                   literally '00';

 23   1         DECLARE NO$SYSTEM$CHANNEL LITERALLY 'OFFFFH,
                                                    00H,
                                                    00H,
                                                    OFFFFH,
                                                    00H,
                                                    00H';

 24   1         declare clr$mb$intr             literally '200h';
 25   1         declare set$mb$intr             literally '201h';

 26   1         DECLARE NO$DEVICE               LITERALLY '00H';
 27   1         DECLARE SLAVE$DEVICE            LITERALLY '01H';
 28   1         DECLARE PEER$DEVICE             LITERALLY '02H';

 29   1         DECLARE NO$INTERRUPT            LITERALLY '00H';
 30   1         DECLARE MB$INTERRUPT            LITERALLY '01H';
 31   1         DECLARE IO$INTERRUPT            LITERALLY '02H';
```

```
32   1        DECLARE MM$INTERRUPT            LITERALLY '03H';

33   1        declare no$intr$cleared         literally '00h';
34   1        declare memory$read$clr         literally '01h';
35   1        declare memory$write$clr        literally '02h';
36   1        declare io$read$clr             literally '03h';
37   1        DECLARE IO$WRITE$CLR            LITERALLY '04h';

38   1        DECLARE NO$SYSTEM$SERVICE LITERALLY 'NO$DEVICE,
                                                  NO$INTERRUPT,
                                                  00H,
                                                  00H,
                                                  0000H,
                                                  0000H';


                 /*
                   MMX SYSTEM SIZE INFORMATION
                 */
39   1           DECLARE     NUMBER_DEVICES   LITERALLY '3';
40   1           DECLARE     CQSKTS_VALUE     LITERALLY '3';
41   1           DECLARE     CQPRTS_VALUE     LITERALLY '3';
                 $eject
```

```
                /*          number of mmx devices          */
42    1           DECLARE cqdvcs BYTE PUBLIC
                    DATA (number_devices);

                /*          sockets local to board         */
43    1           DECLARE cqskts BYTE PUBLIC
                    DATA (cqskts_value);

                /*          system ports addressable       */
44    1           DECLARE cqprts BYTE PUBLIC
                    DATA (cqprts_value);

                /*          time out factor for tx         */
45    1           DECLARE cqmdly WORD PUBLIC
                    DATA (0);

                /*          initial wait time for tx's     */
46    1           DECLARE cqitwt WORD PUBLIC
                    DATA (5);

                /*          socket mapping info            */
47    1           DECLARE DSDT (cqskts_value) DSDTSTYPE PUBLIC
                    DATA (
                          CXU$RETURN$PORT1,
                               spu,          /* CXU FLIPPY RECEIVE PORT      */
                               port01,
                               cxu,
                               reserved,
                               free$space$pool,
                               ids0,
                          CXU$RETURN$PORT2,
                               spu,          /* CXU PRINTER RECEIVE PORT     */
                               port02,
                               cxu,
                               reserved,
                               free$space$pool,
                               ids0,
                          cxu$wraparound,
                               cxu,
                               port03,
                               cxu,
                               reserved,
                               free$space$pool,
                               ids0
                            );

               /*          local port information          */
48    1           DECLARE LPT$ROM (cqprts_value) LPT$ROM$TYPE PUBLIC
                    DATA (CXU$PORT1,free$space$pool,
                          CXU$PORT2,free$space$pool,
                          cxu$wraparound,free$space$pool);

49    1           DECLARE LPT$RAM (CQPRTS_VALUE) LPT$RAM$TYPE PUBLIC;

               /*          request queue descriptions     */
50    1           DECLARE DCM$ROM (NUMBER_DEVICES) DCM$ROM$TYPE PUBLIC
```

```
                DATA (cxu$to$spu$rqd,4,rqe$size,
                      spu$to$cxu$rqd,4,rqe$size, /* SPU - CXU   "       "    */
                      cxu$wraparound,1,rqe$size, /* cxu - cxu                */
                      NO$SYSTEM$CHANNEL          /* cxu - 544 nonexistant    */
                     );

51   1      DECLARE DCM$RAM (NUMBER_DEVICES) DCM$RAM$TYPE PUBLIC;

            /*       god only knows!                */
52   1          DECLARE MCBISTYPE  LITERALLY 'STRUCTURE (
                   ENTRY(23)   BYTE)';

53   1      DECLARE MCBIT (NUMBER_DEVICES) MCBI$TYPE PUBLIC;

            /*       multibus interrupt level?     */
54   1      DECLARE cqsglv BYTE PUBLIC
                 DATA (6); /* ? can be any value from 0 to 7 depending on MMX int level */

55   1      DECLARE RQL6EX (28) BYTE EXTERNAL;

56   1      DECLARE cqlmex POINTER PUBLIC DATA (@RQL6EX);

57   1      DECLARE cqidpd WORD PUBLIC
                 DATA (25);

58   1      DECLARE SFT (NUMBER_DEVICES) SFTSTYPE PUBLIC
                  DATA (PEER$DEVICE,                                  /* spu */
                        IO$INTERRUPT,
                        clr$mb$intr,
                        no$intr$cleared,
                        IO$WRITE$CLR,
                        set$mb$intr,
                        0,
                    peer$device,                                      /* CXU  */
                        no$interrupt,
                        zilch,
                        zilch,
                        no$intr$cleared,
                        zilch,
                        0,
                    NO$SYSTEM$SERVICE                                 /* 544   */
                   );

59   1      DECLARE cqidss BYTE PUBLIC
               DATA (2);

60   1      DECLARE IDST (2) IDS$TYPE PUBLIC
               DATA (0,4,          /* ONBOARD AND SPU - CXU IDS  */
                     0C000H,000EH  /* SPU - 544 IDS               */
                    );

61   1      DECLARE cqplhs BYTE PUBLIC
               DATA (2);

62   1      DECLARE PLHTBL (2) POOL$TABLE$TYPE PUBLIC;

63   1      DECLARE cqblks BYTE PUBLIC
```

```
                  DATA (2);

  64   1        DECLARE BLKTBL (2) BLOCKSTABLESTYPE PUBLIC
                     DATA (0,CXU$POOL$TOKEN,CXU$POOL$LENGTH,
                           1,i544$POOL$TOKEN,i544$POOL$LENGTH
                          );

  65   1      END R3CNFG;


MODULE INFORMATION:

     CODE AREA SIZE     = 0089H    137D
     CONSTANT AREA SIZE = 0000H      0D
     VARIABLE AREA SIZE = 0136H    310D
     MAXIMUM STACK SIZE = 0000H      0D
     367 LINES READ
     0 PROGRAM WARNINGS
     0 PROGRAM ERRORS

END OF PL/M-86 COMPILATION
```