LDL07F18A

LDLABO

2008 年 1 月 1 日

目次

図目次

表目次

第 1 章

概要

1.1 コマンド

1.1.1 メモリ表示

コマンド名は **D** です。この後に空白を挟(はさ)んで要素を 2 個続けて改行を置きます。要素と要素の区切りは**','** を使います。1 番目の要素は開始の番地を 2 番目の要素は終了の番地を 16 進数で示します。

```
>D 1234,5678
1234 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
1244 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
1254 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
1264 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
5644 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
5654 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
5664 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
5674 00 00 00 00 00
>
```

1 行に最大で 16 個のメモリの内容を終了の番地まで表示します。

1.1.2 メモリ変更

コマンド名は **S** です。この後に空白を挟(はさ)んで要素を 1 個続けて改行します。要素は開始の番地を 16 進数で示します。

```
>S 1234
1234 (00) -> 56
1235 (00) -> 78
1236 (00) -> .
>
```

以降、メモリの内容が表示されて入力待ちになるので、変更内容を入力して改行します。変更しない場合は改行のみ行います。終了する場合は**'.'** を入力します。

1.1.3 インテルヘクスファイル受信

コマンド名は **HEX** です。

```
>HEX
>
```

インテルヘクスファイルを受信してバスのメモリに書き込みます。

1.1.4 バスマスタ初期化

コマンド名は **RES** です。

```
>RES
>
```

1.1.5 EEPROM 読み出し

コマンド名は **EEPR** です。範囲を入力して改行します。

```
>EEPR 2
>
```

EEPROM から**範囲**で指定された領域を読み出してバスのメモリに書き込みます。

1.1.6 EEPROM 書き込み

コマンド名は **EEPW** です。範囲を入力して改行します。

```
>EEPW 2
>
```

バスのメモリから**範囲**で指定された領域を読み出して EEPROM に書き込みます。

1.2　文字入力機構

図 1.1　文字入力機構

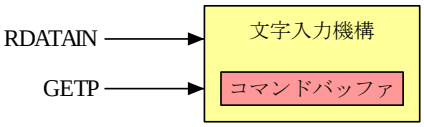
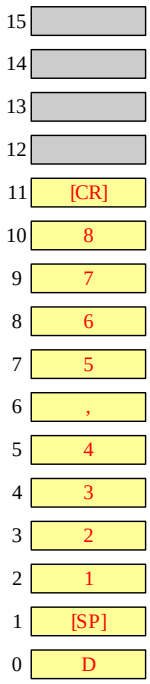


図 1.2　コマンドバッファ



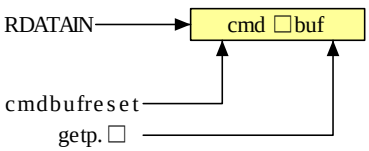
1.2.1　本体

ASCII コードの 1 文字を収容する 8 ビットのレジスタを 16 個使ってコマンドバッファの cmd0buf~cmd15buf を構成します。図 1.2 は `D 1234,5678` を収容した例です。

1.2.2　文字入力制御

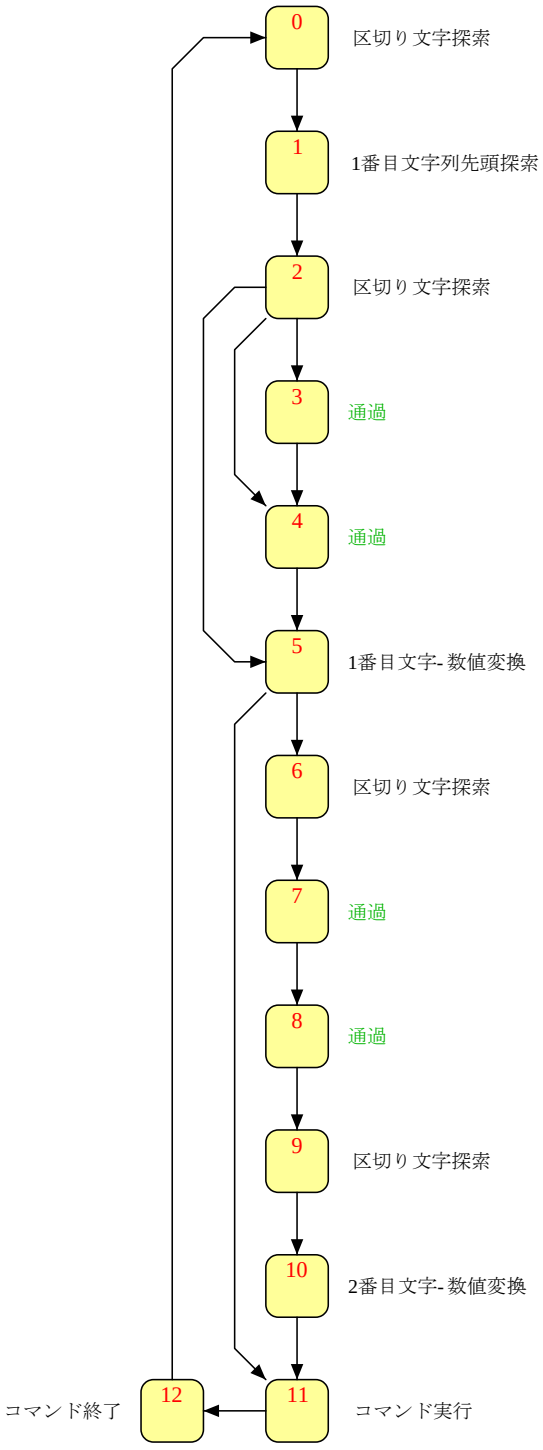
RDATAIN に受信データが確定すると GETP に起点がでます。受信データをどのバッファに記憶するかは point で決定します。

図 1.3　コマンドバッファ 1 個



getp.□ (□=0~15) は point の値によって選択した位置に GETP の起点を出力します。

図 1.4　主行程 (sequence)



1.3　コマンド解析機構

1.3.1　コマンドの検出

受信文字はバッファの先頭から収容されていきます。コマンドはバッファの先頭からコマンド名に該当する文字が配置されて区切り文字が続いていることで認識します。どのコマンドで認識したかは cmdno で示します。

1.3.2　要素の取り出し

コマンドの文字列は改行により確定とみなします。この後に要素を取り出します。コマンド文字列と主行程の図 1.4 との関係は図 1.5 のようになります。
要素を取り出すとき図 1.6 のように位置と文字数を検出します。
コマンドバッファの文字列から図 1.7 のように count を取り出しの位置にして pm0data に数値化します。

1.3.3　コマンド実行

文字が入力されているときは文字の入力行程です、改行によってコマンド解析行程に移ります。cmdinto が 1 のときコマンド解析行程に移ったことを示し、さらに主行程の sequence が 11 のときコマンド実行に移ったことを示します。各行程と指標との関係は図 1.8 のようになっています。

図 1.5　主行程とコマンド文字列

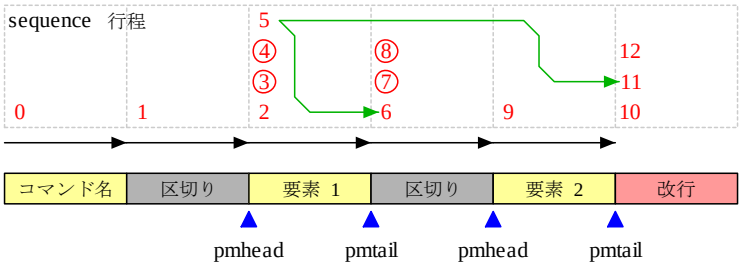


図 1.6　要素の位置と文字数

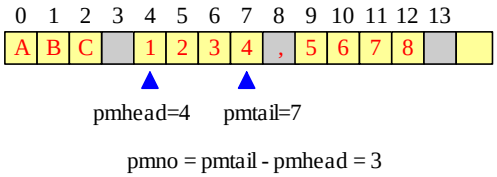


図 1.7　要素の取り出しと数値化

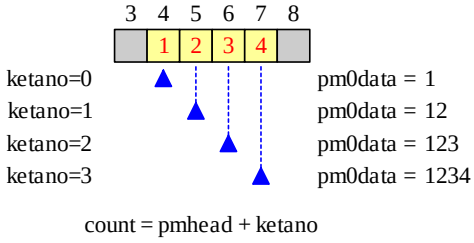
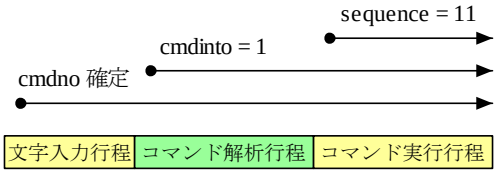
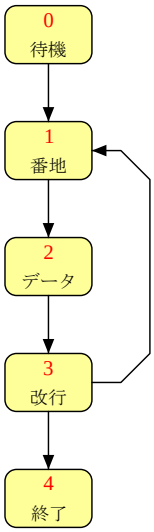


図 1.8　各行程と信号指標



1.4 メモリ表示コマンド

図 1.9 dump 行程 (dumpseq)



メモリの開始番地は pm0data に終了番地は pm1data に示します。dump 行程と表示位置の関係は図 1.10 のようになります。dump の実行における 1 文字づつの送り出しは図 1.11 のようになります。

図 1.10 dump 行程と表示位置

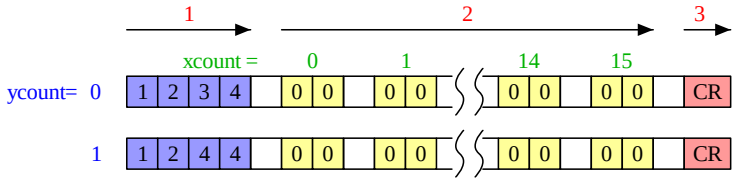
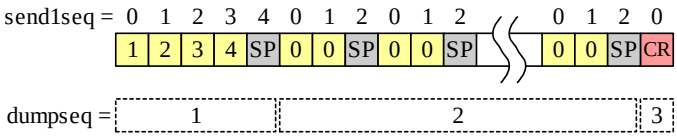


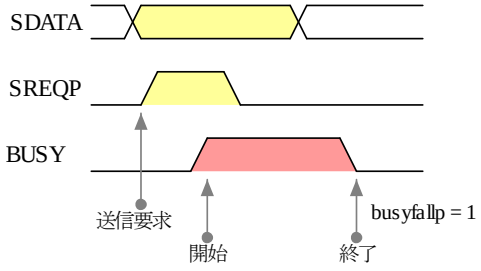
図 1.11 dump の実行と信号指標



1.5 データ送信

文字受信時のエコーやコマンド実行時の文字の送信行程は図 1.12 のようになります。BUSY が 1 であれば送信中を示し busyfallp が 1 であれば送信終了を示します。

図 1.12 送信行程



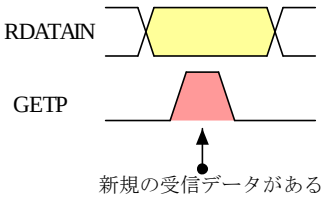
1.6　メモリ変更コマンド

メモリの番地の開始位置は pm0data に示します。コマンドの実行は図 1.16 のようになります。

1.7　データ受信

GETP が 1 のときにデータが受信されたことを示します。

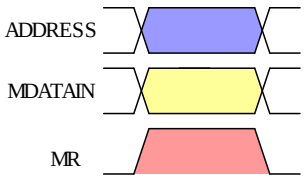
図 1.13　データ受信



1.8　メモリ読み出し

メモリのデータは MR が 1 のときに memdata に記憶します。

図 1.14　メモリ読み出し



1.9　メモリ書き込み

図 1.15　メモリ書き込み

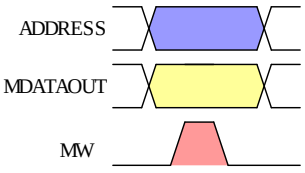
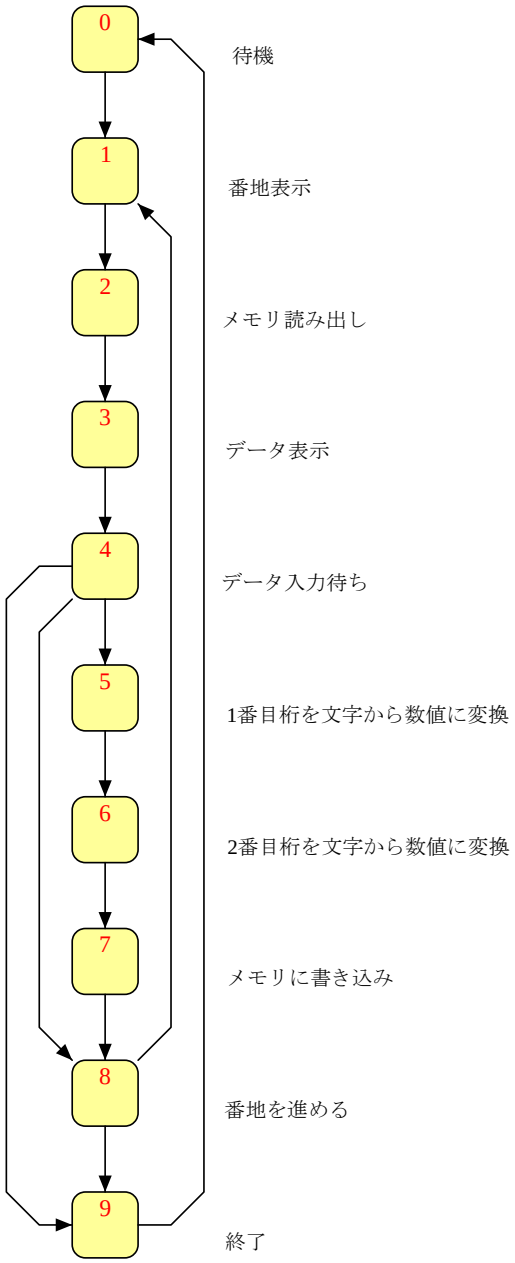


図 1.16　setm 行程 (setmseq)



第 2 章

信号

2.1 信号表

名称	意味	場所
cpureset	CPU 初期化	↪ p.16
point	コマンドバッファ位置	↪ p.18
cmd$x$buf	コマンドバッファ x	↪ p.18
getp	コマンドバッファ記憶	↪ p.18
cmdinto	コマンド実行中	↪ p.20
cmdno	コマンド番号	↪ p.21
pmhead	パラメータ先頭	↪ p.22
pmtail	パラメータ後備	↪ p.22
data	バッファ取り出し	↪ p.23
sequence	主行程	↪ p.23
count	パラメータ読み出し位置	↪ p.25
attr	パラメータ読み出し文字属性	↪ p.25
ketano	パラメータ読み出し桁位置	↪ p.25
pm0data	パラメータ 1 番目数値化	↪ p.27
pm1data	パラメータ 2 番目数値化	↪ p.27
no	数値	↪ p.26
pm0in	パラメータ 1 番目読み出し	↪ p.27
pm1in	パラメータ 2 番目読み出し	↪ p.27
dumpseq	dump 行程	↪ p.28
mr	メモリ読み出し	↪ p.30
address	番地	↪ p.30
xcount	dump 横位置	↪ p.28
ycount	dump 縦位置	↪ p.28
offset1p	番地要素 1	↪ p.30
offset2p	番地要素 2	↪ p.30
sdata	送信データ	↪ p.31
sreqp	送信要求	↪ p.32

名称	意味	場所
busyfallp	送信終了起点	↪ p.34
send1seq	送信行程	↪ p.34
send1seq3p	送信行程 3	↪ p.35
dumpseqdelay	dumpseq 遅延	↪ p.35
dumpseqchg	dumpseq 遷移起点	↪ p.35
cmdreset	コマンド終了による初期化	↪ p.20
cmdbufreset	コマンドバッファ初期化	↪ p.19
setmseq	setm 行程	↪ p.29
addresscount	番地計数	↪ p.27
memdata	メモリ読み出しデータ	↪ p.23
pointreset	コマンドバッファ位置初期化	↪ p.18
crdetect	改行検出	↪ p.??
mojino	文字数	↪ p.19
mw	メモリ書き込み	↪ p.31
echo	受信エコー	↪ p.17
sum	HEX ファイル照合結果	↪ p.16
readpara	引数読み出し実行許可	↪ p.22
pmno	パラメータ文字数	↪ p.25
hexres	HEX 初期化	↪ p.36
busreq	バス要求	↪ p.36
rop	EEPROM 読み出し	↪ p.15
wop	EEPROM 書き込み	↪ p.15
blksel	EEPROM 操作範囲	↪ p.16
crst	改行監視	↪ p.??
cmdwaitmark	コマンド待ち指標	↪ p.14
cpuiniload	CPU 初期プログラム設置	↪ p.15

名称	意味	場所
intvtime	間隔タイマ	↪ p.14
atdetect	@検出	↪ p.??
debugcount	コマンドバッファ位置	↪ p.??
debugdata	コマンドバッファ内容	↪ p.??
debugseq	コマンドバッファ表示進行	↪ p.??
busyfallpdelay	送信終了起点遅延	↪ p.34
dumppagecount	dump 頁計数	↪ p.30
sdata1op	送信データ条件 1	↪ p.31
sdata2op	送信データ条件 2	↪ p.31
send1seq1op	送信行程条件 1	↪ p.34
count1op	要素読み出し位置条件 1	↪ p.25
pmtail1op	要素後尾条件 1	↪ p.22
ketano1op	要素読み出し桁位置条件 1	↪ p.25
pmhead1op	要素先頭 +1	↪ p.22

第 3 章

論理譜

```
{ ========================================================================== }
{    モニタアダプタ                                                        }
{ ========================================================================== }
logicname LDL07F18A

{ ========================================================================== }
{    コマンドバッファ                                                      }
{ ========================================================================== }
procedure cmdbufp
input  RESET;
input  GETP;
input  DATAIN[8];
output Q[8];

bitr cmdbuf[8];

   Q=cmdbuf;

   if (RESET)
      cmdbuf=0;
   else
      if (GETP)
         cmdbuf=DATAIN;
      else
         cmdbuf=cmdbuf;
      endif
   endif

endp

{ ========================================================================== }
{    実効譜                                                                }
{ ========================================================================== }
entity LDL07F18A

{ -------------------------------------------------------------------------- }
{    入力                                                                  }
{ -------------------------------------------------------------------------- }
input RESET;          { 初期化 }
input GETP;           { 受信データ取得起点 }
input RDATAIN[8];     { 受信データ }
input MDATAIN[8];     { メモリデータ }
input BUSY;           { 送信中指標 }
input EOF;            { HEX ファイルロード終了 }
input SUM;            { HEX ファイル照合結果 }
input EOEPP;          { EEPROM 操作終了 }
input BUSACK;         { バス許可 }
```

```
input RE;               { 読み出し有効 }
input WE;               { 書き込み有効 }
input TEST;             { 検証端子 }

{ ------------------------------------------------------------------------ }
{    出力                                                          }
{ ------------------------------------------------------------------------ }
output   MR;            { メモリ読み出し }
output   MW;            { メモリ書き込み }
output   ADDRESS[16];   { 番地 }
output   MDATAOUT[8];   { メモリデータ }
output   SDATA[8];      { 送信データ }
output   SREQP;         { 送信指示起点 }
output   CMDNO[3];      { コマンド番号 }
output   CPURESET;      { CPU 初期化 }
output   HEXRES;        { HEX 初期化 }
output   BUSREQ;        { バス要求 }
output   ROP;           { EEPROM 読み出し }
output   WOP;           { EEPROM 書き込み }
output   BLKSEL[4];     { EEPROM 操作範囲 }

{ ------------------------------------------------------------------------ }
{    内部信号                                                      }
{ ------------------------------------------------------------------------ }
bitn cmd0buf[8];         { コマンドバッファ 0 }
bitn cmd1buf[8];         { コマンドバッファ 1 }
bitn cmd2buf[8];         { コマンドバッファ 2 }
bitn cmd3buf[8];         { コマンドバッファ 3 }
bitn cmd4buf[8];         { コマンドバッファ 4 }
bitn cmd5buf[8];         { コマンドバッファ 5 }
bitn cmd6buf[8];         { コマンドバッファ 6 }
bitn cmd7buf[8];         { コマンドバッファ 7 }
bitn cmd8buf[8];         { コマンドバッファ 8 }
bitn cmd9buf[8];         { コマンドバッファ 9 }
bitn cmd10buf[8];        { コマンドバッファ 10 }
bitn cmd11buf[8];        { コマンドバッファ 11 }
bitn cmd12buf[8];        { コマンドバッファ 12 }
bitn cmd13buf[8];        { コマンドバッファ 13 }
bitn cmd14buf[8];        { コマンドバッファ 14 }
bitn cmd15buf[8];        { コマンドバッファ 15 }

bitr point[4];           { コマンドバッファ位置 }
bitn getp[16];           { コマンドバッファ記憶 }
bitn cmdinto;            { コマンド実行中 }
bitn cmdx[4];            { コマンド番号候補 }
bitr cmdv[4];            { コマンド番号 }
bitr cmdno[4];           { 有効コマンド番号 }
bitr pmhead[4];          { 要素先頭 }
bitn pmhead1op;          { 要素先頭 +1 }
bitr pmtail[4];          { 要素後尾 }
bitn data[8];            { バッファ取り出し }
bitr sequence[4];        { 主行程 }
bitn count[4];           { 要素読み出し位置 }
bitn attr[4];            { 要素読み出し文字属性 }
bitr ketano[4];          { 要素読み出し桁位置 }
bitr pm0data[16];        { 要素 1 番目数値化 }
bitr pm1data[16];        { 要素 2 番目数値化 }
bitn no[8];              { 数値 }
bitn pm0in;              { 要素 1 番目読み出し }
bitn pm1in;              { 要素 2 番目読み出し }
bitr dumpseq[3];         { dump 行程 }
```

```
bitn mr;                  { メモリ読み出し }
bitn address[16];         { 番地 }
bitr xcount[5];           { dump 横位置 }
bitr ycount[5];           { dump 縦位置 }
bitn offset1p[16];        { 番地要素 1 }
bitn offset2p[16];        { 番地要素 2 }
bitn sdata[8];            { 送信データ }
bitr sreqp;               { 送信要求 }
bitr busyfallp[3];        { 送信終了起点 }
bitr send1seq[4];         { 送信行程 }
bitn send1seq3p;          { 送信行程 3 }
bitr dumpseqdelay[3];     { dumpseq 遅延 }
bitn dumpseqchg;          { dumpseq 遷移起点 }
bitn cmdreset;            { コマンド終了による初期化 }
bitn cmdbufreset;         { コマンドバッファ初期化 }
bitr setmseq[4];          { setm 行程 }
bitr addresscount[16];    { 番地計数 }
bitr memdata[8];          { メモリ読み出しデータ }
bitn pointreset;          { コマンドバッファ位置初期化 }
bitn crdetect[16];        { 改行検出 }
bitn mojino[4];           { 文字数 }
bitn mw;                  { メモリ書き込み }
bitn echo;                { 受信エコー }
bitn invecho;             { 否定受信エコー }
bitr sum;                 { HEX ファイル照合結果 }
bitn cpureset;            { CPU 初期化 }
bitn readpara;            { 引数読み出し実行許可 }
bitn pmno[4];             { 要素文字数 }
bitn hexres;              { HEX 初期化 }
bitn busreq;              { バス要求 }
bitr rop[2];              { EEPROM 読み出し }
bitr wop[2];              { EEPROM 書き込み }
bitn blksel[4];           { EEPROM 操作範囲 }
bitr cpuiniload;          { CPU 初期プログラム設置 }
bitr cmdwaitmark[4];      { コマンド待ち指標 }
bitr lfst[2];             { LF 監視 }
bitr intvtime[4];         { 間隔タイマ }
bitr atdetect;            { @検出 }
bitr busyfallpdelay[4];   { 送信終了起点遅延 }
bitr dumppagecount[8];    { dump 頁計数 }
bitn sdata1op;            { 送信データ条件 1 }
bitn sdata2op;            { 送信データ条件 2 }
bitn send1seq1op;         { 送信行程条件 1 }
bitn count1op;            { 要素読み出し位置条件 1 }
bitn pmtail1op;           { 要素後尾条件 1 }
bitn ketano1op;           { 要素読み出し桁位置条件 1 }
bitr lfseq[3];            { 行送り行程 }

{ ------------------------------------------------------------------------ }
{    出力代入                                                            }
{ ------------------------------------------------------------------------ }
   ADDRESS=address;
   MDATAOUT=memdata;
   MR=mr;
   MW=mw;
   SDATA=sdata;
   SREQP=sreqp;

   switch(cmdno)
      case 4: CMDNO=cmdno.0:2;
      case 7: CMDNO=cmdno.0:2;
```

```
      default:
         if (BUSACK)
            CMDNO=cmdno.0:2;
         endif
   endswitch

   CPURESET=cpureset;
   HEXRES=hexres;
   BUSREQ=busreq;
   ROP=rop.0;
   WOP=wop.0;
   BLKSEL=blksel;

{ ------------------------------------------------------------------------ }
{    間隔タイマ                                                          }
{ ------------------------------------------------------------------------ }
   if (RESET|BUSY)
      if (TEST)
         intvtime=0;
      else
         intvtime=10;
      endif
   else
      switch(intvtime)
         case 0:  intvtime=intvtime;
         default: intvtime=intvtime-1;
      endswitch
   endif

{ ------------------------------------------------------------------------ }
{    LF 監視                                                             }
{ ------------------------------------------------------------------------ }
   if (RESET)
      lfst=0;
   else
      switch(lfst)
         case 0:
            switch(sdata)
               case 0x0a: if (sreqp) lfst=1; endif
            endswitch
         case 1:
            if (BUSY) lfst=2; else lfst=lfst; endif
         case 2:
            if (BUSY) lfst=lfst; else lfst=3; endif
         case 3:
            switch(sdata)
               case 0x0a:
                  if (sreqp)
                     lfst=1;
                  else
                     lfst=lfst;
                  endif
               default:
                  if (sreqp)
                     lfst=0;
                  else
                     lfst=lfst;
                  endif
            endswitch
      endswitch
   endif

{ ------------------------------------------------------------------------ }
{    コマンド待ち指標                                                    }
{ ------------------------------------------------------------------------ }
```

```
   if (RESET)
      cmdwaitmark=0;
   else
      switch(cmdwaitmark)
         case 0:
            if (cmdreset)
               switch(cmdno)
                  case 1:  cmdwaitmark=2;
                  case 3:  cmdwaitmark=2;
                  case 7:  cmdwaitmark=2;
                  default: cmdwaitmark=1;
               endswitch
            endif
         case 1:                              { 行送りの送信待ち }
            switch(lfst)
               case 3:  cmdwaitmark=2;
               default: cmdwaitmark=cmdwaitmark;
            endswitch
         case 2: cmdwaitmark=4;               { コマンドプロンプト sreqp }
         case 4: cmdwaitmark=6;               { コマンドプロンプト sdata }
      endswitch
   endif

{ ------------------------------------------------------------------------ }
{    CPU 初期プログラム設置                                  }
{ ------------------------------------------------------------------------ }
   if (RESET)
      cpuiniload=1;
   else
      if (EOEPP)
         cpuiniload=0;
      else
         cpuiniload=cpuiniload;
      endif
   endif

{ ------------------------------------------------------------------------ }
{    EEPROM 読み出し                                      }
{ ------------------------------------------------------------------------ }
   if (RESET)
      rop=0;
   else
      switch(cmdno)
         case 6:
            switch(sequence)
               case 11:
                  switch(rop)
                     case 0:  rop=1;
                     case 1:  rop=2;
                     default: rop=rop;
                  endswitch
            endswitch
      endswitch
   endif

{ ------------------------------------------------------------------------ }
{    EEPROM 書き込み                                      }
{ ------------------------------------------------------------------------ }
   if (RESET)
      wop=0;
   else
      switch(cmdno)
         case 5:
            switch(sequence)
               case 11:
```

```
                switch(wop)
                   case 0:  wop=1;
                   case 1:  wop=2;
                   default: wop=wop;
                endswitch
             endswitch
          endswitch
       endif

{ ------------------------------------------------------------------------ }
{    EEPROM 操作範囲                                                      }
{ ------------------------------------------------------------------------ }
       if (TEST)
          blksel=15;
       else
          switch(cmdno)
             case 7:  blksel=0;
             default: blksel=pm0data.0:3;
          endswitch
       endif

{ ------------------------------------------------------------------------ }
{    CPU 初期化                                                           }
{ ------------------------------------------------------------------------ }
       switch(cmdno)
          case 4: cpureset=1;
       endswitch

{ ------------------------------------------------------------------------ }
{    チェックサム記憶                                                     }
{ ------------------------------------------------------------------------ }
       if (RESET)
          sum=0;
       else
          if (EOF)
             sum=SUM;
          else
             sum=sum;
          endif
       endif

{ ------------------------------------------------------------------------ }
{    行送り行程                                                           }
{ ------------------------------------------------------------------------ }
       if (RESET)
          lfseq=0;
       else
          switch(lfseq)
             case 0:                          { 改行待ち }
                if (echo)
                   if (GETP)
                      switch(RDATAIN)
                         case 0x0d: lfseq=1;
                      endswitch
                   endif
                endif
             case 1:                          { 改行エコー終了待ち }
                if (busyfallp.0)
                   lfseq=2;
                else
                   lfseq=lfseq;
                endif
             case 2:                          { 行送り送信 sreq }
                lfseq=3;
             case 3:                          { 行送り送信 sdata }
```

```
            lfseq=4;
         case 4:                          { 行送り送信終了待ち }
            if (busyfallp.0)
               lfseq=5;
            else
               lfseq=lfseq;
            endif
         case 5:                          { 行送り送信終了 }
            lfseq=0;
      endswitch
   endif

{ ------------------------------------------------------------------------ }
{    受信エコー                                                            }
{ ------------------------------------------------------------------------ }
   switch(cmdno)
      case 3:  echo=0;              { HEX }
      default: echo=!invecho;
   endswitch

{ ------------------------------------------------------------------------ }
{    否定受信エコー                                                        }
{ ------------------------------------------------------------------------ }
   switch(setmseq)
      case 1:                     { 番地 }
         switch(send1seq)
            case 0:
            case 1:
            case 2:
            case 3:
            from: invecho=1;
         endswitch

         switch(send1seq)
            case 4: invecho=1;
         endswitch
      case 3:                        { データ表示 }
         invecho=1;
   endswitch

   switch(dumpseq)
      case 1:                         { 番地 }
         switch(send1seq)
            case 0:
            case 1:
            case 2:
            case 3:
            from: invecho=1;
         endswitch

         switch(send1seq)
            case 4: invecho=1;
         endswitch
      case 2:                          { データ }
         switch(send1seq)
            case 0:
            case 1:
            from: invecho=1;
         endswitch

         switch(send1seq)
            case 2: invecho=1;
         endswitch

      case 3:                          { 改行 }
```

```
        invecho=1;
   endswitch

{ ------------------------------------------------------------------------ }
{    コマンドバッファ位置初期化                                          }
{ ------------------------------------------------------------------------ }
   switch(setmseq)
      case 3: pointreset=1;
   endswitch

{ ------------------------------------------------------------------------ }
{    コマンドバッファ位置                                                }
{ ------------------------------------------------------------------------ }
   if (RESET|cmdreset|pointreset)
      point=0;
   else
      if (GETP)
         switch(RDATAIN)
            case 0x08: point=point-1;   { BS }
            default:   point=point+1;
         endswitch
      else
         point=point;
      endif
   endif

{ ------------------------------------------------------------------------ }
{    コマンドバッファ記憶                                                }
{ ------------------------------------------------------------------------ }
   if (GETP)
      switch(RDATAIN)
         case 0x08:              { BS }
         default:
            switch(point)
               case 0:  getp.0=1;
               case 1:  getp.1=1;
               case 2:  getp.2=1;
               case 3:  getp.3=1;
               case 4:  getp.4=1;
               case 5:  getp.5=1;
               case 6:  getp.6=1;
               case 7:  getp.7=1;
               case 8:  getp.8=1;
               case 9:  getp.9=1;
               case 10: getp.10=1;
               case 11: getp.11=1;
               case 12: getp.12=1;
               case 13: getp.13=1;
               case 14: getp.14=1;
               case 15: getp.15=1;
            endswitch
      endswitch
   endif

{ ------------------------------------------------------------------------ }
{    コマンドバッファ                                                    }
{ ------------------------------------------------------------------------ }
   cmd0buf=cmdbufp(cmdbufreset,getp.0,RDATAIN);
   cmd1buf=cmdbufp(cmdbufreset,getp.1,RDATAIN);
   cmd2buf=cmdbufp(cmdbufreset,getp.2,RDATAIN);
   cmd3buf=cmdbufp(cmdbufreset,getp.3,RDATAIN);
   cmd4buf=cmdbufp(cmdbufreset,getp.4,RDATAIN);
   cmd5buf=cmdbufp(cmdbufreset,getp.5,RDATAIN);
   cmd6buf=cmdbufp(cmdbufreset,getp.6,RDATAIN);
   cmd7buf=cmdbufp(cmdbufreset,getp.7,RDATAIN);
```

```
   cmd8buf=cmdbufp(cmdbufreset,getp.8,RDATAIN);
   cmd9buf=cmdbufp(cmdbufreset,getp.9,RDATAIN);
   cmd10buf=cmdbufp(cmdbufreset,getp.10,RDATAIN);
   cmd11buf=cmdbufp(cmdbufreset,getp.11,RDATAIN);
   cmd12buf=cmdbufp(cmdbufreset,getp.12,RDATAIN);
   cmd13buf=cmdbufp(cmdbufreset,getp.13,RDATAIN);
   cmd14buf=cmdbufp(cmdbufreset,getp.14,RDATAIN);
   cmd15buf=cmdbufp(cmdbufreset,getp.15,RDATAIN);

   cmd0buf.8=1;
   cmd1buf.8=1;
   cmd2buf.8=1;
   cmd3buf.8=1;
   cmd4buf.8=1;
   cmd5buf.8=1;
   cmd6buf.8=1;
   cmd7buf.8=1;
   cmd8buf.8=1;
   cmd9buf.8=1;
   cmd10buf.8=1;
   cmd11buf.8=1;
   cmd12buf.8=1;
   cmd13buf.8=1;
   cmd14buf.8=1;
   cmd15buf.8=1;

{ ------------------------------------------------------------------------ }
{    コマンドバッファ初期化                                              }
{ ------------------------------------------------------------------------ }
   if (RESET | cmdreset)
      cmdbufreset=1;
   else
      switch(setmseq)
         case 3: cmdbufreset=1;
      endswitch
   endif

{ ------------------------------------------------------------------------ }
{    改行検出                                                            }
{ ------------------------------------------------------------------------ }
   switch(cmd0buf) case 0x0d: crdetect.0=1; endswitch
   switch(cmd1buf) case 0x0d: crdetect.1=1; endswitch
   switch(cmd2buf) case 0x0d: crdetect.2=1; endswitch
   switch(cmd3buf) case 0x0d: crdetect.3=1; endswitch
   switch(cmd4buf) case 0x0d: crdetect.4=1; endswitch
   switch(cmd5buf) case 0x0d: crdetect.5=1; endswitch
   switch(cmd6buf) case 0x0d: crdetect.6=1; endswitch
   switch(cmd7buf) case 0x0d: crdetect.7=1; endswitch
   switch(cmd8buf) case 0x0d: crdetect.8=1; endswitch
   switch(cmd9buf) case 0x0d: crdetect.9=1; endswitch
   switch(cmd10buf) case 0x0d: crdetect.10=1; endswitch
   switch(cmd11buf) case 0x0d: crdetect.11=1; endswitch
   switch(cmd12buf) case 0x0d: crdetect.12=1; endswitch
   switch(cmd13buf) case 0x0d: crdetect.13=1; endswitch
   switch(cmd14buf) case 0x0d: crdetect.14=1; endswitch
   switch(cmd15buf) case 0x0d: crdetect.15=1; endswitch

{ ------------------------------------------------------------------------ }
{    改行までの文字数                                                    }
{ ------------------------------------------------------------------------ }
   switch(crdetect)
      case 0b0000000000000001: mojino=0;
      case 0b0000000000000010: mojino=1;
      case 0b0000000000000100: mojino=2;
      case 0b0000000000001000: mojino=3;
```

```
      case 0b0000000000010000: mojino=4;
      case 0b0000000000100000: mojino=5;
      case 0b0000000001000000: mojino=6;
      case 0b0000000010000000: mojino=7;

      case 0b0000000100000000: mojino=8;
      case 0b0000001000000000: mojino=9;
      case 0b0000010000000000: mojino=10;
      case 0b0000100000000000: mojino=11;

      case 0b0001000000000000: mojino=12;
      case 0b0010000000000000: mojino=13;
      case 0b0100000000000000: mojino=14;
      case 0b1000000000000000: mojino=15;

      default: mojino=16;
   endswitch

{ ------------------------------------------------------------------------ }
{    コマンド実行中                                                        }
{ ------------------------------------------------------------------------ }
   switch(cmdno)
      case 0:
      default: cmdinto=1;
   endswitch

{ ------------------------------------------------------------------------ }
{    コマンド終了による初期化                                              }
{ ------------------------------------------------------------------------ }
   switch(dumpseq)                                          { dump }
      case 4: cmdreset=1;
   endswitch

   switch(setmseq)                                           { setm }
      case 9: cmdreset=1;
   endswitch

   switch(cmdno)                                             { hex }
      case 3: if (EOF) cmdreset=1; endif
   endswitch

   switch(cmdno)                                             { res }
      case 4: if (cpureset) cmdreset=1; endif
   endswitch

   switch(cmdno)
      case 5:                                                { eepw }
      case 6:                                                { eepr }
      from: if (EOEPP) cmdreset=1; endif
   endswitch

   if (cpuiniload)
      if (EOEPP) cmdreset=1; endif                           { CPU 初期プログラム設置 }
   endif

   if (GETP)
      switch(RDATAIN)
         case 0x0d:
            switch(cmdno)
               case 0:
                  switch(cmdx)
                     case 0: cmdreset=1;                    { 該当コマンドなし }
                  endswitch
               case 1: cmdreset=1;                          { dump 中断 }
```

```
               case 2: cmdreset=0;                { setm 中は無効 }
               case 5: cmdreset=1;                { EEPW 中断 }
               case 6: cmdreset=1;                { EEPR 中断 }
               case 7: cmdreset=1;                { CPU 初期プログラム設置中断 }
            endswitch
         case 0x2e:
            switch(cmdno)
               case 3: cmdreset=1;                { HEX 中断 }
            endswitch
      endswitch
   endif

{ ------------------------------------------------------------------- }
{    コマンド番号候補                                                 }
{ ------------------------------------------------------------------- }
   switch(cmd0buf,cmd1buf)
      case 0x44,0x20: cmdx=1;                  { D : dump }
      case 0x53,0x20: cmdx=2;                  { S : setm }
   endswitch

   switch(cmd0buf,cmd1buf,cmd2buf,cmd3buf)
      case 0x48,0x45,0x58,0x00: cmdx=3;        { HEX : hex }
      case 0x52,0x45,0x53,0x00: cmdx=4;        { RES : res }
   endswitch

   switch(cmd0buf,cmd1buf,cmd2buf,cmd3buf,cmd4buf)
      case 0x45,0x45,0x50,0x57,0x20: cmdx=5;   { EEPW : eepw }
      case 0x45,0x45,0x50,0x52,0x20: cmdx=6;   { EEPR : eepr }
   endswitch

{ ------------------------------------------------------------------- }
{    有効コマンド番号                                                 }
{ ------------------------------------------------------------------- }
   if (RESET|cmdreset)
      cmdno=0;
   else
      if (cpuiniload)
         cmdno=7;
      else
         switch(cmdno)
            case 0:
               switch(lfseq)
                  case 5: cmdno=cmdv;
                  default: cmdno=cmdno;
               endswitch
            default: cmdno=cmdno;
         endswitch
      endif
   endif

{ ------------------------------------------------------------------- }
{    コマンド番号                                                     }
{ ------------------------------------------------------------------- }
   if (RESET|cmdreset)
      cmdv=0;
   else
      if (GETP)
         switch(RDATAIN)
            case 0x0d:
               switch(cmdv)
                  case 0:  cmdv=cmdx;
                  default: cmdv=cmdv;
            endswitch
            default: cmdv=cmdv;
         endswitch
```

```
      else
         cmdv=cmdv;
      endif
   endif

{ ------------------------------------------------------------------------ }
{    要素読み出し                                                          }
{ ------------------------------------------------------------------------ }
   switch(cmdno)
      case 1:              { dump }
      case 2:              { setm }
      case 5:              { eepw }
      case 6:              { eepr }
      from: readpara=1;
   endswitch

{ ------------------------------------------------------------------------ }
{    要素先頭                                                              }
{ ------------------------------------------------------------------------ }
   if (RESET|cmdreset|!readpara)
      pmhead=0;
   else
      if (cmdinto)
         switch(sequence)
            case 0:                          { 区切り文字探索 }
               switch(attr)
                  case 1:  pmhead=pmhead;     { SP }
                  default: pmhead1op=1;
               endswitch

            case 1:                          { 1 番目文字列先頭探索 }
               switch(attr)
                  case 4:  pmhead=pmhead;     { A-Z,a-z }
                  case 5:  pmhead=pmhead;     { 0-9 }
                  default: pmhead1op=1;
               endswitch

            case 6:                          { 区切り文字探索 }
               switch(attr)
                  case 1:  pmhead=pmhead;     { SP }
                  case 2:  pmhead=pmhead;     { , }
                  default: pmhead1op=1;
               endswitch

            case 7:                          { 2 番目文字列先頭探索 }
               switch(attr)
                  case 4:  pmhead=pmhead;     { A-Z,a-z }
                  case 5:  pmhead=pmhead;     { 0-9 }
                  default: pmhead1op=1;
               endswitch

            default: pmhead=pmhead;
         endswitch
      else
         pmhead=0;
      endif
   endif

   if (pmhead1op) pmhead=pmhead+1; endif

{ ------------------------------------------------------------------------ }
{    要素後尾                                                              }
{ ------------------------------------------------------------------------ }
   if (RESET|cmdreset|!readpara)
      pmtail=0;
```

```
   else
      switch(sequence)
         case 1: pmtail=pmhead;              { 1 番目文字列先頭探索 }
         case 2: pmtail1op=1;                { 区切り文字探索 }
         case 3: pmtail=pmtail-2;            {  }
         case 6: pmtail=pmhead;              { 区切り文字探索 }
         case 7: pmtail=pmhead;              {  }
         case 8: pmtail=pmhead;              {  }
         case 9: pmtail1op=1;                { 区切り文字探索 }
         default: pmtail=pmtail;
      endswitch
   endif

   if (pmtail1op) pmtail=pmtail+1; endif

{ ------------------------------------------------------------------------ }
{    バッファ取り出し                                          }
{ ------------------------------------------------------------------------ }
   switch(count)
      case 0:  data=cmd0buf;
      case 1:  data=cmd1buf;
      case 2:  data=cmd2buf;
      case 3:  data=cmd3buf;
      case 4:  data=cmd4buf;
      case 5:  data=cmd5buf;
      case 6:  data=cmd6buf;
      case 7:  data=cmd7buf;
      case 8:  data=cmd8buf;
      case 9:  data=cmd9buf;
      case 10: data=cmd10buf;
      case 11: data=cmd11buf;
      case 12: data=cmd12buf;
      case 13: data=cmd13buf;
      case 14: data=cmd14buf;
      case 15: data=cmd15buf;
   endswitch

{ ------------------------------------------------------------------------ }
{    メモリ書き込みデータ                                      }
{ ------------------------------------------------------------------------ }
   if (RESET)
      memdata=0;
   else
      if (RE)
         memdata=MDATAIN;
      else
         switch(setmseq)
            case 5:                          { 1 番目桁を文字から数値に変換 }
                     memdata.4:7=no.0:3;
                     memdata.0:3=memdata.0:3;
            case 6:                          { 2 番目桁を文字から数値に変換 }
                     memdata.4:7=memdata.4:7;
                     memdata.0:3=no.0:3;
            default: memdata=memdata;
         endswitch
      endif
   endif

{ ------------------------------------------------------------------------ }
{    主行程                                                }
{ ------------------------------------------------------------------------ }
   if (RESET|cmdreset|!readpara)
      sequence=0;
   else
      switch(sequence)
```

```
      case 0:                                  { 区切り文字探索 }
          switch(attr)
            case 1:  sequence=1;               { SP }
            default: sequence=sequence;
         endswitch

      case 1:                                  { 1 番目文字列先頭探索 }
          switch(attr)
            case 4:  sequence=2;               { A-Z,a-z }
            case 5:  sequence=2;               { 0-9 }
            default: sequence=sequence;
         endswitch

      case 2:                                  { 区切り文字探索 }
         switch(attr)
            case 4:  sequence=sequence;        { A-Z,a-z }
            case 5:  sequence=sequence;        { 0-9 }
            default: sequence=3;
         endswitch
      case 3: sequence=4;                      {  }
      case 4: sequence=5;                      {  }
      case 5:                                  { 1 番目文字-数値変換 }
         if (pmno==ketano)
            switch(cmdno)
               case 1: sequence=6;             { dump }
               case 2: sequence=11;            { setm }
               case 5: sequence=11;            { eepw }
               case 6: sequence=11;            { eepr }
               default: sequence=sequence;
            endswitch
         else
            sequence=sequence;
         endif
      case 6:                                  { 区切り文字探索 }
          switch(attr)
            case 4:  sequence=sequence;        { A-Z,a-z }
            case 5:  sequence=sequence;        { 0-9 }
            default: sequence=7;
         endswitch
      case 7: sequence=8;                      {  }
      case 8: sequence=9;                      {  }
      case 9:                                  { 区切り文字探索 }
          switch(attr)
             case 1:  sequence=10;             { SP }
             case 2:  sequence=10;             { , }
             case 3:  sequence=10;             { CR }
             default: sequence=sequence;
          endswitch
      case 10:                                 { 2 番目文字-数値変換 }
         if (pmno==ketano)
            sequence=11;
         else
            sequence=sequence;
         endif
      case 11:                                 { コマンド実行 }
         if (cmdreset)
            sequence=12;
         else
            sequence=sequence;
         endif
      case 12:                                 { コマンド終了 }
         sequence=0;
      default: sequence=sequence;
   endswitch
endif
```

```
{ ------------------------------------------------------------------------ }
{    要素読み出し位置                                          }
{ ------------------------------------------------------------------------ }
   switch(setmseq)
      case 5:
         switch(mojino)
            case 1:  count=15;
            default: count=0;
         endswitch
      case 6:
         switch(mojino)
            case 1:  count=0;
            default: count=1;
         endswitch
      default:
         switch(sequence)
            case 0:  count=pmhead;            { 区切り文字探索 }
            case 1:  count=pmhead;            { 1 番目文字列先頭探索 }
            case 2:  count=pmtail;            { 区切り文字探索 }
            case 4:  count=pmhead;            {  }
            case 5:  count1op=1;              { 1 番目文字-数値変換 }
            case 6:  count=pmhead;            { 区切り文字探索 }
            case 7:  count=pmhead;            {  }
            case 8:  count=pmhead;            {  }
            case 9:  count=pmtail;            { 区切り文字探索 }
            case 10: count1op=1;              { 2 番目文字-数値変換 }
         endswitch
   endswitch

   if (count1op) count=pmhead+ketano; endif

{ ------------------------------------------------------------------------ }
{    要素読み出し文字属性                                        }
{ ------------------------------------------------------------------------ }
   switch(data)
      case 0x20: attr=1;   { SP }
      case 0x2c: attr=2;   { , }
      case 0x0d: attr=3;   { CR }
   endswitch

   if ((data>=0x41)&(data<=0x5a)) attr=4; endif { A-Z }
   if ((data>=0x61)&(data<=0x7a)) attr=4; endif { a-z }
   if ((data>=0x30)&(data<=0x39)) attr=5; endif { 0-9 }

{ ------------------------------------------------------------------------ }
{    要素文字数                                           }
{ ------------------------------------------------------------------------ }
   switch(sequence)
      case 5:  pmno=pmtail-pmhead;              { 1 番目文字-数値変換 }
      case 10: pmno=pmtail-pmhead-2;            { 2 番目文字-数値変換 }
   endswitch

{ ------------------------------------------------------------------------ }
{    要素読み出し桁位置                                          }
{ ------------------------------------------------------------------------ }
   if (RESET|cmdreset|!readpara)
      ketano=0;
   else
      switch(sequence)
         case 4:  ketano=0;                 {  }
         case 5:  ketano1op=1;              { 1 番目文字-数値変換 }
         case 9:  ketano=0;                 { 区切り文字探索 }
```

```
         case 10: ketano1op=1;              { 2 番目文字-数値変換 }
         default: ketano=ketano;
      endswitch
   endif

   if (ketano1op) ketano=ketano+1;
 endif

{ ------------------------------------------------------------------------ }
{    数値                                                                  }
{ ------------------------------------------------------------------------ }
{ dump }
   switch(dumpseq)
      case 1:
         switch(send1seq)
            case 0: no.0:3=address.12:15;   { 番地 1 桁 }
            case 1: no.0:3=address.8:11;    { 番地 2 桁 }
            case 2: no.0:3=address.4:7;     { 番地 3 桁 }
            case 3: no.0:3=address.0:3;     { 番地 4 桁 }
         endswitch

      case 2:
         switch(send1seq)
            case 0: no.0:3=MDATAIN.4:7;     { 値 1 桁 }
            case 1: no.0:3=MDATAIN.0:3;     { 値 2 桁 }
         endswitch
   endswitch

{ setm }
   switch(setmseq)
      case 1:
         switch(send1seq)
            case 0: no.0:3=address.12:15;   { 番地 1 桁 }
            case 1: no.0:3=address.8:11;    { 番地 2 桁 }
            case 2: no.0:3=address.4:7;     { 番地 3 桁 }
            case 3: no.0:3=address.0:3;     { 番地 4 桁 }
         endswitch
      case 3:
         switch(send1seq)
            case 1: no.0:3=memdata.4:7;     { 値 1 桁 }
            case 2: no.0:3=memdata.0:3;     { 値 2 桁 }
         endswitch
   endswitch

   switch(setmseq)
      case 5:                       { 1 番目桁を文字から数値に変換 }
      case 6:                       { 2 番目桁を文字から数値に変換 }
      from:
         switch(attr)
            case 4:
               if (data>=0x61)
                  no=data-0x57;
               else
                  no=data-0x37;
               endif
            case 5: no=data-0x30;
         endswitch
   endswitch

   switch(sequence)
      case 11:
      default:                      { 全コマンド共通の要素読み出し }
         switch(attr)
            case 4:
```

```
            if (data>=0x61)
               no=data-0x57;
            else
               no=data-0x37;
            endif
         case 5: no=data-0x30;
      endswitch
   endswitch

{ ------------------------------------------------------------------------ }
{    要素 1 番目読み出し                                                  }
{ ------------------------------------------------------------------------ }
   switch(sequence)
      case 5: pm0in=1;      { 1 番目文字-数値変換 }
   endswitch

{ ------------------------------------------------------------------------ }
{    要素 2 番目読み出し                                                  }
{ ------------------------------------------------------------------------ }
   switch(sequence)
      case 10: pm1in=1;     { 2 番目文字-数値変換 }
   endswitch

{ ------------------------------------------------------------------------ }
{    要素 1 番目数値化                                                    }
{ ------------------------------------------------------------------------ }
   if (RESET|cmdreset)
      pm0data=0;
   else
      if (pm0in)
         pm0data.0:3=no.0:3;
         pm0data.4:7=pm0data.0:3;
         pm0data.8:11=pm0data.4:7;
         pm0data.12:15=pm0data.8:11;
      else
         pm0data=pm0data;
      endif
   endif

{ ------------------------------------------------------------------------ }
{    要素 2 番目数値化                                                    }
{ ------------------------------------------------------------------------ }
   if (RESET|cmdreset)
      pm1data=0;
   else
      if (pm1in)
         pm1data.0:3=no.0:3;
         pm1data.4:7=pm1data.0:3;
         pm1data.8:11=pm1data.4:7;
         pm1data.12:15=pm1data.8:11;
      else
         pm1data=pm1data;
      endif
   endif

{ ------------------------------------------------------------------------ }
{    番地計数                                                             }
{ ------------------------------------------------------------------------ }
   if (RESET|cmdreset)
      addresscount=0;
   else
      switch(setmseq)
         case 0:  addresscount=pm0data;
         case 8:  addresscount=addresscount+1;
         default: addresscount=addresscount;
```

```
      endswitch
   endif

{ ------------------------------------------------------------------------ }
{    dump 横位置                                                          }
{ ------------------------------------------------------------------------ }
   if (RESET|cmdreset)
      xcount=0;
   else
      switch(dumpseq)
         case 2:
            if (send1seq3p)
               xcount=xcount+1;
            else
               xcount=xcount;
            endif
         case 3:
            xcount=xcount;
      endswitch
   endif

{ ------------------------------------------------------------------------ }
{    dump 縦位置                                                          }
{ ------------------------------------------------------------------------ }
   if (RESET|cmdreset)
      ycount=0;
   else
      switch(dumpseq)
         case 3:
            if (dumpseqchg)
               ycount=ycount+1;
            else
               ycount=ycount;
            endif
         case 5: ycount=0;
         default:
            ycount=ycount;
      endswitch
   endif

{ ------------------------------------------------------------------------ }
{    dump 行程                                                            }
{ ------------------------------------------------------------------------ }
   if (RESET|cmdreset)
      dumpseq=0;
   else
      switch(dumpseq)
         case 0:                              { 待機 }
            switch(cmdno,sequence)
               case 1,11:
                  switch(busyfallp)
                     case 0: dumpseq=1;
                  endswitch
            endswitch
         case 1:                              { 番地 }
            switch(send1seq)
               case 5:  dumpseq=2;
               default: dumpseq=dumpseq;
            endswitch
         case 2:                              { データ }
            switch(xcount)
               case 2:
                  if (TEST)
                     dumpseq=3;               { 検証用桁端到着 }
                  else
```

```
                  dumpseq=dumpseq;
               endif
            case 16: dumpseq=3;          { 桁端到着 }
            default: dumpseq=dumpseq;
         endswitch

      case 3:                            { 改行と行送り }
         switch(send1seq)
            case 2:  dumpseq=6;
            default: dumpseq=dumpseq;
         endswitch
      case 4:                            { 終了 }
         dumpseq=0;
      case 5:                            { 番地閾値更新 }
         dumpseq=1;
      case 6:                            { 継続判定 }
         if (address>pm1data)
            dumpseq=4;
         else
            dumpseq=7;
         endif
      case 7:                            { 越頁判定 }
            switch(ycount)
               case 2:
                  if (TEST)
                     dumpseq=5;
                  else
                     dumpseq=1;
                  endif
               case 16: dumpseq=5;
               default: dumpseq=1;
            endswitch
   endswitch
endif

{ ------------------------------------------------------------------------ }
{    setm 行程                                                           }
{ ------------------------------------------------------------------------ }
if (RESET|cmdreset)
   setmseq=0;
else
   switch(setmseq)
      case 0:                            { 待機 }
         switch(cmdno,sequence)
            case 2,11:
               switch(busyfallp)
                  case 0: setmseq=1;
               endswitch
         endswitch
      case 1:                            { 番地表示 }
         switch(send1seq)
            case 5:  setmseq=2;
            default: setmseq=setmseq;
         endswitch
      case 2:                            { メモリ読み出し }
         setmseq=3;
      case 3:                            { データ表示 }
         switch(send1seq)
            case 8:  setmseq=4;
            default: setmseq=setmseq;
         endswitch
      case 4:                            { データ入力待ち }
         switch(crdetect)
            case 0:
               switch(cmd0buf)
```

```
                  case 0x2e: setmseq=11;
                  default:   setmseq=setmseq;
               endswitch
            default:
               switch(mojino)
                  case 0:  setmseq=8;    { 改行のみのとき }
                  default: setmseq=5;
               endswitch
         endswitch
      case 5: setmseq=6;                { 1 番目桁を文字から数値に変換 }
      case 6: setmseq=7;                { 2 番目桁を文字から数値に変換 }
      case 7: setmseq=8;                { メモリに書き込み }
      case 8: setmseq=10;               { 番地を進める }
      case 9: setmseq=0;                { 終了 }
      case 10:                          { 行送りの終了を待つ }
         switch(lfseq)
            case 5:  setmseq=1;
            default: setmseq=setmseq;
         endswitch
      case 11:                          { 改行待ち }
         switch(crdetect)
            case 0:  setmseq=setmseq;
            default: setmseq=9;
         endswitch
   endswitch
endif

{ ------------------------------------------------------------------------ }
{    番地要素 1                                                          }
{ ------------------------------------------------------------------------ }
offset1p.0:3=xcount.0:3;
offset1p.4:7=ycount.0:3;
offset1p.8:15=dumppagecount;

{ ------------------------------------------------------------------------ }
{    dump 頁計数                                                         }
{ ------------------------------------------------------------------------ }
if (RESET|cmdreset)
   dumppagecount=0;
else
   switch(dumpseq)
      case 5:  dumppagecount=dumppagecount+1;
      default: dumppagecount=dumppagecount;
   endswitch
endif

{ ------------------------------------------------------------------------ }
{    番地要素 2                                                          }
{ ------------------------------------------------------------------------ }
offset2p=pm0data;

{ ------------------------------------------------------------------------ }
{    番地                                                                }
{ ------------------------------------------------------------------------ }
if (dumpseq>0)
   address=offset1p+offset2p;
endif

if (setmseq>0)
   address=addresscount;
endif

{ ------------------------------------------------------------------------ }
{    メモリ読み出し                                                      }
```

```
{ ------------------------------------------------------------------------ }
   switch(dumpseq)
      case 1: mr=1;
      case 2: mr=1;
   endswitch

   switch(setmseq)
      case 2: mr=1;
   endswitch

{ ------------------------------------------------------------------------ }
{    メモリ書き込み                                          }
{ ------------------------------------------------------------------------ }
   switch(setmseq)
      case 7: mw=1;
   endswitch

{ ------------------------------------------------------------------------ }
{    送信データ                                              }
{ ------------------------------------------------------------------------ }
   if (echo)                              { エコー }
      switch(cmdwaitmark)
         case 4:                          { コマンド待ち表示 }
            if (TEST)
               sdata=0x25;                { % }
            else
               sdata=0x3e;                { > }
            endif
         default:
            switch(lfseq)
               case 3:  sdata=0x0a;       { 行送り }
               default: sdata=RDATAIN;    { 受信エコー }
            endswitch
      endswitch
   else
      switch(setmseq)                  { setm }
         case 1:                       { 番地 }
            switch(send1seq)
               case 0:
               case 1:
               case 2:
               case 3:
               from:
                  if (no<10)
                     sdata2op=1;
                  else
                     if (no<16)
                        sdata1op=1;
                     endif
                  endif
            endswitch

            switch(send1seq)
               case 4: sdata=0x20;
            endswitch
         case 3:                       { データ表示 }
            switch(send1seq)
               case 0: sdata=0x28;     { ( }
               case 3: sdata=0x29;     { ) }
               case 4: sdata=0x20;
               case 5: sdata=0x2d;     { - }
               case 6: sdata=0x3e;     { > }
               case 7: sdata=0x20;
               default:
                  if (no<10)
```

```
                    sdata2op=1;
                 else
                    if (no<16)
                       sdata1op=1;
                    endif
                 endif
           endswitch
        endswitch

        switch(dumpseq)                 { dump }
           case 1:                      { 番地 }
              switch(send1seq)
                 case 0:
                 case 1:
                 case 2:
                 case 3:
                 from:
                    if (no<10)
                       sdata2op=1;
                    else
                       if (no<16)
                          sdata1op=1;
                       endif
                    endif
              endswitch

              switch(send1seq)
                 case 4: sdata=0x20;
              endswitch
           case 2:                      { データ }
              switch(send1seq)
                 case 0: sdata=0;
                 case 1: sdata=0;
                 from:
                    if (no<10)
                       sdata2op=1;
                    else
                       if (no<16)
                          sdata1op=1;
                       endif
                    endif
              endswitch

              switch(send1seq)
                 case 2: sdata=0x20;
              endswitch
           case 3:                      { 改行と行送り }
              switch(send1seq)
                 case 0: sdata=0x0d;    { 改行 }
                 case 1: sdata=0x0a;    { 行送り }
              endswitch
        endswitch

        switch(cmdno)
           case 3: sdata=0x0d;          { HEX }
        endswitch
     endif

     if (sdata1op) sdata=no+0x41-10; endif
     if (sdata2op) sdata=no+0x30; endif

{ ------------------------------------------------------------------- }
{    送信要求                                                       }
{ ------------------------------------------------------------------- }
        switch(dumpseq)
```

```
   case 1:
      switch(busyfallp)
         case 0:
            switch(send1seq)
               case 5:
               default: sreqp=1;
            endswitch
      endswitch
   case 2:
      switch(busyfallp)
         case 0:
            switch(send1seq)
               case 5:
               case 3:
               default:
                  if (TEST)
                     switch(xcount)
                        case 2:
                        default: sreqp=1;
                     endswitch
                  else
                     switch(xcount)
                        case 16:
                        default: sreqp=1;
                     endswitch
                  endif
            endswitch
      endswitch
   case 3:
      switch(send1seq)
         case 0:                    { 改行 }
         case 1:                    { 行送り }
         from:
            switch(busyfallp)
               case 0: sreqp=1;
            endswitch
      endswitch
endswitch

switch(setmseq)
   case 1:
      switch(busyfallp)
         case 0:
            switch(send1seq)
               case 5:
               default: sreqp=1;
            endswitch
      endswitch
   case 3:
      switch(busyfallp)
         case 0:
            switch(send1seq)
               case 8:
               default: sreqp=1;
            endswitch
      endswitch
endswitch

if (echo)
   if (GETP) sreqp=1; endif    { ローカルエコー }
endif

switch(cmdwaitmark)
   case 2: sreqp=1;   { コマンド待ち表示 }
endswitch
```

```
      switch(lfseq)
         case 2: sreqp=1;
      endswitch

{ ------------------------------------------------------------------------ }
{    送信終了起点                                                          }
{ ------------------------------------------------------------------------ }
   if (RESET)
      busyfallp=0;
   else
      switch(busyfallp)
         case 0: if (BUSY) busyfallp=2; endif
         case 1: busyfallp=0;
         case 2:
            if (BUSY)
               busyfallp=busyfallp;
            else
               busyfallp=4;
            endif
         case 4:
            switch(intvtime)
               case 0:  busyfallp=1;
               default: busyfallp=busyfallp;
            endswitch
      endswitch
   endif

{ ------------------------------------------------------------------------ }
{    送信終了起点遅延                                                      }
{ ------------------------------------------------------------------------ }
   if (RESET)
      busyfallpdelay=0;
   else
      busyfallpdelay.0=busyfallp.0;
      busyfallpdelay.1:3=busyfallpdelay.0:2;
   endif

{ ------------------------------------------------------------------------ }
{    送信行程                                                              }
{ ------------------------------------------------------------------------ }
   if (RESET)
      send1seq=0;
   else
      if (dumpseqchg)
         send1seq=0;
      else
         { dump }
         switch(dumpseq)
            case 1:                              { 番地 }
               switch(send1seq)
                  case 5:
                     send1seq=send1seq;
                  default:
                     if (busyfallp.0)
                        send1seq1op=1;
                     else
                        send1seq=send1seq;
                     endif
               endswitch
            case 2:                              { データ }
               switch(send1seq)
                  case 3:
                     send1seq=0;
                  default:
                     if (busyfallp.0)
```

```
                    send1seq1op=1;
                 else
                    send1seq=send1seq;
                 endif
              endswitch
           case 3:                          { 改行と行送り }
              switch(send1seq)
                 case 2:
                    send1seq=2;
                 default:
                    if (busyfallp.0)
                       send1seq1op=1;
                    else
                       send1seq=send1seq;
                    endif
              endswitch
        endswitch

        { setm }
        switch(setmseq)
           case 1:                          { 番地 }
              switch(send1seq)
                 case 5:
                    send1seq=send1seq;
                 default:
                    if (busyfallp.0)
                       send1seq1op=1;
                    else
                       send1seq=send1seq;
                    endif
              endswitch
           case 3:                          { メモリ内容 }
              switch(send1seq)
                 case 8:
                    send1seq=send1seq;
                 default:
                    if (busyfallp.0)
                       send1seq1op=1;
                    else
                       send1seq=send1seq;
                    endif
              endswitch
        endswitch
     endif
  endif

  if (send1seq1op) send1seq=send1seq+1; endif

{ ------------------------------------------------------------------- }
{    dumpseq 遅延                                                     }
{ ------------------------------------------------------------------- }
  dumpseqdelay=dumpseq;

{ ------------------------------------------------------------------- }
{    dumpseq 遷移起点                                                 }
{ ------------------------------------------------------------------- }
  dumpseqchg=dumpseqdelay!=dumpseq;

{ ------------------------------------------------------------------- }
{    送信行程 3                                                       }
{ ------------------------------------------------------------------- }
  switch(send1seq)
     case 3: send1seq3p=1;
  endswitch
```

```
{ ------------------------------------------------------------------------ }
{    HEX 初期化                                                           }
{ ------------------------------------------------------------------------ }
   switch(cmdno)
      case 3:              { hex }
      default: hexres=1;
   endswitch

{ ------------------------------------------------------------------------ }
{    バス要求                                                             }
{ ------------------------------------------------------------------------ }
   switch(cmdno)
      case 0:
      case 4:              { res }
      default: busreq=1;
   endswitch

ende
```

```
{ ========================================================================== }
{    機能実行                                                              }
{ ========================================================================== }
entity sim
output RESET;
output GETP;
output RDATAIN[8];
output MDATAIN[8];
output BUSY;
output EOF;
output SUM;
output EOEPP;
output BUSACK;
output TEST;
output MR;
output MW;
output ADDRESS[16];
output MDATAOUT[8];
output SDATA[8];
output SREQP;
output CMDNO[3];
output CPURESET;
output HEXRES;
output BUSREQ;
output ROP;
output WOP;
output BLKSEL[4];

input  simres;

{ ------------------------------------------------------------------------- }
{    検査出力                                                               }
{ ------------------------------------------------------------------------- }
output TBOP[16];

{ ------------------------------------------------------------------------- }
{    内部信号                                                               }
{ ------------------------------------------------------------------------- }
bitn node_ADDRESS[16];
bitn node_SREQP;
bitr times[4];
bitr busy;
bitr tc[16];

{ ------------------------------------------------------------------------- }
{    出力代入                                                               }
{ ------------------------------------------------------------------------- }
   ADDRESS=node_ADDRESS;
   SREQP=node_SREQP;
   BUSY=busy;

{ ------------------------------------------------------------------------- }
{    検査位置                                                               }
{ ------------------------------------------------------------------------- }
   TBOP=tc;

{ ------------------------------------------------------------------------- }
{    実効譜導入                                                             }
{ ------------------------------------------------------------------------- }
   part main(RESET,GETP,RDATAIN,MDATAIN,BUSY,EOF,SUM,EOEPP,BUSACK,TEST
            ,MR,MW,node_ADDRESS,MDATAOUT,SDATA,node_SREQP,CMDNO,CPURESET
            ,HEXRES,BUSREQ,ROP,WOP,BLKSEL);

{ ------------------------------------------------------------------------- }
```

```
{    検査譜                                                    }
{ ------------------------------------------------------------------------ }
   simres=0;
   if (!simres) tc=tc+1; endif

   if (tc<5) RESET=1; endif

   switch(tc)
      case 110: GETP=1; RDATAIN=0x44;    { D }
      case 115: GETP=1; RDATAIN=0x20;    { SP }
      case 120: GETP=1; RDATAIN=0x31;    { 1 }
      case 125: GETP=1; RDATAIN=0x32;    { 2 }
      case 130: GETP=1; RDATAIN=0x33;    { 3 }
      case 135: GETP=1; RDATAIN=0x34;    { 4 }
      case 140: GETP=1; RDATAIN=0x2c;    { , }
      case 145: GETP=1; RDATAIN=0x35;    { 5 }
      case 150: GETP=1; RDATAIN=0x41;    { 6 }
      case 155: GETP=1; RDATAIN=0x62;    { 7 }
      case 160: GETP=1; RDATAIN=0x38;    { 8 }
      case 165: GETP=1; RDATAIN=0x0d;    { CR }
   endswitch

   switch(tc)
      case 800: GETP=1; RDATAIN=0x53;    { S }
      case 805: GETP=1; RDATAIN=0x20;    { SP }
      case 810: GETP=1; RDATAIN=0x31;    { 1 }
      case 815: GETP=1; RDATAIN=0x32;    { 2 }
      case 820: GETP=1; RDATAIN=0x33;    { 3 }
      case 825: GETP=1; RDATAIN=0x34;    { 4 }
      case 830: GETP=1; RDATAIN=0x0d;    { CR }

      case 940: GETP=1; RDATAIN=0x33;    { 3 }
      case 945: GETP=1; RDATAIN=0x34;    { 4 }
      case 950: GETP=1; RDATAIN=0x0d;    { CR }

      case 1040: GETP=1; RDATAIN=0x35;    { 5 }
      case 1045: GETP=1; RDATAIN=0x36;    { 6 }
      case 1050: GETP=1; RDATAIN=0x37;    { 7 }
      case 1055: GETP=1; RDATAIN=0x38;    { 8 }
      case 1060: GETP=1; RDATAIN=0x0d;    { CR }

      case 1150: GETP=1; RDATAIN=0x2e;    { . }
   endswitch

   switch(tc)
      case 1250: GETP=1; RDATAIN=0x48;    { H }
      case 1255: GETP=1; RDATAIN=0x45;    { E }
      case 1260: GETP=1; RDATAIN=0x58;    { X }
      case 1265: GETP=1; RDATAIN=0x0d;    { CR }
      case 1300: EOF=1; SUM=1;            { 通信終了 }

      case 1320: GETP=1; RDATAIN=0x52;    { R }
      case 1325: GETP=1; RDATAIN=0x45;    { E }
      case 1330: GETP=1; RDATAIN=0x53;    { S }
      case 1335: GETP=1; RDATAIN=0x0d;    { CR }

      case 1350: GETP=1; RDATAIN=0x45;    { E }
      case 1355: GETP=1; RDATAIN=0x45;    { E }
      case 1360: GETP=1; RDATAIN=0x50;    { P }
      case 1365: GETP=1; RDATAIN=0x57;    { W }
      case 1370: GETP=1; RDATAIN=0x20;    { SP }
      case 1375: GETP=1; RDATAIN=0x33;    { 3 }
```

```
      case 1380: GETP=1; RDATAIN=0x34;   { 4 }
      case 1385: GETP=1; RDATAIN=0x0d;   { CR }
      case 1410: EOEPP=1;                { EEPROM 操作終了 }

      case 1500: GETP=1; RDATAIN=0x45;   { E }
      case 1505: GETP=1; RDATAIN=0x45;   { E }
      case 1510: GETP=1; RDATAIN=0x50;   { P }
      case 1515: GETP=1; RDATAIN=0x52;   { R }
      case 1520: GETP=1; RDATAIN=0x20;   { SP }
      case 1525: GETP=1; RDATAIN=0x35;   { 5 }
      case 1530: GETP=1; RDATAIN=0x36;   { 6 }
      case 1535: GETP=1; RDATAIN=0x0d;   { CR }
      case 1560: EOEPP=1;                { EEPROM 操作終了 }

   endswitch

   switch(node_ADDRESS)
      case 0x1234: MDATAIN=0x56;
      case 0x1235: MDATAIN=0xab;
      case 0x5678: MDATAIN=0x12;
   endswitch

   switch(tc)
      case 100: EOEPP=1;
   endswitch

   if (tc<5)
      busy=0;
   else
      switch(busy)
         case 0:
            if (node_SREQP) busy=1; endif
         case 1:
            switch(times)
               case 5:  busy=0;
               default: busy=busy;
            endswitch
      endswitch
   endif

   if (busy)
      times=times+1;
   else
      times=0;
   endif

   TEST=1;

ende

endlogic
```