LDL07K19A

LDLABO

2007 年 12 月 22 日

目次

図目次

表目次

第 1 章

論理譜

```
{ ========================================================================== }
{    メモリモニタ                                                          }
{ ========================================================================== }
logicname LD07K19A

library LDL07F02A  { シリアル受信アダプタ }
library LDL07E16A  { シリアル送信アダプタ }
library LDL06M17A  { HEX ローダ }
library LDL06L08A  { I2C ローダ }
library LDL07F18A  { モニタアダプタ }
library LDL07H06A  { バス操作アダプタ }
library LDL06L17A  { I2C メモリ }

{ ========================================================================== }
{    実効譜                                                                }
{ ========================================================================== }
entity main

{ -------------------------------------------------------------------------- }
{    入力                                                                  }
{ -------------------------------------------------------------------------- }
input RESET;
input SC;
input RXD;
input CTS;
input BUSACK;

input Z80MREQ;
input Z80IORQ;
input Z80RD;
input Z80WR;

{ -------------------------------------------------------------------------- }
{    出力                                                                  }
{ -------------------------------------------------------------------------- }
output BUSREQ;
output TXD;
output RTS;
output WE;
output RD;
output WP;
output SCL;
output CPURESET;
output SCOUT;

{ -------------------------------------------------------------------------- }
{    双方向                                                                }
{ -------------------------------------------------------------------------- }
```

```
inout ADDRESS[16];
inout DATA[8];
inout SDA;

{ ------------------------------------------------------------------------ }
{    引き出し                                                              }
{ ------------------------------------------------------------------------ }

output EXPprobBUSY;
output EXPoutADDRESS[16];
output EXPoutDATA[8];
output EXPoutSDA;

{ ------------------------------------------------------------------------ }
{    内部信号                                                              }
{ ------------------------------------------------------------------------ }
{ バス操作アダプタ }
bitn proa[32];
bitn proaMADDRESS0P[16];      { HEX - 番地 }
bitn proaMDATA0P[8];          { HEX - 値 }
bitn proaMWE0P;               { HEX - 書き込み }
bitn proaADDRESS1P[16];       { EEPROM - 番地 }
bitn proaDATAOUT1P[8];        { EEPROM - 値 }
bitn proaRD1P;                { EEPROM - 読み出し }
bitn proaWR1P;                { EEPROM - 書き込み }
bitn proaBUSREQ1P;            { EEPROM - バス要求 }
bitn proaADDRESSCNT1P;        { EEPROM - 番地制御 }
bitn proaDATACNT1P;           { EEPROM - 値制御 }
bitn proaCPURESET1P;          { EEPROM - バスマスタ初期化 }
bitn proaMR3P;                { モニタアダプタ - 読み込み }
bitn proaMW3P;                { モニタアダプタ - 書き込み }
bitn proaMADDRESS3P[16];      { モニタアダプタ - 番地 }
bitn proaMDATAOUT3P[8];       { モニタアダプタ - 値 }
bitn proaCMDNO3P[3];          { モニタアダプタ - コマンド番号 }
bitn proaCPURESET3P;          { モニタアダプタ - バスマスタ初期化 }
bitn proaBUSREQ3P;            { モニタアダプタ - バス要求 }

{ モニタアダプタ }
bitn prob[47];
bitn probGETP;                { 受信データ取得起点 }
bitn probRDATAIN[8];          { 受信データ }
bitn probMDATAIN[8];          { メモリデータ }
bitn probBUSY;                { 送信中指標 }
bitn probEOF;                 { HEX ファイルロード終了 }
bitn probEOEPP;               { EEPROM 操作終了 }
bitn probBUSACK;              { バス許可 }
bitn probRE;                  { 読み取り有効 }
bitn probTEST;                { 検査端子 }

{ I2C ローダ }
bitn proc[35];
bitn procRESET;               { 初期化 }
bitn procDATAIN[8];           { BUS IF }
bitn procSDAIN;               { I2C IF }
bitn procBLKSEL[4];           { 範囲選択 }
bitn procWOP;                 { 書き込み指示 }
bitn procROP;                 { 読み出し指示 }
bitn procBUSACK;              { BUS 開放 }
bitn procRE;                  { 読み取り有効 }

{ HEX ローダ }
bitn prod[51];
```

```
bitn prodRESET;                 { 初期化 }
bitn prodRDYP;                  { データパルス }
bitn prodDATAIN[8];             { データ }

{ シリアル送信アダプタ }
bitn proe[2];
bitn proeCTS;                   { 送信許可 }
bitn proeSDATA[8];              { 送信データ }
bitn proeSC;                    { 送信 CLK }
bitn proeSREQP;                 { 送信要求起点 }

{ シリアル受信アダプタ }
bitn prof[10];
bitn profRXD;                   { 受信データ }
bitn profSC;                    { 受信パリティ }

{ 双方向信号 }
bitn inADDRESS[16];
bitn inDATA[8];
bitn inSDA;
bitn outDATA[8];
bitn outSDA;
bitn outADDRESS[16];
bitn cntDATA;
bitn cntSDA;
bitn cntADDRESS;

{ 極性変更 }
bitn busack;

{ 出力 }
bitn rd;                        { Z80MRD }
bitn we;                        { Z80MWE }
bitr scount[4];                 { 13 分周 }
bitr scout;                     { RS232C 通信 CLK }
bitn cts;

{ Z80 通信 }
bitr receflg;                   { Z80 受信フラグ }
bitr z80senddata[8];            { Z80 要求の送信データ }
bitr z80sendseq[2];             { Z80 要求の送信行程 }
bitr z80cpuonly;                { ローカルエコー抑制 }

{ ------------------------------------------------------------------------ }
{    検査端子                                                    }
{ ------------------------------------------------------------------------ }

output T0P[3];    T0P=proaCMDNO3P;
output T1P[8];    T1P=probRDATAIN;
output T2P;       T2P=probGETP;
output T3P;       T3P=probBUSY;
output T4P;       T4P=proc.29;
output T5P[16];   T5P=outADDRESS;
output T6P;       T6P=cntADDRESS;
output T7P;       T7P=proeSREQP;
output T8P[8];    T8P=proeSDATA;
output T9P[8];    T9P=proa.17:24;
output T10P;      T10P=cntDATA;
output T11P[16];  T11P=prob.2:17;
output T12P;      T12P=proaWR1P;
output T13P;      T13P=procWOP;
output T16P;      T16P=procROP;
output T17P;      T17P=probEOEPP;
output T18P[4];   T18P=prob.43:46;
```

```
{ ========================================================================= }
{     固有譜                                                               }
{ ========================================================================= }
{ ------------------------------------------------------------------------- }
{    出力極性変更                                                          }
{ ------------------------------------------------------------------------- }
   BUSREQ=!proa.27;
   WE=!we;
   RD=!rd;
   RTS=!proa.0;
   CPURESET=!proa.30;

{ ------------------------------------------------------------------------- }
{    入力極性変更                                                          }
{ ------------------------------------------------------------------------- }
   busack=!BUSACK;
   cts=!CTS;

{ ------------------------------------------------------------------------- }
{    出力代入                                                              }
{ ------------------------------------------------------------------------- }
   SCOUT=scout;

{ ------------------------------------------------------------------------- }
{    Z80MWE                                                                }
{ ------------------------------------------------------------------------- }
   if (busack)
      we=proa.28;
   else
      if (Z80MREQ)
         we=0;
      else
         if (Z80WR)
            we=0;
         else
            we=1;
         endif
      endif
   endif

{ ------------------------------------------------------------------------- }
{    Z80MRD                                                                }
{ ------------------------------------------------------------------------- }
   if (busack)
      rd=proa.29;
   else
      if (Z80MREQ)
         rd=0;
      else
         if (Z80RD)
            rd=0;
         else
            rd=1;
         endif
      endif
   endif

{ ------------------------------------------------------------------------- }
{    ローカルエコー抑制                                                    }
{ ------------------------------------------------------------------------- }
   if (RESET)
      z80cpuonly=0;
```

```
   else
      if (!Z80IORQ&!Z80WR)
         switch(inADDRESS.0:1)
            case 3:  z80cpuonly=inDATA.0;
            default: z80cpuonly=z80cpuonly;
         endswitch
      else
         z80cpuonly=z80cpuonly;
      endif
   endif

{ ------------------------------------------------------------------------ }
{    Z80 要求の送信行程                                                    }
{ ------------------------------------------------------------------------ }
   if (RESET)
      z80sendseq=0;
   else
      switch(z80sendseq)
         case 0:
            if (!Z80IORQ&!Z80WR)
               switch(inADDRESS.0:1)
                  case 2: z80sendseq=1;
               endswitch
            endif
         case 1:
            if (proe.1)
               z80sendseq=z80sendseq;
            else
               z80sendseq=2;
            endif
         case 2:
            if (proe.1)
               z80sendseq=3;
            else
               z80sendseq=z80sendseq;
            endif
         case 3:
            z80sendseq=0;
      endswitch
   endif

{ ------------------------------------------------------------------------ }
{    Z80 要求の送信データ                                                  }
{ ------------------------------------------------------------------------ }
   if (RESET)
      z80senddata=0;
   else
      if (!Z80IORQ&!Z80WR)
         switch(inADDRESS.0:1)
            case 2:  z80senddata=inDATA;
            default: z80senddata=z80senddata;
         endswitch
      else
         z80senddata=z80senddata;
      endif
   endif

{ ------------------------------------------------------------------------ }
{    Z80 受信フラグ                                                        }
{ ------------------------------------------------------------------------ }
   if (RESET)
      receflg=0;
   else
      if (prof.9)
         receflg=1;
```

```
      else
         if (!Z80IORQ&!Z80RD)
            switch(inADDRESS.0:1)
               case 1:  receflg=0;
               default: receflg=receflg;
            endswitch
         else
            receflg=receflg;
         endif
      endif
   endif

{ ------------------------------------------------------------------------ }
{    RS232C 通信 CLK                                                      }
{ ------------------------------------------------------------------------ }
   if (RESET)
      scout=0;
   else
      switch(scount)
         case 0:  scout=1;
         default: scout=0;
      endswitch
   endif

{ ------------------------------------------------------------------------ }
{    13 分周                                                              }
{ ------------------------------------------------------------------------ }
   if (RESET)
      scount=0;
   else
      switch(scount)
         case 12: scount=0;
         default: scount=scount+1;
      endswitch
   endif

{ ======================================================================== }
{    共通譜                                                               }
{ ======================================================================== }
{ ------------------------------------------------------------------------ }
{    双方向                                                               }
{ ------------------------------------------------------------------------ }
   enable(ADDRESS,inADDRESS,outADDRESS,cntADDRESS)
   enable(DATA,inDATA,outDATA,cntDATA)
   enable(SDA,inSDA,outSDA,cntSDA)

   inADDRESS.16=1;
   inDATA.8=1;
   inSDA.1=1;

{ ------------------------------------------------------------------------ }
{    双方向制御                                                           }
{ ------------------------------------------------------------------------ }
   if (!Z80IORQ&!Z80RD)
      cntDATA=1;
   else
      cntDATA=proa.26;
   endif

   cntSDA=proc.29;
   cntADDRESS=proa.25;

{ ------------------------------------------------------------------------ }
{    出力代入                                                             }
{ ------------------------------------------------------------------------ }
```

```
   outADDRESS=proa.1:16;

   if (!Z80IORQ&!Z80RD)
      switch(inADDRESS.0:1)
         case 0:
            outDATA.0=receflg;

            switch(z80sendseq)
               case 0:  outDATA.1=proe.1;
               default: outDATA.1=1;
            endswitch
         case 1:
            outDATA=prof.0:7;
      endswitch
   else
      outDATA=proa.17:24;
   endif

   outSDA=proc.28;

   WP=proc.30;
   SCL=proc.27;

   TXD=proe.0;

{ ------------------------------------------------------------------------ }
{    引き出し代入                                                          }
{ ------------------------------------------------------------------------ }

   EXPprobBUSY=proe.1;
   EXPoutADDRESS=proa.1:16;
   EXPoutDATA=proa.17:24;
   EXPoutSDA=proc.28;

{ ------------------------------------------------------------------------ }
{    バス操作アダプタ                                                      }
{ ------------------------------------------------------------------------ }
   proa=LDL07H06A(
        RESET
       ,proaMADDRESS0P
       ,proaMDATA0P
       ,proaMWE0P
       ,proaADDRESS1P
       ,proaDATAOUT1P
       ,proaRD1P
       ,proaWR1P
       ,proaBUSREQ1P
       ,proaADDRESSCNT1P
       ,proaCPURESET1P
       ,proaMR3P
       ,proaMW3P
       ,proaMADDRESS3P
       ,proaMDATAOUT3P
       ,proaCMDNO3P
       ,proaCPURESET3P
       ,proaBUSREQ3P
       );

   proa.32=1;

   proaMADDRESS0P=prod.16:31;
   proaMDATA0P=prod.32:39;
   proaMWE0P=prod.40;
   proaADDRESS1P=proc.9:24;
   proaDATAOUT1P=proc.0:7;
```

```
    proaRD1P=proc.31;
    proaWR1P=proc.32;
    proaBUSREQ1P=proc.33;
    proaADDRESSCNT1P=proc.25;
    proaDATACNT1P=proc.8;
    proaCPURESET1P=proc.26;
    proaMR3P=prob.0;
    proaMW3P=prob.1;
    proaMADDRESS3P=prob.2:17;
    proaMDATAOUT3P=prob.18:25;
    proaCMDNO3P=prob.35:37;
    proaCPURESET3P=prob.38;
    proaBUSREQ3P=prob.40;


    proa.0=RTS;
    proa.1:16=MADDRESS;
    proa.17:24=MDATAOUT;
    proa.25=ADDRESSCNT;
    proa.26=MDATACNT;
    proa.27=BUSREQ;
    proa.28=MWE;
    proa.29=MRD;
    proa.30=CPURESET;
    proa.31=HEXRES;


{ ------------------------------------------------------------------------ }
{    モニタアダプタ                                          }
{ ------------------------------------------------------------------------ }
    prob=LDL07F18A(
        RESET
       ,probGETP
       ,probRDATAIN
       ,probMDATAIN
       ,probBUSY
       ,probEOF
       ,probEOEPP
       ,probBUSACK
       ,probRE
       ,probTEST
       );

    prob.47=1;

    probGETP=prof.9;
    probRDATAIN=prof.0:7;
    probMDATAIN=inDATA;
    probBUSY=proe.1;
    probEOF=prod.49;
    probEOEPP=proc.34;
    probBUSACK=busack;
    probRE=proa.29;
    probTEST=0;                 { 検証設定 }


    prob.0=MR;
    prob.1=MW;
    prob.2:17=ADDRESS;
    prob.18:25=MDATAOUT;
    prob.26:33=SDATA;
    prob.34=SREQP;
    prob.35:37=CMDNO;
    prob.38=CPURESET;
    prob.39=HEXRES;
    prob.40=BUSREQ;
```

```
   prob.41=ROP;
   prob.42=WOP;
   prob.43:46=BLKSEL;


{ ------------------------------------------------------------------------ }
{    I2C ローダ                                                          }
{ ------------------------------------------------------------------------ }
   proc=LDL06L08A(
       RESET
      ,procDATAIN
      ,procSDAIN
      ,procBLKSEL
      ,procWOP
      ,procROP
      ,procBUSACK
      ,procRE
      );

   proc.35=1;

   procRESET=proa.32;
   procDATAIN=inDATA;
   procSDAIN=inSDA;
   procBLKSEL=prob.43:46;
   procWOP=prob.42;
   procROP=prob.41;
   procBUSACK=busack;
   procRE=proa.29;


   proc.0:7=DATAOUT;
   proc.8=DATACNT;
   proc.9:24=ADDRESS;
   proc.25=ADDRESSCNT;
   proc.26=CPURESET;
   proc.27=SCL;
   proc.28=SDAOUT;
   proc.29=SDACNT;
   proc.30=WP;
   proc.31=RD;
   proc.32=WR;
   proc.33=BUSREQ;
   proc.34=EOP;


{ ------------------------------------------------------------------------ }
{    HEX ローダ                                                          }
{ ------------------------------------------------------------------------ }
   prod=LDL06M17A(
       prodRESET
      ,prodRDYP
      ,prodDATAIN
      );

   prod.51=1;

   prodRESET=prob.39;
   prodRDYP=prof.9;
   prodDATAIN=prof.0:7;


   prod.0:15 =ememaddress;
   prod.16:31=memaddress;
   prod.32:39=memdata;
   prod.40=memwe;
```

```
   prod.41:48=rtype;
   prod.49=regeof;
   prod.50=chksum;


{ ------------------------------------------------------------------------ }
{    シリアル送信アダプタ                                  }
{ ------------------------------------------------------------------------ }
   proe=LDL07E16A(
       RESET
      ,proeCTS
      ,proeSDATA
      ,proeSC
      ,proeSREQP
      );

   proe.2=1;

   proeCTS=cts;

   switch(z80sendseq)
      case 0:  proeSDATA=prob.26:33;
      default: proeSDATA=z80senddata;
   endswitch

   proeSC=SC;

   switch(z80sendseq)
      case 2:  proeSREQP=1;
      default:
         if (z80cpuonly)
         else
            proeSREQP=prob.34;
         endif
   endswitch


   proe.0=TXD;
   proe.1=BUSY;


{ ------------------------------------------------------------------------ }
{    シリアル受信アダプタ                                  }
{ ------------------------------------------------------------------------ }
   prof=LDL07F02A(
       RESET
      ,profRXD
      ,profSC
      );

   prof.10=1;

   profRXD=RXD;
   profSC=SC;


   prof.0:7=DATA;
   prof.8=PARITY;
   prof.9=GETP;


ende
```

```
{ ========================================================================= }
{    Z80 信号検証                                                          }
{ ========================================================================= }
entity sim
output RESET;
output SC;
output RXD;
output CTS;
output BUSACK;
output Z80MREQ;
output Z80IORQ;
output Z80RD;
output Z80WR;

output BUSREQ;
output TXD;
output RTS;
output WE;
output RD;
output WP;
output SCL;
output CPURESET;
output SCOUT;

output ADDRESS[16];
output DATA[8];
output SDA;

input  simres;

{ ------------------------------------------------------------------------- }
{    検査端子                                                              }
{ ------------------------------------------------------------------------- }
output TB0P[16];

{ ------------------------------------------------------------------------- }
{    内部信号                                                              }
{ ------------------------------------------------------------------------- }
bitr  tc[16];

{ ------------------------------------------------------------------------- }
{    出力代入                                                              }
{ ------------------------------------------------------------------------- }

{ ------------------------------------------------------------------------- }
{    検査端子代入                                                          }
{ ------------------------------------------------------------------------- }
   TB0P=tc;

{ ------------------------------------------------------------------------- }
{    実効譜引用                                                            }
{ ------------------------------------------------------------------------- }
   part main(RESET,SC,RXD,CTS,BUSACK,Z80MREQ,Z80IORQ,Z80RD,Z80WR
            ,BUSREQ,TXD,RTS,WE,RD,WP,SCL,CPURESET,SCOUT
            ,ADDRESS,DATA,SDA)

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

   if (tc<5) RESET=1; endif
```

```
   switch(tc)
      case 10: Z80MREQ=0; Z80RD=0; Z80WR=1; BUSACK=1;
      case 11: Z80MREQ=0; Z80RD=1; Z80WR=0; BUSACK=1;
      case 12: Z80MREQ=0; Z80RD=1; Z80WR=1; BUSACK=1;
      case 13: Z80MREQ=1; Z80RD=0; Z80WR=0; BUSACK=1;
      case 14: Z80MREQ=1; Z80RD=0; Z80WR=1; BUSACK=1;
      case 15: Z80MREQ=1; Z80RD=1; Z80WR=0; BUSACK=1;
      case 16: Z80MREQ=1; Z80RD=1; Z80WR=1; BUSACK=1;
      case 17: Z80MREQ=0; Z80RD=0; Z80WR=0; BUSACK=1;
      case 18: Z80MREQ=0; Z80RD=0; Z80WR=0; BUSACK=0;
      default: Z80MREQ=1; Z80RD=1; Z80WR=1; BUSACK=0;
   endswitch

ende
```

```
{ ========================================================================== }
{    バスマスタ初期化                                                      }
{ ========================================================================== }
entity sim
output RESET;
output SC;
output RXD;
output CTS;
output BUSACK;
output Z80MREQ;
output Z80IORQ;
output Z80RD;
output Z80WR;

output BUSREQ;
output TXD;
output RTS;
output WE;
output RD;
output WP;
output SCL;
output CPURESET;
output SCOUT;

output ADDRESS[16];
output DATA[8];
output SDA;

input  simres;

{ -------------------------------------------------------------------------- }
{    検査端子                                                              }
{ -------------------------------------------------------------------------- }
output TB0P;
output TB1P[2];
output TB2P[8];
output TB3P[3];
output TB4P[8];
output TB5P;
output TB6P[4];
output TB7P[11];
output TB8P[8];
output TB9P;
output TB10P[8];
output TB11P[8];
output TB12P[8];
output TB13P;
output TB14P[20];

{ -------------------------------------------------------------------------- }
{    内部信号                                                              }
{ -------------------------------------------------------------------------- }
bitr  tc[20];
bitn  nodeRESET;
bitn  nodeSC;
bitn  nodeSREQP;
bitn  nodeSDATA[8];
bitn  nodeCTS;
bitn  nodeSEND[2];
bitr  sp[2];
bitr  sdp[3];
bitr  scount[8];
bitn  EXPprobBUSY;
bitn  EXPoutADDRESS[16];
```

```
bitn  rxd;
bitn  nodeTXD;
bitn  proRECEIVE[10];
bitn  nodeWE;
bitn  EXPoutDATA[8];
bitn  EXPoutSDA;

{ ------------------------------------------------------------------------ }
{    出力代入                                                             }
{ ------------------------------------------------------------------------ }
   RESET  =nodeRESET;
   SC     =nodeSC;
   CTS    =!nodeCTS;
   TXD    =nodeTXD;
   WE     =nodeWE;

{ ------------------------------------------------------------------------ }
{    検査端子代入                                                         }
{ ------------------------------------------------------------------------ }
   TB0P=nodeSEND.1;
   TB1P=sp;
   TB2P=scount;
   TB3P=sdp;
   TB4P=nodeSDATA;
   TB5P=nodeSREQP;
   TB8P=proRECEIVE.0:7;
   TB9P=proRECEIVE.9;
   TB10P=0;
   TB11P=0;
   TB12P=0;
   TB13P=EXPoutSDA;
   TB14P=tc;

{ ------------------------------------------------------------------------ }
{    実効譜引用                                                           }
{ ------------------------------------------------------------------------ }
   part main(RESET,SC,RXD,CTS,BUSACK,Z80MREQ,Z80IORQ,Z80RD,Z80WR
            ,BUSREQ,nodeTXD,RTS,nodeWE,RD,WP,SCL,CPURESET,SCOUT
            ,ADDRESS,DATA,SDA
            ,EXPprobBUSY,EXPoutADDRESS,EXPoutDATA,EXPoutSDA)

{ ------------------------------------------------------------------------ }
{    送信手続き譜導入                                                     }
{ ------------------------------------------------------------------------ }
   nodeSEND=LDL07E16A(nodeRESET,nodeCTS,nodeSDATA,nodeSC,nodeSREQP);

   nodeSEND.2=1;

{ ------------------------------------------------------------------------ }
{    受信手続き譜導入                                                     }
{ ------------------------------------------------------------------------ }
   proRECEIVE=LDL07F02A(nodeRESET,rxd,nodeSC);

   proRECEIVE.10=1;

{ ------------------------------------------------------------------------ }
{    送信受付行程                                                         }
{ ------------------------------------------------------------------------ }
   if (nodeRESET)
      sp=0;
   else
      switch(sp)
         case 0:
            if (nodeSEND.1) sp=1; endif
         case 1:
```

```
            if (nodeSEND.1)
               sp=sp;
            else
               sp=2;
            endif
         case 2:
            sp=0;
      endswitch
   endif

{ ------------------------------------------------------------------------ }
{    ローカルエコー待ち行程                                              }
{ ------------------------------------------------------------------------ }
   switch(sdp)
      case 0: sdp.0=sp.1;
      case 1:
         if (EXPprobBUSY)
            sdp=sdp;
         else
            sdp=2;
         endif

      case 2: sdp=4;
      case 4: sdp=0;
   endswitch

{ ------------------------------------------------------------------------ }
{    送信計数                                                            }
{ ------------------------------------------------------------------------ }
   if (nodeRESET)
      scount=0;
   else
      if (sdp.1)
         scount=scount+1;
      else
         scount=scount;
      endif
   endif

{ ------------------------------------------------------------------------ }
{    送信データ                                                          }
{ ------------------------------------------------------------------------ }
   switch(scount)
      case 0:  nodeSDATA=0x52;     { R }
      case 1:  nodeSDATA=0x45;     { E }
      case 2:  nodeSDATA=0x53;     { S }
      case 3:  nodeSDATA=0x0d;     { cr }
   endswitch

{ ------------------------------------------------------------------------ }
{    送信要求                                                            }
{ ------------------------------------------------------------------------ }
   if (scount<=3)
      nodeSREQP=sdp.2;
   endif

   switch(tc)
      case 2100: nodeSREQP=1;
   endswitch

   switch(scount)
      case 1: nodeSREQP=sdp.2;
      case 2: nodeSREQP=sdp.2;
      case 3: nodeSREQP=sdp.2;
   endswitch
```

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

   if (tc<5) nodeRESET=1; endif

   nodeSC=tc.0;
   nodeCTS=1;
   RXD=nodeSEND.0;
   BUSACK=0;
   rxd=nodeTXD;

ende
```

```
{ ========================================================================== }
{     メモリ変更コマンド検証                                                 }
{ ========================================================================== }
entity sim
output RESET;
output SC;
output RXD;
output CTS;
output BUSACK;
output Z80MREQ;
output Z80IORQ;
output Z80RD;
output Z80WR;

output BUSREQ;
output TXD;
output RTS;
output WE;
output RD;
output WP;
output SCL;
output CPURESET;
output SCOUT;

output ADDRESS[16];
output DATA[8];
output SDA;

input  simres;

{ ------------------------------------------------------------------------- }
{     検査端子                                                               }
{ ------------------------------------------------------------------------- }
output TB0P;
output TB1P[2];
output TB2P[8];
output TB3P[3];
output TB4P[8];
output TB5P;
output TB6P[4];
output TB7P[11];
output TB8P[8];
output TB9P;
output TB10P[8];
output TB11P[8];
output TB12P[8];
output TB13P;
output TB14P[16];

{ ------------------------------------------------------------------------- }
{     内部信号                                                               }
{ ------------------------------------------------------------------------- }
bitr  tc[16];
bitn  nodeRESET;
bitn  nodeSC;
bitn  nodeSREQP;
bitn  nodeSDATA[8];
bitn  nodeCTS;
bitn  nodeSEND[2];
bitr  sp[2];
bitr  sdp[3];
bitr  scount[8];
bitn  EXPprobBUSY;
bitn  EXPoutADDRESS[16];
```

```
bitn  rxd;
bitn  nodeTXD;
bitn  proRECEIVE[10];
bitn  nodeWE;
bitn  EXPoutDATA[8];
bitr  mem0DATA[8];
bitr  mem1DATA[8];
bitr  mem2DATA[8];
bitn  EXPoutSDA;

{ ------------------------------------------------------------------------ }
{    出力代入                                                             }
{ ------------------------------------------------------------------------ }
   RESET  =nodeRESET;
   SC     =nodeSC;
   CTS    =!nodeCTS;
   TXD    =nodeTXD;
   WE     =nodeWE;

{ ------------------------------------------------------------------------ }
{    検査端子代入                                                         }
{ ------------------------------------------------------------------------ }
   TB0P=nodeSEND.1;
   TB1P=sp;
   TB2P=scount;
   TB3P=sdp;
   TB4P=nodeSDATA;
   TB5P=nodeSREQP;
   TB8P=proRECEIVE.0:7;
   TB9P=proRECEIVE.9;
   TB10P=mem0DATA;
   TB11P=mem1DATA;
   TB12P=mem2DATA;
   TB13P=EXPoutSDA;
   TB14P=tc;

{ ------------------------------------------------------------------------ }
{    実効譜引用                                                           }
{ ------------------------------------------------------------------------ }
   part main(RESET,SC,RXD,CTS,BUSACK,Z80MREQ,Z80IORQ,Z80RD,Z80WR
            ,BUSREQ,nodeTXD,RTS,nodeWE,RD,WP,SCL,CPURESET,SCOUT
            ,ADDRESS,DATA,SDA
            ,EXPprobBUSY,EXPoutADDRESS,EXPoutDATA,EXPoutSDA)

{ ------------------------------------------------------------------------ }
{    送信手続き譜導入                                                     }
{ ------------------------------------------------------------------------ }
   nodeSEND=LDL07E16A(nodeRESET,nodeCTS,nodeSDATA,nodeSC,nodeSREQP);

   nodeSEND.2=1;

{ ------------------------------------------------------------------------ }
{    受信手続き譜導入                                                     }
{ ------------------------------------------------------------------------ }
   proRECEIVE=LDL07F02A(nodeRESET,rxd,nodeSC);

   proRECEIVE.10=1;

{ ------------------------------------------------------------------------ }
{    送信受付行程                                                         }
{ ------------------------------------------------------------------------ }
   if (nodeRESET)
      sp=0;
   else
      switch(sp)
```

```
         case 0:
            if (nodeSEND.1) sp=1; endif
         case 1:
            if (nodeSEND.1)
               sp=sp;
            else
               sp=2;
            endif
         case 2:
            sp=0;
      endswitch
   endif

{ ------------------------------------------------------------------------ }
{    ローカルエコー待ち行程                                    }
{ ------------------------------------------------------------------------ }
   switch(sdp)
      case 0: sdp.0=sp.1;
      case 1:
         if (EXPprobBUSY)
            sdp=sdp;
         else
            sdp=2;
         endif

      case 2: sdp=4;
      case 4: sdp=0;
   endswitch

{ ------------------------------------------------------------------------ }
{    送信計数                                                }
{ ------------------------------------------------------------------------ }
   if (nodeRESET)
      scount=0;
   else
      if (sdp.1)
         scount=scount+1;
      else
         scount=scount;
      endif
   endif

{ ------------------------------------------------------------------------ }
{    送信データ                                                }
{ ------------------------------------------------------------------------ }
   switch(scount)
      case 0:  nodeSDATA=0x53;     { S }
      case 1:  nodeSDATA=0x20;     { sp }
      case 2:  nodeSDATA=0x31;     { 1 }
      case 3:  nodeSDATA=0x32;     { 2 }
      case 4:  nodeSDATA=0x33;     { 3 }
      case 5:  nodeSDATA=0x34;     { 4 }
      case 6:  nodeSDATA=0x0d;     { cr }

      case 7:  nodeSDATA=0x32;     { 2 }
      case 8:  nodeSDATA=0x33;     { 3 }
      case 9:  nodeSDATA=0x0d;     { cr }

      case 10: nodeSDATA=0x34;     { 4 }
      case 11: nodeSDATA=0x35;     { 5 }
      case 12: nodeSDATA=0x0d;     { cr }

      case 13:  nodeSDATA=0x2e;     { . }
      case 14:  nodeSDATA=0x0d;     { cr }
```

```
      case 15:  nodeSDATA=0x52;     { R }
      case 16:  nodeSDATA=0x45;     { E }
      case 17:  nodeSDATA=0x53;     { S }
      case 18:  nodeSDATA=0x0d;     { cr }
   endswitch

{ ------------------------------------------------------------------------ }
{    送信要求                                                             }
{ ------------------------------------------------------------------------ }
   if (scount<=6)
      nodeSREQP=sdp.2;
   endif

   switch(tc)
      case 2100:  nodeSREQP=1;
      case 16200: nodeSREQP=1;
      case 27500: nodeSREQP=1;
      case 38800: nodeSREQP=1;
      case 41700: nodeSREQP=1;
   endswitch

   switch(scount)
      case 8:  nodeSREQP=sdp.2;
      case 9:  nodeSREQP=sdp.2;
      case 11: nodeSREQP=sdp.2;
      case 12: nodeSREQP=sdp.2;
      case 14: nodeSREQP=sdp.2;
      case 16: nodeSREQP=sdp.2;
      case 17: nodeSREQP=sdp.2;
      case 18: nodeSREQP=sdp.2;
   endswitch


{ ------------------------------------------------------------------------ }
{    メモリ出力                                                           }
{ ------------------------------------------------------------------------ }
   switch(EXPoutADDRESS)
      case 0x1234: DATA=mem0DATA;
      case 0x1235: DATA=mem1DATA;
      case 0x1236: DATA=mem2DATA;
   endswitch

{ ------------------------------------------------------------------------ }
{    メモリ 0                                                             }
{ ------------------------------------------------------------------------ }
   if (nodeRESET)
      mem0DATA=0x56;
   else
      if (nodeWE)
         switch(EXPoutADDRESS)
            case 0x1234: mem0DATA=EXPoutDATA;
            default: mem0DATA=mem0DATA;
         endswitch
      else
         mem0DATA=mem0DATA;
      endif
   endif

{ ------------------------------------------------------------------------ }
{    メモリ 1                                                             }
{ ------------------------------------------------------------------------ }
   if (nodeRESET)
      mem1DATA=0x78;
   else
```

```
      if (nodeWE)
         switch(EXPoutADDRESS)
            case 0x1235: mem1DATA=EXPoutDATA;
            default: mem1DATA=mem1DATA;
         endswitch
      else
         mem1DATA=mem1DATA;
      endif
   endif

{ ------------------------------------------------------------------------ }
{    メモリ 2                                                             }
{ ------------------------------------------------------------------------ }
   if (nodeRESET)
      mem2DATA=0x9a;
   else
      if (nodeWE)
         switch(EXPoutADDRESS)
            case 0x1236: mem2DATA=EXPoutDATA;
            default: mem2DATA=mem2DATA;
         endswitch
      else
         mem2DATA=mem2DATA;
      endif
   endif

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

   if (tc<5) nodeRESET=1; endif

   nodeSC=tc.0;
   nodeCTS=1;
   RXD=nodeSEND.0;
   BUSACK=0;
   rxd=nodeTXD;

ende
```

```
{ ========================================================================== }
{    メモリ表示コマンド検証                                                }
{ ========================================================================== }
entity sim
output RESET;
output SC;
output RXD;
output CTS;
output BUSACK;
output Z80MREQ;
output Z80IORQ;
output Z80RD;
output Z80WR;

output BUSREQ;
output TXD;
output RTS;
output WE;
output RD;
output WP;
output SCL;
output CPURESET;
output SCOUT;

output ADDRESS[16];
output DATA[8];
output SDA;

input  simres;

{ ------------------------------------------------------------------------ }
{    検査端子                                                              }
{ ------------------------------------------------------------------------ }
output TB0P;
output TB1P[2];
output TB2P[8];
output TB3P[3];
output TB4P[8];
output TB5P;
output TB6P[4];
output TB7P[11];
output TB8P[8];
output TB9P;
output TB10P[8];
output TB11P[8];
output TB12P[8];
output TB13P;
output TB14P[20];

{ ------------------------------------------------------------------------ }
{    内部信号                                                              }
{ ------------------------------------------------------------------------ }
bitr  tc[20];
bitn  nodeRESET;
bitn  nodeSC;
bitn  nodeSREQP;
bitn  nodeSDATA[8];
bitn  nodeCTS;
bitn  nodeSEND[2];
bitr  sp[2];
bitr  sdp[3];
bitr  scount[8];
bitn  EXPprobBUSY;
bitn  EXPoutADDRESS[16];
```

```
bitn  rxd;
bitn  nodeTXD;
bitn  proRECEIVE[10];
bitn  nodeWE;
bitn  EXPoutDATA[8];
bitr  mem0DATA[8];
bitr  mem1DATA[8];
bitr  mem2DATA[8];
bitn  EXPoutSDA;
bitn  nodeRD;

{ ------------------------------------------------------------------------ }
{    出力代入                                                             }
{ ------------------------------------------------------------------------ }
   RESET  =nodeRESET;
   SC     =nodeSC;
   CTS    =!nodeCTS;
   TXD    =nodeTXD;
   WE     =nodeWE;
   RD     =nodeRD;

{ ------------------------------------------------------------------------ }
{    検査端子代入                                                         }
{ ------------------------------------------------------------------------ }
   TB0P=nodeSEND.1;
   TB1P=sp;
   TB2P=scount;
   TB3P=sdp;
   TB4P=nodeSDATA;
   TB5P=nodeSREQP;
   TB8P=proRECEIVE.0:7;
   TB9P=proRECEIVE.9;
   TB10P=mem0DATA;
   TB11P=mem1DATA;
   TB12P=mem2DATA;
   TB13P=EXPoutSDA;
   TB14P=tc;

{ ------------------------------------------------------------------------ }
{    実効譜引用                                                           }
{ ------------------------------------------------------------------------ }
   part main(RESET,SC,RXD,CTS,BUSACK,Z80MREQ,Z80IORQ,Z80RD,Z80WR
            ,BUSREQ,nodeTXD,RTS,nodeWE,nodeRD,WP,SCL,CPURESET,SCOUT
            ,ADDRESS,DATA,SDA
            ,EXPprobBUSY,EXPoutADDRESS,EXPoutDATA,EXPoutSDA)

{ ------------------------------------------------------------------------ }
{    送信手続き譜導入                                                     }
{ ------------------------------------------------------------------------ }
   nodeSEND=LDL07E16A(nodeRESET,nodeCTS,nodeSDATA,nodeSC,nodeSREQP);

   nodeSEND.2=1;

{ ------------------------------------------------------------------------ }
{    受信手続き譜導入                                                     }
{ ------------------------------------------------------------------------ }
   proRECEIVE=LDL07F02A(nodeRESET,rxd,nodeSC);

   proRECEIVE.10=1;

{ ------------------------------------------------------------------------ }
{    送信受付行程                                                         }
{ ------------------------------------------------------------------------ }
   if (nodeRESET)
      sp=0;
```

```
   else
      switch(sp)
         case 0:
            if (nodeSEND.1) sp=1; endif
         case 1:
            if (nodeSEND.1)
               sp=sp;
            else
               sp=2;
            endif
         case 2:
            sp=0;
      endswitch
   endif

{ ------------------------------------------------------------------------ }
{    ローカルエコー待ち行程                                    }
{ ------------------------------------------------------------------------ }
   switch(sdp)
      case 0: sdp.0=sp.1;
      case 1:
         if (EXPprobBUSY)
            sdp=sdp;
         else
            sdp=2;
         endif

      case 2: sdp=4;
      case 4: sdp=0;
   endswitch

{ ------------------------------------------------------------------------ }
{    送信計数                                                }
{ ------------------------------------------------------------------------ }
   if (nodeRESET)
      scount=0;
   else
      if (sdp.1)
         scount=scount+1;
      else
         scount=scount;
      endif
   endif

{ ------------------------------------------------------------------------ }
{    送信データ                                                }
{ ------------------------------------------------------------------------ }
   switch(scount)
      case 0:  nodeSDATA=0x44;     { D }
      case 1:  nodeSDATA=0x20;     { sp }
      case 2:  nodeSDATA=0x31;     { 1 }
      case 3:  nodeSDATA=0x32;     { 2 }
      case 4:  nodeSDATA=0x33;     { 3 }
      case 5:  nodeSDATA=0x34;     { 4 }
      case 6:  nodeSDATA=0x20;     { sp }
      case 7:  nodeSDATA=0x31;     { 5 }
      case 8:  nodeSDATA=0x32;     { 6 }
      case 9:  nodeSDATA=0x38;     { 7 }
      case 10: nodeSDATA=0x30;     { 8 }
      case 11: nodeSDATA=0x0d;     { cr }
      case 12: nodeSDATA=0x0d;     { cr }
   endswitch

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

```
{    送信要求                                                    }
{ ------------------------------------------------------------------------ }
   if (scount<=11)
      nodeSREQP=sdp.2;
   endif

   switch(tc)
      case 2100:  nodeSREQP=1;
      case 32000:  nodeSREQP=1;
   endswitch

{ ------------------------------------------------------------------------ }
{    メモリ出力                                                  }
{ ------------------------------------------------------------------------ }
   if (!nodeRD)
      switch(EXPoutADDRESS)
         case 0x1234: DATA=0x12;
         case 0x1235: DATA=0x34;
         case 0x1236: DATA=0x56;
      endswitch
   else
      DATA=0xff;
   endif

{ ------------------------------------------------------------------------ }
{    メモリ 0                                                    }
{ ------------------------------------------------------------------------ }
   if (nodeRESET)
      mem0DATA=0x56;
   else
      if (nodeWE)
         switch(EXPoutADDRESS)
            case 0x1234: mem0DATA=EXPoutDATA;
            default: mem0DATA=mem0DATA;
         endswitch
      else
         mem0DATA=mem0DATA;
      endif
   endif

{ ------------------------------------------------------------------------ }
{    メモリ 1                                                    }
{ ------------------------------------------------------------------------ }
   if (nodeRESET)
      mem1DATA=0x78;
   else
      if (nodeWE)
         switch(EXPoutADDRESS)
            case 0x1235: mem1DATA=EXPoutDATA;
            default: mem1DATA=mem1DATA;
         endswitch
      else
         mem1DATA=mem1DATA;
      endif
   endif

{ ------------------------------------------------------------------------ }
{    メモリ 2                                                    }
{ ------------------------------------------------------------------------ }
   if (nodeRESET)
      mem2DATA=0x9a;
   else
      if (nodeWE)
         switch(EXPoutADDRESS)
            case 0x1236: mem2DATA=EXPoutDATA;
```

```
            default: mem2DATA=mem2DATA;
         endswitch
      else
         mem2DATA=mem2DATA;
      endif
   endif

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

   if (tc<5) nodeRESET=1; endif

   nodeSC=tc.0;
   nodeCTS=1;
   RXD=nodeSEND.0;
   BUSACK=0;
   rxd=nodeTXD;

ende
```

```
{ ========================================================================== }
{    EEPROM 書き込み                                                       }
{ ========================================================================== }
entity sim
output RESET;
output SC;
output RXD;
output CTS;
output BUSACK;
output Z80MREQ;
output Z80IORQ;
output Z80RD;
output Z80WR;

output BUSREQ;
output TXD;
output RTS;
output WE;
output RD;
output WP;
output SCL;
output CPURESET;
output SCOUT;

output ADDRESS[16];
output DATA[8];
output SDA;

input  simres;

{ -------------------------------------------------------------------------- }
{    検査端子                                                              }
{ -------------------------------------------------------------------------- }
output TB0P;
output TB1P[2];
output TB2P[8];
output TB3P[3];
output TB4P[8];
output TB5P;
output TB6P[4];
output TB7P[11];
output TB8P[8];
output TB9P;
output TB10P[8];
output TB11P[8];
output TB12P[8];
output TB13P;
output TB14P[16];
output TB15P[2];

{ -------------------------------------------------------------------------- }
{    内部信号                                                              }
{ -------------------------------------------------------------------------- }
bitr  tc[16];
bitn  nodeRESET;
bitn  nodeSC;
bitn  nodeSREQP;
bitn  nodeSDATA[8];
bitn  nodeCTS;
bitn  nodeSEND[2];
bitr  sp[2];
bitr  sdp[3];
bitr  scount[8];
bitn  EXPprobBUSY;
```

```
bitn  EXPoutADDRESS[16];
bitn  rxd;
bitn  nodeTXD;
bitn  proRECEIVE[10];
bitn  nodeWE;
bitn  nodeRD;
bitn  EXPoutDATA[8];
bitn  cs[3];
bitn  nodeSCL;
bitn  proI2CMEM[6];
bitn  EXPoutSDA;
bitr  mem0DATA[8];
bitr  mem1DATA[8];
bitr  mem2DATA[8];
bitn  memmode;
bitn  memreset;

{ ------------------------------------------------------------------- }
{    出力代入                                                         }
{ ------------------------------------------------------------------- }
   RESET  =nodeRESET;
   SC     =nodeSC;
   CTS    =!nodeCTS;
   TXD    =nodeTXD;
   WE     =nodeWE;
   RD     =nodeRD;
   SCL    =nodeSCL;

{ ------------------------------------------------------------------- }
{    検査端子代入                                                     }
{ ------------------------------------------------------------------- }
   TB0P=nodeSEND.1;
   TB1P=sp;
   TB2P=scount;
   TB3P=sdp;
   TB4P=nodeSDATA;
   TB5P=nodeSREQP;
   TB8P=proRECEIVE.0:7;
   TB9P=proRECEIVE.9;
   TB10P=mem0DATA;
   TB11P=mem1DATA;
   TB12P=mem2DATA;
   TB13P=EXPoutSDA;
   TB14P=tc;

{ ------------------------------------------------------------------- }
{    実効譜引用                                                       }
{ ------------------------------------------------------------------- }
   part main(RESET,SC,RXD,CTS,BUSACK,Z80MREQ,Z80IORQ,Z80RD,Z80WR
            ,BUSREQ,nodeTXD,RTS,nodeWE,nodeRD,WP,nodeSCL,CPURESET,SCOUT
            ,ADDRESS,DATA,SDA
            ,EXPprobBUSY,EXPoutADDRESS,EXPoutDATA,EXPoutSDA)

{ ------------------------------------------------------------------- }
{    送信手続き譜導入                                                 }
{ ------------------------------------------------------------------- }
   nodeSEND=LDL07E16A(nodeRESET,nodeCTS,nodeSDATA,nodeSC,nodeSREQP);

   nodeSEND.2=1;

{ ------------------------------------------------------------------- }
{    受信手続き譜導入                                                 }
{ ------------------------------------------------------------------- }
   proRECEIVE=LDL07F02A(nodeRESET,rxd,nodeSC);
```

```
   proRECEIVE.10=1;

{ ------------------------------------------------------------------------ }
{    I2C メモリ手続き譜導入                                    }
{ ------------------------------------------------------------------------ }
   proI2CMEM=ldl06l17a(nodeRESET,cs,nodeSCL,EXPoutSDA);

   proI2CMEM.6=1;

   cs=0;

{ ------------------------------------------------------------------------ }
{    送信受付行程                                              }
{ ------------------------------------------------------------------------ }
   if (nodeRESET)
      sp=0;
   else
      switch(sp)
         case 0:
            if (nodeSEND.1) sp=1; endif
         case 1:
            if (nodeSEND.1)
               sp=sp;
            else
               sp=2;
            endif
         case 2:
            sp=0;
      endswitch
   endif

{ ------------------------------------------------------------------------ }
{    ローカルエコー待ち行程                                    }
{ ------------------------------------------------------------------------ }
   switch(sdp)
      case 0: sdp.0=sp.1;
      case 1:
         if (EXPprobBUSY)
            sdp=sdp;
         else
            sdp=2;
         endif

      case 2: sdp=4;
      case 4: sdp=0;
   endswitch

{ ------------------------------------------------------------------------ }
{    送信計数                                                  }
{ ------------------------------------------------------------------------ }
   if (nodeRESET)
      scount=0;
   else
      if (sdp.1)
         scount=scount+1;
      else
         scount=scount;
      endif
   endif

{ ------------------------------------------------------------------------ }
{    送信データ                                                }
{ ------------------------------------------------------------------------ }
  switch(scount)
      case 0:  nodeSDATA=0x45;     { E }
```

```
      case 1:  nodeSDATA=0x45;     { E }
      case 2:  nodeSDATA=0x50;     { P }
      case 3:  nodeSDATA=0x57;     { W }
      case 4:  nodeSDATA=0x20;     { sp }
      case 5:  nodeSDATA=0x46;     { f }
      case 6:  nodeSDATA=0x0d;     { cr }

      case 7:  nodeSDATA=0x45;     { E }
      case 8:  nodeSDATA=0x45;     { E }
      case 9:  nodeSDATA=0x50;     { P }
      case 10: nodeSDATA=0x52;     { R }
      case 11: nodeSDATA=0x20;     { sp }
      case 12: nodeSDATA=0x46;     { f }
      case 13: nodeSDATA=0x0d;     { cr }
   endswitch

{ ------------------------------------------------------------------------ }
{    送信要求                                                 }
{ ------------------------------------------------------------------------ }
   if (scount<=6)
      nodeSREQP=sdp.2;
   endif

   if ((scount>=8)&(scount<=13))
      nodeSREQP=sdp.2;
   endif

   switch(tc)
      case 2100: nodeSREQP=1;
      case 9400: nodeSREQP=1;
   endswitch

{ ------------------------------------------------------------------------ }
{    メモリ出力                                               }
{ ------------------------------------------------------------------------ }
   if (!nodeRD)
      switch(EXPoutADDRESS)
         case 0x1234: DATA=mem0DATA;
         case 0x1235: DATA=mem1DATA;
         case 0x1236: DATA=mem2DATA;
         default:  DATA=0x55;
      endswitch
   else
      DATA=0xff;
   endif

{ ------------------------------------------------------------------------ }
{    メモリ 0                                                 }
{ ------------------------------------------------------------------------ }
   if (nodeRESET|memreset)
      switch(memmode)
         case 0:  mem0DATA=0;
         default: mem0DATA=0x56;
      endswitch
   else
      if (!nodeWE)
         switch(EXPoutADDRESS)
            case 0x1234: mem0DATA=EXPoutDATA;
            default: mem0DATA=mem0DATA;
         endswitch
      else
         mem0DATA=mem0DATA;
      endif
   endif
```

```
{ ------------------------------------------------------------------------ }
{    メモリ 1                                                             }
{ ------------------------------------------------------------------------ }
   if (nodeRESET|memreset)
      switch(memmode)
         case 0:  mem1DATA=0;
         default: mem1DATA=0x78;
      endswitch
   else
      if (!nodeWE)
         switch(EXPoutADDRESS)
            case 0x1235: mem1DATA=EXPoutDATA;
            default: mem1DATA=mem1DATA;
         endswitch
      else
         mem1DATA=mem1DATA;
      endif
   endif

{ ------------------------------------------------------------------------ }
{    メモリ 2                                                             }
{ ------------------------------------------------------------------------ }
   if (nodeRESET|memreset)
      switch(memmode)
         case 0:  mem2DATA=0;
         default: mem2DATA=0x9a;
      endswitch
   else
      if (!nodeWE)
         switch(EXPoutADDRESS)
            case 0x1236: mem2DATA=EXPoutDATA;
            default: mem2DATA=mem2DATA;
         endswitch
      else
         mem2DATA=mem2DATA;
      endif
   endif

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

   if (tc<5) nodeRESET=1; memmode=1; endif

   switch(tc)
      case 1800: memmode=1; memreset=1;
   endswitch

   switch(tc)
      case 9400: memmode=0; memreset=1;
   endswitch

   nodeSC=tc.0;
   nodeCTS=1;
   RXD=nodeSEND.0;
   BUSACK=0;
   rxd=nodeTXD;
   SDA=proI2CMEM.0;

   if ((tc>8413)&(tc<8425)) SDA=1; endif

ende
```

endlogic