# LDL08B10A

LDLABO

2008 年 6 月 13 日

# 目次

# 図目次

# 表目次

# 第 1 章

# 論理譜

```
{ ========================================================================= }
{    ディスク・リード                                    }
{ ========================================================================= }
logicname LDL08B10A

{
```

```
}
{ ========================================================================== }
{    16 ビット CRC                                                          }
{ ========================================================================== }
procedure crc

{ -------------------------------------------------------------------------- }
{    入力                                                                   }
{ -------------------------------------------------------------------------- }
input RESET;
input crcin;
input ep;

{ -------------------------------------------------------------------------- }
{    出力                                                                   }
{ -------------------------------------------------------------------------- }
output Q[16];

{ -------------------------------------------------------------------------- }
{    内部信号                                                               }
{ -------------------------------------------------------------------------- }
bitr q[16];
bitn xor1;
bitn xor2;
bitn xor3;

{ -------------------------------------------------------------------------- }
{    出力代入                                                               }
{ -------------------------------------------------------------------------- }
   Q=q;

{ -------------------------------------------------------------------------- }
{    CRC 演算                                                               }
{ -------------------------------------------------------------------------- }
   if (RESET)
      q=0;
   else
      if (ep)
         q.1:4=q.0:3;
         q.0=xor1;
         q.6:11=q.5:10;
         q.5=xor2;
         q.13:15=q.12:14;
         q.12=xor3;
      else
         q=q;
      endif
   endif

   switch(crcin,q.15)
      case 1,0: xor1=1;
      case 0,1: xor1=1;
   endswitch

   switch(q.4,xor1)
      case 1,0: xor2=1;
      case 0,1: xor2=1;
   endswitch

   switch(q.11,xor1)
      case 1,0: xor3=1;
      case 0,1: xor3=1;
   endswitch

endp
```

{

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

{ -------------------------------------------------------------------------- }
{    入力                                                                  }
{ -------------------------------------------------------------------------- }
input  RESET;              { 初期化 }

input  MFM;                { CPU : 単密/倍密 切り替え }
input  READON;             { CPU : ディスク読み出し指示 }
input  WRITEON;            { CPU : ディスク書き込み指示 }
input  SPT[2];             { CPU : トラック当たりのセクタ数 }
input  READSECNO[5];       { CPU : 目的セクタ番号 }
input  TRYCOUNT[4];        { CPU : 再試行回数 }
input  CERRST;             { CPU : エラー初期化 }
input  DRVTYPE[2];         { CPU : ドライブ種類 }
input  INDEXP;             { FDCF : インデックスパルス }
input  BYTEND;             { FDCF : 書き込みバイト最終ビット }
input  WINDOW;             { DSEP : データ窓 }
input  LOCK;               { DSEP : 同期確立 }
input  RDATA;              { FDD : ディスク読み出しデータ }

{ -------------------------------------------------------------------------- }
{    出力                                                                  }
{ -------------------------------------------------------------------------- }
output Q[98];

bitn   RELOCK ;            { 再同期指示 }
bitn   RMC[4];             { トラック行程 }
bitn   RDDATA[8];          { データパルス記憶シフトレジスタ }
bitn   RCHKSEC;            { セクタ番号一致指標 }
bitn   RCRC4P[2];          { データフィールド CRC 検査位置 }
bitn   RERROR[4];          { 読み出しエラー }
bitn   RCYNO[8];           { シリンダ番号 }
bitn   RHEDNO[8];          { ヘッド番号 }
bitn   RSECNO[8];          { セクタ番号 }
bitn   RSECL[8];           { セクタ長 }
bitn   RICRC1[8];          { 1 番目 IDCRC }
bitn   RICRC2[8];          { 2 番目 IDCRC }
bitn   RDCRC1[8];          { データフィールド CRC1 番目 }
bitn   RDCRC2[8];          { データフィールド CRC2 番目 }
bitn   RAM2;               { データマーク位置 }
bitn   RBIT8D;             { 読み出しバイト最終ビット }
bitn   RGAP3;              { セクタ終了から最初の GAP 位置 }
bitn   RCRC3;              { データフィールド CRC1 番目位置 }
bitn   RCRC4;              { データフィールド CRC2 番目位置 }
bitn   RSDATA[8];          { セクタ読み出しデータ }
bitn   RDEND;              { データフィールド最終データ位置 }

{ -------------------------------------------------------------------------- }
{    内部信号                                                              }
{ -------------------------------------------------------------------------- }
bitn   Relock ;            { 再同期位置 }
bitr   rddata[8];          { データパルス記憶シフトレジスタ }
bitn   rchksec;            { セクタ番号一致指標 }
bitr   rcrc4p[2];          { データフィールド CRC 検査位置 }
bitr   rerror[4];          { 読み出しエラー }
bitr   rcyno[8];           { シリンダ番号 }
```

```
bitr   rhedno[8];         { ヘッド番号 }
bitr   rsecno[8];         { セクタ番号 }
bitr   rsecl[8];          { セクタ長 }
bitr   ricrc1[8];         { 1 番目 IDCRC }
bitr   ricrc2[8];         { 2 番目 IDCRC }
bitr   rdcrc1[8];         { データフィールド CRC1 番目 }
bitr   rdcrc2[8];         { データフィールド CRC2 番目 }
bitr   ram2[2];           { データマーク位置 }
bitn   rbit8d;            { 読み出しバイト最終ビット }
bitn   rgap3;             { セクタ終了から最初の GAP 位置 }
bitn   rcrc3;             { データフィールド CRC1 番目位置 }
bitn   rcrc4;             { データフィールド CRC2 番目位置 }
bitr   rsdata[8];         { セクタ読み出しデータ }
bitn   rdend;             { データフィールド最終データ位置 }

bitr   readen;            { 同期確立後読み出し実行 }
bitr   rcs[2];            { クロック取得期間終了 }
bitr   rds[2];            { データ取得期間終了 }
bitr   rbc[5];            { ビット計数 }
bitr   rdset;             { データ記憶パルス }
bitr   rcset;             { クロック記憶パルス }
bitn   rbit7c;            { 最終ビットクロック取得期間終了 }
bitr   rcdata[8];         { クロックパルス記憶シフトレジスタ }
bitr   rct[12];           { バイト計数 }
bitn   rid1;              { ID フィールドシリンダ番号位置 }
bitn   rid2;              { ID フィールドヘッド番号位置 }
bitn   rid3;              { ID フィールドレコード番号位置 }
bitn   rid4;              { ID フィールドレコード長位置 }
bitn   rcrc1;             { ID フィールド CRC1 番目位置 }
bitn   rcrc2;             { ID フィールド CRC2 番目位置 }
bitr   rcrc2p[2];         { ID フィールド CRC 検査位置 }
bitr   rcrcdata[16];      { CRC 演算器 }
bitr   rindexc[4];        { インデックス計数 }
bitr   readon2;           { 読み出し開始 }
bitn   rchkindex;         { インデックス計数検査 }
bitr   rchkcrc;           { CRC 結果 }
bitr   rsdata[8];         { セクタ読み出しデータ }
bitr   trkseq[5];         { トラック行程 }
bitr   delaytrkseq[5];    { 遅れトラック行程 }
bitn   bitsyncro;         { ビット同期指標 }
bitn   crcen;             { CRC 演算実行 }

{ ------------------------------------------------------------------------ }
{    検査端子                                                    }
{ ------------------------------------------------------------------------ }
output T0P[8];   T0P=rddata;
output T1P;      T1P=rdset;
output T2P[2];   T2P=rds;
output T3P[8];   T3P=rcdata;
output T4P;      T4P=rbit8d;
output T5P[5];   T5P=trkseq;
output T6P[2];   T6P=rcs;
output T7P;      T7P=rcset;
output T8P;      T8P=readen;
output T9P;      T9P=readon2;
output T10P[12]; T10P=rct;
output T11P[8];  T11P=rcyno;
output T12P[8];  T12P=rsdata;
output T13P;     T13P=rcrc2;
output T14P[16]; T14P=rcrcdata;
output T15P;     T15P=crcen;
```

```
{ ------------------------------------------------------------------------ }
{    出力代入                                                            }
{ ------------------------------------------------------------------------ }
   Q.0=RELOCK;
   Q.1:4=RMC;
   Q.5:12=RDDATA;
   Q.13=RCHKSEC;
   Q.14:15=RCRC4P;
   Q.16:19=RERROR;
   Q.20:27=RCYNO;
   Q.28:35=RHEDNO;
   Q.36:43=RSECNO;
   Q.44:51=RSECL;
   Q.52:59=RICRC1;
   Q.60:67=RICRC2;
   Q.68:75=RDCRC1;
   Q.76:83=RDCRC2;
   Q.84=RAM2;
   Q.85=RBIT8D;
   Q.86=RGAP3;
   Q.87=RCRC3;
   Q.88=RCRC4;
   Q.89:96=RSDATA;
   Q.97=RDEND;

   RELOCK=Relock;
   RMC=trkseq.0:3;
   RDDATA=rddata.7:0;
   RCHKSEC=rchksec;
   RCRC4P=rcrc4p;
   RERROR=rerror;
   RCYNO=rcyno;
   RHEDNO=rhedno;
   RSECNO=rsecno;
   RSECL=rsecl;
   RICRC1=ricrc1;
   RICRC2=ricrc2;
   RDCRC1=rdcrc1;
   RDCRC2=rdcrc2;
   RAM2=ram2.0;
   RBIT8D=rbit8d;
   RGAP3=rgap3;
   RCRC3=rcrc3;
   RCRC4=rcrc4;
   RSDATA=rsdata;
   RDEND=rdend;

{ ------------------------------------------------------------------------ }
{    インデックス計数                                                    }
{ ------------------------------------------------------------------------ }
   if (RESET)
      rindexc=0;
   else
      if (READON|WRITEON)
         switch(rindexc)
            case 15: rindexc=rindexc;
            default:
               if (INDEXP)
                  rindexc=rindexc+1;
               else
                  rindexc=rindexc;
               endif
         endswitch
      endif
   endif
```

```
{ ------------------------------------------------------------------------ }
{    インデックス計数検査                                                  }
{ ------------------------------------------------------------------------ }
   if (READON)
      if (rindexc==TRYCOUNT) rchkindex=1; endif
   endif

{ ------------------------------------------------------------------------ }
{    読み出し開始                                                          }
{ ------------------------------------------------------------------------ }
   if (RESET)
      readon2=0;
   else
      if (READON|WRITEON)
         if (readon2)
            readon2=readon2;
         else
            switch(rindexc)
               case 1: readon2=1;
            endswitch
         endif
      endif
   endif

{ ------------------------------------------------------------------------ }
{    同期確立後読み出し実行                                                }
{ ------------------------------------------------------------------------ }
   if (RESET)
      readen=0;
   else
      if (LOCK)
         if (bitsyncro)
            readen=1;
         else
            if (Relock)
               readen=0;
            else
               readen=readen;
            endif
         endif
      endif
   endif

{ ------------------------------------------------------------------------ }
{    ビット計数                                                            }
{ ------------------------------------------------------------------------ }
   if (RESET|bitsyncro)
      rbc=30;
   else
      if (readen)
         if (rbit8d) rbc=15;
         else
            if (rcs.0)
               switch(rbc)
                  case 15: rbc=0;
                  default: rbc=rbc+1;
               endswitch
            else
               rbc=rbc;
            endif
         endif
      endif
   endif

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

```
{    データ取得期間終了                                    }
{ ------------------------------------------------------------------------ }
   switch(rds)
      case 0: if (WINDOW) rds=2; endif
      case 1: rds=0;
      case 2: if (WINDOW) rds=rds; else rds=1; endif
   endswitch

{ ------------------------------------------------------------------------ }
{    クロック取得期間終了                                  }
{ ------------------------------------------------------------------------ }
   switch(rcs)
      case 0: if (WINDOW) rcs=1; endif
      case 1: rcs=2;
      case 2: if (WINDOW) rcs=rcs; endif
   endswitch

{ ------------------------------------------------------------------------ }
{    読み出しバイト最終ビット                              }
{ ------------------------------------------------------------------------ }
   switch(rbc)
      case 7: if (rds.0) rbit8d=1; endif
   endswitch

{ ------------------------------------------------------------------------ }
{    最終ビットクロック取得期間終了                        }
{ ------------------------------------------------------------------------ }
   switch(rbc)
      case 6: if (rcs.0) rbit7c=1; endif
   endswitch

{ ------------------------------------------------------------------------ }
{    クロック記憶パルス                                    }
{ ------------------------------------------------------------------------ }
   if (RESET|rcs.0)
      rcset=0;
   else
      if (WINDOW)
         rcset=rcset;
      else
         if (RDATA)
            rcset=1;
         else
            rcset=rcset;
         endif
      endif
   endif

{ ------------------------------------------------------------------------ }
{    データ記憶パルス                                      }
{ ------------------------------------------------------------------------ }
   if (RESET|rds.0)
      rdset=0;
   else
      if (WINDOW)
         if (RDATA)
            rdset=1;
         else
            rdset=rdset;
         endif
      else
         rdset=rdset;
      endif
   endif
```

```
{ ------------------------------------------------------------------------ }
{    クロックパルス記憶シフトレジスタ                                    }
{ ------------------------------------------------------------------------ }
   if (rcs.0)
      rcdata.1:7=rcdata.0:6;
      rcdata.0=rcset;
   else
      rcdata=rcdata;
   endif

{ ------------------------------------------------------------------------ }
{    データパルス記憶シフトレジスタ                                      }
{ ------------------------------------------------------------------------ }
   if (rds.0)
      rddata.1:7=rddata.0:6;
      rddata.0=rdset;
   else
      rddata=rddata;
   endif

{ ------------------------------------------------------------------------ }
{    セクタ読み出しデータ                                                }
{ ------------------------------------------------------------------------ }
   if (RESET)
      rsdata=0;
   else
      if (rbit8d)
         rsdata=rddata.7:0;
      else
         rsdata=rsdata;
      endif
   endif

{ ------------------------------------------------------------------------ }
{    バイト計数                                                          }
{ ------------------------------------------------------------------------ }
   if (RESET)
      rct=0;
   else
      if (readen)
         if (bitsyncro)
            rct=0;
         else
            if (rbit8d)
               rct=rct+1;
            else
               rct=rct;
            endif
         endif
      endif
   endif

{ ------------------------------------------------------------------------ }
{    ID フィールド期間                                                   }
{ ------------------------------------------------------------------------ }
   if (rbit8d)
      switch(trkseq)
         case 2:
            switch(rct)
               case 0: rid1=1;  { ID  no.1 }
               case 1: rid2=1;  { ID  no.2 }
               case 2: rid3=1;  { ID  no.3 }
               case 3: rid4=1;  { ID  no.4 }
               case 4: rcrc1=1; { CRC no.1 }
```

```
                case 5: rcrc2=1; { CRC no.2 }
              endswitch
        endswitch
     endif

  { ------------------------------------------------------------------------ }
  {    シリンダ番号                                          }
  { ------------------------------------------------------------------------ }
     if (readon2)
        if (rid1)
           rcyno=rddata.7:0;
        else
           rcyno=rcyno;
        endif
     endif

  { ------------------------------------------------------------------------ }
  {    ヘッド番号                                          }
  { ------------------------------------------------------------------------ }
     if (readon2)
        if (rid2)
           rhedno=rddata.7:0;
        else
           rhedno=rhedno;
        endif
     endif

  { ------------------------------------------------------------------------ }
  {    セクタ番号                                          }
  { ------------------------------------------------------------------------ }
     if (readon2)
        if (rid3)
           rsecno=rddata.7:0;
        else
           rsecno=rsecno;
        endif
     endif

  { ------------------------------------------------------------------------ }
  {    セクタ長                                          }
  { ------------------------------------------------------------------------ }
     if (readon2)
        if (rid4)
           rsecl=rddata.7:0;
        else
           rsecl=rsecl;
        endif
     endif

  { ------------------------------------------------------------------------ }
  {    1 番目 IDCRC                                          }
  { ------------------------------------------------------------------------ }
     if (readon2)
        if (rcrc1)
           ricrc1=rddata.7:0;
        else
           ricrc1=ricrc1;
        endif
     endif

  { ------------------------------------------------------------------------ }
  {    2 番目 IDCRC                                          }
  { ------------------------------------------------------------------------ }
     if (readon2)
        if (rcrc2)
```

```
         ricrc2=rddata.7:0;
      else
         ricrc2=ricrc2;
      endif
   endif

{ ------------------------------------------------------------------------ }
{    CRC 演算実行                                                         }
{ ------------------------------------------------------------------------ }
   if (LOCK)
      if (rds.0) crcen=1; endif
   endif

{ ------------------------------------------------------------------------ }
{    CRC 演算器                                                           }
{ ------------------------------------------------------------------------ }
   rcrcdata=crc(Relock,rdset,crcen);

   rcrcdata.16=1;

{ ------------------------------------------------------------------------ }
{    CRC 結果                                                             }
{ ------------------------------------------------------------------------ }
   if (RESET)
      rchkcrc=0;
   else
      if (Relock)
         switch(rcrcdata)
            case 0: rchkcrc=1;
         endswitch
      else
         rchkcrc=rchkcrc;
      endif
   endif

{ ------------------------------------------------------------------------ }
{    データフィールド位置                                                 }
{ ------------------------------------------------------------------------ }
   switch(trkseq)
      case 3:
         switch(DRVTYPE)
            case 2:
               switch(SPT)
                  case 0:
                     switch(rct)
                        case 127: rdend=1; { last data }
                        case 128: rcrc3=1; { CRC no.1  }
                        case 129: rcrc4=1; { CRC no.2  }
                        case 130: rgap3=1; { GAP3      }
                     endswitch

                  case 1:
                     switch(rct)
                        case 255: rdend=1; { last data }
                        case 256: rcrc3=1; { CRC no.1  }
                        case 257: rcrc4=1; { CRC no.2  }
                        case 258: rgap3=1; { GAP3      }
                     endswitch

                  case 2:
                     switch(rct)
                        case 511: rdend=1; { last data }
                        case 512: rcrc3=1; { CRC no.1  }
                        case 513: rcrc4=1; { CRC no.2  }
```

```
                  case 514: rgap3=1; { GAP3      }
               endswitch
            endswitch

         case 1:
            switch(SPT)
               case 0:
                  switch(rct)
                     case 127: rdend=1; { last data }
                     case 128: rcrc3=1; { CRC no.1  }
                     case 129: rcrc4=1; { CRC no.2  }
                     case 130: rgap3=1; { GAP3      }
                  endswitch
               case 1:
                  switch(rct)
                     case 255: rdend=1; { last data }
                     case 256: rcrc3=1; { CRC no.1  }
                     case 257: rcrc4=1; { CRC no.2  }
                     case 258: rgap3=1; { GAP3      }
                  endswitch
               case 2:
                  switch(rct)
                     case 511: rdend=1; { last data }
                     case 512: rcrc3=1; { CRC no.1  }
                     case 513: rcrc4=1; { CRC no.2  }
                     case 514: rgap3=1; { GAP3      }
                  endswitch
            endswitch

         case 0:
            switch(SPT)
               case 0:
                  switch(rct)
                     case 127: rdend=1; { last data }
                     case 128: rcrc3=1; { CRC no.1  }
                     case 129: rcrc4=1; { CRC no.2  }
                     case 130: rgap3=1; { GAP3      }
                  endswitch
               case 1:
                  switch(rct)
                     case 511: rdend=1; { last data }
                     case 512: rcrc3=1; { CRC no.1  }
                     case 513: rcrc4=1; { CRC no.2  }
                     case 514: rgap3=1; { GAP3      }
                  endswitch
               case 2:                { 検証用 }
                  switch(rct)
                     case 4: rdend=1; { last data }
                     case 5: rcrc3=1; { CRC no.1  }
                     case 6: rcrc4=1; { CRC no.2  }
                     case 7: rgap3=1; { GAP3      }
                  endswitch
            endswitch

         case 3:
            switch(rct)
               case  3: rdend=1; { last data }
               case  4: rcrc3=1; { CRC no.1  }
               case  5: rcrc4=1; { CRC no.2  }
               case  6: rgap3=1; { GAP3      }
            endswitch
      endswitch
endswitch
```

```
{ ------------------------------------------------------------------------ }
{    ID フィールド CRC 検査位置                                            }
{ ------------------------------------------------------------------------ }
   if (READON)
      switch(rcrc2p)
         case 0: if (rcrc2) rcrc2p=1; endif
         case 1: if (rcrc2) rcrc2p=rcrc2p; else rcrc2p=2; endif
         case 2: rcrc2p=0;
      endswitch
   endif

{ ------------------------------------------------------------------------ }
{    データフィールド CRC 検査位置                                         }
{ ------------------------------------------------------------------------ }
   if (READON)
      switch(rcrc4p)
         case 0: if (rcrc4) rcrc4p=1; endif
         case 1: if (rcrc4) rcrc4p=rcrc4p; else rcrc4p=2; endif
         case 2: rcrc4p=0;
      endswitch
   endif

{ ------------------------------------------------------------------------ }
{    セクタ番号一致指標                                                    }
{ ------------------------------------------------------------------------ }
   if (READSECNO==rsecno.0:4) rchksec=1; endif

{ ------------------------------------------------------------------------ }
{    データマーク位置                                                      }
{ ------------------------------------------------------------------------ }
   switch(trkseq)
      case 2:
         switch(ram2)
            case 0: if (rds.0) ram2=1; endif
            case 1: if (BYTEND) ram2=2; else ram2=ram2; endif
            case 2: ram2=ram2;
         endswitch
   endswitch

{ ------------------------------------------------------------------------ }
{    データフィールド CRC1 番目                                            }
{ ------------------------------------------------------------------------ }
   if (rcrc3)
      rdcrc1=rddata.7:0;
   else
      rdcrc1=rdcrc1;
   endif

{ ------------------------------------------------------------------------ }
{    データフィールド CRC2 番目                                            }
{ ------------------------------------------------------------------------ }
   if (rcrc4)
      rdcrc2=rddata.7:0;
   else
      rdcrc2=rdcrc2;
   endif

{ ------------------------------------------------------------------------ }
{    再同期指示                                                            }
{ ------------------------------------------------------------------------ }
   switch(trkseq)
      case 1: Relock=1;                  { INDEX MARK }
      case 2:                            { ADDRESS MARK }
         switch(rct)
```

```
            case 5: if (rbit8d) Relock=1; endif
         endswitch
      case 3:                          { DATA MARK }
         if (rbit8d)
            switch(DRVTYPE)
               case 3:                 { 検証用 }
                  switch(rct)
                     case 5: Relock=1;
                  endswitch
               case 2:
                  switch(SPT)
                     case 0: switch(rct) case 130: Relock=1; endswitch
                     case 1: switch(rct) case 258: Relock=1; endswitch
                     case 2: switch(rct) case 514: Relock=1; endswitch
                  endswitch
               case 1:
                  switch(SPT)
                     case 0: switch(rct) case 130: Relock=1; endswitch
                     case 1: switch(rct) case 258: Relock=1; endswitch
                     case 2: switch(rct) case 514: Relock=1; endswitch
                  endswitch
               case 0:
                  switch(SPT)
                     case 0: switch(rct) case 514: Relock=1; endswitch
                     case 1: switch(rct) case 514: Relock=1; endswitch
                  endswitch
            endswitch
         endif
   endswitch

   if (INDEXP) Relock=1; endif

{ ------------------------------------------------------------------------ }
{    インデックスマーク補足不能                                          }
{ ------------------------------------------------------------------------ }
   if (CERRST|RESET)
      rerror.0=0;
   else
      if (rchksec)
         if (rchkindex)
            rerror.0=rerror.0;
         else
            rerror.0=1;
         endif
      else
         rerror.0=rerror.0;
      endif
   endif

{ ------------------------------------------------------------------------ }
{    ID フィールド CRC 不一致                                            }
{ ------------------------------------------------------------------------ }
   if (CERRST|RESET)
      rerror.1=0;
   else
      if (rchksec)
         if (rchkcrc)
            if (rcrc2p.1)
               rerror.1=1;
            else
               rerror.1=rerror.1;
            endif
         else
            rerror.1=rerror.1;
         endif
      else
```

```
         rerror.1=rerror.1;
      endif
   endif

{ ------------------------------------------------------------------------ }
{    データフィールド CRC 不一致                                          }
{ ------------------------------------------------------------------------ }
   if (CERRST|RESET)
      rerror.2=0;
   else
      if (rchksec)
         if (!rchkcrc)
            if (rcrc4p.1)
               rerror.2=1;
            else
               rerror.2=rerror.2;
            endif
         else
            rerror.2=rerror.2;
         endif
      else
         rerror.2=rerror.2;
      endif
   endif

{ ------------------------------------------------------------------------ }
{    Deleted Data Mark 検出                                               }
{ ------------------------------------------------------------------------ }
   if (CERRST|RESET)
      rerror.3=0;
   else
      if (rchksec)
         switch(trkseq)
            case 4: rerror.3=1;
            default: rerror.3=rerror.3;
         endswitch
      else
         rerror.3=rerror.3;
      endif
   endif

{ ------------------------------------------------------------------------ }
{    トラック行程                                                         }
{ ------------------------------------------------------------------------ }
   if (RESET|INDEXP)
      trkseq=0;
   else
      if (MFM)
         switch(trkseq)
            case 0:                                  { IM 0 待ち }
               switch(rddata.7:0,rcdata.7:0)
                  case 0xc2,0x14: trkseq=11;         { INDEX MARK }
               endswitch
            case 1:                                  { IM 通過 }
               if (Relock)
                  trkseq=8;
               else
                  trkseq=trkseq;
               endif
            case 2:                                  { ID フィールド }
               if (Relock)
                  trkseq=9;
               else
                  trkseq=trkseq;
               endif
```

```
case 3:                                  { データフィールド }
   if (Relock)
      trkseq=10;
   else
      trkseq=trkseq;
   endif
case 4:                                  { 削除データフィールド }
   if (Relock)
      trkseq=10;
   else
      trkseq=trkseq;
   endif
case 8:                                  { AM 0 待ち }
   switch(rddata.7:0,rcdata.7:0)
      case 0xa1,0x0a: trkseq=17;         { ADDRESS MARK }
      default: trkseq=trkseq;
   endswitch
case 9:                                  { DM 0 待ち }
   switch(rddata.7:0,rcdata.7:0)
      case 0xa1,0x0a: trkseq=23;         { DATA MARK }
      default: trkseq=trkseq;
   endswitch
case 10:                                 { セクタ終了 }
   trkseq=8;

case 11:                                 { IM 0 通過 }
   switch(rddata.7:0,rcdata.7:0)
      case 0xc2,0x14: trkseq=trkseq;     { INDEX MARK }
      default: trkseq=12;
   endswitch
case 12:                                 { IM 1 待ち }
   switch(rddata.7:0,rcdata.7:0)
      case 0xc2,0x14: trkseq=13;         { INDEX MARK }
      default: trkseq=trkseq;
   endswitch
case 13:                                 { IM 1 通過 }
   switch(rddata.7:0,rcdata.7:0)
      case 0xc2,0x14: trkseq=trkseq;     { INDEX MARK }
      default: trkseq=14;
   endswitch
case 14:                                 { IM 2 待ち }
   switch(rddata.7:0,rcdata.7:0)
      case 0xc2,0x14: trkseq=15;         { INDEX MARK }
      default: trkseq=trkseq;
   endswitch
case 15:                                 { IM 2 通過 }
   switch(rddata.7:0,rcdata.7:0)
      case 0xc2,0x14: trkseq=trkseq;     { INDEX MARK }
      default: trkseq=16;
   endswitch
case 16:                                 { IM 3 待ち }
   switch(rddata.7:0,rcdata.7:0)
      case 0xfc,0x01: trkseq=1;          { INDEX MARK }
      default: trkseq=trkseq;
   endswitch

case 17:                                 { AM 1 待ち }
   switch(rddata.7:0,rcdata.7:0)
      case 0xa1,0x0a: trkseq=trkseq;     { ADDRESS MARK }
      default: trkseq=18;
   endswitch
case 18:                                 { AM 1 }
   switch(rddata.7:0,rcdata.7:0)
      case 0xa1,0x0a: trkseq=19;         { ADDRESS MARK }
      default: trkseq=trkseq;
   endswitch
```

```
      case 19:                                  { AM 2 待ち }
         switch(rddata.7:0,rcdata.7:0)
            case 0xa1,0x0a: trkseq=trkseq;      { ADDRESS MARK }
            default: trkseq=20;
         endswitch
      case 20:                                  { AM 2 }
         switch(rddata.7:0,rcdata.7:0)
            case 0xa1,0x0a: trkseq=21;          { ADDRESS MARK }
            default: trkseq=trkseq;
         endswitch
      case 21:                                  { AM 3 待ち }
         switch(rddata.7:0,rcdata.7:0)
            case 0xa1,0x0a: trkseq=trkseq;      { ADDRESS MARK }
            default: trkseq=22;
         endswitch
      case 22:                                  { AM 3 }
         switch(rddata.7:0,rcdata.7:0)
            case 0xfe,0x00: trkseq=2;           { ADDRESS MARK }
            default: trkseq=trkseq;
         endswitch

      case 23:                                  { DM 0 通過 }
         switch(rddata.7:0,rcdata.7:0)
            case 0xa1,0x0a: trkseq=trkseq;      { DATA MARK }
            default: trkseq=24;
         endswitch
      case 24:                                  { DM 1 待ち }
         switch(rddata.7:0,rcdata.7:0)
            case 0xa1,0x0a: trkseq=25;          { DATA MARK }
            default: trkseq=trkseq;
         endswitch
      case 25:                                  { DM 1 通過 }
         switch(rddata.7:0,rcdata.7:0)
            case 0xa1,0x0a: trkseq=trkseq;      { DATA MARK }
            default: trkseq=26;
         endswitch
      case 26:                                  { DM 2 待ち }
         switch(rddata.7:0,rcdata.7:0)
            case 0xa1,0x0a: trkseq=27;          { DATA MARK }
            default: trkseq=trkseq;
         endswitch
      case 27:                                  { DM 2 通過 }
         switch(rddata.7:0,rcdata.7:0)
            case 0xa1,0x0a: trkseq=trkseq;      { DATA MARK }
            default: trkseq=28;
         endswitch
      case 28:                                  { DM 3 待ち }
         switch(rddata.7:0)
            case 0xfb: trkseq=3;                { DATA MARK }
            case 0xf8: trkseq=4;                { DATA MARK }
            default: trkseq=trkseq;
         endswitch


   endswitch
else
   switch(trkseq)
      case 0:                                { IM 待ち }
         switch(rddata.7:0,rcdata.7:0)
            case 0xfc,0xd7: trkseq=1;        { INDEX MARK }
         endswitch
      case 1:                                { IM 通過 }
         if (Relock)
            trkseq=8;
         else
```

```
                  trkseq=trkseq;
               endif
            case 2:                                { ID フィールド }
               if (Relock)
                  trkseq=9;
               else
                  trkseq=trkseq;
               endif
            case 3:                                { データフィールド }
               if (Relock)
                  trkseq=10;
               else
                  trkseq=trkseq;
               endif
            case 4:                                { 削除データフィールド }
               if (Relock)
                  trkseq=10;
               else
                  trkseq=trkseq;
               endif
            case 8:                                { AM 待ち }
               switch(rddata.7:0,rcdata.7:0)
                  case 0xfe,0xc7: trkseq=2;        { ID ADDRESS MARK (AM1) }
                  default: trkseq=trkseq;
               endswitch
            case 9:                                { DM 待ち }
               switch(rddata.7:0,rcdata.7:0)
                  case 0xfb,0xc7: trkseq=3;        { DATA MARK (AM2) }
                  case 0xf8,0xc7: trkseq=4;        { DELETE MARK (AM2) }
                  default: trkseq=trkseq;
               endswitch
            case 10:                               { セクタ終了 }
               trkseq=8;
            default: trkseq=trkseq;
         endswitch
      endif
   endif

{ ------------------------------------------------------------------------ }
{    遅れトラック行程                                                      }
{ ------------------------------------------------------------------------ }
   if (RESET)
      delaytrkseq=0;
   else
      delaytrkseq=trkseq;
   endif

{ ------------------------------------------------------------------------ }
{    ビット同期指標                                                        }
{ ------------------------------------------------------------------------ }
   if (delaytrkseq!=trkseq) bitsyncro=1; endif

ende

{ ======================================================================== }
{    機能実行譜                                                            }
{ ======================================================================== }
entity sim

{ ------------------------------------------------------------------------ }
{    出力                                                                  }
{ ------------------------------------------------------------------------ }
output  RESET;
output  MFM;
output  READON;
```

```
output  WRITEON;
output  SPT[2];
output  READSECNO[5];
output  TRYCOUNT[4];
output  CERRST;
output  DRVTYPE[2];
output  INDEXP;
output  BYTEND;
output  WINDOW;
output  LOCK;
output  RDATA;
output  Q[98];

{ ------------------------------------------------------------------------ }
{    検査端子                                                              }
{ ------------------------------------------------------------------------ }
output T0BP[3];
output T1BP[8];
output T2BP[8];
output T3BP[8];
output T4BP;

{ ------------------------------------------------------------------------ }
{    内部信号                                                              }
{ ------------------------------------------------------------------------ }
bitr tc[16];
bitr window[4];
bitn rdata;
bitn cdata;
bitr bitcount[3];
bitr bytecount[8];
bitn data[8];
bitn clkdata[8];
bitr lastbit;

{ ------------------------------------------------------------------------ }
{    内部接続                                                              }
{ ------------------------------------------------------------------------ }
bitn node_RESET;
bitn node_MFM;
bitn node_RELOCK;
bitr node_LOCK;
bitn node_Q[98];
bitn node_READSECNO[5];
bitn node_SPT[2];
bitn node_DRVTYPE[2];

{ ------------------------------------------------------------------------ }
{    実効譜引用                                                            }
{ ------------------------------------------------------------------------ }
   part main(node_RESET,MFM,READON,WRITEON,node_SPT,node_READSECNO,TRYCOUNT
            ,CERRST,node_DRVTYPE,INDEXP,BYTEND,WINDOW,node_LOCK,RDATA,node_Q)

{ ------------------------------------------------------------------------ }
{    検査端子代入                                                          }
{ ------------------------------------------------------------------------ }
   T0BP=bitcount;
   T1BP=bytecount;
   T2BP=clkdata;
   T3BP=data;
   T4BP=lastbit;

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

```
   RESET=node_RESET;
   MFM=node_MFM;
   LOCK=node_LOCK;
   Q=node_Q;
   READSECNO=node_READSECNO;
   SPT=node_SPT;
   DRVTYPE=node_DRVTYPE;

{ ------------------------------------------------------------------------ }
{    検査位置                                                             }
{ ------------------------------------------------------------------------ }
   tc=tc+1;

{ ------------------------------------------------------------------------ }
{    初期化                                                               }
{ ------------------------------------------------------------------------ }
   if (tc<5) node_RESET=1; endif

{ ------------------------------------------------------------------------ }
{    データ窓計数                                                         }
{ ------------------------------------------------------------------------ }
   if (node_RESET)
      window=0;
   else
      switch(window)
         case 15: window=0;
         default: window=window+1;
      endswitch
   endif

{ ------------------------------------------------------------------------ }
{    ビット計数                                                           }
{ ------------------------------------------------------------------------ }
   if (node_RESET)
      bitcount=0;
   else
      switch(window)
         case 15:
            switch(bitcount)
               case 7: bitcount=0;
               default: bitcount=bitcount+1;
            endswitch
         default: bitcount=bitcount;
      endswitch
   endif

{ ------------------------------------------------------------------------ }
{    バイト計数                                                           }
{ ------------------------------------------------------------------------ }
   if (node_RESET)
      bytecount=0;
   else
      switch(window)
         case 15:
            switch(bitcount)
               case 7: bytecount=bytecount+1;
               default: bytecount=bytecount;
            endswitch
         default: bytecount=bytecount;
      endswitch
   endif

{ ------------------------------------------------------------------------ }
{    データバイト                                                         }
{ ------------------------------------------------------------------------ }
```

```
if (node_MFM)
   switch(bytecount)
      case 0:  data=0x00;     { SYNC }
      case 1:  data=0xc2;     { IM 0 }
      case 2:  data=0xc2;     { IM 1 }
      case 3:  data=0xc2;     { IM 2 }
      case 4:  data=0xfc;     { IM 3 }
      case 5:  data=0x4e;     { GAP }
      case 6:  data=0x00;     { SYNC }
      case 7:  data=0xa1;     { AM 0 }
      case 8:  data=0xa1;     { AM 1 }
      case 9:  data=0xa1;     { AM 2 }
      case 10: data=0xfe;     { AM 3 }
      case 11: data=0x12;     { ID 0 }
      case 12: data=0x34;     { ID 1 }
      case 13: data=0x56;     { ID 2 }
      case 14: data=0x78;     { ID 3 }
      case 15: data=0xe1;     { CRC 1 }
      case 16: data=0x9b;     { CRC 2 }
      case 17: data=0x4e;     { GAP }
      case 18: data=0x00;     { SYNC }
      case 19: data=0xa1;     { DM 0 }
      case 20: data=0xa1;     { DM 1 }
      case 21: data=0xa1;     { DM 2 }
      case 22: data=0xfb;     { DM 3 }
      case 23: data=0x12;     { DATA 0 }
      case 24: data=0x34;     { DATA 1 }
      case 25: data=0x56;     { DATA 2 }
      case 26: data=0x78;     { DATA 3 }
      case 27: data=0xb5;     { CRC 1 }
      case 28: data=0xbd;     { CRC 2 (bf) }
      case 29: data=0xff;     { GAP }
      default: data=0x4e;
   endswitch
else
   switch(bytecount)
      case 0:  data=0x00;     { SYNC }
      case 1:  data=0xfc;     { IM }
      case 2:  data=0xff;     { GAP }
      case 3:  data=0x00;     { SYNC }
      case 4:  data=0xfe;     { AM1 }
      case 5:  data=0x12;     { ID 0 }
      case 6:  data=0x34;     { ID 1 }
      case 7:  data=0x56;     { ID 2 }
      case 8:  data=0x78;     { ID 3 }
      case 9:  data=0xee;     { CRC 1 }
      case 10: data=0x99;     { CRC 2 }
      case 11: data=0xff;     { GAP }
      case 12: data=0x00;     { SYNC }
      case 13: data=0xfb;     { AM2 }
      case 14: data=0x12;     { DATA 0 }
      case 15: data=0x34;     { DATA 1 }
      case 16: data=0x56;     { DATA 2 }
      case 17: data=0x78;     { DATA 3 }
      case 18: data=0xba;     { CRC 1 }
      case 19: data=0xbe;     { CRC 2 (bf) }
      case 20: data=0xff;     { GAP }
      default: data=0xff;
   endswitch
endif
```

```
{ ------------------------------------------------------------------------ }
{    同期確立                                                            }
{ ------------------------------------------------------------------------ }
   if (node_MFM)
      switch(bytecount)
         case 0:                       { SYNC }
            switch(bitcount)
               case 0:
               default: node_LOCK=1;
            endswitch
         case 6:                       { SYNC }
            switch(bitcount)
               case 0:
               default: node_LOCK=1;
            endswitch
         case 18:                      { SYNC }
            switch(bitcount)
               case 0:
               default: node_LOCK=1;
            endswitch
         default:
            if (node_RELOCK)
               node_LOCK=0;
            else
               node_LOCK=node_LOCK;
            endif
      endswitch
   else
      switch(bytecount)
         case 0:                       { SYNC }
            switch(bitcount)
               case 0:
               default: node_LOCK=1;
            endswitch
         case 3:                       { SYNC }
            switch(bitcount)
               case 0:
               default: node_LOCK=1;
            endswitch
         case 12:                      { SYNC }
            switch(bitcount)
               case 0:
               default: node_LOCK=1;
            endswitch
         default:
            if (node_RELOCK)
               node_LOCK=0;
            else
               node_LOCK=node_LOCK;
            endif
      endswitch
   endif

{ ------------------------------------------------------------------------ }
{    データビット                                                        }
{ ------------------------------------------------------------------------ }
   switch(bitcount)
      case 0: rdata=data.0;
      case 1: rdata=data.1;
      case 2: rdata=data.2;
      case 3: rdata=data.3;
      case 4: rdata=data.4;
      case 5: rdata=data.5;
      case 6: rdata=data.6;
      case 7: rdata=data.7;
```

```
   endswitch

{ ------------------------------------------------------------------------ }
{    クロックビット                                                        }
{ ------------------------------------------------------------------------ }
   switch(bitcount)
      case 0: cdata=clkdata.0;
      case 1: cdata=clkdata.1;
      case 2: cdata=clkdata.2;
      case 3: cdata=clkdata.3;
      case 4: cdata=clkdata.4;
      case 5: cdata=clkdata.5;
      case 6: cdata=clkdata.6;
      case 7: cdata=clkdata.7;
   endswitch

{ ------------------------------------------------------------------------ }
{    最終データビット                                                      }
{ ------------------------------------------------------------------------ }
   if (node_RESET)
      lastbit=0;
   else
      switch(window)
         case 15:
            switch(bitcount)
               case 7: lastbit=rdata;
               default: lastbit=lastbit;
            endswitch
         default: lastbit=lastbit;
      endswitch
   endif

{ ------------------------------------------------------------------------ }
{    クロックバイト                                                        }
{ ------------------------------------------------------------------------ }
   if (node_MFM)
      switch(bytecount)
         case 1:  clkdata=0x14;  { IM 0 }
         case 2:  clkdata=0x14;  { IM 1 }
         case 3:  clkdata=0x14;  { IM 2 }
         case 7:  clkdata=0x0a;  { AM 0 }
         case 8:  clkdata=0x0a;  { AM 1 }
         case 9:  clkdata=0x0a;  { AM 2 }
         case 19: clkdata=0x0a;  { DM 0 }
         case 20: clkdata=0x0a;  { DM 1 }
         case 21: clkdata=0x0a;  { DM 2 }

         default:
            if (lastbit|data.0) clkdata.0=0; else clkdata.0=1; endif
            if (data.0|data.1) clkdata.1=0; else clkdata.1=1; endif
            if (data.1|data.2) clkdata.2=0; else clkdata.2=1; endif
            if (data.2|data.3) clkdata.3=0; else clkdata.3=1; endif
            if (data.3|data.4) clkdata.4=0; else clkdata.4=1; endif
            if (data.4|data.5) clkdata.5=0; else clkdata.5=1; endif
            if (data.5|data.6) clkdata.6=0; else clkdata.6=1; endif
            if (data.6|data.7) clkdata.7=0; else clkdata.7=1; endif
      endswitch
   else
      switch(bytecount)
         case 1:  clkdata=0xD7;   { IM }
         case 4:  clkdata=0xC7;   { AM }
         case 13: clkdata=0xC7;   { DM }
         default: clkdata=0xff;
      endswitch
```

```
   endif

{ ------------------------------------------------------------------------ }
{    FDD 出力パルス                                                        }
{ ------------------------------------------------------------------------ }
   switch(window)
      case 3:  RDATA=cdata;
      case 4:  RDATA=cdata;
      case 11: RDATA=rdata;
      case 12: RDATA=rdata;
   endswitch

{ ------------------------------------------------------------------------ }
{    データ窓                                                              }
{ ------------------------------------------------------------------------ }
   if ((window>=8)&(window<=15)) WINDOW=1; endif

{ ------------------------------------------------------------------------ }
{    ディスク読み出し指示                                                  }
{ ------------------------------------------------------------------------ }
   READON=1;

{ ------------------------------------------------------------------------ }
{    インデックスパルス                                                    }
{ ------------------------------------------------------------------------ }
   switch(tc)
      case 6: INDEXP=1;
   endswitch

{ ------------------------------------------------------------------------ }
{    単密/倍密 切り替え                                                    }
{ ------------------------------------------------------------------------ }
   node_MFM=1;

{ ------------------------------------------------------------------------ }
{    目的セクタ番号                                                        }
{ ------------------------------------------------------------------------ }
   node_READSECNO=0x16;

{ ------------------------------------------------------------------------ }
{    トラック当たりのセクタ数                                              }
{ ------------------------------------------------------------------------ }
   node_SPT=3;

{ ------------------------------------------------------------------------ }
{    ドライブ種類                                                          }
{ ------------------------------------------------------------------------ }
   node_DRVTYPE=3;

{ ------------------------------------------------------------------------ }
{    再同期指示                                                            }
{ ------------------------------------------------------------------------ }
   node_RELOCK=node_Q.0;

ende

endlogic
```