論理設計自動化システム
マクロ展開

和田義弥，木下光彦，伊藤誠
（山梨大学工学部）

はじめに

論理設計の自動化を目的としたマクロ展開とマクロ・ライブラリ保守システムが完成したので報告する．このシステムは，HDL/Yトランスレータ[1]の出力または，直接エディタで作成したマクロ素子を素子レベルへ展開するものである．本システムにより，ブロック図レベルの回路入力が可能となる他，マクロ展開ライブラリを変更するだけで，種々のICファミリに展開できるため，実装部品との柔軟な対応が可能となる．

1. 設計自動化システムの構成

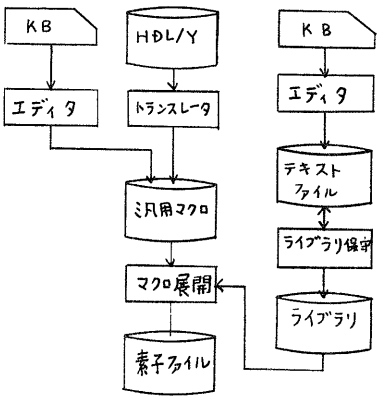


図.1 設計自動化システム

本設計自動化システムの全体[2]を図1に示す．汎用マクロは，HDL/Yトランスレータの出力形式であるが，文字型ファイルであるのでKB(鍵盤)からエディタを使って直接記述できる．

ライブラリは，KBからエディタによりテキストファイルが作られ，ライブラリ保守システムによって登録される．修正は，テキストファイルに戻してからエディタにより行なわれ，再登録する．

マクロ展開は，汎用マクロをライブラリの記述にしたがい素子ファイルに変換する．以後は，実装設計の段階である．

2. マクロ素子

マクロ素子は，デコーダ，カウンタ，レジスタ等の特定の論理機能に対応するものであり，実装部品と1対1に対応する必要はない[3]．マクロ素子記述の形式を図2に示す．最初にある'.'(ピリオド)がマクロ素子を示し，'('と')'の中を'/'で区切り入力実引数と出力実引数を区別して記述する．

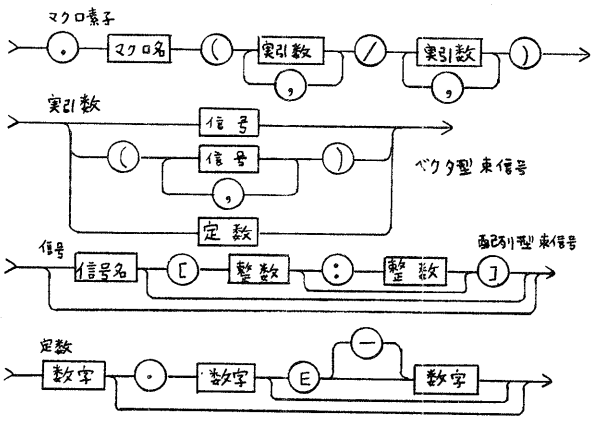
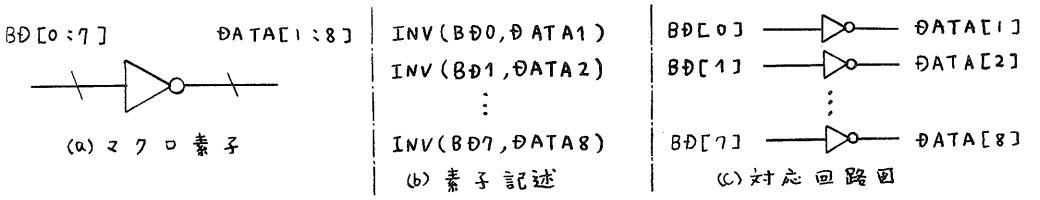


図2 マクロ素子記述

実引数として，単信号の他に配列型およびベクタ型の束信号および定数を記述できる．定数は，遅延時間や抵抗値などを表わすのに使う．

配列型束信号は，信号名に添字を付けたものであり，上限と下限を指定する．たとえば，BD[0:7]は，BD0，BD1，…，BD7の8本の信号からなる束信号である．ベクタ型束信号は，複数の信号を括弧でくくったものである．たとえば，(SYNC, RD, WT)は，SYNC, RD, WTからなる束信号であり，また，(D[1:3], D7, D[9:10])は，D1, D2, D3, D7, D9, D10の6本の信号からなる束信号である．



(a) マクロ素子　(b) 素子記述　(c) 対応回路図

図3 マクロ素子の展開例

マクロ素子 .INV(BD[0:7]/DATA[1:8]) とその展開例を図3に示す．BD[0]～BD[7]の8本の信号の否定をとりDATA[1]～DATA[8]に出力することを示すマクロ素子である(図3(a))．(b)はマクロ展開の出力，(c)は対応する回路図である．

3. マクロ素子定義

マクロ素子定義は，各マクロ素子記述を素子記述に展開する手順を示すものである．マクロ定義構文を図4(次ページ)に示す．一つのマクロ素子定義は，def(define)で始まり，マクロ名，仮引数，文が続きeod(end of define)で終わる．仮引数は，英字1文字とし，I,J,…,Nを整数型，O,P,Q,Rを実数型，他を信号型とし，区別して使う．文は，マクロ素子の展開手順を記述する．文には，素子ファイルへの出力を行なう出力文，代入文，IF文，WHILE文，エラーメッセージ用のERROR文がある．

マクロ素子定義記述の例として，先に述べた.INVのマクロ素子定義を図5に示す．INVはマクロ名であり，A,Bは仮引数である．Aが入力信号で，Bが出力信号を表わす．A,Bは信号型であるが，式の中で使われるときは信号の数を意味する．Aの実引数がBD[0:7]ならA=8であり，Bの実引数が(SYNC,RD,WT)ならB=3である．

```
def INV(A/B)
  if A<>B then
    error('INV INVALID',A,B) eoi
  I=0
  while I<A do
    I=I+1,
    "INV(&A[I],&B[I]);" eow
eod
```

図5 .INVのマクロ素子定義

ERROR文はエラーメッセージを出力するためのものであり，例の場合は入力信号数と出力信号数が異なるとき，'INV INVALID 8,3' をディスプレイに表示する．&A[I]は仮引数と実引数の対応を表わすものであり，仮引数Aに対応する実引数のI番目の信号を表わす．I=3のとき，上例ではBD[2]であり，素子レベルでは，BD2を表わす．また，&B[2]はRDを表わす．対応する信号が存在しないときは，空信号として処理する．&A[10],&B[4]はいずれも空信号である．式処理の中に，組み込み関数として LOG($\log_2 x$), PWR(2^x), ODD(整数の偶奇の判定)がある．

素子ファイルへの出力を行なう出力文について，例を用いて説明する．出力文は，'"'でくくられた出力列という構成となっている．'&','"'以外の文字を順次出力していき，'&'が来たら次にくる文字により対応する処理を行なう．これを'"'が来るまで繰り返す．たとえば，"TEST M"はそのまま TEST M と出力され，"TEST &A" は，仮引数に対応する実引数が SYNC なら TEST SYNC と出力される．&*は，通し番号処理であり，&*を使用しているマクロに一つの番号が対応し，使用されるごとに更新される．一つのマクロ内では同じ番号が対応している．その他の例を図6に示す．

```
M=31   "A&M"  ⟶  A31
P=34.2E5  "(&P)"  ⟶  (3.42E6)
Cの実引数がBA[3:5]のとき
  "&C"  ⟶  BA3,BA4,BA5
```

図6 出力文の例

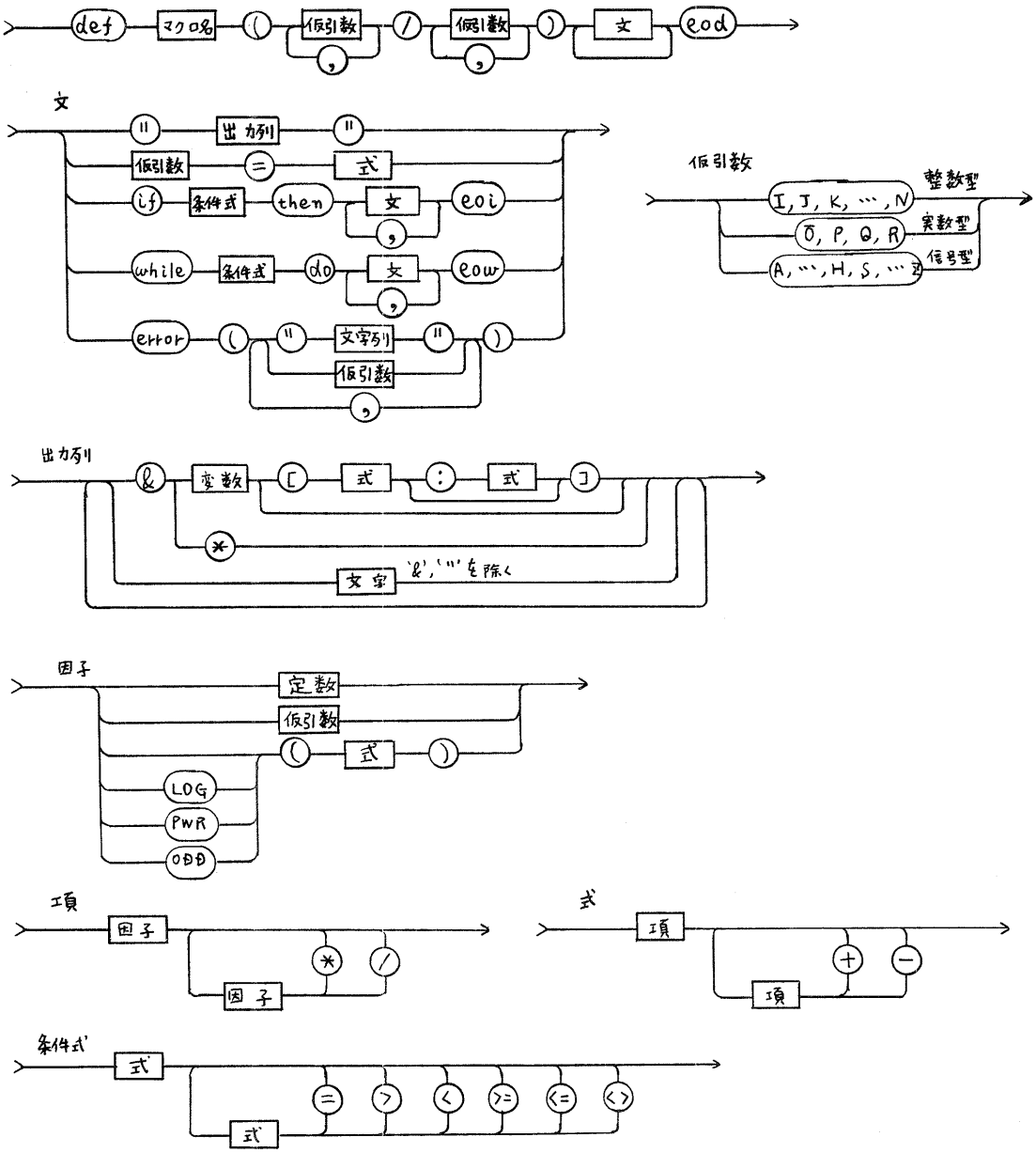
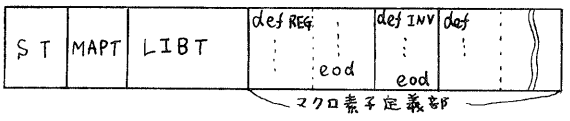


図4 マクロ素子定義構文

4. ライブラリ保守システム

ライブラリ保守システムは，マクロ素子定義を管理するものである．ライブラリファイルは図7に示す形式であり，ステータス部(ST)，MAP表部(MAPT)，ライブラリ名表部(LIBT)，マクロ素子定義部から構成される．レコードと呼ぶ128文字を最小単位として，大きなものはレコードに区切って記憶する．ライブラリファイルは乱ファイル形式であり，マクロ展開時に動的にメモリに読み出される．



ST　：最大レコード数，使用レコード数，最大要素数，使用要素数

MAPT：ライブラリのレコードの使用状況

LIBT：マクロ素子名，スタートレコード番号，サイズ

NAME	SRN	SIZE
REG	003	002
INV	005	001
⋮	⋮	⋮

図7 ライブラリファイルの形式

ライブラリ保守システムには，操作性を考え以下に示す6つのコマンドが用意されている．

Create：新しいライブラリの生成．
Open　：ライブラリファイルのオープン．
Insert：マクロ定義の追加．
List　：ライブラリ名表とマクロ定義の表示．
Delete：マクロ定義の消去．
Merge：複数のライブラリの合成．

まず，マクロ定義記述ファイルをテキストエディタ(RT-11)で作成する．次に，このファイルをライブラリファイルに，Insertすることにより，定義を登録する．ライブラリの修正は，Listコマンドにより定義をテキストファイルにコピーした後，エディタで修正し，再登録する．

ライブラリは先頭に定義名と登録ブロック位置を示すテーブル(LIBT)を持つため，展開時に動的にライブラリの定義読みとりができる．

5. マクロ展開

マクロ展開システムの構成を図8に示す．全体は，表の初期化部(INIT)，マクロ定義の読み込み部(RDMCR)，マクロ展開部(EXPAND)の3つから成る．EXPANDは，RDMCRで読み込んだマクロを，仮引数表と実引数表の対応をとりながらマクロ素子定義に従い，素子記述に展開する．ライブラリ名表は，ライブラリファイルのオープン時にメモリに読み込まれる．表にはマクロ素子定義の記憶位置，サイズがある．マクロ展開時に，ライブラリファイルから，マクロ素子定義がマクロ素子定義バッファに読み出される．マクロ素子定義バッファは，リスト構造になっていて，利用された順にならびかえられ，バッファに空領域がなくなったときに，"最近利用されていない領域を空けて利用する(LRUアルゴリズム)"ようにしている．これは，フロッピーディスクへのアクセス回数を少なくさせるためである．

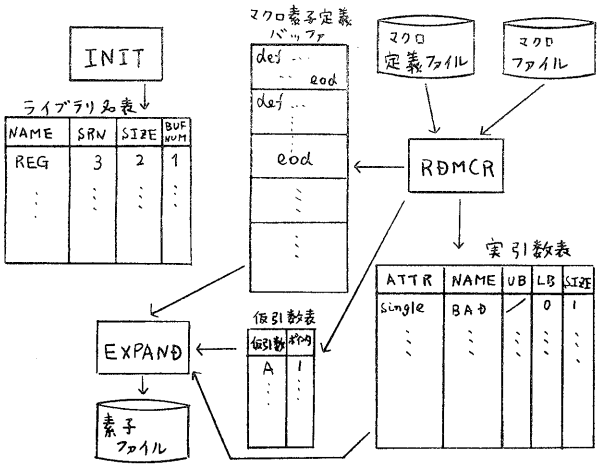


図8 マクロ展開の構成

仮引数	ポインタ
A	1
B	2
C	3
D	4
⋮	
Z	0

(a) 仮引数表

	ATTR	NAME	UB	LB	SIZE
1	sigarray	AD	7	0	8
2	single	CP		0	1
3	single	CNT	3	1	1
4	sigrecord				8
5	sigarray	UA	0	3	4
6	sigarray	LA	0	3	4
7	eor				0

(b) 実引数表

図9 仮引数表と実引数表

次に，信号名の対応を行なう仮引数表と実引数表(図9)を例を用いて説明する．

マクロ素子定義が

def REG(A,B,C,/D) …… eod

マクロ素子記述が

.REG(AD[7:0],CP,CNT[3]/(UA[0:3],LA[0:3]))

であるとき，RDMCR によって，マクロ素子記述が実引数表に展開される．それから，マクロ素子定義の仮引数との対応がとられる．上例の場合は，図9に示すようになる．マクロ素子定義の出力文が，"&A[3]" なら，仮引数表から 'A' のポインタ '1' を読み，実引数表から配列型束信号(sigarray)，信号名が 'AD'，上限が7，下限が0，サイズが8と，読み取れる．だから，素子ファイルへは，AD5 を出力する．他に，"&C" は 'CNT3'，"&D[2:5]" は 'UA1, UA2, UA3, LA0' となり，"&B[2]" のように実引数の範囲を出た場合は，空信号として出力する．

6. マクロ定義と展開例

a) ANDS マクロ

ANDSマクロは，入力信号のすべての論理積をとり出力するものである．ANDSのマクロ素子定義を図10に，マクロ素子図を図11に示す．

入力が4より多いときに，NAND4(4入力NAND)を基底として切り出し，整数変数Mを用いて，合成処理をしている．このANDSマクロは，16入力までしか展開できないが，マクロ素子定義を変更することにより，基底となる素子の入力数や全体の入力の制限は変えられる．"W&MA&*" は内部信号線名であり，'&*' のところが展開時に通し番号処理され，M=0なら，W0A11 のようになる．

```
DEF ANDS(A/B)
  N=1 M=0
  IF A<=2 THEN "NAND2(&A[1:2],^&B);",N=A+1 EOI
  IF A=3 THEN "NAND3(&A[1:3],^&B);",N=A+1 EOI
  IF A=4 THEN "NAND4(&A[1:4],^&B);",N=A+1 EOI
  WHILE A>N+3 DO
    "NAND4(&A[N:N+3],W&MA&*);",
    N=N+4,M=M+1 EOW
  IF A=N THEN "INV(&A[N],W&MA&*);",M=M+1 EOI
  IF A=N+1 THEN "NAND2(&A[N:N+1],W&MA&*);",M=M+1 EOI
  IF A=N+2 THEN "NAND3(&A[N:N+2],W&MA&*);",M=M+1 EOI
  IF M=2 THEN "NOR2(W0A&*,W1A&*,&B);" EOI
  IF M=3 THEN "NOR3(W0A&*,W1A&*,W2A&*,&B);" EOI
  IF M=4 THEN "NOR4(W0A&*,W1A&*,W2A&*,W3A&*,&B);" EOI
  IF M>4 THEN ERROR("ANDS TOO MANY INPUT",A) EOI
EOD
```

図10 ANDSマクロのマクロ素子定義

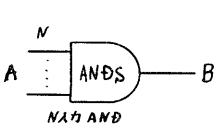
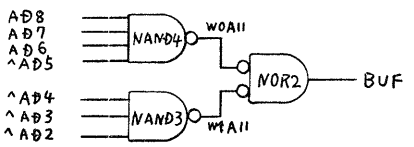


図11 ANDSのマクロ素子図




```
NAND4(AD8,AD7,AD6,^AD5,W0A11)
NAND3(^AD4,^AD3,^AD2,W1A11)
NOR2(W0A11,W1A11,BUF)
```

図12 ANDSの展開結果

マクロ素子記述が

.ANDS((AD[8:6],^AD[5:2])/BUF)

のときには，入力が7本なので，NAND4 と NAND3 で受け，NOR2で合成され出力される．展開結果を図12に示す．

b) PULSE マクロ

PULSE マクロは，入力信号の立ち上がりで，0 nS のパルスを出力するものである．PULSE のマクロ素子定義を図13に，マクロ素子図を図14に示す．

```
DEF PULSE(A,O/B)
  P=1.0E-10
  R=O*0.14E-8/P
  WHILE (R<1.4E3)+(R>4.0E4) DO
    IF R<1.4E3 THEN P=P/10 EOI,
    IF R>4.0E4 THEN P=P*10 EOI,
    R=O*0.14E-8/P EOW
  "SHOT(&A,GND,VCC,CXRX&*,CX&*,&B,^&B);"
  "RG(VCC,CXRX&*,&R);"
  "CAP(CXRX&*,CX&*,&P);"
EOD
```

図13 PULSE のマクロ素子定義

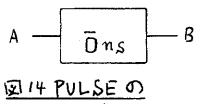
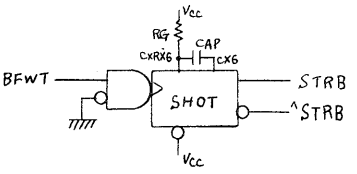


図14 PULSE のマクロ素子図




```
SHOT(BFWT,GND,VCC,CXRX6,CX6,STRB,^STRB)
RG(VCC,CXRX6,2.79E3)
CAP(CXRX6,CX6,1.00E-10)
```

図15 PULSE の展開結果

パルスの幅が0 nS になるよう，抵抗とコンデンサの値を計算している．抵抗が1.4KΩ以上，40KΩ以下になるようにコンデンサの値を定め，その後抵抗値を決定する．P,R は，実数変数でありパルス幅の計算に使用している．このPULSE マクロは，ワンショットと抵抗とコンデンサの3つの素子に展開される．マクロ素子記述が

.PULSE(BFWT,2.0E2/STRB)

のときには，図15に示す通りに展開される．

7. LPインターフェース回路への適用

マクロ展開システムの使用例として，LSI-11 のバスにおけるLPインターフェース回路の展開を行なった．これは，設計した回路を直接マクロレベルでスカし展開したものである．17マクロ素子が95素子に展開された．LPインターフェース回路のマクロ素子記述と展開素子記述を付録に示す．ファイルのス出力（LPへの打ち出し）を含め，約30秒である．また，LPインターフェース回路はIC数で23個である

あとがき

論理設計CADシステムのマクロ展開について報告した．システムは，PASCALで記述され，ライブラリ保守と展開部で約2500行で，LSI-11（56 KB メモリ，フロッピーディスク付）上で実行される．

今後，マクロ素子および展開した素子の回路図の自動作成，また図でのスカ，素子記述に対するIC,ピン割当て等の実装システム，および，機能シミュレータ等を開発する予定である．

参考文献

(1) 伊藤，河野：論理設計自動化システム
電子装置設計技術 4-3 (80, 3, 18)

(2) 伊藤，河野：会話型設計援助システム
電子装置設計技術 2-3 (79, 9, 18)

(3) Mcwilliams T.M : SCALD Structured Computer Aided Logic Design
P271～277 15th DAC (1978)

```
.BRECV(^BDAL[0:12]/DA[0:12]);
.BRECV((^BBS7,^BSYNC,^BDOUT,^BDIN,^BIAKI,^BINIT)/(IO,SYNC,DOUT,DIN,IAKI,INIT));
.TST(DA[1:12],2003/CALL);
DFF(CALL,SYNC, , /CALED,^CALED);          DFF(DA1,SYNC, , /BUFFER,^BUFFER);
AND2(DOUT,SYNC,WT);                       AND2(DIN,SYNC,RD);
.DECORD((BUFFER,CALED),WT/( , ,^STWT,^BFWT));
.DECORD((BUFFER,CALED),RD/( , ,^STRD,^BFRD));
OR2(ERR,DONE,RQST);
.INTM(DA6,STWT,RQST,IAKI,DIN,INIT/IRQ,VEC,IAKO,INTENB);
.ORS((STWT,BFWT,STRD,BFRD,VEC)/RPLY);
.DELAY((LPEMG,LPPALL,LPBUSY,LPOFL,^LPTIME),1000/LPS[1:5]);
.RECV(LPS[1:5]/(^ERR,^PALL,^BUSY,^OFL,TIME));
AND2(^OFL,^TIME,WAIT);                    AND2(^BUSY,WAIT,DONE);
.REG((ERR,PAAL,DONE,INTENB),STRD, /ST[1:4]);
.BDRIV((IRQ,IAKO,VEC,RPLY), /(^BIRQ,^BIAKO,^BDAL7,^BRPLY));
.BDRIV(ST[1:4],STRD/(^BDAL[15:14],^BDAL[7:6]));
.PULSE(BFWT,3.0E2/ROMCS);
.PULSE(BFWT,2.0E2/STB);
.FSROM(DA[0:7],^ROMCS/O[1:8]);
.DRIV((INIT,^STB,^STB),1000/(^RESET,START,STRB));
.DRIV(O[1:7],1000/D[6:0]);
```

付録　LPインターフェース回路のマクロ素子記述

```
BRECV(^BDAL0,DA0)
BRECV(^BDAL1,DA1)
BRECV(^BDAL2,DA2)
BRECV(^BDAL3,DA3)
BRECV(^BDAL4,DA4)
BRECV(^BDAL5,DA5)
BRECV(^BDAL6,DA6)
BRECV(^BDAL7,DA7)
BRECV(^BDAL8,DA8)
BRECV(^BDAL9,DA9)
BRECV(^BDAL10,DA10)
BRECV(^BDAL11,DA11)
BRECV(^BDAL12,DA12)
BRECV(^BBS7,IO)
BRECV(^BSYNC,SYNC)
BRECV(^BDOUT,DOUT)
BRECV(^BDIN,DIN)
BRECV(^BIAKI,IAKI)
BRECV(^BINIT,INIT)
NAND4(^DA1,^DA2,DA3,DA4,W0A1)
NAND4(^DA5,DA6,^DA7,^DA8,W1A1)
NOR2(W0A1,W1A1,CALL)
DFF(CALL,SYNC, , /CALED,^CALED)
DFF(DA1,SYNC, , /BUFFER,^BUFFER)
AND2(DOUT,SYNC,WT)
AND2(DIN,SYNC,RD)
DEC02(BUFFER,CALED,^WT, , ,^STWT,^BFWT)
DEC02(BUFFER,CALED,^RD, , ,^STRD,^BFRD)
OR2(ERR,DONE,RQST)
DEF(DA6,STWT, ,^INIT,INTENB,^INTENB)
AND2(INTENB,RQST,I1M2)
DFF(VCC,I1M2,^INIT,VEC,I2M2,^I2M2)
AND2(I3M2,IAKI,VEC)
DFF(IRQ,DIN, ,^INIT,I3M2,^I3M2)
AND2(I2M2,I1M2,IRQ)
AND2(^I3M2,IAKI,INTENB)
NOR4(STWT,BFWT,STRD,BFRD,W0A3)
INV(VEC,W1A3)
NAND2(W0A3,W1A3,RPLY)
RG(LPEMG,LPS1,100)
CAP(LPS1,GND,1.42E-8)
RG(LPPALL,LPS2,100)
CAP(LPS2,GND,1.42E-8)
RG(LPBUSY,LPS3,100)
CAP(LPS3,GND,1.42E-8)
RG(LPOFL,LPS4,100)
CAP(LPS4,GND,1.42E-8)
RG(^LPTIME,LPS5,100)
CAP(LPS5,GND,1.42E-8)
RECV(LPS1,^ERR)
RECV(LPS2,^PALL)
RECV(LPS3,^BUSY)
RECV(LPS4,^OFL)
RECV(LPS5,TIME)
AND2(^OFL,^TIME,WAIT)
AND2(^BUSY,WAIT,DONE)
REG4(ERR,PAAL,DONE,INTENB,R4, ,ST1,ST2,ST
      3,ST4,^ST1,^ST2,^ST3,^ST4)
AND2(STRD, ,R4)
BDRIV(IRQ, ,^BIRQ)
BDRIV(IAKO, ,^BIAKO)
BDRIV(VEC, ,^BDAL7)
BDRIV(RPLY, ,^BRPLY)
BDRIV(ST1,STRD,^BDAL15)
BDRIV(ST2,STRD,^BDAL14)
BDRIV(ST3,STRD,^BDAL7)
BDRIV(ST4,STRD,^BDAL6)
SHOT(BFWT,GND,VCC,CXRX5,CX5,ROMCS,^ROMCS)
RG(VCC,CXRX5,4.19E3)
CAP(CXRX5,CX5,1.00E-10)
SHOT(BFWT,GND,VCC,CXRX6,CX6,STB,^STB)
RG(VCC,CXRX6,2.79E3)
CAP(CXRX6,CX6,1.00E-10)
FS256(DA0,DA1,DA2,DA3,DA4,DA5,DA6,DA7,^RO
      MCS,O1,O2,O3,O4,O5,O6,O7,O8)
DRIV(INIT,^RESET)        RG(VCC,^RESET,1000)
DRIV(^STB,START)         RG(VCC,START,1000)
DRIV(^STB,STRB)          RG(VCC,STRB,1000)
DRIV(O1,D6)              RG(VCC,D6,1000)
DRIV(O2,D5)              RG(VCC,D5,1000)
DRIV(O3,D4)              RG(VCC,D4,1000)
DRIV(O4,D3)              RG(VCC,D3,1000)
DRIV(O5,D2)              RG(VCC,D2,1000)
DRIV(O6,D1)              RG(VCC,D1,1000)
DRIV(O7,D0)              RG(VCC,D0,1000)
```

付録　LPインター フェース 回路の展開素子記述