構造化論理設計を支援する論理図編集システムLODE

平石裕実　進藤尉一　矢島脩三
（京都大学工学部）

1．はじめに

近年のLSI技術の進歩に伴い，設計すべき回路の規模，複雑さは増大してきている。このような大規模で複雑な回路を効率よく設計する為に多くの計算機援用設計（CAD）システムが開発されてきた。ブロック図や論理回路図は論理設計に不可欠である為，通常CADシステムには論理図編集システムが含まれる。論理図編集システムの役割はCADシステム本体への回路情報の入力提供であり，その方法は使用する入力装置により次の3種類に大別される。

（1）　素子及び接続線の座標をタブレットやディジタイザで指定する[1][2]。
（2）　グラフィックスを用いて会話的に論理図を編集する[3][4][5]。
（3）　スキャナを用いて論理図を読む[6]。

これらの方法は設計した結果である論理図を扱うものであり，設計過程を支援するものではない。しかし論理設計の規模，複雑さが増大してくるにつれて，単に論理図の編集のみならず論理設計そのものを支援することができる論理図編集システムが望まれる。大規模回路の設計を支援する為には構造化論理設計の手法が適していると考えられ，又，設計の各段階において検証が行えることが要求される。これらの要求を満たす為に階層的論理図の取り扱い，検証の為のシミュレータへの入力提供，高度な図形編集機能といった機能の実現が重要である。本稿では，これらの点を考慮して我々の研究室で開発した論理図編集システムLODE（Logic Diagram Editing System for Structured Logic Design）について述べる。LODEを通じて設計者はテレビモニタ，ライトペンを用いて会話的に論理設計を進めることができる。又，端子間の自動配線等の編集機能により設計者は容易に論理図の編集を行うことができ，設計した回路を任意の時点でシミュレートすることができる。以下，第2章でLODEの概要を，第3章でLODEの持つコマンドやデータ構造について述べる。第4章でLODEの使用例を挙げ評価を行う。

2．LODEの概要

2．1　構造化論理設計

構造化論理設計は大規模，複雑な回路を設計するのに適したトップダウン的設計手法である。構造化論理設計では図1に示すようにブロックを分割することによって設計が進められる。設計すべき回路は最初はその動作仕様のみが与えられた1個のブロックとみなされる。まずブロックを複数個のブロックに分割しブロック間のインタフェースを設計し，次に各ブロックの詳細設計を行う。各ブロックは更に複数個のブロックに分割され，すべてのブロックがNANDゲートやフリップフロップ等の基本ゲートで構成された時点で設計は終了する。

構造化論理設計ではこのようにブロックの分割と詳細化を繰り返して設計が進められるが，各ブロックの設計は次の2段階をふむと考えられる。

（段階1）　機能レベル設計　　（段階2）　構造レベル設計

段階1で設計者はブロックの機能を設計し，段階2でその機能を満足するように

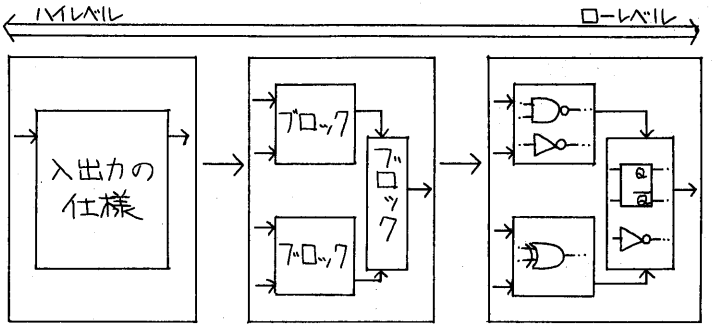


図1. 構造化論理設計

ブロックの構造を設計する。このような設計過程の信頼性を向上させる為には，各設計段階においてシミュレーション等を行うことにより設計の誤りがないことを検証しながら設計を進めることが望ましい。例えば機能レベル設計の後に機能レベルシミュレーションを行うことにより分割前のブロックの機能を満足することを確かめる。もし誤りが検出されれば機能設計をやり直す。これにより設計誤りを初期段階で検出することができ能率のよい設計が可能になる。

2．2　LODEの機能と特徴

既に述べたように構造化論理設計を支援する論理図編集システムにおいては，階層的な論理図の取り扱い，シミュレーションによる検証，強力な図形編集機能等が要求される。これらの要求を考慮してLODEでは以下の機能，特徴を持つ。

（1）ブロック図による階層的な回路表現　構造化論理設計ではブロックの分割を繰り返すことによって設計が進められる。設計の初期の段階では回路はブロック図のみで表現され，中間段階では回路はブロック図と基本素子を併用することにより表現される。LODEではブロック図を設計の各レベルで用いることができ，基本シンボルと同様，図面への追加，削除，他のシンボルとの配線等の編集を施せる。又，LODEでは画面に表示された図が1個のブロック図に対応しており，そのブロックの構造を他のブロックを用いて画面上で定義することによりブロックの分割が行われる（ブロックの詳細化）。又，コマンドにより1レベル上のブロック図を表示することもできる（ブロックの抽象化）。これらのブロックの詳細化，抽象化の機能により設計者は階層的な論理図を取り扱うことができる。

（2）ブロックの機能表現　構造化論理設計では，ブロックの設計は機能レベル設計と構造レベル設計から成る。従来の論理図編集システムが回路の機能表現を取り扱っていないのに対して，LODEはブロックの構造記述のみならず機能記述も取り扱うことにより一貫した設計を支援する。機能の記述手法としてタイミングチャート，フローチャート，状態遷移図，ハードウェア記述言語等の言語が考えられる。設計過程で記述に要求される点が記述の厳密性（あいまい性のない表現），理解しやすさであるという観点からLODEではブロックの機能記述の手法として状態遷移図と我々の研究室で開発したハードウェア記述言語SHDL[7]を採用した。設計者は論理図と同様に状態遷移図を画面で編集できる。またSHDLは機能レベルからゲートレベルまでの記述が可能であり，多レベルシミュレータを中心とした論理設計検証支援システムESS[8]のハードウェア記述言語としてそのまま使用できる。

（3）論理図の編集機能　論理設計において論理図の作成は重要であるが，

端子間の配線等の煩わしい作成行為を伴う。ＬＯＤＥでは設計者の負担を軽減する為に多くの強力な編集機能を提供している。例えば，接続したい2端子をライトペンで指定するだけで配線はＬＯＤＥが行う自動配線機能や，シンボルの削除や移動に伴う配線の自動修正機能などが挙げられる。

（4）シミュレータへの入力提供　既に述べたように設計の各段階で設計の検証を行うことは誤りのない設計を行うという点で重要である。検証を目的として設計した回路のシミュレーションを可能にする為に，ＬＯＤＥでは編集された論理図から結線情報や機能記述情報等の回路情報を抽出し，ＩＳＳが取り扱えるデータに変換してＩＳＳを起動することが可能である。

3．ＬＯＤＥの構成

3．1　システム構成

ＬＯＤＥのシステム構成図を図2に示す。ＬＯＤＥは小型計算機PDP 11/40上でMUーＢＡＳＩＣの拡張版を用いて開発された。論理図はテレビ走査型計算機グラフィックス装置ＤＩＧＲＡＰＨ[9][10]上でライトペンを用いて編集され，そのハードコピーは高精度プリンタHyTypeによって出力される。編集に必要な内部データ（3．3節参照）はすべてディスク上に格納されている。又，PDP11/40は京都大学大型計算機センターM200と300ボーの電話回線で結合されており，ＬＯＤＥで作成した論理図の回路情報をM－200上で開発されたＩＳＳに送り，シミュレーションを行うことができる。

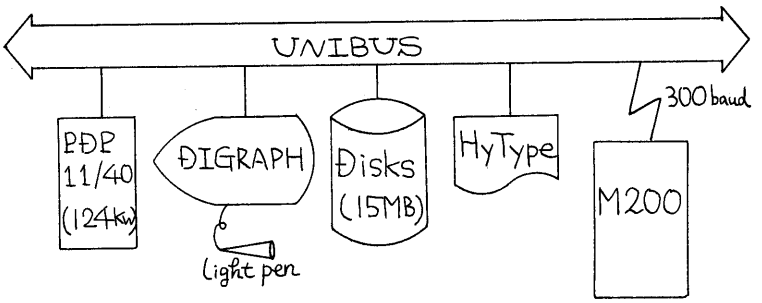


図2．システム構成

3．2　動作モード

ブロックの機能設計と構造設計が中心である構造化論理設計を支援する為にＬＯＤＥはその各々を編集する機能を持つ必要がある。このためＬＯＤＥでは図3に示すようにブロックの定義，ブロックの機能記述の編集，ブロックの構造記述の編集といった3つのモードを持ち，各モードの画面には固有のコマンドメニューが表示される。コマンドの選択や修正箇所の指定はライトペンを用いて会話的に行われる。以下，各動作モードについて説明する。

シンボル定義モード（MD）　ブロック図によって回路を表現する場合，ブロックの形や大きさを前もって設計者が定義しなければならない。このモードではブロックの名前，端子，表示の大きさなどを定義する。定義されたブロックはＬＯＤＥが提供する基本的な論理素子と共にシンボルライブラリに登録され，構造編集モードで利用できる。

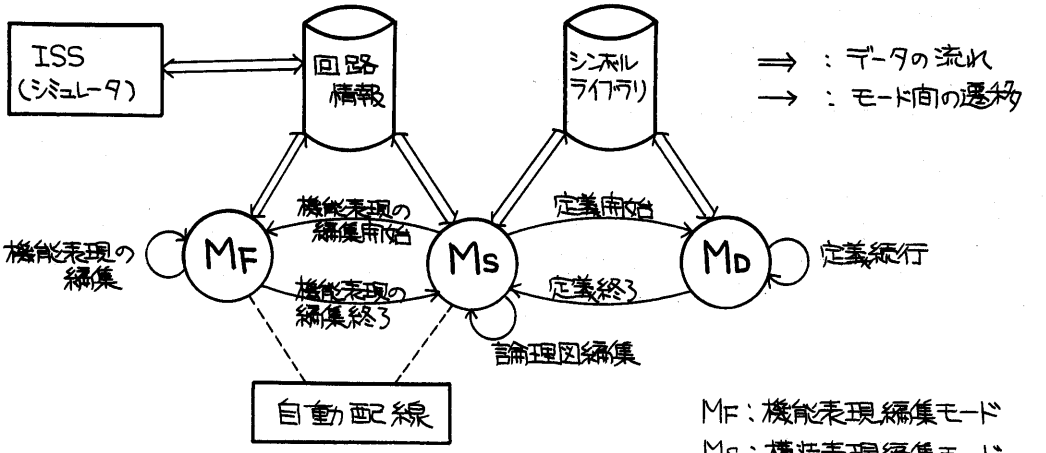


図3　全体の構成

構造編集モード(Ms)　構造化論理設計で用いられる論理図の編集機能としては，ゲートレベルの回路図の編集のみならずブロック間の配線やブロックの詳細化，抽象化等も要求される。このモードでは画面にシンボルを追加しながら論理図を編集するが，設計者の負担を軽減する為にシンボル間の配線に際しては，指定された2端子間の経路は自動的に決定される。又，配線の削除は図4に示すように単一削除と全削除の2種類が指定できる。シンボルの削除に際しては図5に示すように，入力端子の配線に関しては単一削除，出力端子の配線に関しては

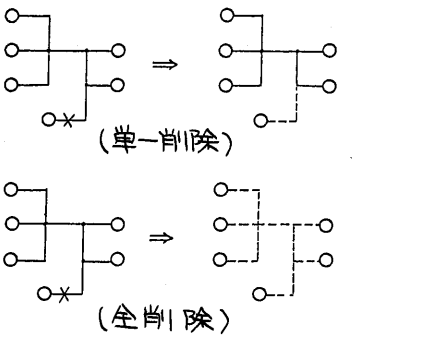


図4. 接続線の削除

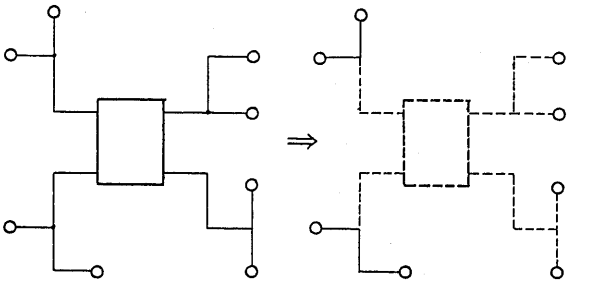


図5　シンボルの削除

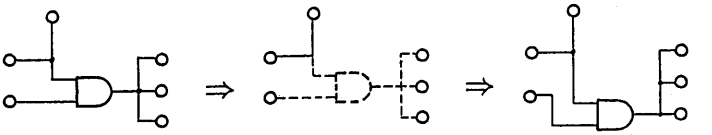


図6　シンボルの移動

全削除が行われる。更にシンボルの移動においては，まず指定されたシンボルが削除され，新たな位置にそのシンボルが追加され，元の接続関係を保つように配線の修正が行われる。

機能表現編集モード(MF)　ブロックの機能記述を編集するモードであり，状態遷移図及びハードウェア記述言語を用いることができる。状態と遷移がMsモードにおけるシンボルと配線に対応しており，Msモードと同様の編集機能が提供される。又、ハードウェア記述言語の場合はSHDLのテキスト編集を行える。

3、3　データ構造

LODEが管理する内部データとしては (1) 画面情報であるディスプレイファイル (2) 図中のシンボルの名前，位置，端子間の結線情報等の論理図の情報 の2種類がある。これらのデータはLODEの会話的応答性を重視して構成されており，多少の冗長性を持つものの，効率のよい実現を可能にしている。

ディスプレイファイルの構造　会話的な論理図編集を支援する為にはライトペンで指された図形の同定やコマンドによる処理結果の表示を迅速に行う必要がある。そのため，ディスプレイファイルは，ライトペンで指された図形の同定，及び，図の変更に対応するディスプレイファイルの変更を容易に行えるように構成されている。例を図7に示す。一画面に対応するディスプレイファイルはシンボル表示部と接続線表示部が分離されている。LODEの各コマンドでは，次にライトペンで指示される図形がシンボルか接続線か一意的に決定できるので，この分離により，効率のよい図形同定が可能である。各シンボルのディスプレイファイルはサブルーチン形式でシンボル表示部に結合され，シンボルの表示位置とサブルーチンの開始番地の指定から成り，1シンボルにつき5語から成る。又、1接続線の表示命令は始点と終点の位置の指定から成り，4語から成る。このような固定長構成により、ライトペンで指示された接続線やシンボルを容易に知ることができる。又、削除されたシンボルや接続線に対応するディスプレイファイルの領域の動的管理を効率よく行うことができる。

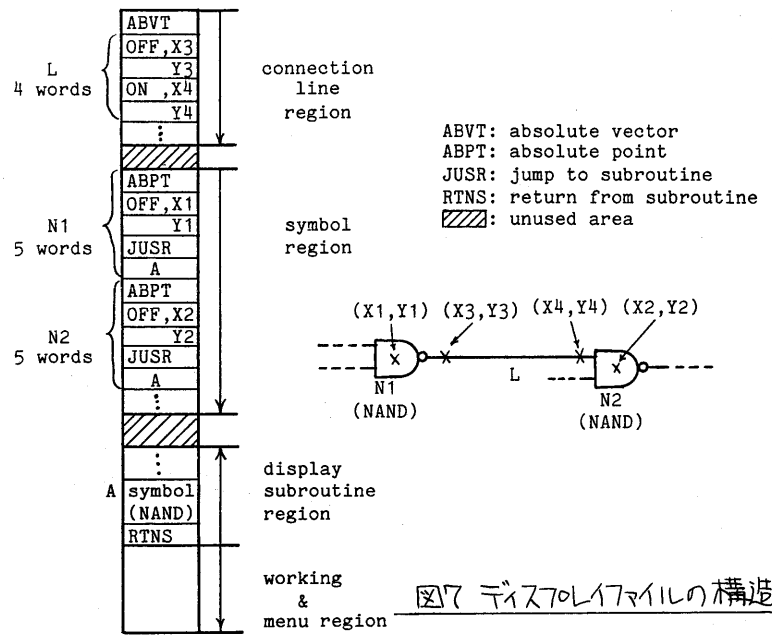
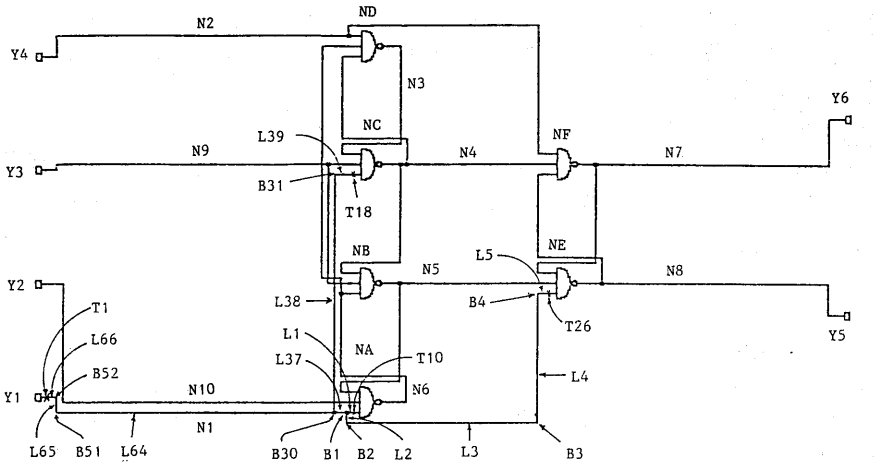


図7 ディスプレイファイルの構造



論理図

Name	Module	Type	Size	Position	#Terminals	Address	Pointer
Y1	1EXT	S	1,1 -1,-1	-1 22	1	32016	1
Y2	1EXT	S	1,1 -1,-1	-1 44	1	32016	2
Y3	1EXT	S	1,1 -1,-1	-1 66	1	32016	3
Y4	1EXT	S	1,1 -1,-1	-1 88	1	32016	4
Y5	3EXT	S	1,1 -1,-1	124 38	1	31996	5
Y6	3EXT	S	1,1 -1,-1	124 76	1	31996	6
NA	3NAND	S	3,3 -2,-3	50 21	4	31868	7
NB	3NAND	S	3,3 -2,-3	50 44	4	31868	11
NC	3NAND	S	3,3 -2,-3	50 67	4	31868	15
ND	3NAND	S	3,3 -2,-3	50 90	4	31868	19
NE	3NAND	S	3,3 -2,-3	80 44	4	31868	23
NF	3NAND	S	3,3 -2,-3	80 67	4	31868	27

シンボル表

	Degree	Position	LP	NP
1	3	47 19	1	1
2	2	47 17	2	1
3	2	76 17	3	1
4	2	76 42	4	1
5	3	47 92	6	2
30	3	45 19	37	1
31	2	45 65	38	1
51	2	2 19	64	1
52	2	2 22	65	1
53	2	46 21	67	10

分岐点表

	Pointer1	Start Point	End Point	Pointer2
1	0	10	-1	37
2	1	-1	-2	3
3	2	-2	-3	4
4	3	-3	-4	5
5	0	26	-4	4
37	2	-1	-30	64
38	37	-30	-31	39
39	0	18	-31	38
64	38	-30	-51	65
65	64	-51	-52	66
66	0	1	-52	65

接続線表

	Name	Characteristic	Position	LP	NP
1	CLR	EXT INPUT	1 22	66	1
2	DATA	EXT INPUT	1 44	71	10
3	CLK	EXT INPUT	1 66	74	9
4	PR	EXT INPUT	1 88	63	2
5	QN	EXT OUTPUT	122 38	84	8
6	Q	EXT OUTPUT	122 76	81	7
7	O1	OUTPUT	54 21	36	6
8	I1	INPUT	47 23	26	5
9	I2	INPUT	47 21	67	10
10	I3	INPUT	47 19	1	1
11	O1	OUTPUT	54 44	22	5
12	I1	INPUT	47 46	30	4
13	I2	INPUT	47 44	52	9
14	I3	INPUT	47 42	31	6
15	O1	OUTPUT	54 67	21	4
16	I1	INPUT	47 69	15	3
17	I2	INPUT	47 67	55	9
18	I3	INPUT	47 65	40	1
19	O1	OUTPUT	54 90	11	3
20	I1	INPUT			
21	I2	INP			
22	I3				

端子点表

図8. 論理図を表す各表

	Output Terminal	Input Terminals	Branching Points
1	1	10,26,18	1,2,3,4,30,31,32,51,52
2	4	20,28	5,6,7,8,49,50
3	19	16	9,10,11,12
4	15	22,12,29	13,14,15,16,17,22,23,24,59
5	11	8,25	18,19,20,21,60
6	7	14,21	25,26,27,28,29,45,46,47,48
7	27	24,6	33,34,35,36,61,62
8	23	30,5	37,38,39,40,41,63,64
9	3	13,17	42,43,44,57,58
10	2	9	53,54,55,56

ネット表

図8　論理図を表す各表（続き）

論理図の情報

論理図を編集する為にはLODEはシンボル，接続線，分岐点（角の点も含む）等のデータを持つ必要がある。LODEではこれらのデータをすべて表の形で管理しており，更に図の編集に必要なデータの検索，データの修正を効率よく行う為に表の間はポインタで結ばれている。図8に回路図とそれに対応する表の例を示す。

- シンボル表：表示画面中のシンボルの各々について，その名前，型，大きさ，位置，端子点の数，ディスプレイサブルーチンの先頭番地を格納する。更に端子点へのポインタを各シンボルについて持つ。
- 端子点表：シンボルの持つ端子点の各々について，その名前，属性（入力か出力か）を格納する。更に各端子点に対してその端子点を端点とする接続線へのポインタ（LP），その端子点が含まれるネットへのポインタ（NP）を持つ。
- 接続線表：接続線の各々についてその始点及び終点のインデックスを格納する。更に両端点における他の接続線との接続関係は循環リストで保持される。例えば図8で接続線3と分岐点ー3でつながっている接続線は接続線表の3，4行目のポインタをたどることにより，接続線4のみであるとわかる。循環リストは接続線の全削除の処理などで有用である。

その他、分岐点に関する情報（位置，度数）を格納する分岐点表や論理的に接続された点を各行毎にまとめたネット表などがある。ネット表はシミュレータに回路情報を提供する時に有用である。

3.4 コマンドの処理

LODEではブロックの展開，論理図の編集，シミュレータへの入力提供といった設計行為はすべてコマンドによって指示される。コマンドが与えられるとLODEはコマンドの処理を行う為に前節で述べたデータの検索，修正を行う。ここで自動配線及びISSのシミュレータへの入力提供を例にとってどのような処理をとるか説明する。

自動配線の処理　LODEで行なわれる論理図の自動配線は会話的配線を対象にしており，従来のプリント基板等を対象とする配線システムと比較すると配線長や配線密度に対する制約は弱く，むしろ会話的応答性が重要であり，実行速度の速いアルゴリズムが必要である。そこでLODEでは新たな2段階配線アルゴリズムを導入している。第1段階は，現在の探索方向と目標点の方向より図9の優先度に基づいて深さ優先探索を行う。これは迷路法[11]を改良したものであり，配線可能な場合は必ず経路をみつけることができる。しかも配線禁止領域が存在しない場合は2端子間の直角距離のオーダーで経路を発見することが

A	B	C			A	B	C		
		1	2	3			1	2	3
→	→	→	↑	↓	←	→	↓	↑	←
	↗	→	↑	↓		↗	↑	←	↓
	↑	↑	→	↓		↑	↑	←	↓
	↖	↑	→	↓		↖	←	↑	↓
	←	↑	↓	←		←	←	↑	↓
	↙	↓	→	↑		↙	←	↓	↑
	↓	↓	→	↑		↓	↓	←	↑
	↘	→	↓	↑		↘	↓	←	↑
↑	→	→	↑	←	↓	→	→	↓	←
	↗	↑	→	←		↗	→	↓	←
	↑	↑	→	←		↑	→	←	↓
	↖	↑	←	→		↖	←	↓	→
	←	←	↑	→		←	←	↓	→
	↙	←	↑	→		↙	↓	←	→
	↓	←	→	↑		↓	↓	←	→
	↘	→	↑	←		↘	↓	→	←

A：現在の探索方向
B：目標点の方向
C：次の探索方向の優先度

図9　探索優先度表

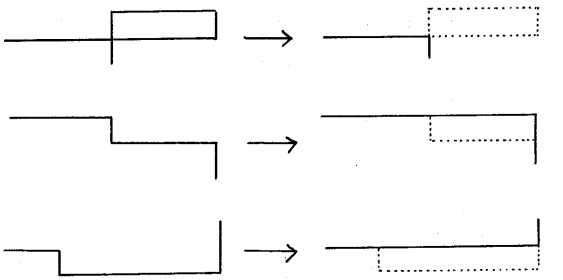


図10　配線の修正

できる。第2段階では図10で示されるように，第1段階で求めた経路に対して修正を施し，ループの削除や曲り角の除去を行う。これにより高速で見易い配線が可能になる。本アルゴリズムでは必ずしも最短経路は得られないが，図13の例では配線長の増加は最短経路長に対して約25%におさまっている。

ＩＳＳへの入力提供　ＩＳＳのシミュレータが必要とする回路情報はシンボルの機能及び端子間の接続に関する情報であり，シンボルの位置に関する情報は不要である。ＬＯＤＥからシミュレータへ回路情報を送る際にＬＯＤＥではシンボル表より各シンボルの使用モジュールの情報を得，各シンボルにその機能を対応付けると共に，端子点表，ネット表より各シンボル間の論理的な接続関係を抽出する。又，ＩＳＳではシンボルをＩＣ番号によって区別しているので各ゲートのＩＣ番号を与える必要がある。

4　ＬＯＤＥの使用例と評価

図11～図13にＬＯＤＥを用いた設計例を示す。最初，回路は図11のSLINTFで表現され，SLINTFは図12で示されるようにADRDEC，PSCNV及びINTCNTの3つのブロックに分割される。INTCNTは図13で示されるようにゲートレベルで構成される。又，実際の編集を通じて以下の問題点が明らかになった。

(1) ブロックの形状定義に際して，端子名等の文字列の位置の指定が煩雑で時間がかかる。

(2) 端子間の接続はＬＯＤＥが自動的に行う為に常に見易い配線が得られるとは限らない。仮想端子や仮想禁止領域の導入により大まかな配線経路を指定する機能の実現が必要である。

ＬＯＤＥはBASICで開発されている為，必ずしも動作速度は速くなく，論理図を新たに画面に表示するＬＯＡＤコマンドでは2次記憶（ディスク）より表

示される論理図のディスプレイファイルや内部データをロードする為に1分弱の時間がかかる。しかしシンボルの追加や削除，配線等に対する応答時間は遅くとも10秒以内であり，各論理図の編集を30分程度で行うことができ，十分会話的利用に耐え得るものと考える。

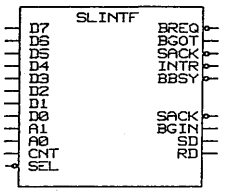


図11. 設計される回路

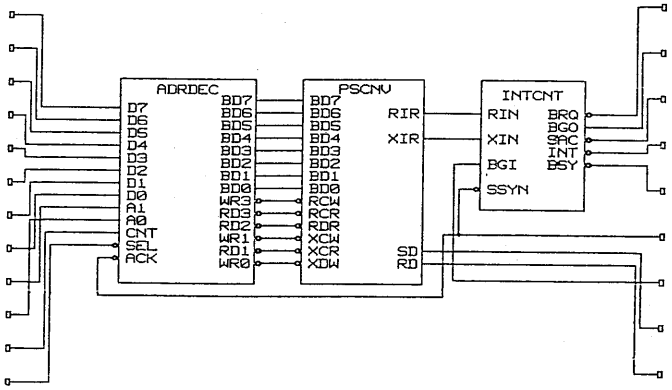


図12 SLINTFのブロック図

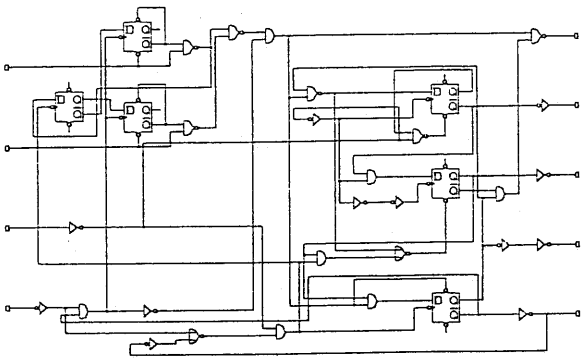


図13 INTCNTの論理回路図

5. おわりに

論理図を編集するのみならず，論理設計をも支援する論理図編集システムLODEについて述べてきた。LODEでは階層的なブロック図を取り扱い，又，ブロックの機能記述も取り扱うことができる。これらの記述を用いて設計者は回路の機能，構造の設計を階層的に進めることができる。LODEは自動配線やシンボルの削除，移動に伴う配線の自動修正等の編集機能を提供することにより，設計者の負担を大巾に軽減することができる。更にシミュレータへの入力を提供

することにより，誤りのない設計の支援が可能である。

構造化論理設計をより強力に支援する為にしのHDLの拡張すべき点としては回路全体の階層性や設計状況を設計者に明示する為，回路の階層構造を表すトリーを表示する機能や，回路の作成を容易にする為にバス等の複数本の信号線を1本の線で代表させる機能，シミュレーション結果の論理図上への表示や信号線に基づく関連部分の検索等の機能が考えられ，検討中である。

謝辞

種々貴重な御助言御討論いただいた上林彌彦助教授，安浦寛人助手，萩野博幸氏をはじめとする矢島研究室各位，ならびにしのHDLの設計，開発に携った青木豊氏，近田信彦氏，伊東幹雄氏に深謝いたします。

文献

[1] R. Rutman "Non-Gridded Graphic Input" Proc. 13th DA conference June 1976 pp 392～398

[2] P. Villers "A Minicomputer-Based Interactive Graphics System as Used for Electronic Design and Automation" Proc. 15th DA conference June 1978 pp 446～453

[3] H. M. Bayegan "An Integrated System for Interactive Editing of Schematics, Logic Simulation and PCB Layout Design" Proc. 15th DA conference, June 1978, pp1～8

[4] T. M. McWilliams, L. C. Widdoes, Jr. "SCALD: Structured Computer-Aided Logic Design" Proc of 15th DA conference, June 1978, pp 271～277

[5] W. M. vanCleemput "A Structured Design Automation Environment for Digital Systems" Digest of papers, COMPCON, Feb 1978., pp 139～142

[6] M. Ishii et al "Automatic Input and Interactive Editing System of Logic Circuit Diagrams" Papers of Technical Group on Computers, IECEJ, EC80-16 June, 1980

[7] Y. Ono "Development of Translator of a Hardware Description Language for Structured Logic Design" Bachelor Thesis, Dpt of Information Science, Kyoto Univ. Feb. 1981

[8] 酒井，他 "会話型論理シミュレータSIM/D-ISを用いた論理回路の検証" 信学会計算機研究会資料 EC81-54 1981.12.

[9] H. Hiraishi et al "The Design of a Microprogrammed Graphic Display Processor with Pipeline Organization" Proc. 3rd Euromicro, Oct 1977 pp 66～73

[10] H. Hiraishi et al "Realization of a Raster-Scan Computer Graphic Display Device with Random-Scan Functions Using a Device Simulator" Trans. of IPSJ, Vol. 22, No1, Jan 1981, pp 36-43

[11] C. Y. Lee "An Algorithm for Path Connection and Its Application" IRE Trans. Electronics Computers, vol EC-10, 1961 pp 346～365