平成 13 年度　舞鶴工業高等専門学校地域テクノセンター公開講座

「CAD による IC 回路設計と FPGA による実現 」

電子制御工学科　町田秀和
E-mail machida@maizuru-ct.ac.jp , TEL(FAX) 0773-62-8957

目次

1.はじめに

今日、電子制御機器のコントローラ等を実装する電子回路は、1 チップの専用 IC で実現されるのが主流になってきていますが、現実には、所望の応用に対して適当な IC が存在しなかったり入手が困難であることが多いという問題があります。

そこで、他社よりも速く製品化して利益をあげるいわゆる「タイムツーマーケット」を達成できるようになることを目指し、比較的大規模な書きこみ可能な IC である FPGA で所望の IC を実現する技法を習得します。そこでは、IC 回路設計用 CAD (EDA)である MAX+PLUS2 を利用し，信号発生器を実現する IC を設計した後、FPGA で実現するために書き込みケーブルおよび FPGA 評価ボードを製作し、実機実験を行う技術について講習します。

表１　講座スケジュール

講習日時		講義内容	講師
11 月 8 日(木)	19:00 ～ 21:00	書きこみ用ケーブルの製作 EDA ツールの基礎	町田
11 月 13 日(火)	19:00 ～ 21:00	評価用 FPGA ボードの製作 EDA ツールの応用	町田
11 月 15 日(木)	19:00 ～ 21:00	信号発生器の製作	町田

2. FPGA とは何か？

FPGA(Field Programmable Gate Array)とは、その名前のとおり「現場で書き込み可能なゲートアレーIC」です。それでは一体、従来の IC によるディジタル回路の実現方法とどのように異なるかを**表 2** で比較してみましょう。

表 2. FPGA とは何か

	従来の実現	FPGA による実現	FPGA のメリット
実現形態	74 シリーズなど小規模 IC の組み合わせ	1 チップに収まる	信頼性の向上
セキュリティ	回路を読み取られてしまう。	対策可能	情報を盗まれない
設計手段	紙と鉛筆	パソコン	パソコンで動作確認が行える。
設計方針	しばしば非同期回路設計	同期回路設計	パソコンの設計ツールが使用できる。
回路の流用	設計者にしか分かりにくい	モジュール(ライブラリ、IP コア)として再利用可	設計管理が容易
実装	ブレッドボードあるいは、ﾕﾆﾊﾞｰｻﾙ基板に半田付け	FPGA 評価ボードを用意すれば書き込みだけ	半田付け等不要、検証だけに集中できる。
入手容易性	しばしば納期が遅れる。また廃品種となることがある。	何度でも書き換えられるので、ﾃﾞｯﾄﾞｽﾄｯｸにならない	納期、在庫のリスクを回避できる。
開発期間	設計、動作検証、部品の手配に手間がかかりすぎる。	パソコンだけですぐに取りかかれる。	タイムツーマーケットを実現できる。
回路の変更	ほとんど不可能	基板に実装したままで可能	本来の機能を生かせる。
書き込み装置	不要	必要	余分な回路が必要だが、簡単である。またフラッシュタイプもある。

これらの特徴から FPGA を活用するキーポイントは次のとおりであることが指摘できます。

1.パソコンと設計ツール(CAD あるいは EDA)の操作法の習得すること。
2. FPGA 評価ボードと、書き込み装置を用意すること。
3.同期回路設計をマスターすること。

それでは、一つづつ見ていきましょう。

2.1 開発の流れ

FPGA の開発は専用の設計ソフトウェア EDA(Electronics Design Automation)ツールを用いて行います。現在では Web から入手でき、規模や機能を限定したものならほぼ無料で使用できます。次のページの**図 1** に EDA ツールを用いた FPGA の開発の流れを示します。

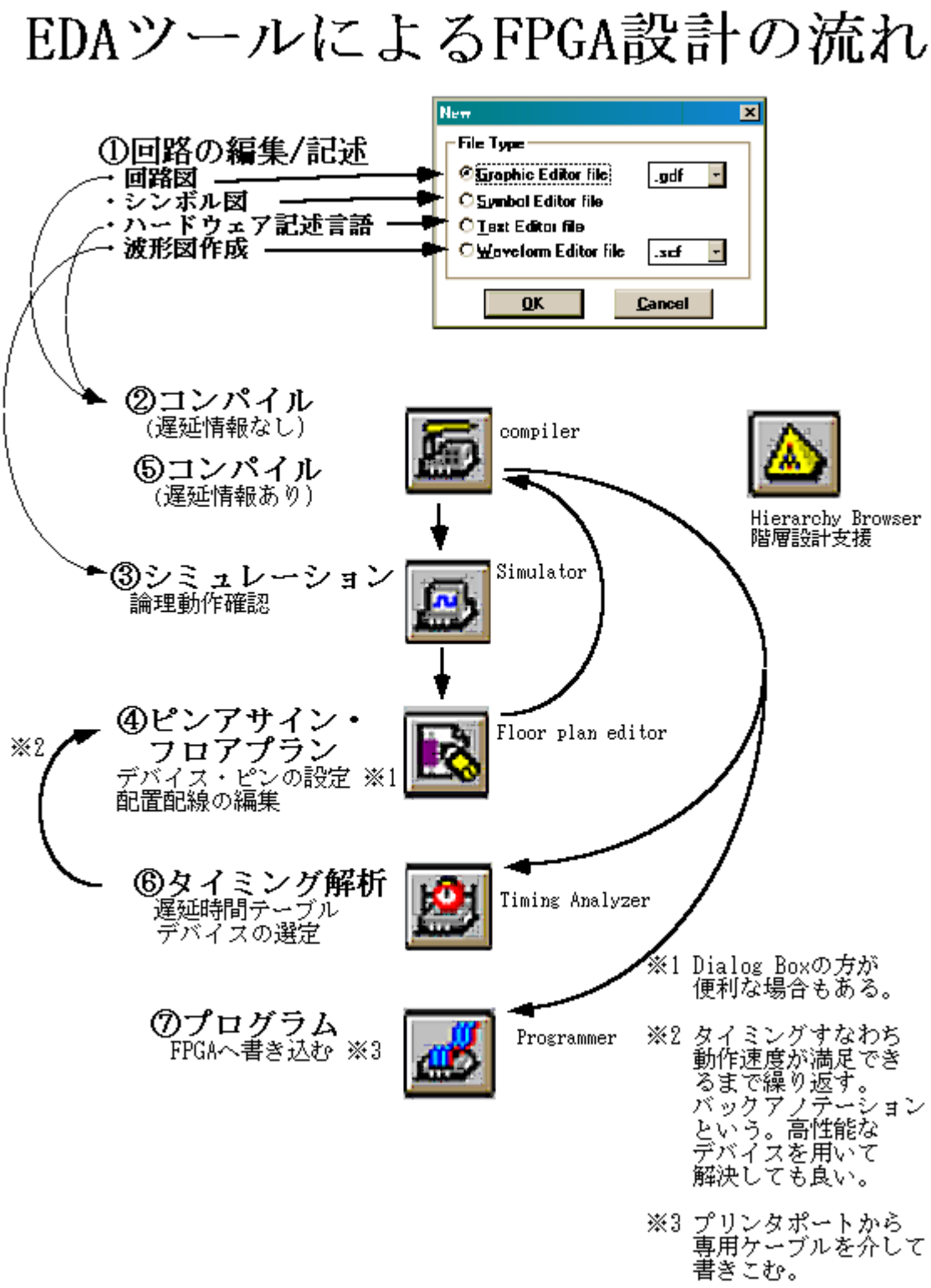


図 1　EDA ツールによる FPGA 設計の流れ

本講習会では Altera 社の Free 版の MAX+PLUS2 という EDA ツールを用います。これは、http://www.altera.com からダウンロードすることができます。この図の各アイコンは MAX+PLUS2 のものです。3 万ゲート規模の FPGA まで実際に開発することができます。本校電子制御工学科 5 年生の CAD 演習Ⅲの授業で使用しています。

2.2 開発装置

写真 1 に FPGA 開発装置の様子を紹介します。

最近の FPGA は専用の書き込み器で書き込むのではなく、パソコンのプリンタポート(あるいは、シリアル、USB)から専用ケーブル(ダウンロードケーブルと呼ばれる)を介して、書き込まれます。また、FPGA はプリント基板に実装したままで書き込めます(In System Programming)。プログラミング時間はほんの数秒で終了します。

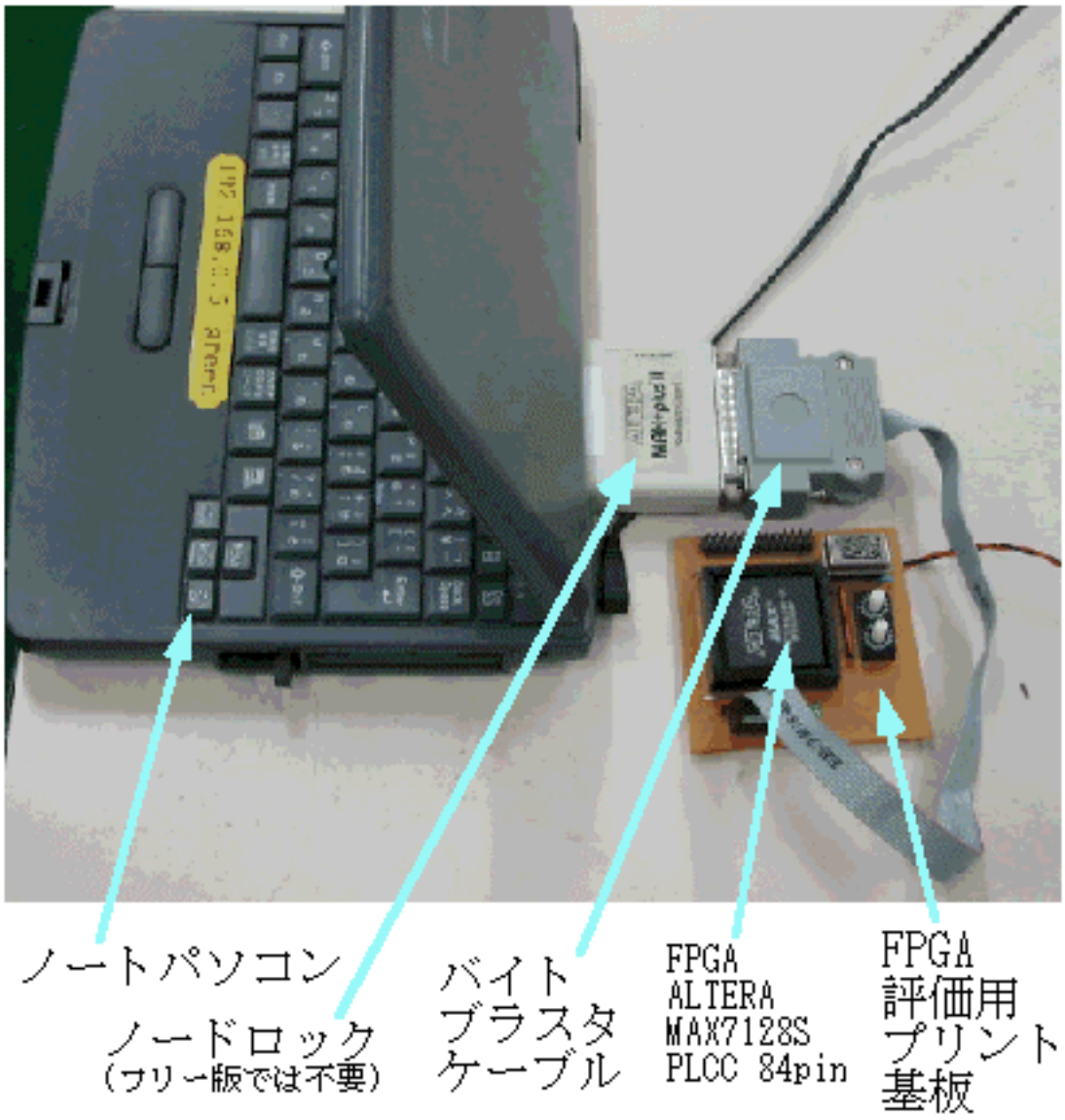


写真 1　FPGA 開発装置全景

なお、FPGA には S-RAM タイプ(例：FLEX10K)と EEP-ROM タイプ(例:MAX7000)があり、前者は電源を切れば回路内容が失われてしまうため、それを避けるためにはコンフィグレーション用のシリアル ROM が必要になります。ただし、S-RAM タイプの方が大容量の回路を実現できます。

2.2.1 ダウンロードケーブルの製作

Altera 社から公開されている**図 2** よりダウンロードケーブル(バイトブラスタ)は簡単に自作できます。本講習会では次ページ**図 4** のようにこの回路を試作します。

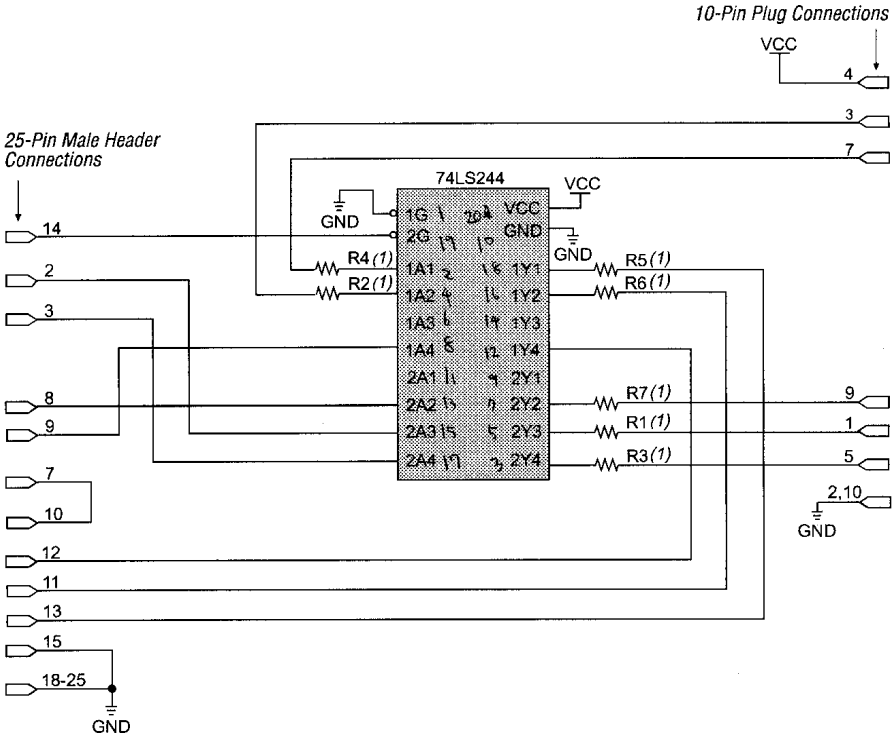


図 2 バイトブラスタ回路図(抵抗は全て 33Ω)

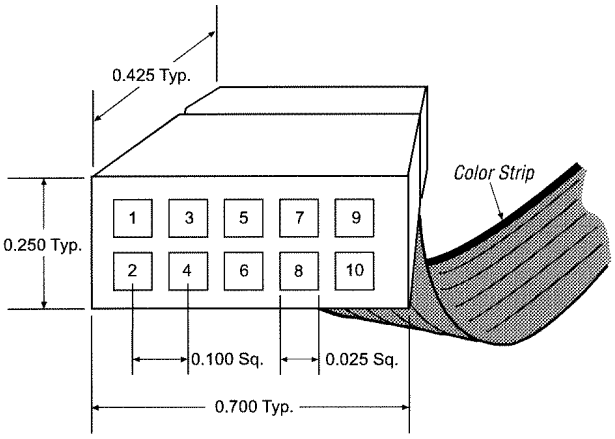


図 3 バイトブラスタ・メールヘッダ・コネクタ

表 3　バイトブラスタ・メールヘッダ・コネクタ信号

Pin	Signal Name	Description	Pin	Signal Name	Description
1	TCK	Clock	6	-	No Connecnt
2	GND	Signal GND	7	-	No Connecnt
3	TCO	Data from device	8	-	No Connecnt
4	VCC	Power supply	9	TDI	Data to device
5	TMS	Control	10	GND	Signal GND

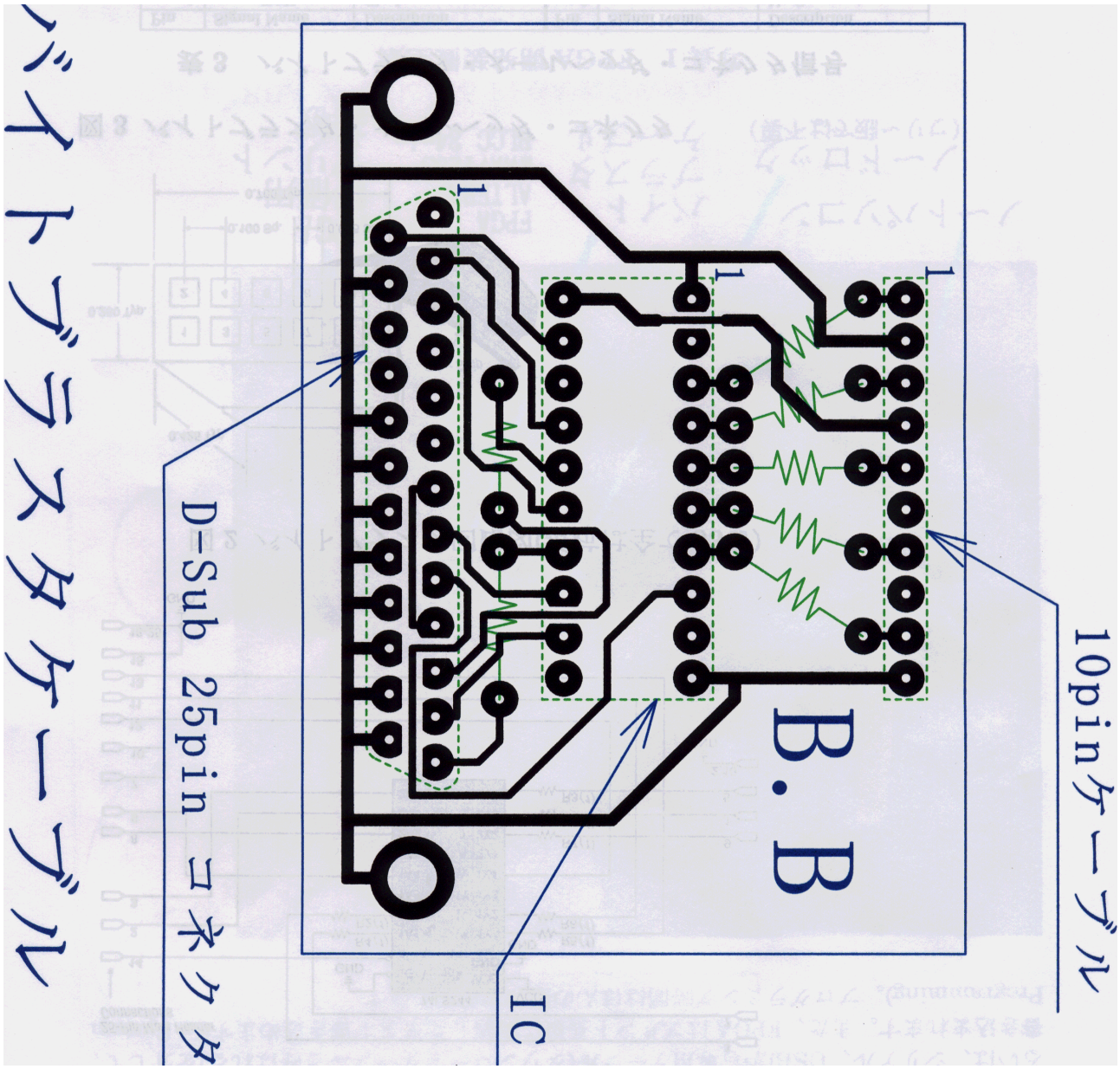


図 4 バイトブラスタ基板実体配線図

2.2.2 FPGA 評価ボード(書き込み器)の製作

本講習会では Altera 社の MAX7128S という FPGA(CPLD)をターゲットとします。ここでは、この FPGA の評価ボードを製作します。MAX7128S は 84 ピン PLCC ソケットタイプで、また EEP-ROM タイプなので一度書き込めば、電源を切っても、ソケットから抜いても回路情報は消去されません。すなわち、この評価ボードは書き込みボードとしても活用することができます。 本講習会では次ページ**図 6** のようにこの回路を試作します。

表 4. FPGA 評価ボード部品表

品名	個数	品名	個数
プリント基板	1	ヘッダピン 2列10pin	1
PLCC 84pin ソケット	1	ヘッダピン 2 列 26pin	1
DIP 14pin ソケット	1	水晶発振モジュール	1
アレー抵抗 103×8	2	抵抗 470Ω	1
バイナリスイッチ	2	LED 緑	1

表 5　MAX7128S ピンアサイン表

ピン	名称	ピン	名称	ピン	名称	ピン	名称	ピン	名称
1	GCLR	21	I/O	41	I/O	61	I/O	81	I/O
2	OE2/GCLK2	22	I/O	42	GND	62	TCK	82	GND
3	Vcc	23	TMS	43	Vcc	63	I/O	83	GCLK1
4	I/O	24	I/O	44	I/O	64	I/O	84	OE1
5	I/O	25	I/O	45	I/O	65	I/O		
6	I/O	26	Vcc	46	I/O	66	Vcc		
7	GND	27	I/O	47	GND	67	I/O		
8	I/O	28	I/O	48	I/O	68	I/O		
9	I/O	29	I/O	49	I/O	69	I/O		
10	I/O	30	I/O	50	I/O	70	I/O		
11	I/O	31	I/O	51	I/O	71	TDO		
12	I/O	32	GND	52	I/O	72	GND		
13	Vcc	33	I/O	53	Vcc	73	I/O		
14	TDI	34	I/O	54	I/O	74	I/O		
15	I/O	35	I/O	55	I/O	75	I/O		
16	I/O	36	I/O	56	I/O	76	I/O		
17	I/O	37	I/O	57	I/O	77	I/O		
18	I/O	38	Vcc	58	I/O	78	Vcc		
19	GND	39	I/O	59	GND	79	I/O		
20	I/O	40	I/O	60	I/O	80	I/O		

表 6 試作 FPGA 評価ボードの外部接続用ピンアサイン

ピン	名称	ピン	名称	ピン	名称	ピン	名称
1	GND	8	I/O58	15	I/O67	22	I/O76
2	I/O55	9	I/O61	16	I/O68	23	Vcc
3	GND	10	I/O60	17	I/O69	24	I/O75
4	I/O54	11	I/O63	18	I/O70	25	Vcc
5	I/O52	12	I/O64	19	I/O73	26	I/O79
6	I/O56	13	I/O65	20	I/O74		
7	I/O57	14	Vcc	21	I/O77		

表 6b バイナリロータリスイッチ　ピン番号

SW 桁	上位 H	下位 L
1	40	49
2	41	51
4	39	45
8	37	44

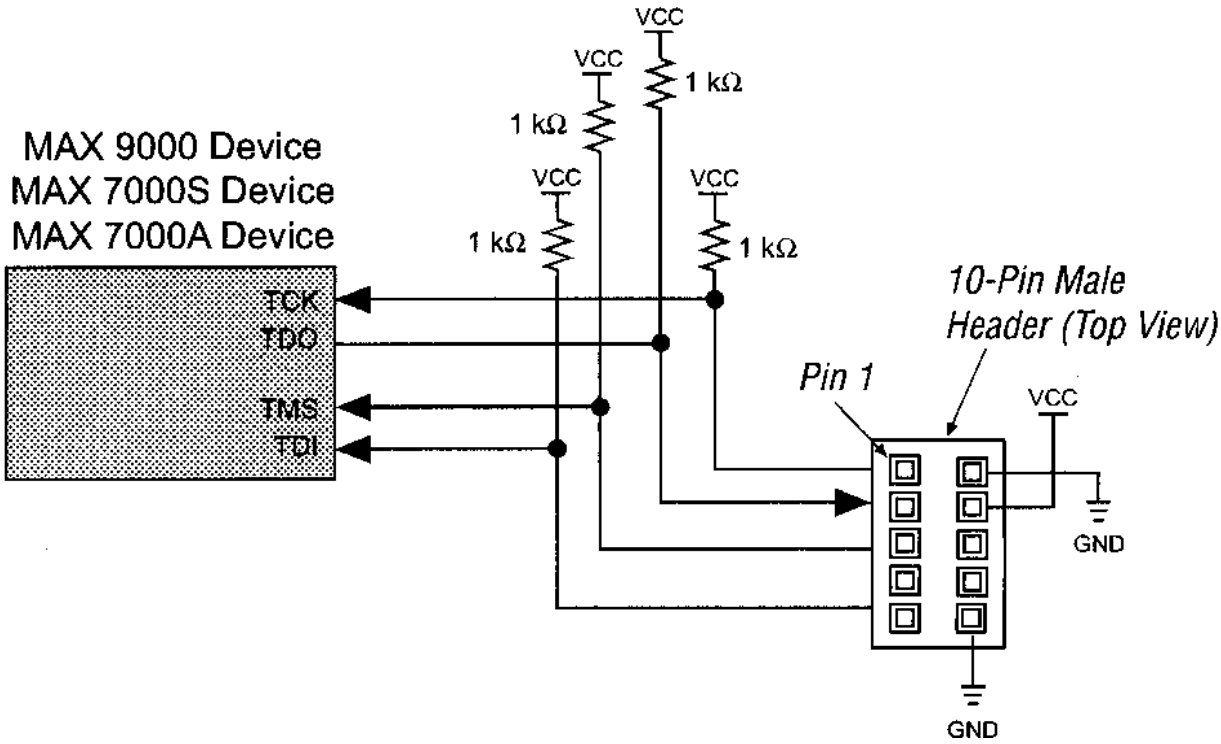


図 5. メールヘッダ部分回路図

図 6　FPGA 評価ボード実体配線図(最終ページにあります。)

2.3　同期回路設計について

ASIC の設計では、フリップフロップを使用する順序回路は、すべて**システムクロックの立ち上がり**に同期設計すべきである。同期設計ならば、タイミング計算も容易であり、無理がないので高速化も達成しやすい。もちろん、設計自体も統一的な指針で望める。

図 7 は順序回路の基本的な機能である、データ、リセット、イネーブルが全てクロックの立ち上がりに同期して行なえる基本回路(プリミティブ回路)である。
同期回路の特徴は、クロック線が全てのフリップフロップに共通(グローバル)で、途中にゲート等が入らない(ゲーテッドクロック)ことである。 なお、クロック信号の遅延を避けるための配線(クロックツリー)は EDA ツールで評価できる。

表 7　同期回路の基本回路入出力端子

入出力	機能	信号本数(値)
入力	クロック　clock	1
	データ　d	1
	イネーブル　e	1
	リセット　r	1
出力	状態　q	1
	反転状態　nq	1

表 8 同期回路の基本回路の真理値表

入　力				出　力	
clock	d	e	r	q	nq
↑	×	×	0	0	1
↑	×	0	1	q	nq
↑	0	1	1	0	1
↑	1	1	1	1	0

注意：E+MAX の D-F/F プリミティブの非同期プリセット(PRN),リセット(CLRN)は使用しない。

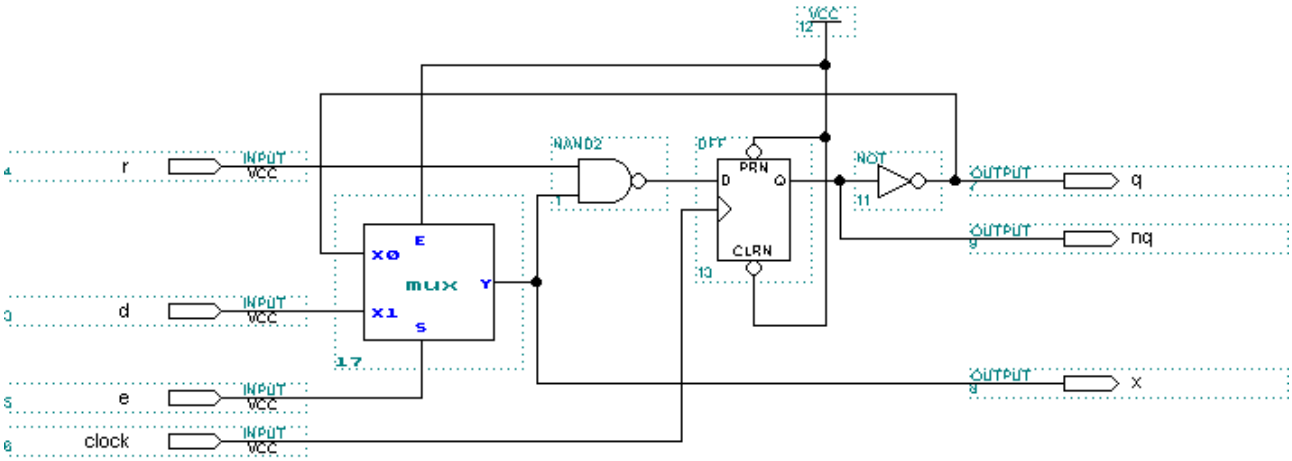


図 7　同期回路のプリミティブ (ファイル名 sprim.gdf)

3.EDA ツールの操作演習

3.1 組み合わせ論理回路

課題 1　2 入力 4 出力イネーブル付デコーダ

手順 1 開発するシステムの機能の確認

デコーダとは、**入力のある組み合わせ(コード)**が与えられた時に、**出力のどれか一つだけ**が'1'になる組み合わせ論理回路である。またコントロール入力端子として**許可(イネーブル)入力**がある。イネーブル入力が'0'の時は入力(コード)の値に関わらず、出力は全て'0'となる。　すなわち、汎用的なデコーダの入出力端子は**表 9** のとおりである。

表 9　デコーダの入出力端子

入出力	機能	信号本数(値)
入力	入力コード	n
	イネーブル	1
出力	選択出力	2 の n 乗

表 10 デコーダプリミティブ真理値表

入　力		出　力	
e	x	y1	y0
0	×	0	0
1	0	0	1
1	1	1	0

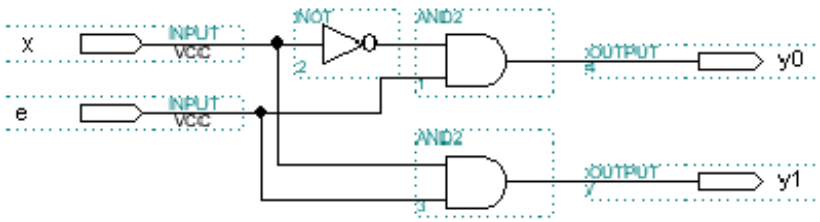


図 8 デコーダ・プリミティブ回路図(ファイル名 dec.gdf)

表 11 2-4 デコーダ真理値表

入　力			出　力			
e	x1	x0	y3	y2	y1	y0
0	×	×	0	0	0	0
1	0	0	0	0	0	1
1	0	1	0	0	1	0
1	1	0	0	1	0	0
1	1	1	1	0	0	0

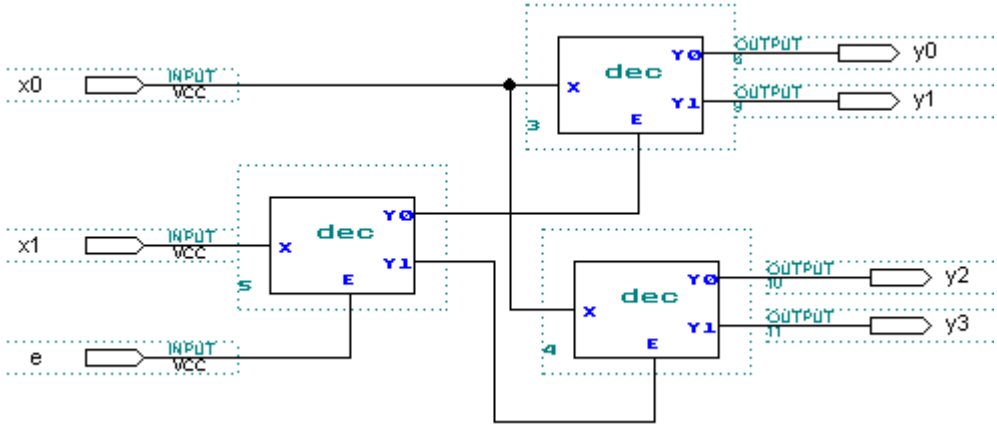


図 9 デコーダ・ネットワーク接続回路図（ファイル名 dec2_4.gdf)

手順 2 基本回路(プリミティブ)の回路図入力

図 10 のデコーダ・プリミティブの回路図を作成する。

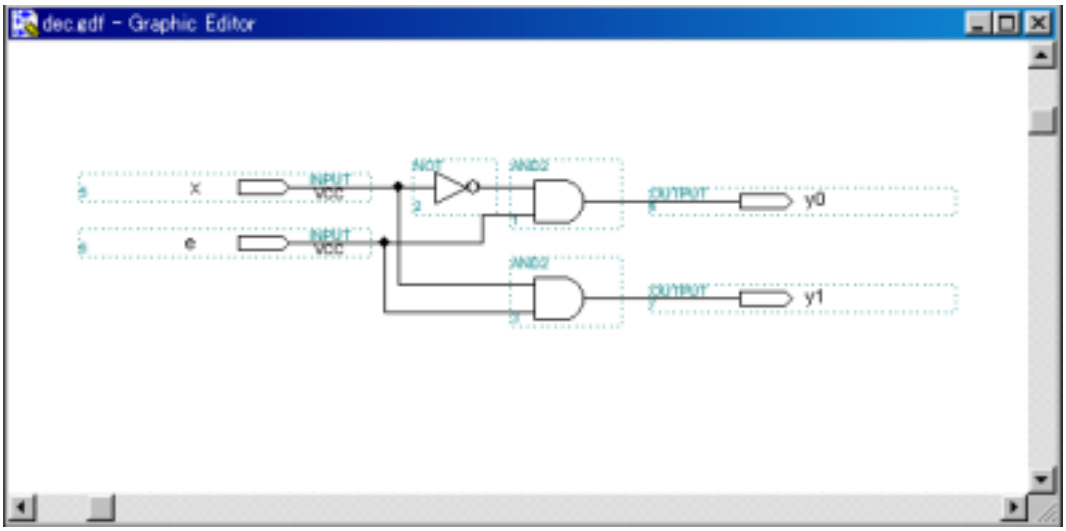


図 10：デコーダのプリミティブ

ステップ 1 グラフィックエディタの起動

図 11 の新規作成ボタンを左クリックして、**図 12** のように **Graphic Editor File(gdf)**を選択して、**OK** をクリックする。**図 10** の **Untitled 1-Graphic Editor** という画面が出てくる。

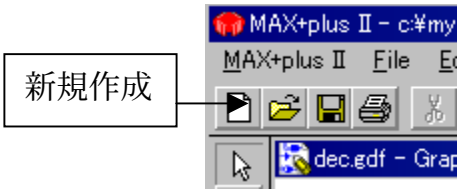


図 11：新規作成

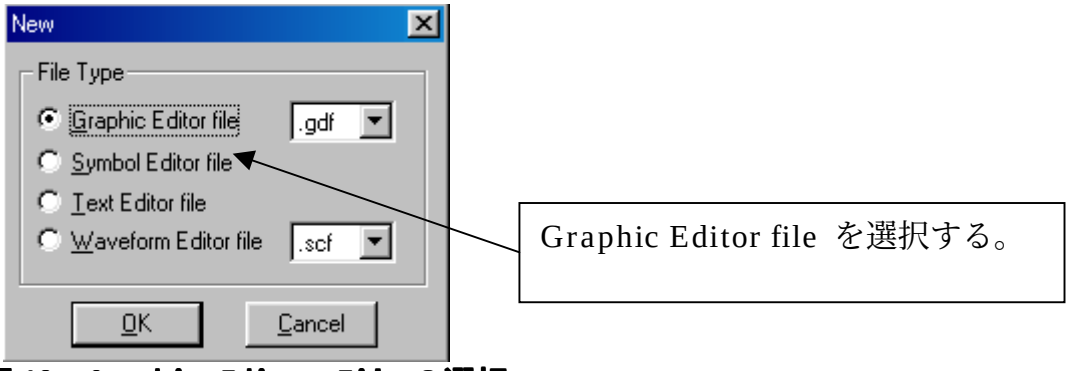


図 12：Graphic Editor File の選択

ステップ 2 部品シンボルのライブラリからの取り出し

まず **AND ゲート**を配置する。**図 13** ようにマウスカーソルを **Untitled 1-Graphic Editor**

① 画面の適当な場所に移動して左ダブルクリックする。

そうすると、**図 14** のような **Enter Symbol** ダイアログボックスが出てくる。そこで、

② **Symbol libraries 欄**の **c:¥baseline¥maxplus2¥max2ib¥prim** をダブルクリックする。

③ **Symbol Files 欄**の中に **and12,and2,and3,...**などが見えてくる。そこで、**and2** を選択し、ダブルクリックすると、**図 13 の**よう **2 入力 AND ゲート**が配置される。

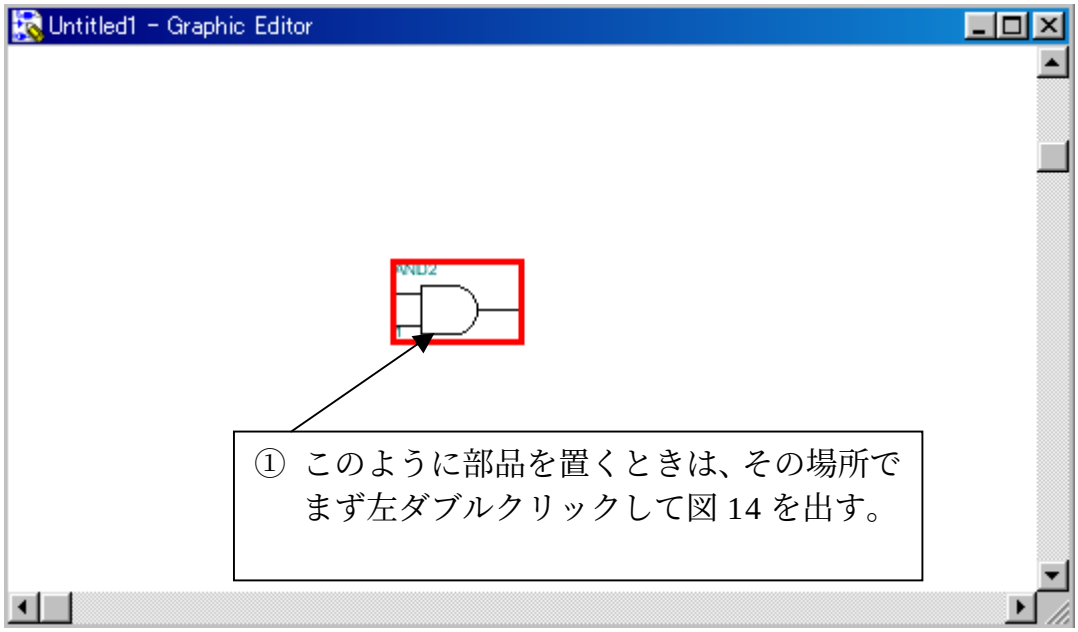


図 13：ライブラリからの部品選択と配置

図 14：Enter Symbol ダイログボックス

ステップ 3 部品のコピー＆ペーストによる配置

デコーダ・プリミティブを構成するためには、**and2,not,input** と **output** の 4 種類が必要である。そこで、それぞれを 1 個づつ**ステップ 2** のやり方で配置する。

それから、同じ部品を選択したい場合は該当の部品を左クリックして、**コピー(Ctrl+c)**して、 **ペースト(Ctrl+v)**すると**ステップ 2** のやり方を繰り返さなくても良い。

このようにして、**図 15** のように全ての部品を配置する。

ところで、**部品を移動**するには、その部品に左クリックしたまま、画面内(Untitled 1-Graphic Editor)のお好みの場所へ移動(**ドラッグ**)させる。また、全図形またはある部分を移動したい場合は移動したい部分の左上から右下までドラッグして離して一旦範囲を指定してから、その指定した部分をドラッグすることができる。

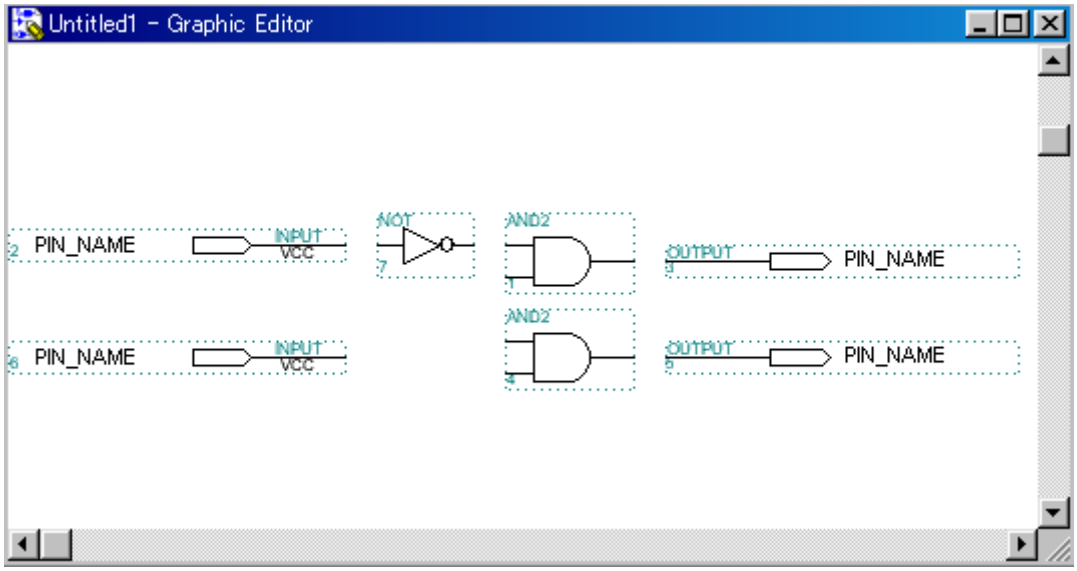


図 15：全部品の配置

ステップ 4 配線

図 8 のように部品間を配線をする。配線は**図 16** のように、その部品の端子を左クリックして、目的の部品の端子までドラッグして離す。ただし、配線作業をするときは画面を拡大した方ががやりやすい。

ポイント 1 ほんとうに**配線できているかどうか**は、部品を少しドラッグしてみて配線がいっしょに移動すれば大丈夫と確認できる。見た目は接続されてるように見えても、実際にはつながっていない場合があるので注意せよ。

ポイント 2 **図 17** に示すように**配線の途中から**さらに配線したい場合でも、**目的の部品の端子から配線**する。また曲がり配線（L 配線）をする場合は直線ごとに配線する。(**図 14** に参照)

ポイント 3 **配線間違い**があれば、間違った配線の部分をクリックして、**'Delete' キー**を押す　と間違っている部分が削除できる。要らない部品の削除も同様である。

ある部品の端子から他の部品の端子まで左クリックし、ドラッグして、離すと部品配線ができる。このときマウスカーソルは点線の十字となる。

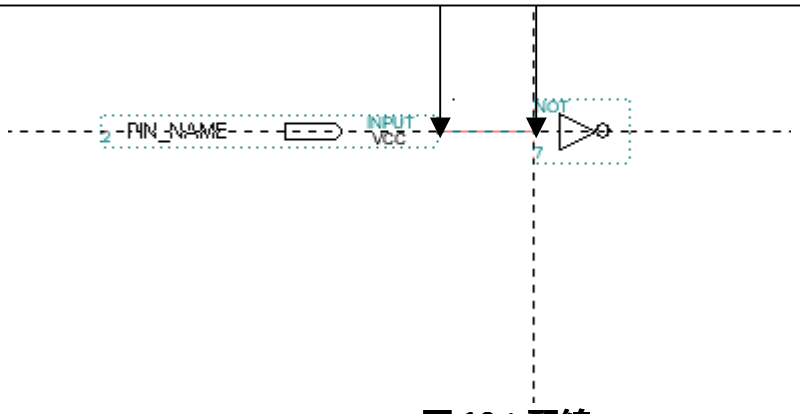
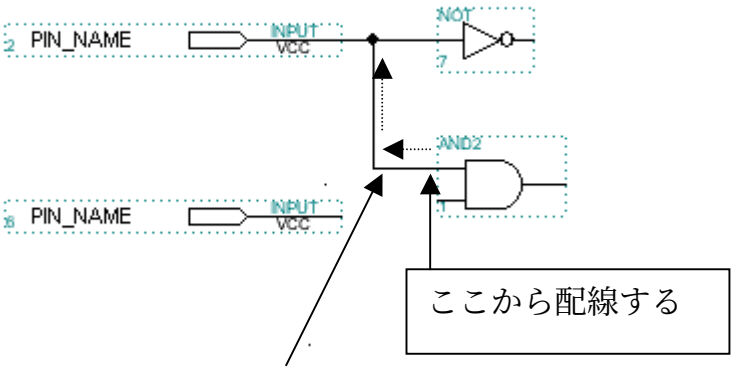


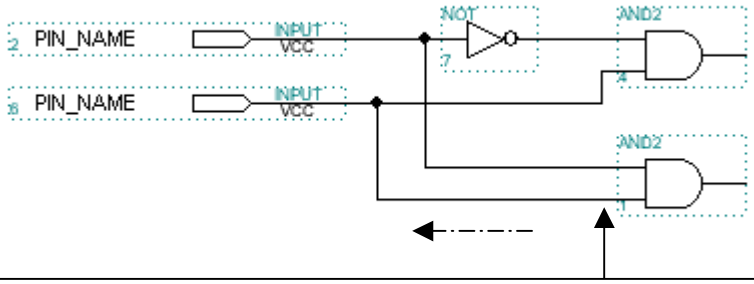
図 16：配線



ここから配線する

こういう配線をするときは部品の端子から配線を始める。

図 17：分岐配線



配線は部品の端子から引っ張り、一度、曲がるところで止め(クリックを離す)、そこからもう一度ドラッグして再び配線する。

図 18：L 配線

ステップ 5　入出力端子名を名付ける

図 19 のように、入出力端子名の場所をダブルクリックして、黒く反転させ、**手順 1** の**真理値表**どおりの名前に変更する。変更後、エンターキーを押すと、次の端子の名前変更に自動移動する(自動移動は、入力端子あるいは出力端子についてだけ)。

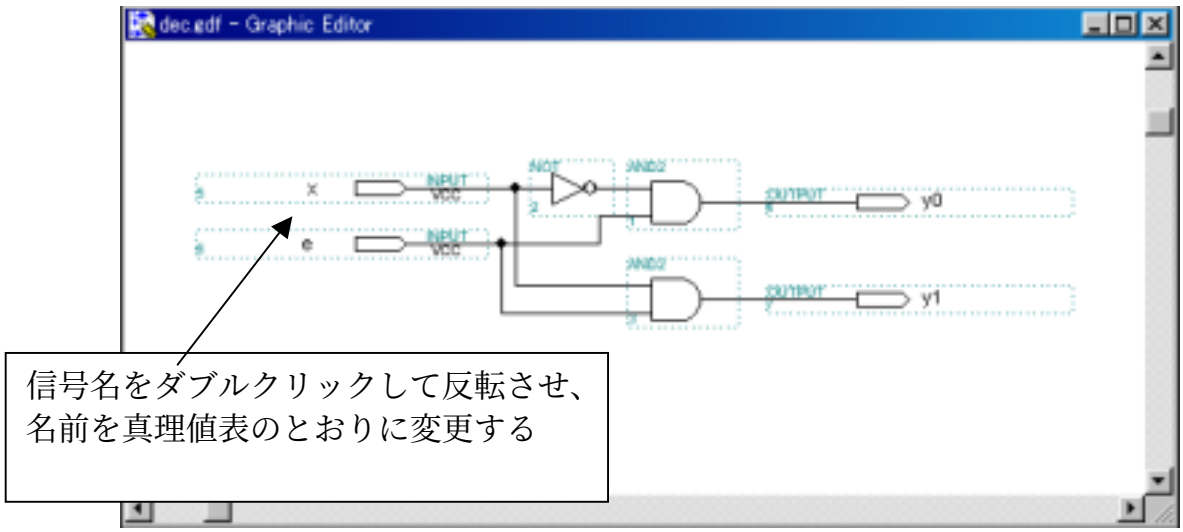


図 19：入出力端子名の更新

ステップ 6 回路図の保存

図 8 の回路図が完成したら、配線を確認してから、**図 20** の手順で保存する。保存先は自分のホームディレクトリ、ファイル名は**手順 1** で指定している **dec.gdf** とせよ。

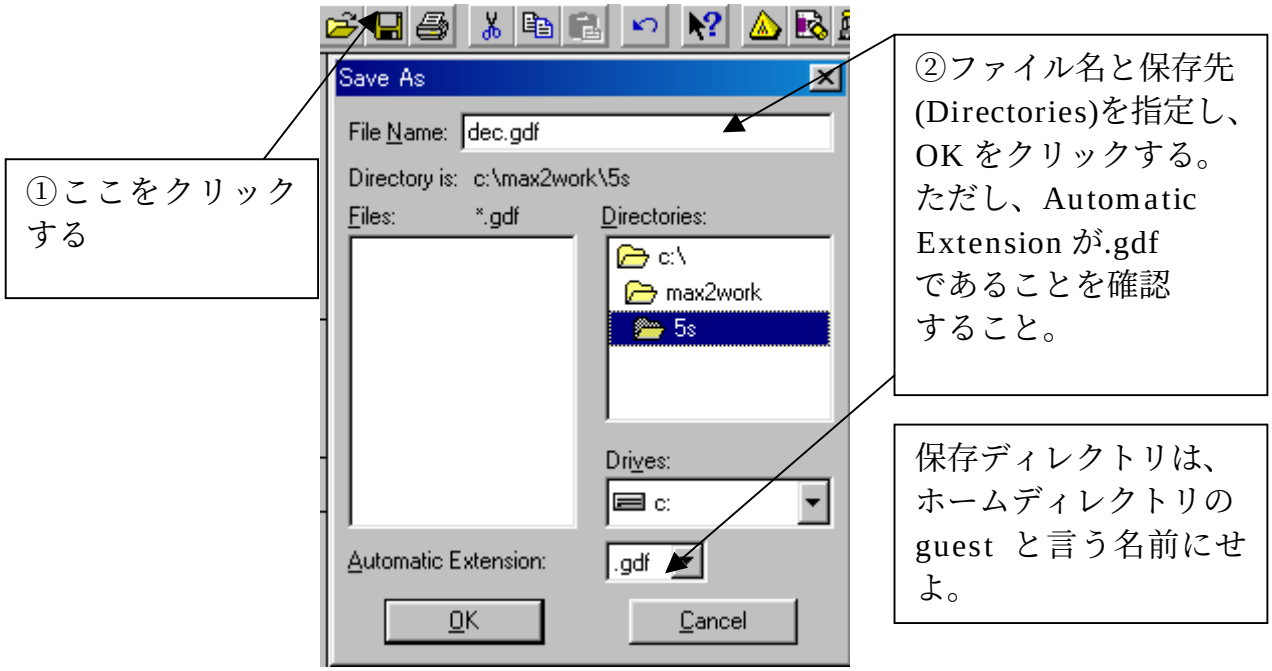


図 20：図形の保存

手順 3 プリミティブのシンボル(部品記号)の編集

ステップ 7　プリミティブのシンボル作成

完成した**図 8** のプリミティブ回路自体を一つの部品シンボルとする。**図 21** に示すように、メニューの **File** の **Create Default Symbol** にクリックする。この操作は一瞬で終わる。

これでグラフィックエディタウィンドウを右上の**×ボタン**をクリックして閉じる。

ステップ 8　シンボルエディタの起動

作成したシンボルを編集するためにシンボルエディタを起動する。**図 22** に示すように、**ファイルオープン**して、**Symbol Editor files の欄にチェック**が付いていることを確認する。そして、先ほど作成したシンボル名(dec.sym)を選択し、**OK** をクリックすると、**図 23** に示すようなシンボルが表示される。

図 21：シンボル作成

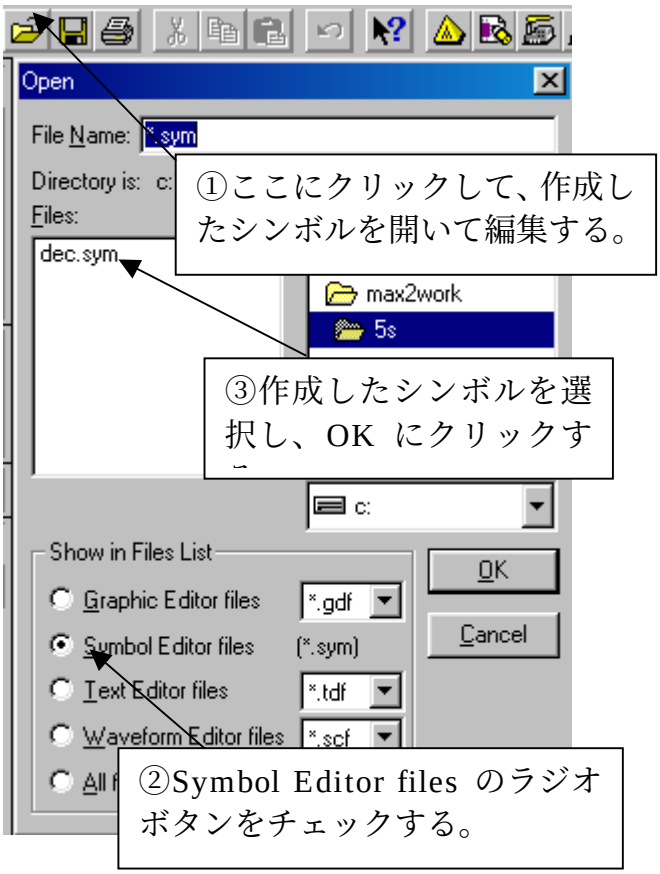


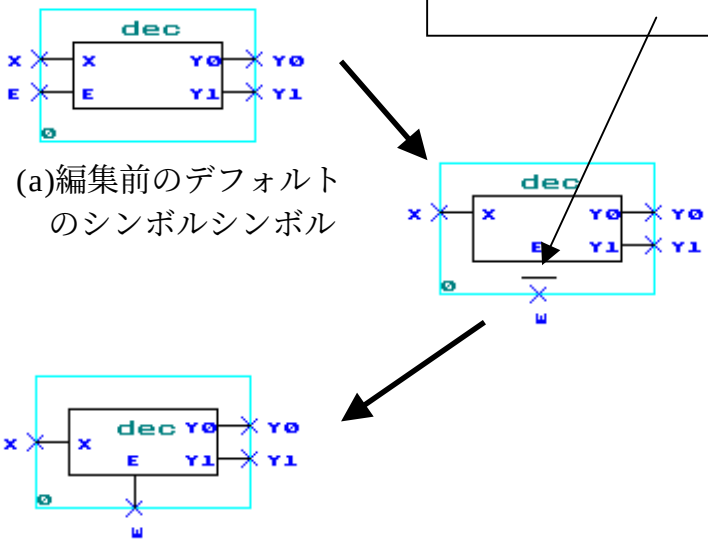
図 22：シンボルエディタ起動

ステップ 8　シンボルの編集

図 23 のシンボルのピン(入出力端子)の位置はデフォルト位置になっている。
これを手順 1 のネットワーク接続回路のようにピンの位置になるように編集する。

シンボルの編集(ピン位置または文字位置)は、編集したい部分を右クリックすると、編集メニューが出てくる。そのメニューを利用して、シンボルの編集ができる。
ここでは、次のように操作する。

① 内側の黒色の箱内の左側にある **E** という文字をドラッグして下に移動する。
② 外側の空色の箱の左縦線上にある　**×** というシンボルをドラッグして下に移動する。
③ ①と②を接続していた横向きの線を図の位置にドラッグし、マウスの右メニューを出して、**Rotate** をクリックし 270 ° を選択する。



(a)編集前のデフォルトのシンボルシンボル

(b)編集後のシンボル

図 23：作成したシンボル

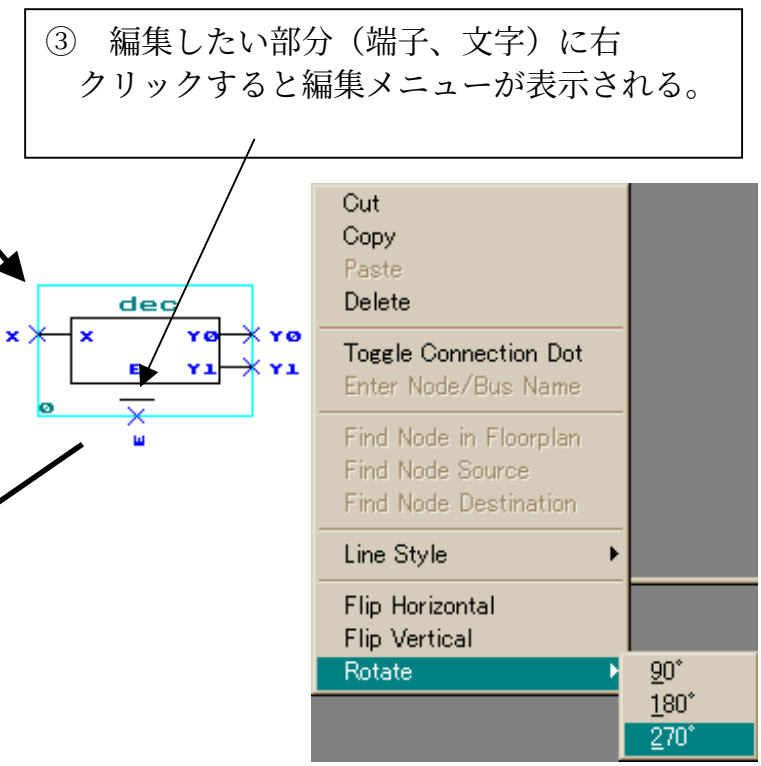


図 24：シンボルの編集

同様にして、端子 x の位置と、シンボル名 dec の位置を図 23(b)のように編集する。

シンボルの編集が完了すれば、**セーブボタンで上書き保存**する。またシンボルエディタウィンドウの右上の**×ボタン**をクリックして閉じる。

手順 4 プリミティブのネットワーク接続によるシステム回路図入力

ステップ 9　ネットワーク回路図の作成

ネットワーク回路の作成は**手順 2** のプリミティブの作成と全く同様である。ただし、作成したシンボルを部品として配置する場合は**図 21** のように **Symbol Libraries** のところにホームディレクトリを指定してから選ぶ。 保存名は**手順 1** で指定している(**dec2_4.gdf**)。

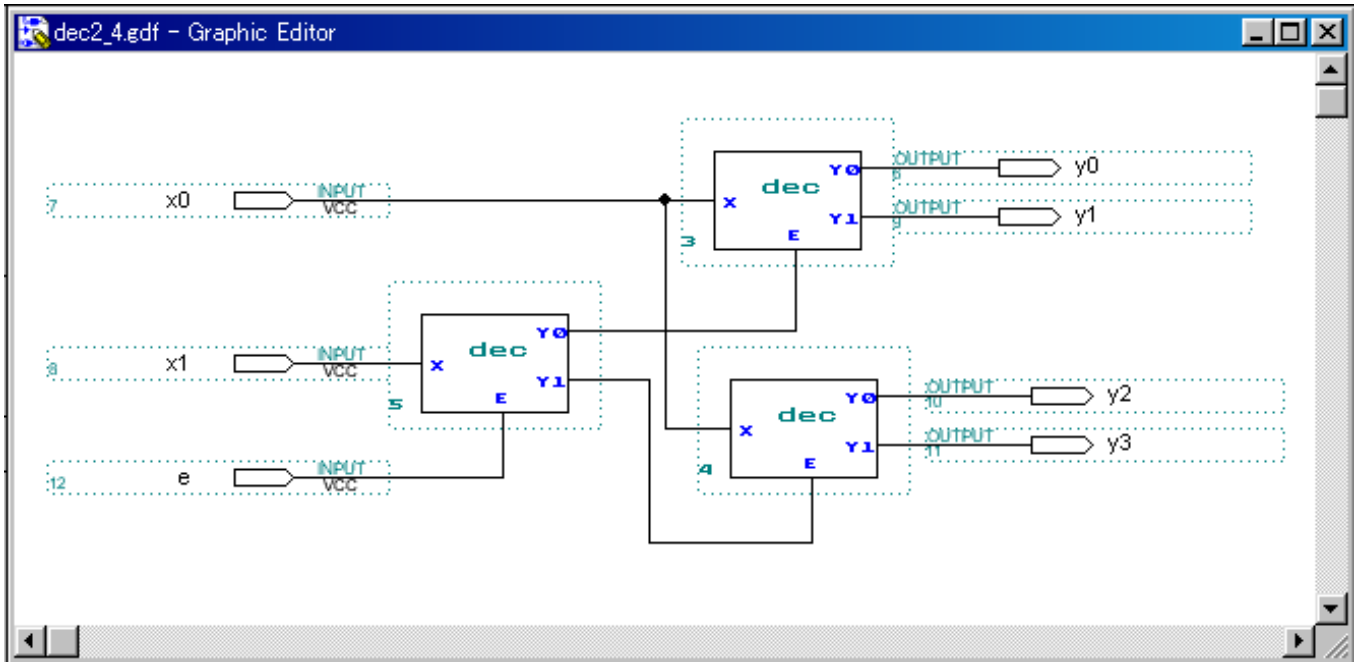


図 25：2 入力 4 出力イネーブル付デコーダのネットワーク接続

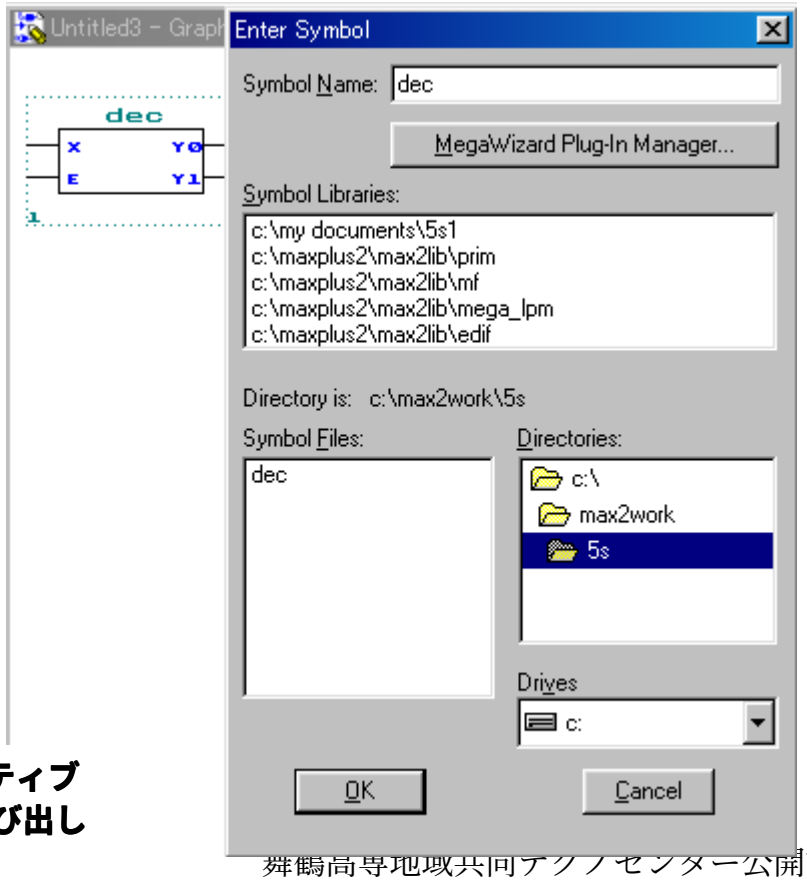


図 26 プリミティブシンボルの呼び出し

手順 5 コンパイル

ステップ 10 プロジェクトの登録

これから、作成した回路をコンパイルする作業にはいる。まず、**Ctrl+Shift+J** キーを押す。これは作成した**現在の回路をプロジェクトに登録**することを意味する。メニューからプロジェクトを登録することもできる。(メニューの **File→Project→Set Project to Current File** を選択する。)

ステップ 11 コンパイラの起動

コンパイラの起動は**図 27** の工場の形のボタンをクリックするか、メニューMAX+PLUS2 →Compiler を選択する。また Processing メニューの Function..をチェックする。

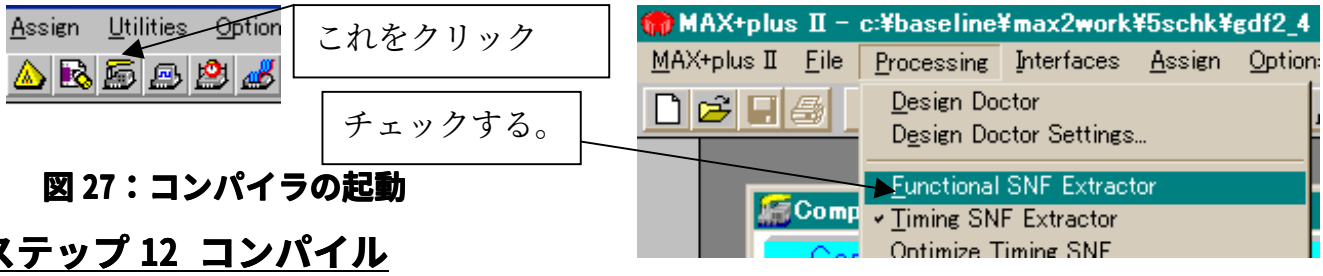


図 27：コンパイラの起動

ステップ 12 コンパイル

図 28 の **Start** をクリックするとコンパイルが始まる。エラーがある場合は、Messages 画面にエラーが表示される。エラーmessage をクリックして、間違っているところが表示される。間違っている部分を訂正し、セーブしてから、再びコンパイルする。

配線がつながっていないことをはじめ、ピン名を指定していないことなどがエラーの主な原因となる。配線をする際に、画面を拡大することがポイントである。

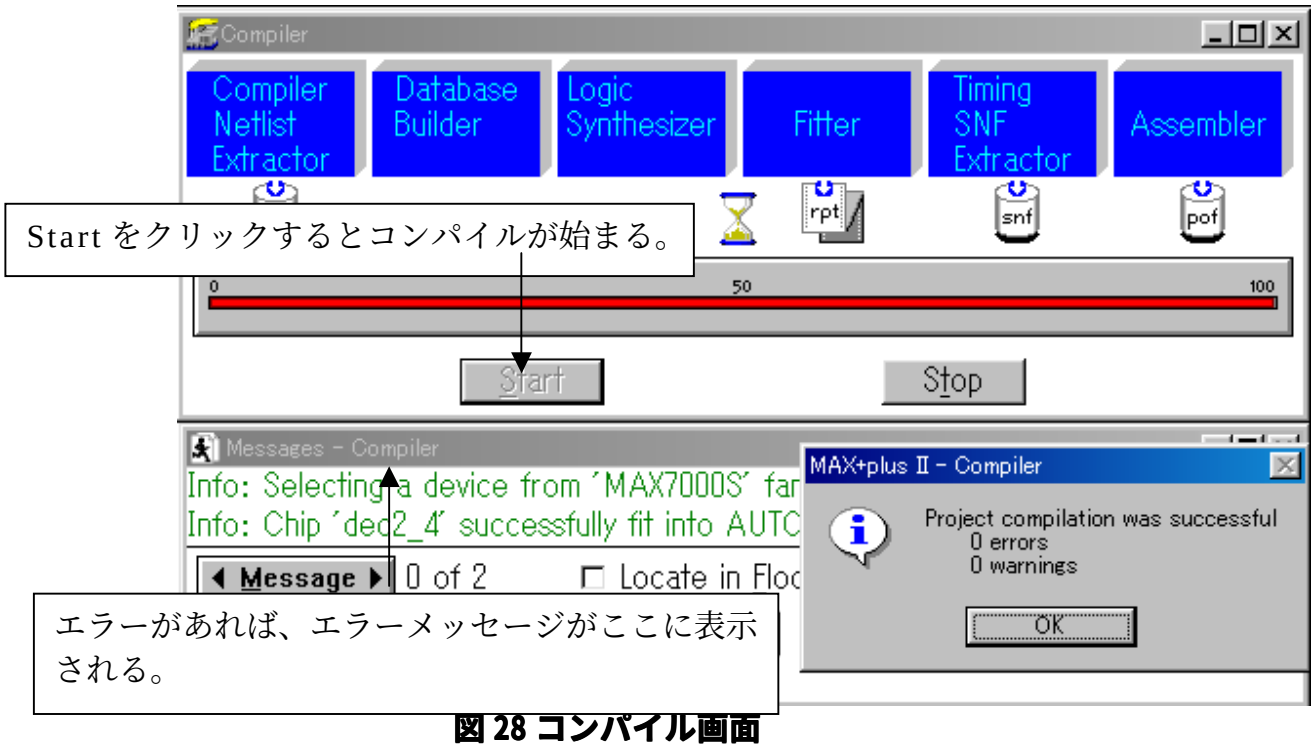


図 28 コンパイル画面

手順 6 シミュレーション

コンパイル過程がエラーなく終了したら、本回路のシミュレーションをやってみよう。シミュレーションによって、本回路が正しく動作しているかどうか確かめることができる。

ステップ 13 波形エディタの起動

図 29 に示すように新規作成から、Waveform Editor file(.scf)を起動する。**図 30** のように Waveform Editor の画面が表示される。

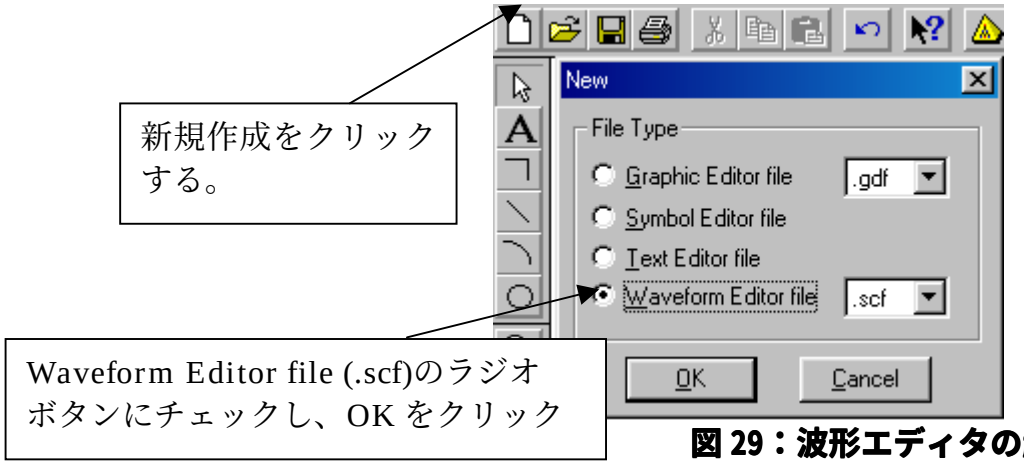


図 29：波形エディタの起動

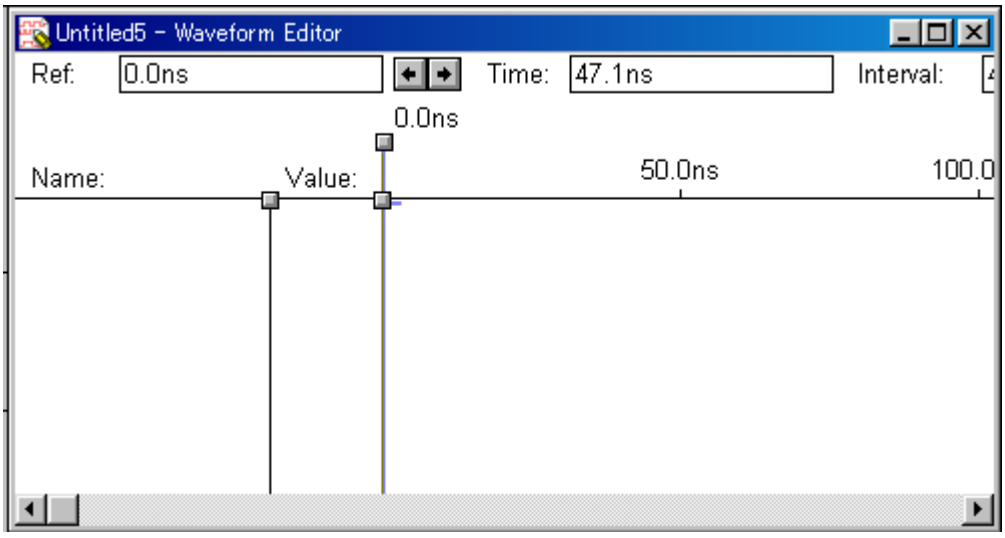


図 30：Waveform Editor の画面

ステップ 14 入出力端子情報の取り込み

図 31 のように**メニューの Node** をクリックし、**Enter Nodes from SNF** を選択する。

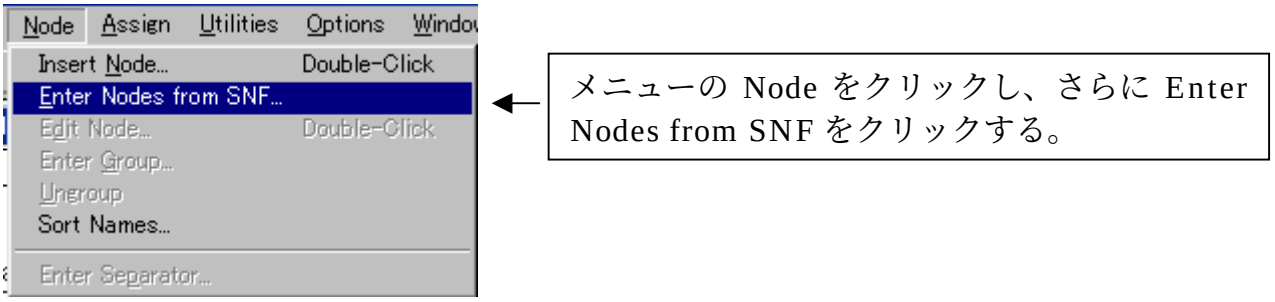


図 31：.snf ファイルから入出力の挿入

そうすると**図 32** が現れるので、図中の手順で入出力端子情報を取り込む。OK ボタンをクリックしたら、**図 33** に示すように **Untitled Waveform Editor** の画面が出てくる。

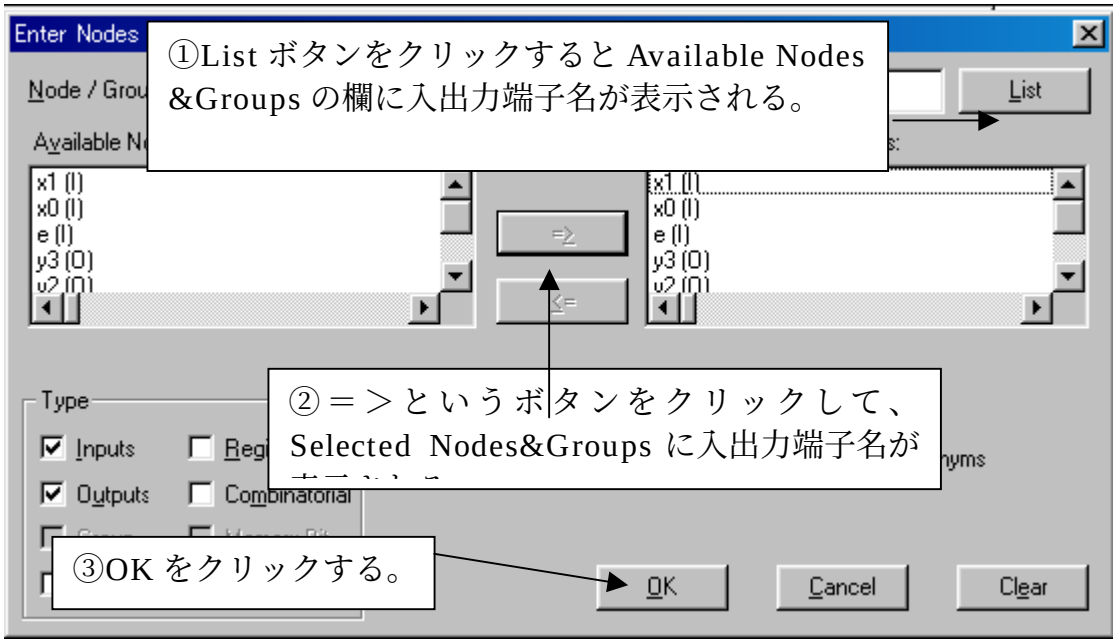


図 32:シミュレーションに表示される入出力端子名の選択

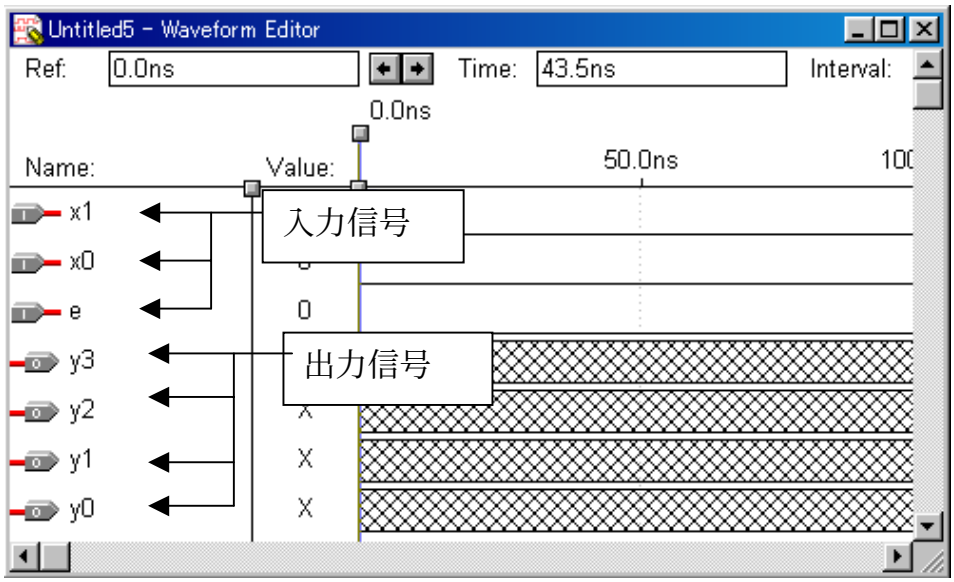


図 33：シミュレーションに用いる入出力信号

ステップ 15　入力波形の編集

ネットワーク回路のシミュレーションする際に、テスト入力信号パターンを与える。このテスト入力信号パターンは、**手順 1** で与えられた**真理値表**に基づくものとする。

ステップ 15-1　全ての組み合わせが必要な入力の編集

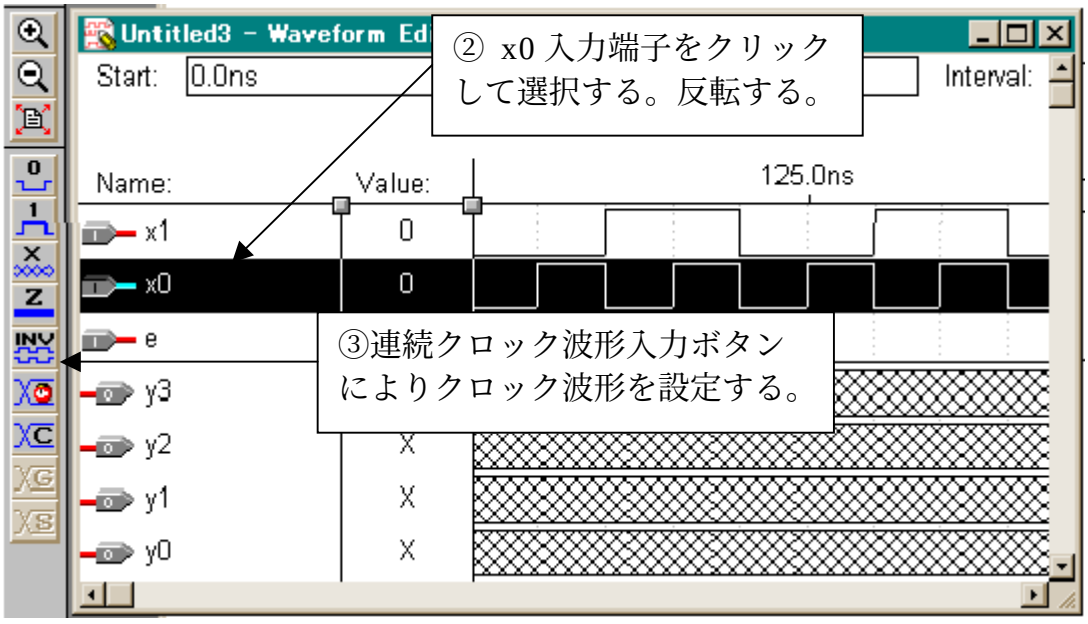
まず、**図 34** のように入力信号 **x0** と **x1** と **e** の波形を作成する。これは真理値表どおりに、**x0** と **x1** と **e** の全ての組み合わせについて与える必要があるので、連続的な方形波として入力する。

X0 の入力

① **図 35** のように、メニューの **Option→Grid Size** で **Grid Size を 25.0ns** に設定する。
② **WaveForm Editor** ウィンドウの **Name** の下の **x0** の端子をクリックして、**x0 全体**を選択する。
③ **WaveForm Editor** ウィンドウの**左側の時計マークの周期的パルス入力ボタン**をクリックすると、全時間にわたって、周期 50ns(25ns×2)の方形派が設定される。

x1 の入力

x1 も **x0** と同様に入力する。ただし、①で **Grid Size を 50.0ns** に設定する。この 2 倍づつに設定するのが、全ての組み合わせを入力するコツである。



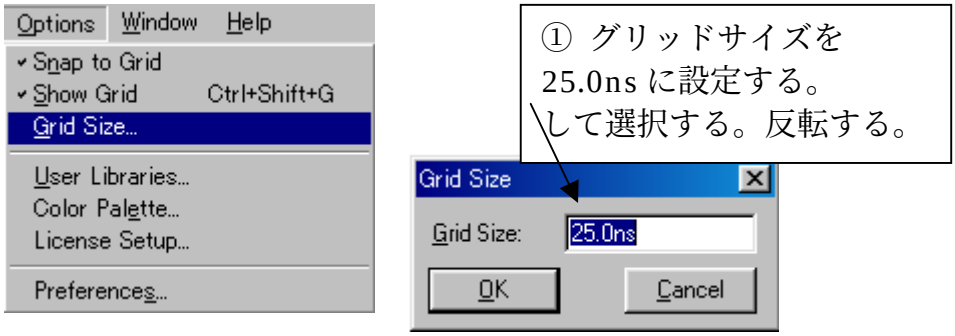


図 34：全ての組み合わせが必要な入力の編集

図 35：グリッドサイズの設定

ステップ 15-2 ある特定の時間だけ'0'か'1'を指定する信号の入力

例えば、**図 36** に示すように **e**(イネーブル)信号の最初の **50ns** までは'**0**'状態に設定するには、e 端子をクリックし、全体を反転させ、編集メニューの'**1**'**のボタン**をクリックして、全体を 1 に設定した後で、**0ns** の時点から **100ns** の時点まで、**左ドラッグ**して離す。すると **e** 信号の 0ns から 100ns までの領域が黒くなり、編集メニューの'**0**'**のボタン**をクリックするとその領域が'0'に変わる。

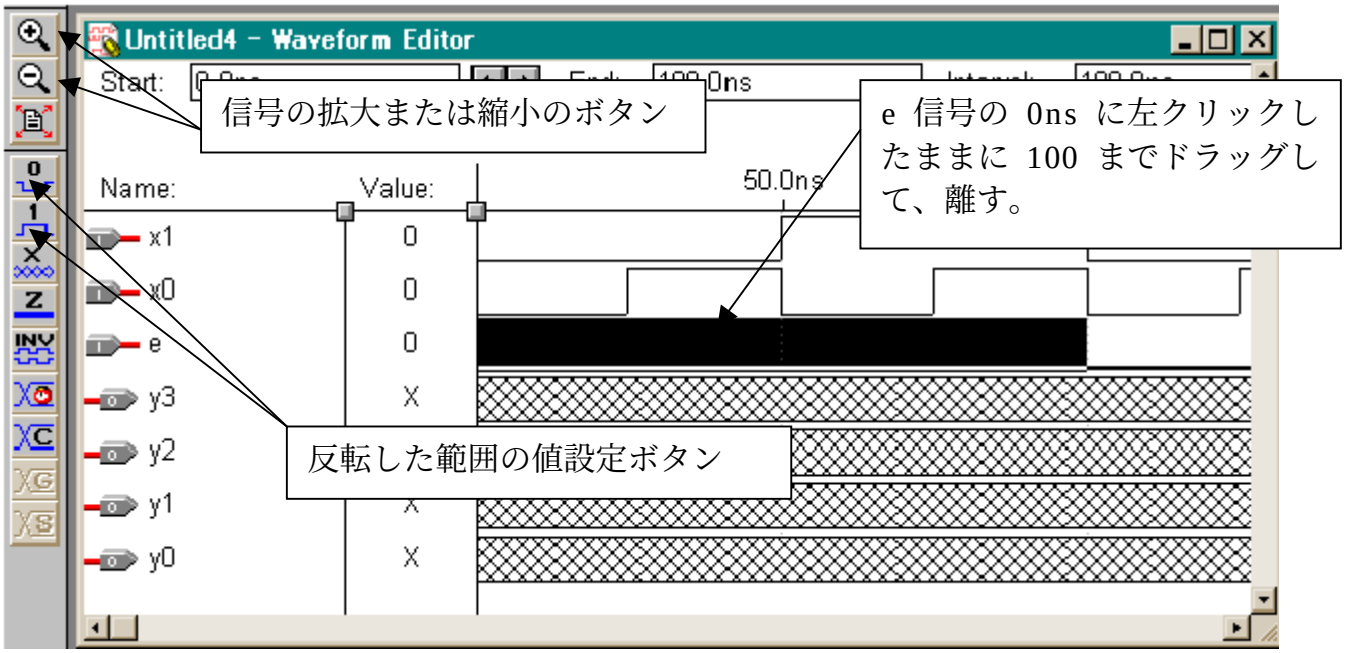


図 36：ある特定の時間だけ'0'か'1'を指定する信号の入力

ステップ 16 入力波形の編集

入力信号が編集できたら、**図 37** のようにして**.scf** ファイルとして保存する。

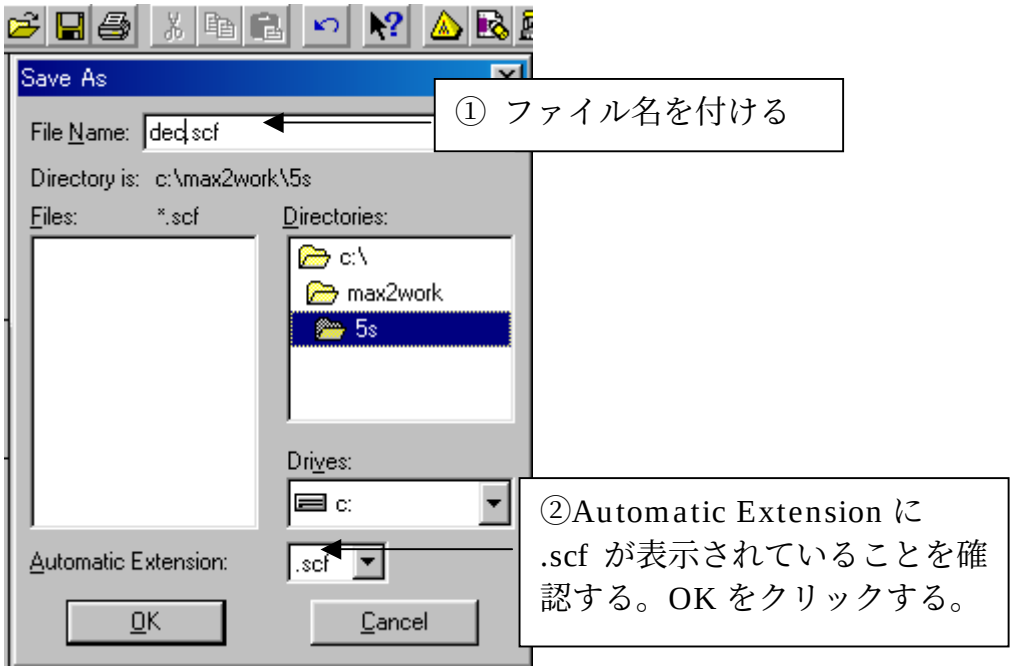


図 37：作成したシミュレーション信号の保存

ステップ 17 シミュレーションの実行と確認

図 38 に示すようにシミュレーションボタン(**Simulator**)をクリックすると、**図 39** のような画面(**Simulator Timing Simulation**)が表示される。

①その画面の **Start** ボタンをクリックする。

②エラーが発生しなければ、Max+plus Ⅱ-Simulator の **OKボタン**をクリックする。

③波形シミュレーションの結果を見るために、Simulator Timing Simulation 画面の **Open SCF ボタン**をクリックすると、**図 40** のような画面が表示される。これは本回路のシミュレーション結果である。**真理値表の出力結果と合致していることを確認する。**

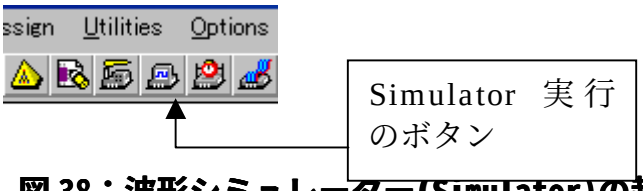


図 38：波形シミュレーター(Simulator)の起動

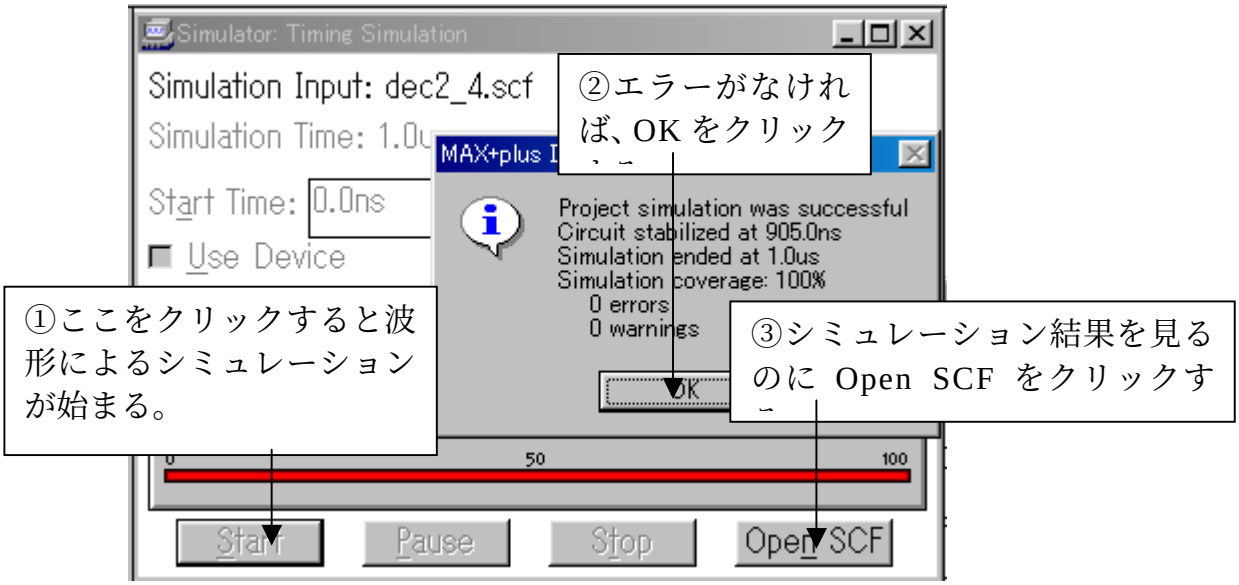


図 39：Simulator 画面

エラーが表示される場合は、エラー・メッセージに従い、エラーを修正する。

もし、シミュレーション結果が真理値表の結果と異なれば、もう一度、回路を見直すこと。

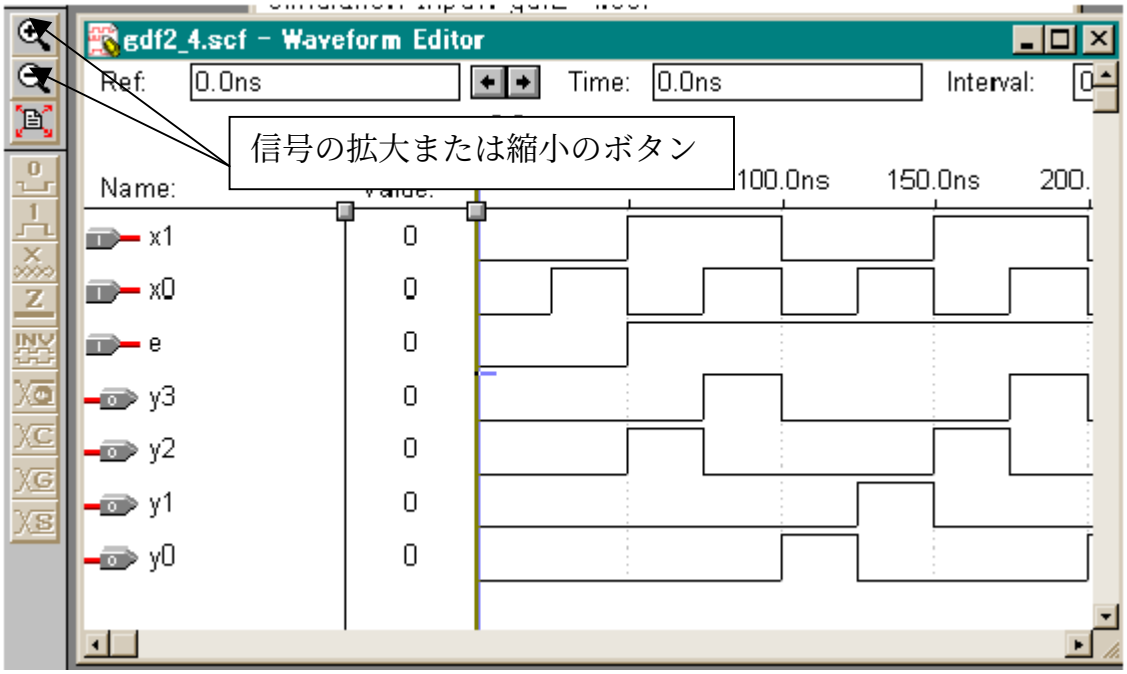


図 40：シミュレーション結果

シミュレーション結果を観測するには左の虫眼鏡のボタンで縮小して見やすくすれば良い。

ステップ 18 ピンアサイン

シミュレーションが成功すれば、Assign →Device でﾀｰｹﾞｯﾄ FPGA を選び、Assign → Pin/Location/Chip から図 41 のようにピン番号に入出力信号を割り振る。

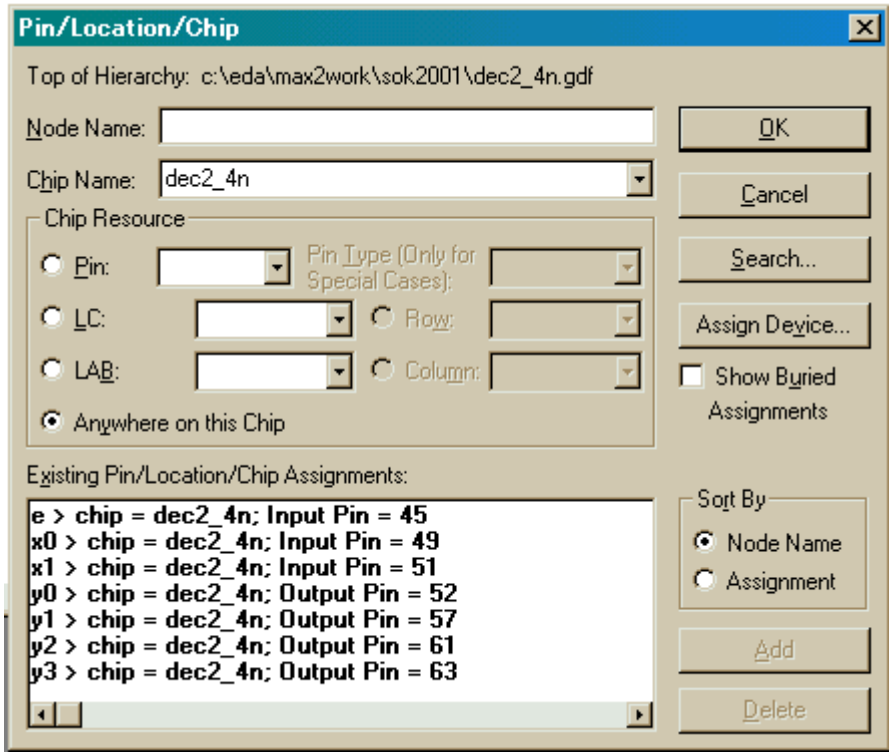


図 41：ピンアサイン

ステップ 18 プログラミング(書き込み)

ピンアサインが済めばもう一度コンパイルする。そして、速度的に問題がなければ、いよいよ FPGA に回路を書き込む。パソコンのプリンタポートからバイトブラスタケーブルを介してターゲットの FPGA ボードに接続し、MAX+PLUS2→Programming から図 42 の Program のボタンを押す。わずか数秒で書き込みは終了する。

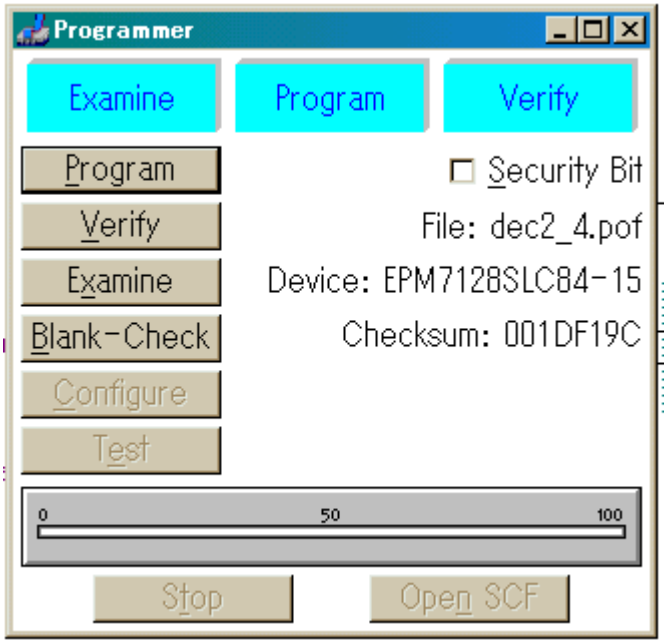


図 42：プログラム

課題 2　4 入力 1 出力マルチプレクサ

手順 1 開発するシステムの機能の確認

マルチプレクサとは **2 の n 乗.対.1** の切り替えスイッチである。このスイッチの切り替えのための **n 本**のコントロール入力端子を**セレクト入力**という。また**イネーブル入力**はこれが'0'の時は、入力がいかなる値であっても、出力が'0'となる。

すなわち、汎用的なマルチプレクサの入出力端子は以下のとおりである

表 12 マルチプレクサの信号

入出力	機能	信号本数(値)
入力	入力信号	1
	セレクト入力	n
	イネーブル	1
出力	分岐選択出力	2 の n 乗

表 13 マルチプレクサ基本回路真理値表

入　力				出　力
e	s	x1	x0	y
0	×	×	×	0
1	0	0	×	0
1	0	1	×	1
1	1	×	0	0
1	1	×	1	1

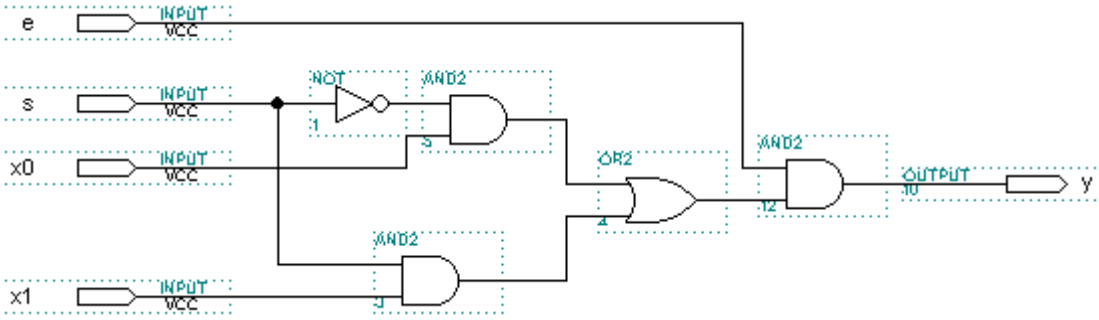


図 42　マルチプレクサ基本回路(プリミティブ)（ファイル名 mux.gdf)

表 14　4-1 マルチプレクサネットワーク接続　ファイル名 mux41.gdf

入　力							出力
e	**s1**	**s0**	**x3**	**x2**	**x1**	**x0**	**y**
0	×	×	×	×	×	×	0
1	0	0	x3	x2	x1	x0	x0
1	0	1	x3	x2	x1	x0	x1
1	1	0	x3	x2	x1	x0	x2
1	1	1	x3	x2	x1	x0	x3

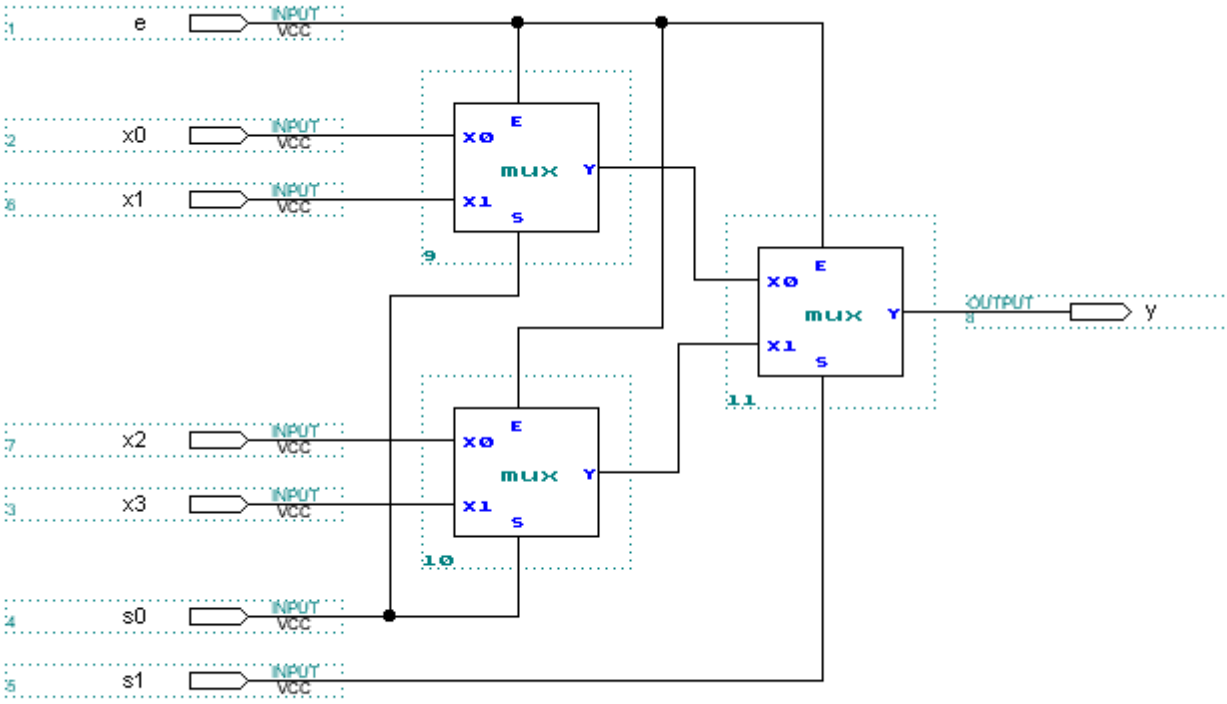


図 43　4-1 マルチプレクサネットワーク接続　ファイル名（mux41.gdf)

3.2 順序回路

課題 3　シンクロナイザの回路図入力

手順 1 開発するシステムの機能の確認

シンクロナイザすなわち同期化器とは、クロックの立ち上がりに非同期な入力……例えば人間のスイッチ入力を同期化させる順序回路である。

この場合、同期化出力は、図のように非同期入力が 1 になった直後のクロック立ち上がりで 1 となり、その次の立ち上がりで 0 に戻る。このとき、同期化出力がクロック立ち上がりで 1 になっているのは、0 に戻るときであることに注意が必要である。このことは、E+MAX のシミュレーションで確認できる。

表 15 シンｋロナイザの入出力信号

入出力	機能	信号本数(値)
入力	クロック　clk	1
	非同期入力　unsyunc	1
出力	同期化出力　insyunc	1

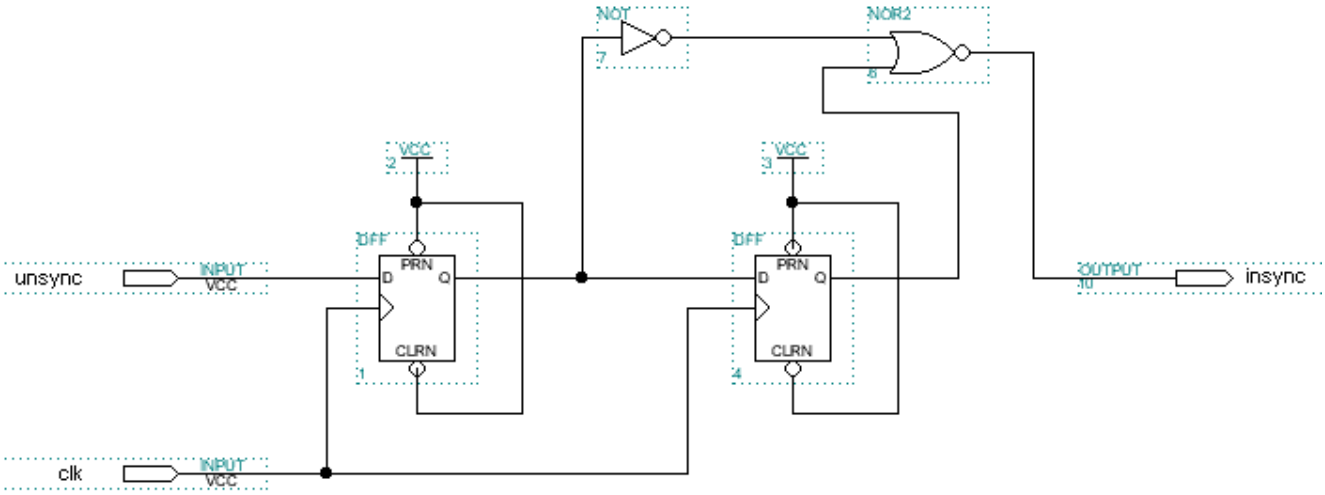


図 44 シンクロナイザの基本回路(プリミティブ)　(ファイル名 sync.gdf)

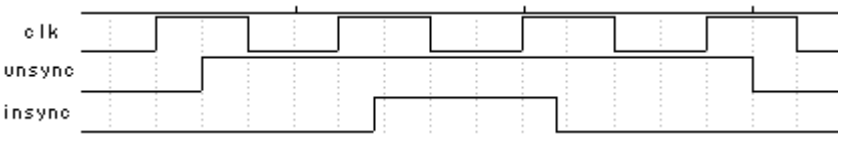


図 45 シンクロナイザのシミュレーション結果

課題 4　アップダウンカウンタの VHDL 記述

　　複雑な順序回路の回路図を作図するのは相当の知識が必要です。
ここでは、ハードウェア記述言語 VHDL によるアップダウンカウンタの記述について説明します。　VHDL とは米国国防総省で規格化された現在最も標準的なハードウェア記述言語の一つです。VHDL の仕様は大変幅広く、とても全部を把握することは不可能ですが、FPGA で実現できる程度の文法であれば、数週間で学習することも可能です。

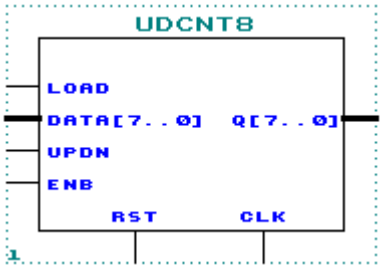


図 46　カウンタの代表的なシンボル

```
 1 library IEEE;
 2 use IEEE.std_logic_1164.all;
 3 use IEEE.std_logic_unsigned.all;
 4
 5 entity udcnt8 is
 6   port( clk,rst,enb,updn,load : in    std_logic;
 7             data : in std_logic_vector (7 downto 0);
 8             q : out std_logic_vector (7 downto 0));
 9 end udcnt8;
10
11 architecture RTL of udcnt8 is
12 signal reg:std_logic_vector (7 downto 0);
13 begin
14     q <= reg;
15     process (clk) begin
16         if (clk'event and clk='1') then
17             if (rst='1') then
18                 reg <= (others => '0');
19             elsif (load='1') then
20                 reg <= data;
21             elsif (enb='1') then
22                 if (updn='1') then
23                     reg <= reg+'1';
24                 else
25                     reg <= reg-'1';
26                 end if;
27             end if;
28         end if;
29     end process;
30 end RTL;
```

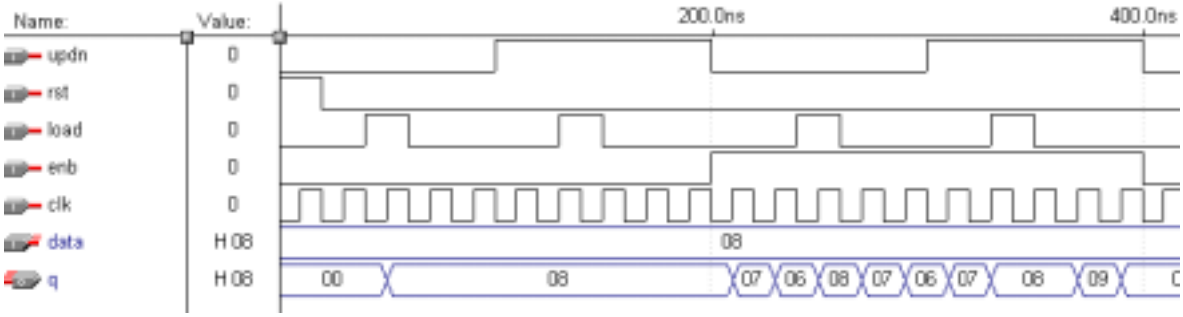


図 47　アップダウンカウンタのシミュレーション例

4.任意周波数信号発生器の実現

最後に、任意の周波数を発生できる発振器を実現してみる。すなわち、入力としてディジタル・バイナリ・スイッチで設定した周波数の方形波を出力する回路である。これは、V/F コンバータとしていろいろな用途に使用できる。

このようないわゆる「周波数シンセサイザ」は PLL(Phase-Locked Loop)で実現するのが一般的であるが、ここではもっと簡単にΔΣ変調器の構成により、開ループで実現する。

図 48 にその回路図を示す．詳細については講義中に解説する。

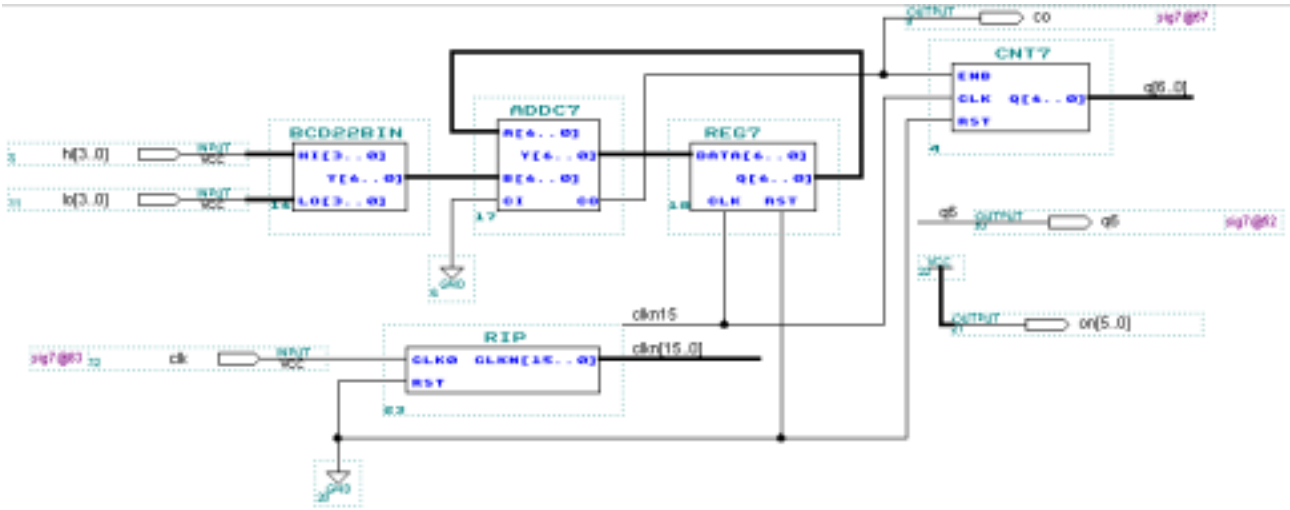


図 48 任意周波数発生回路

5.おわりに

CAD(EDA)ツールによる IC の設計と FPGA による実現について、一通りの作業手順を紹介しました。踏まなければならない手順は数多いのですが、一度経験してしまえば、きっと思ったより簡単であると感じてもらえたのではないでしょうか。

最近では、巷のパーツショップの少なくなり、小口の IC を入手するのは困難になってきています。そこで、このように FPGA で実現できればそのようなリスクを避けることができます。それよりも、ユーザ自身が新しい発想で IC を実現するということが、現実のものとなってきたということです。実際に、FPGA はもう最新のネットワーク機器内に数多く使用されており、大きな評価を得ています。

ただ、この分野は技術の進歩が本当に日進月歩であるため、インターネットや専門雑誌(CQ 出版社、DESIGN WAVE MAGAZINE)等で情報収集を続けることが大変重要です。この公開講座をきっかけに、今後、制御系や IC 開発について技術的なご相談がございましたら、またぜひ舞鶴高専地域共同テクノセンターまでご連絡をお待ちしています。

付録 ピンアサインと実験条件

1. デコーダ

表 A1 ピンアサイン

信号名	ピン番号
e	45
x0	49
x1	51
y0	52
y1	57
y2	61
y3	63

表 A2 実験条件

SW 入力	2 進数値	反転信号値	出力信号値	LED 点滅
L	e,x1,x0	e,x1,x0	y3,y2,y1,y0	y3,y2,y1,y0
0	000	111	1000	0111
1	001	110	0100	1011
2	010	101	0010	1101
3	011	100	0001	1110
4	100	011	0000	1111
5	101	010	0000	1111
6	110	001	0000	1111
7	111	000	0000	1111

2.同期回路プリミティブ

表 A2 ピンアサイン

信号名	ピン番号
clock	83
d	49
e	51
nq	61
q	57
r	45
e	52

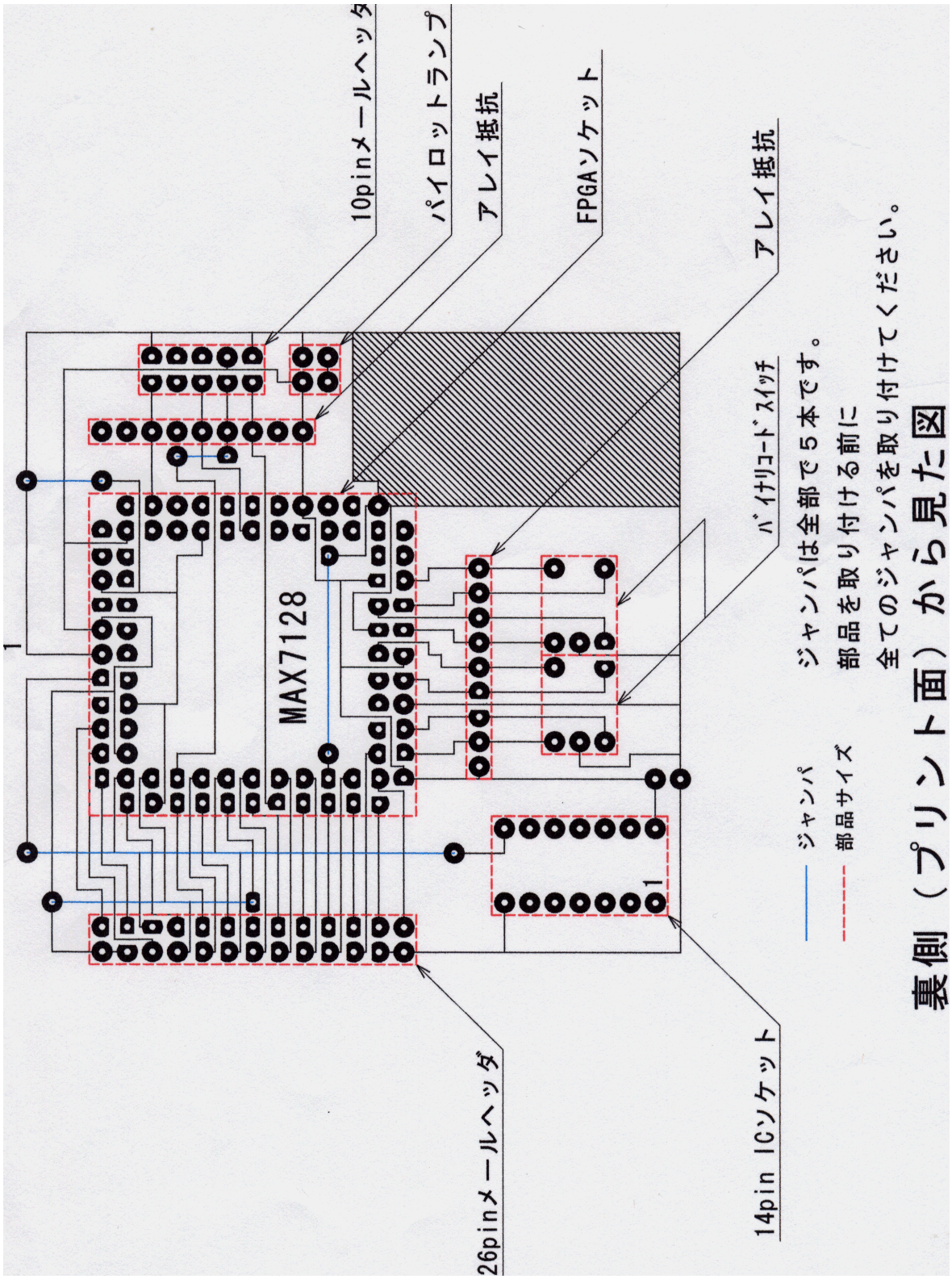
表 A3 実験条件

SW 入力	2 進数値	反転信号値	出力信号値	LED 点滅
L	r,e,d	r,e,d	nq,q,x	nq,q,x
0	000	111	011	100
1	001	110	100	011
2	010	101	100	011
3	011	100	100	011
4	100	011	101	010
5	101	010	100	011
6	110	001	100	011
7	111	000	100	011

3.周波数シンセサイザ

信号名	ピン番号
clk	83
co	57
hi0	40
hi1	41
hi2	39
hi3	37
lo0	49
lo1	51

信号名	ピン番号
lo2	45
lo3	44
on0	61
on1	63
on2	65
on3	67
on4	69
on5	73
q6	52


10pinメールヘッダ
パイロットランプ
アレイ抵抗
FPGAソケット
アレイ抵抗
ﾊﾞｲﾅﾘｺｰﾄﾞｽｲｯﾁ
MAX7128
1
1
ジャンパ
部品サイズ
ジャンパは全部で5本です。
部品を取り付ける前に
全てのジャンパを取り付けてください。
26pinメールヘッダ
14pin ICソケット
裏側（プリント面）から見た図