実験3 HDLによるハードウェア設計

［目的］

ハードウェア記述言語（HDL: Hardware Description Language）を使用したハードウェア（ディジタル論理回路）の設計手法を学習する。また、学習用 FPGA ボードと HDL シミュレータによる回路の動作検証を通して、実際にハードウェアがどのように動作するのか理解する。

［概要］

HDL の一つとして広く利用されている **Verilog-HDL** を使用して、ディジタル論理回路、すなわち組合せ回路や順序回路を記述する。記述した回路が正しく動作することを、学習用 FPGA ボードおよび Verilog-HDL シミュレータを用いて確認する。

HDL によるディジタル論理回路の設計は、一般的には図 1に示す設計フロー[1]で行われる。

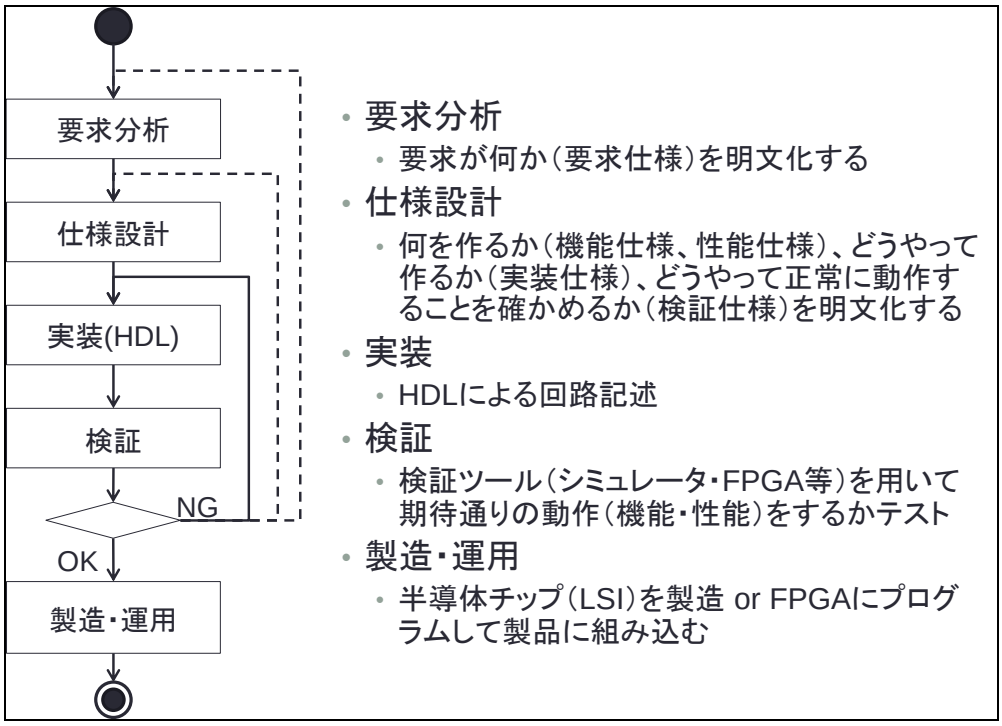


図 1　HDL によるディジタル論理回路設計フローの例（単純化したもの）

本実験では、仕様に基づいて HDL による記述を行い（実装）、期待通りの動作をすることをシミュレータ・FPGA[2]を用いてテストする（検証）という工程を、受講生が各自行う。一連の設計フローを経験し、それをレポートという形に明文化することを通じて、HDL によるハードウェア設計の実際を理解する。

---

[1] この設計フローは、単純化して図示したものであり、実際の設計においては、実装と検証は様々なレベルで繰り返し行われる。また点線のフローは生じないことが望ましい。

[2] FPGA（Field Programmable Gate Array）は、開発者が任意のディジタル論理回路を、ソフトウェアの様にプログラムして実現することが可能な、半導体 LSI チップである。

［使用する教材］

・実験指導書（この資料）

・副教材：キットで学ぶ FPGA チャレンジャー　Xilinx Spartan 3E 版

＜副教材内容＞

・テキスト冊子

・学習用 FPGA ボード　Basys2 ボード

　・FPGA：　Xilinx 社製 Spartan 3E XCS100E

　・メモリ：　SDRAM　16MByte、　EEPROM 16MByte

　・ユーザーI/O デバイス　：

　　LED8 個、プッシュスイッチ 4 個、スライドスイッチ 8 個

　　7 セグメント LED　4 ケタ、PS2 ポート、VGA ポート、I/O ピン 6pin(Pmod) x 4 個

［実験内容］

以下の 3 項目について実験を行い、HDL で記述し合成した回路が期待通りに動作することを、FPGA ボードとシミュレータを用いて確認する。

1. 組合せ回路を FPGA ボードで動かす
2. 順序回路を FPGA ボードで動かす・シミュレーションで何が起こっているか確認する
3. 合成した回路の性能を知る・課題回路を作成して動作を検証する

1. 組合せ回路を FPGA ボードで動かす

組合せ回路は、AND/OR/NOT から構成される論理関数に相当するディジタル論理回路である。すなわち、出力はその時の入力によってのみ決まる。入力の組合せで出力が決まるので、組合せ回路という。

本項目においては FPGA 上に組合せ回路を実現して期待通りに動作することを確認する。

手順

1-1. STEP01 を読み、学習用 FPGA ボードと FPGA について理解する。

1-2. STEP02 のうちライセンス導入(補足 1) と P.22 の Basys2 ボード動作チェックを行う。

1-3. STEP03 に従い、ISE のプロジェクトを作成する。プロジェクトの場所は、各自のマイドキュメント内とする。最も単純な Verilog 記述を作成する。(P.29 ON_Circuit.v)

1-4. STEP04 に従い、文法チェック・ピン配置決定・配置配線・コンパイル（コンフィギュレーションファイルの作成）・コンフィギュレーションファイル転送を行うことで、FPGA ボードに作成した Verilog 記述に相当する回路をプログラムし動作を確認する。

1-5. STEP05 に従い、**課題 05（AND 回路）**を作成し動作を確認する。

1-6. STEP06 に従い、**課題 06（OR 回路）**を作成し動作を確認する。

1-7. **レポート課題 1「組合せ回路の実装・テスト」**を行う。

　→　仕様（何を作ったか）、実装（どうやって作ったか）、およびテスト結果についてレポートに記載すること。

1-8. （オプション）STEP07 に従い、セレクタ回路を作成し動作を確認する。

2. 順序回路を FPGA ボードで動かす・シミュレーションで何が起こっているか確認する

順序回路は、メモリ要素（フリップフロップなど）を含むディジタル論理回路である。すなわち、出力はその時の入力と、以前の状態から決まる。入力の順序により出力が決まるので、順序回路という。本項目においては、FPGA 上に順序回路を実現して、期待通りに動作することを確認する。また、回路の動作は非常に高速（50MH z で動作）であるため、シミュレーションにより動作の詳細な様子を観察して理解する。

手順

2-1. STEP08 に従い、**課題 08-1（非同期リセット）**および**課題 08-2（同期リセット）**を実施する。フリップフロップ(DFF)の動作を FPGA ボードにおいて確認する。

2-2. STEP09 に従い、**課題 08-1** と**課題 08-2** のフリップフロップ(DFF)の動作をシミュレーション(ISim)で確認する。

→　非同期リセットと同期リセットの違いを理解する。（班内でお互いに説明せよ）

2-3. STEP10 に従い、**課題 10-1（カウンタ）**、**課題 10-2（50MHz クロック分周して 1Hz を作成）**および**課題 10-3（1 秒ごとにカウントアップして LED 表示）**を実施する。各回路の動作を確認する（FPGA および ISim）。

2-4. **レポート課題２「順序回路の実装・テスト」**を行う。

→　仕様（何を作ったか）、実装（どうやって作ったか）、およびテスト結果についてレポートに記載すること。

2-5. (オプション)STEP11 に従い、スイッチ入力を数えるカウンタ回路の動作を確認する。

3. 作成した回路の性能を知る・課題回路を作成して動作を検証する

本項目においては、これまで作成した回路の性能を調べる。また、課題回路を作成する。

FPGA は任意の回路をプログラムできる半導体 LSI チップであるが、内部にはメモリ要素である Flip-Flop と、論理関数を実現するための真理値表に相当する LUT(Look Up Table)が数多く存在しており、それらの間をプログラマブルな配線スイッチで接続することで任意の回路を実現する。数多くの配線スイッチを通過して信号が伝播する為、一般的に専用 LSI チップを作るよりも回路の動作速度は遅くなる。以上を踏まえて、FPGA 上に作成した回路の性能を調べる。

手順

3-1. 組合せ回路（レポート課題１など）および順序回路（レポート課題２、カウンタなど）の、フリップフロップ使用量、LUT 使用量、動作可能周波数(Fmax)を調べる。（レポート中に、性能に関する調査結果を記載すること。）

3-2. STEP12 を読み、**課題 12-1**：7 セグ LED デコーダ回路の動作を理解する。
（P.128 課題 12-2 の、7 セグ LED カウンタ回路はオプションとする）

3-3. 任意の 4 ケタ数字を 7 セグメント LED に表示する回路を、講義資料 WEB サイトよりダウンロードし、FPGA ボードにおいて動作を確認する。

3-4. **レポート課題３「すこし複雑なディジタルシステムの実装・テスト」**を行う。

［課 題］

ソースコードなどの参考資料は、講義資料 WEB サイト内の以下の URI を参照すること。

http://www.ced.is.utsunomiya-u.ac.jp/lecture/2013/jikken2/hdl/

レポート課題１：組合せ回路の実装・テスト

4 ビット入力､5 ビット(キャリー付)出力の**加算器**を Verilog-HDL で実装･テストする。

◆　機能仕様

入力：　FPGA ボード上のスライドスイッチ(SW)の、左 4 個を入力 A[3:0]、右 4 個を入力 B[3:0]とする。

出力：　A と B を加算した値を S (Sum の略)とし、FPGA ボード上の LED の右から 5 個に S[4:0]の値を出力する。

◆　実装仕様

・ファイル名は Adder4.v とする。

・加算器は Verilog-HDL の加算演算子を使って記述する。

◆　検証仕様

・FPGA ボードを用いて検証を行う。以下の入力値の組合せに対する出力値を記録し、期待値と一致するかを確認する。結果は OK か NG かで示す。

テスト番号	入力		出力期待値	出力値	結果
	A[3:0]	B[3:0]	S[4:0]	S[4:0]	OK/NG
1	0	0	0		
2	1	0	1		
3	0	1	1		
4	1	1	2		
5	2	2	4		
6	4	4	8		
7	8	8	16		
8	15	15	30		

・自分の学籍番号に基づくテスト入力値をスイッチに設定した際の、出力値(LED 状態)を、カメラで撮影してレポートに画像として貼る。撮影の際、加算器への入力値は、自分の学籍番号の下二桁目の数字を入力 A、下一桁目の数字を入力 B とし、LED の出力が期待値と同じになることを示す。

例）学籍番号：102985　入力 A=8, 入力 B=5　出力期待値　O=13

・フリップフロップ使用量、LUT 使用量、動作可能周波数(Fmax)を調べる事。

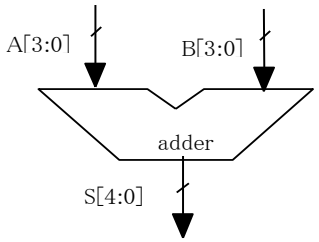


図 2　加算器のブロック図

レポート課題２：　順序回路の実装・テスト

自分の学籍番号の数字を、順番に 1 秒間に 1 桁ずつ LED に表示する順序回路を Verilog-HDL にて実装・テストする。

◆　機能仕様

入力：　プッシュボタン(BTN3 = start)を押すことで学籍番号の表示を開始する合図とする。リセットボタン(BTN0 = reset)を押すと初期状態に戻ることとする。

出力：　FPGA ボード上の 4 つの LED に自分の学籍番号を順番に 1 秒間に 1 桁ずつ、4 ビットの 2 進数として表示する。1 秒間の時間は、FPGA ボードの 50MHz クロック信号を 50M 回カウントすることで計測する。

◆　実装仕様

・ファイル名は NumberDisplay.v とする。

・以下の状態遷移図および状態遷移表に従った、ステートマシン[3]を作成する。

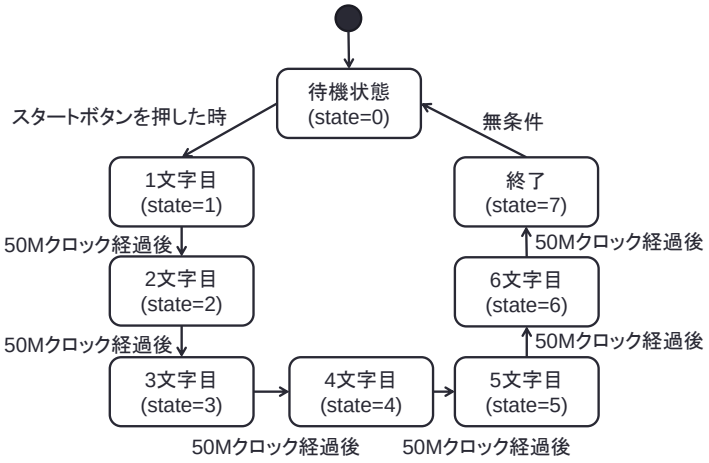


図 3　課題２の状態遷移図

表 1　課題２の状態遷移表（学籍番号：102985 の場合）

状態 (state)	入力		次状態 (state')	出力	備考
	BTN0	BTN3		LED[3:0]	
X	1	X	0	X	-
0	0	0	0	0	-
	0	1	1	0	-
1	0	X	2	1	50M クロック経過後
2	0	X	3	0	50M クロック経過後
3	0	X	4	2	50M クロック経過後
4	0	X	5	9	50M クロック経過後
5	0	X	6	8	50M クロック経過後
6	0	X	7	5	50M クロック経過後
7	0	X	0	0	-

表中の X は、任意の値を示す（Don't care という）。すなわち 1 行目は任意の状態において BTN0 の値が 1 だったら、BTN3 の値に関わらず状態 0 に遷移することを示す。

[3] カウンタを用いることで、任意の状態遷移を持つ順序回路を構成することができる。回路の制御部となる順序回路をステートマシンと呼ぶ。

◆　検証仕様

・シミュレータを用いて検証を行う。実時間（数秒間＝数百万クロック）のシミュレーション結果を確認するのは時間がかかるので、シミュレータを用いた検証の際は、2 クロックに 1 桁ずつ LED への出力信号を変化させることとする。

・以下の入力値の組合せに対する出力値を記録し、期待値と一致するかを確認する。結果は OK か NG かで示す。

クロック時刻	入力		出力期待値	出力値	結果
	BTN0 (reset)	BTN3 (start)	LED[3:0]	LED[3:0]	OK/NG
0	1	0	0		
1	0		0		
2		1	0		
3		0	1		
4			1		
5			0		
6			0		
7			2		
8			2		
9			9		
10			9		
11			8		
12			8		
13			5		
14			5		
15			0		
16			0		

・シミュレーション時に、入力と出力の値をテキストで出力しレポートに貼り付ける事（補足 4 を参照）。

・シミュレーション波形の画面キャプチャを、レポートに貼り付ける事。

・フリップフロップ使用量、LUT 使用量、動作可能周波数(Fmax)を調べる事。

<u>レポート課題３：　少し複雑なディジタルシステムの実装・テスト</u>

以下の 3 つの課題のうち 1 つ以上を選択して、機能仕様・実装仕様・検証仕様を作成し（すなわち設計を行い）、HDL による実装と FPGA ボードもしくはシミュレーションによる検証を行う。

課題 3-1　4 ビット算術論理演算装置(ALU)　　◆組合せ回路

課題 3-2　8 ビット簡易電卓（加算専用）　　◆順序回路

課題 3-3　8 ビット 2 進数トレーニングゲーム　　◆順序回路

検証結果として、以下の 3 つを含める事。

・シミュレーション時の入力と出力の値をテキストで出力

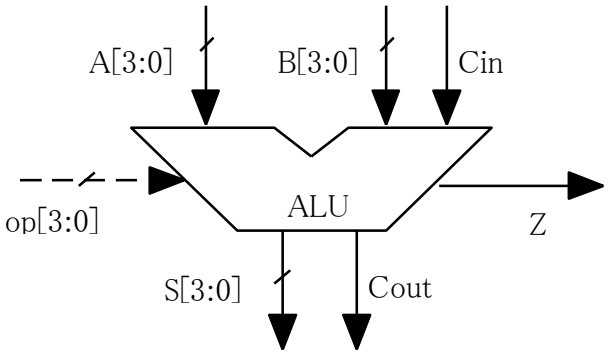
・シミュレーション波形の画面写真

・動作中の FPGA ボードの写真

課題 3-1 4 ビット算術論理演算装置(ALU)

図 4に示す 4 ビット**算術論理演算装置** (ALU) を設計・実装・テストする。この ALU は、4 ビットデータ A, B、及びキャリーCin を入力とし、op で指定される演算を行い、演算結果 S、キャリーCout を出力する。また、S がゼロの時、Z に 1 を出力する。

入力は、8 ビットのスライドスイッチで A[3:0], B[3:0]を設定し、4 ビットのプッシュスイッチで op[3:0]を設定する。出力は、S[3:0]を 7 セグメント LED に表示し、Cout と Z を LED に表示する。



op	演 算 内 容	Z	Cout
0	S <= ~A	*	
1	S <= A & B	*	
2	S <= A | B	*	
3	S <= A ^ B	*	
4	S <= A << B	*	
5	S <= A >> B	*	
8	S <= A + B	*	*
9	S <= A + B + Cin	*	*
a	S <= A - B	*	*
b	S <= A - B + Cin	*	*

演算結果は * 印の付いた出力に反映

図 4　4 ビット算術論理演算装置

課題 3-2　8 ビット簡易電卓（加算専用）

8 ビットの数字を加算可能な、簡易電卓を設計・実装・テストする。

計算結果は、16 進数の数字として 7 セグメント LED に表示する。ユーザは、値を設定するときはスライドスイッチ(8 ビット)を設定する。操作方法は以下の通りである。

・BTN1 を押すと値 1 として記憶し、7 セグメント LED に表示する。

・BTN2 を押すと値 2 として記憶し、7 セグメント LED に表示する。

・BTN3 を押すと、2 つの値の和を 7 セグメント LED に表示する。

課題 3-3　8 ビット 2 進数トレーニングゲーム

システムは 7 セグメント LED に 16 進数の数字 2 ケタ (4 + 4 = 8 ビット) を表示し、ユーザはその数字に相当する 2 進数(8 ビット)をスライドスイッチを用いて設定し、プッシュボタンを押す。それに対して、システムは 16 進数と 2 進数の数字が一致しているかどうかを判定し、結果を 7 セグメント LED もしくは LED に表示するゲームを設計・実装・テストする。

［補足資料］

補足 1. 開発環境の構築について

演習用 PC への Xilinx 社製開発環境(ISE)のインストールは完了しているが、各ユーザ(受講生)が以下の手順でライセンスファイルをインストールする必要がある。

A. Xilinx ライセンス設定マネージャーを起動する。（スタートメニュー→すべてのプログラム→Xilinx Design Tools→ISE Design Suite 14.2→Accessories→Manage Xilinx Licenses）

B. Xilinx ライセンス設定マネージャー（副教材 P.13 の②）の画面で、Manage Xilinx Licenses タブをクリックして選択する。

C. 副教材 P.17⑲の画面が表示されていることを確認する。

D. Copy License…のボタンをクリックして、ライセンスファイルを指定する。

ライセンスファイルの場所:「¥¥fs1.ced.is.utsunomiya-u.ac.jp¥vol1¥share¥学部授業関連¥2013 年度前期¥情報工学実験 II¥HDL¥Xilinx.lic」

補足 2. Xilinx ISE 設計ツールの起動方法

実験で用いる Xilinx ISE 設計ツールは、「スタートメニュー→すべてのプログラム→Xilinx Design Tools→ISE Design Suite 14.2→ISE Design Tools→Project Navigator」から起動する。

補足 3. Digilent Adept ツールの起動方法

FPGA 基板の動作テストや回路書き換えの際に用いる、Digilent Adept ツールは、「スタートメニュー→すべてのプログラム→Digilent→Adept→Adept」から起動する。

補足 4. Verilog-HDL シミュレーションにおいて信号値を出力する方法

レポート作成時、動作の様子を示すためには、画面写真をキャプチャして貼りつけるか、信号値の変化をテキストとして貼りつける必要がある。シミュレーションの際に、テストベンチの Verilog のモジュール内に**図 5**に示す記述を加えると、信号値の変化をテキストで出力することが出来る。（%b: 2 進数、 %d:10 進数、%o:8 進数、%h:16 進数）

```
initial
        $monitor($stime, "in0=%b in1=%b out=%b", in0, in1, out);
```

図 5　シミュレーションにおける信号値を出力するための記述例

補足 5. FPGA のコンパイル時に動作周波数の制約を記述する方法

FPGA のコンパイル（合成・配置配線）を行う際に、図 6の記述を UCF ファイル内にピン配置と併せて行う事で、動作周波数に関する制約をツールに知らせることが出来る。なお、クロック信号は CLK とすること。

コンパイルの結果、50MHz で動作する回路が合成できたかどうかを確認する必要がある。

```
NET CLK TNM_NET = USER_CLOCK;
TIMESPEC TS_USER_CLOCK = PERIOD USER_CLOCK 50 MHz;
```

図 6　動作周波数制約(50MHz) UCF ファイル記述の例

補足 6. FPGA のコンパイル結果・リソース使用量・性能を知る方法

FPGA のコンパイル（合成・配置配線）を行った後、ISE の Design Summary を見ることで、コンパイルの結果としてエラーや警告があるかどうか、リソース使用量や性能を知ることが出来る。

DisplayStudentNumber Project Status (03/27/2013 - 08:08:10)			
Project File:	Timing.xise	Parser Errors:	No Errors
Module Name:	Timing	Implementation State:	Programming File Generated
Target Device:	xc3s100e-5cp132	• Errors:	No Errors
Product Version:	ISE 14.2	• Warnings:	2 Warnings (0 new)
Design Goal:	Balanced	• Routing Results:	All Signals Completely Routed
Design Strategy:	Xilinx Default (unlocked)	• Timing Constraints:	All Constraints Met
Environment:	System Settings	• Final Timing Score:	0 (Timing Report)

エラーが有れば
ここに表示される

Device Utilization Summary				
Logic Utilization	Used	Available	Utilization	Note(s)
Number of Slice Flip Flops	27	1,920	1%	
Number of 4 input LUTs	41	1,920	2%	
Number of occupied Slices	33	960	3%	
Number of Slices containing only related logic	33	33	100%	
Number of Slices containing unrelated logic	0	33	0%	
Total Number of 4 input LUTs	64	1,920	3%	
Number used as logic	41			
Number used as a route-thru	23			
Number of bonded IOBs	7	83	8%	
Number of BUFGMUXs	1	24	4%	
Average Fanout of Non-Clock Nets	2.41			

リソース使用量は
ここに表示される
FlipFlopの数と
LUTの数を確認する

Performance Summary			
Final Timing Score:	0 (Setup: 0, Hold: 0, Component Switching Limit: 0)	Pinout Data:	Pinout Report
Routing Results:	All Signals Completely Routed	Clock Data:	Clock Report
Timing Constraints:	All Constraints Met		

性能に関する概要は
ここに表示される
詳細な性能を知るには
Timing Constraints
のリンクをクリックする

図 7　ISE の Design Summary の読み方

補足 7. FPGA の最大動作周波数を知る方法

Timing Constraints のリンクをクリックすることで、図 8のタイミング制約結果レポートが表示される。赤枠で囲んだ部分に、フリップフロップからフリップフロップの間の遅延時間の最大値がここに表示される。この値の逆数が、最大動作周波数である。

	Constraint	Check	Worst Case Slack	Best Case Achievable	Timing Errors	Timing Score
1	Yes TS_USER_CLOCK = PERIOD TIMEGRP "USER_CLOCK" 50 MHz …	SET…	14.078ns	5.922ns	0 0	0 0

図 8　タイミング制約結果レポートの例

［参考文献］

［1］小林優，“入門 Verilog-HDL 記述”，CQ 出版.

［2］小林優，“初めてでも使える HDL 文法概要① Verilog-HDL 編”，デザインウェーブマガジン，No.13, pp.150－159.

［3］小林優，“初めてでも使える Verilog HDL 文法ガイド ―― 記述スタイル編”，http://www.kumikomi.net/archives/2009/07/verilog_hdl.php

［4］内田智久，“Verilog-HDL 入門”，http://research.kek.jp/people/uchida/educations/verilogHDL/

［改訂履歴］

日付	氏名	修正内容
2012 年度まで	-	実験内容：シミュレータによる論理回路設計
2013/3/27	大川 猛	教育用 FPGA ボード（Basys2）およびシミュレータを用いた実験内容への変更
2013/4/3	大川 猛	語句の間違いなどの微修正