テストベンチとは？

- テストベンチはハードウェアの実体ではなく、シミュレーションのための記述
- ハードウェアモジュールをテストするには絶対に必要
- しかし、記述が分かりにくく入門者の障壁になっている
- 大体パターンは同じなので、記述の詳細にこだわらないで使うのが良い

テストベンチの典型的構造

タイムスケール文

module宣言

　　配線、レジスタ宣言

　　クロック発生　←　組み合わせ回路では不要

　　テスト対象モジュールの生成

　　initial文で入力設定

endmodule

タイムスケール文

- 一番上に書く
- これがないとシミュレーションができない場合もある

`timescale X/Y

X：Verilog記述上の単位時間

　ここを1nsにしておくと、コード中で10と書くと10nsになる

Y:シミュレーションの刻み時間

例）`timescale 1ns/1ps

加算モジュール用のテストベンチ例

```
module test;
  parameter STEP=10;
  reg ina, inb;
  wire outs;
  adder adder_1(.a(ina), .b(inb), .s(outs));
  initial begin
    $dumpfile(“adder.vcd”);
    $dumpvars(0,adder_1);
    ina <= 1’b0;
    inb <= 1’b0;
  #STEP
    $display(“a:%b b:%b s:%b”, ina,inb,outs);
    ina <= 1’b0;
    inb <= 1’b1;
 #STEP
…
```

配線、レジスタ、モジュール実体生成

parameter文はdefine文と似ているがより柔軟

```
parameter STEP=10;
  reg ina, inb;
  wire outs;
  adder adder_1(.a(ina), .b(inb), .s(outs));
```

reg文での宣言では値を記憶できる：モジュールの入力に使う

wire文は信号に名前を付けるだけ：モジュールの出力に使う

インスタンス名

別ファイルで宣言したモジュール名

入出力への接続
ピリオド以下はローカルな名前を使う

モジュール実体生成

モジュール名　実体名　(

.ローカル信号名（信号名）, .ローカル信号名（信号名）, ….);

- モジュール名は、通常他のファイルに定義されているモジュールの名前（ここではadder)
- 実体名は、このテストベンチで使う名前で、適当につける（ここではadder_1）
- ローカル信号名は、adderの中で定義した入出力信号名、前にピリオドを付ける
- 信号名はこのローカル信号に接続する信号を示す。()の中に入れる。

シミュレーションの制御

```
initial begin
    $dumpfile(“adder.vcd”);
    $dumpvars(0,adder_1);
    ina <= 1’b0;
    inb <= 1’b0;
  #STEP
    $display(“a:%b b:%b s:%b”, ina,inb,outs);
    ina <= 1’b0;
    inb <= 1’b1;
 #STEP
     $display(“a:%b b:%b s:%b”, ina,inb,outs);
    ….
     $finish
end
```

initial文はシミュレーションを一回実行

波形ファイルを指定

記録する範囲を指定

reg文に値をブロッキング代入<=

10nsec 時間消費

値の表示、プリント文と似ている
%bで2進数表示
リターンは自動的に入る

10nsec 時間消費

シミュレーションの終了

波形の保存

$dumpfile(“adder.vcd”);

$dumpvars(0,adder_1);

$が付くとシミュレーション実行上のタスクを実行する特殊な文になる

$dumpfileはvcd形式という形式で波形を記録するファイル名を指定　gtkwaveはこの形式を読む。

vcd形式は良く使われるテキストによる波形記録形式で、便利だがサイズが大きい欠点がある

dumpvars(レベル,モジュール実体名）

ここで書いたモジュールのインスタンスの信号名を記録する。ここではadder_1

値の設定と時間の消費

ina <= 1’b0; ina,inbはreg宣言されているので値を記憶できる
inb <= 1’b0; ここでは、ina=0,inb=0になる
#STEP ここで10ns時間消費
inb <= 1’b1; ina=0,inb=1になるina=0は覚えている
#STEP
ina <= 1’b1;
inb <= 1’b0; ina=1, inb=0
#STEP
inb <= 1’b1; ina=1,inb=1
#STEP

これで全ての入力の組み合わせを試す

display文

シミュレーション実行時の信号名を表示

$display(“a:%b b:%b s:%b”,

ina,inb,outs);

C言語のprintfに似ていて“”内にフォーマット、その後に表示信号名を指定する

%b: 2進数　%x:16進数　%d:10進数を利用

¥nを入れなくても自動的に改行される

というか、改行しないことができない

$finish文

- シミュレーションを終了させる
- これがなくても、最後の時間まで来ると終わるのだが、ちゃんと付けておこう