Verilog撰寫簡介

Verilog module

module module_name (port_name);

Port宣告

Data Type宣告

電路描述

endmodule

Module名稱

- 開頭不可為數字或符號
- module_name不可與Reserved Keywords完全相同
  - Ex. Module, reg, input ……

Port 宣告

- Port種類
  - Input
  - Output
  - Inout(極少用)
- 宣告時若超過1bit則需加[x:0](x+1bits)
- Port_name有出現過的全需宣告
- Ex.

  input cin; //1bit 的input 名字cin

  output [3:0] cout; //4bits的output 名字cout

Data Type宣告

- Data type種類
  - Wire
  - Reg
- 同Port宣告時若超過1bit則需加[x:0](x+1bits)
- 在Data type宣告的Reg跟Wire不一定是Port
- 一個Data只能宣告一種Data type
- Ex.

```
wire  a;
reg [3:0] b;
```

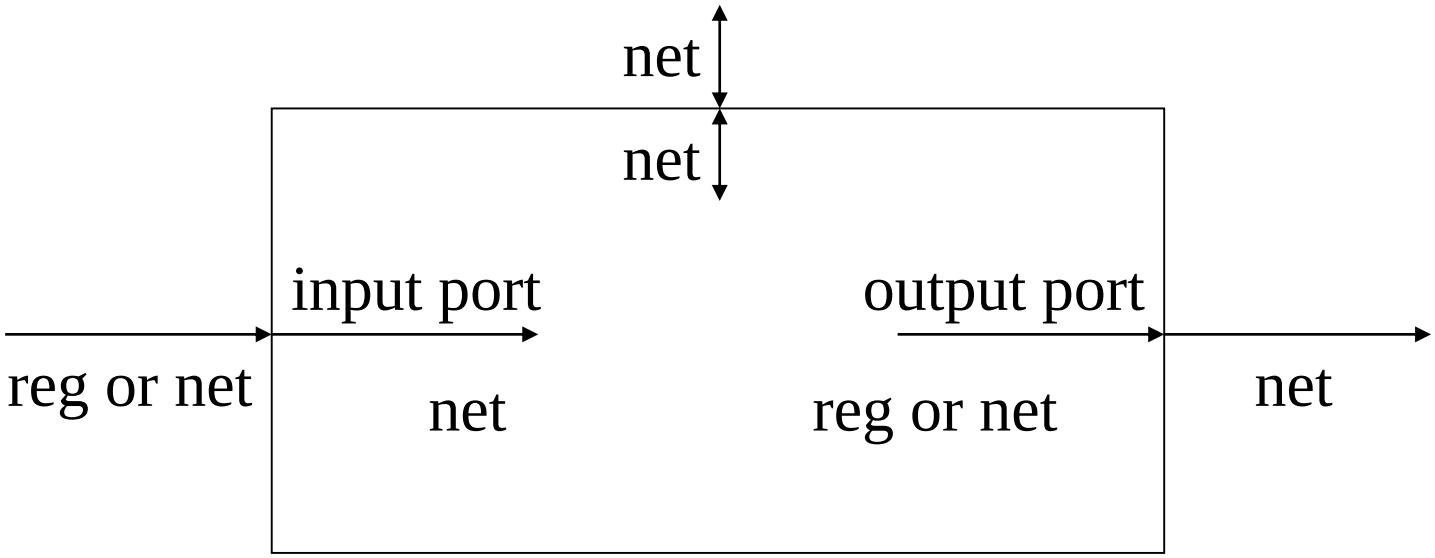
在使用Data type 要注意下列幾點：

(1)一個 input port 可以用 net 或 reg 來驅動，而進到電路內部這個 input port 只能去驅動另一個 net，不可去驅動 register。

(2)在電路內部一個 output port 可以用 net 或 register 來驅動，而這個 output port 只能去驅動另一個 net，不可去驅動 register。

(3)一個 inout port只能由 net 驅動，也只能去驅動 net。

示意圖



電路描述

- 分為Behavior和Structure(下兩頁為完整的範例)
  - Structure為直觀硬體描述
    - Ex. xor U0（sum, a, b, cin）……
  - Behavior為行為描述
    - Ex. If, else, always@(xxx) ……
- 一個module可共存兩種方法

◆ Structure Description

```
module fadder (sum, cout, a, b, cin);
   //---------- port declaration
  output sum, cout；
   input a, b, cin；
  // ---------- gate connection
   xor U0（sum, a, b, cin）;
   and U1（net1, a, b）；
   and U2（net2, a, cin）；
   and U3（net3, b, cin）；
   or U4（cout, net1, net2, net3）；
endmodule
```

◆ Behavior Description

```
module fadder (sum, cout, a, b, cin);
// -------- port declaration
   output sum, cout；
   input a, b, cin；
// -------- data type delcaration
   reg sum, cout；
// -------- behavior description
      always @（a or b or cin）
          begin
            sum = a^b^cin；
            cout = (a&b)|(b&cin)|(cin&a)；
          end
endmodule
```

可用的運算元

Arithmetic Operators	+, -, *, /, %
Relational Operators	<, <=, >, >=
Equality Operators	==, !=, ===, !==
Logical Operators	!, &&, ||
Bitwise and Unary Reduction Operators	~, &, |, ^, ~^
Shift Operators	>>, <<
Conditional Operators	?:
Concatenations	{ }
Replication	{{ }}

▲ Number

整數（integer），實數（real number）。

<size> '<base> <value>

<size>：指定integer的size有多大，以bit為單位。

<base>：指定integer的基底，可以是b（binary）二進位，o（octal）八進位，d（decimal）十進位，或h（hexadecimal）十六進位。

<value>:指定integer的值。

Ex. 3’b001

4’d7

16’hffff

簡單程式範例

```
module example1(clk,cin1,cin2,cout1,cout2);
input clk;
input [1:0]cin1,cin2;
output cout1;
output [3:0] cout2;
reg [3:0]cout2;
wire cc;

assign cc =(clk==1'b0)?cin1[0]:cin2[0];
assign cout1 =cc;
always@(posedge clk)
begin
    cout2<={cin1,cin2};
end

endmodule
```