Verilog 硬體描述語言

Verilog HDL

A Guide to Digital Design and Synthesis

IEEE 1364-2001 Compliant

第4章 模組與輸出入埠

4.1 模　組

- Verilog模組定義的開頭一定是關鍵字module，接下來是模組名稱、輸出入埠列表、輸出入埠的型態宣告，接著可能是選擇性使用的參數列表(parameter)。
- 只有當模組需要與外界連接時，才有埠列表與型態宣告。

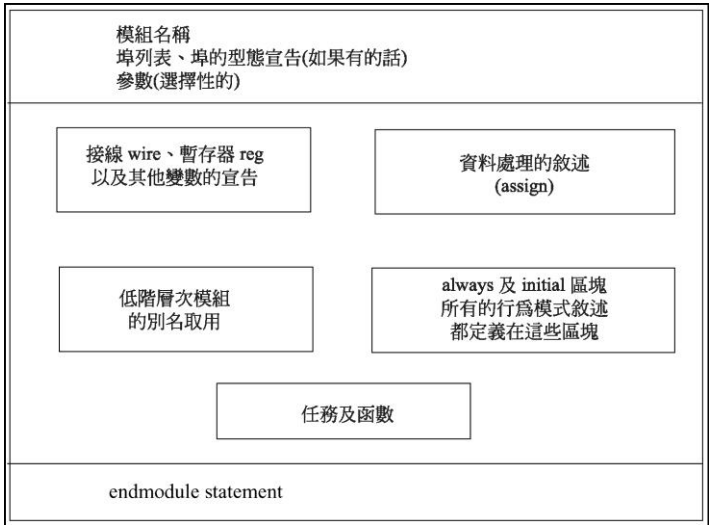

圖4-1　Verilog 模組的組成元件

4.1 模　組

- 圖4-1的其他五個部份沒有一定的編排順序，可以出現在一個模組的任何地方，模組定義最後是以關鍵字endmodule來表示模組結束。
- 在Verilog的檔案中可以擁有許多的模組，每個模組在檔案中，模組宣告的順序可以是任意的，不受限於引用層次的高低。
- 範例: SR閂(SR latch)
  - SR閂有S與R兩個輸入埠，和Q與Qbar兩個輸出埠

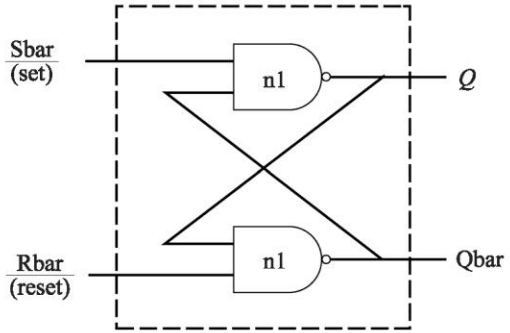

圖4-2　SR閂

4.1 模　組

- **範例4-1 SR閂的各個部分**

```
// 這個範例主要是在說明一個模組中的不同元件
// 模組名稱與輸出入埠的列表
// 模組名稱為 SR_latch
module SR_latch(Q,Qbar,Sbar,Rbar) ;

// 埠的宣告
output      Q,Qbar;
input       Sbar,Rbar;

// 取用低階層次模組的別名
// 在本例中引用的是 Verilog 提供的主要元件反及閘
// 注意訊號線連接的方式
nand n1(Q,Sbar,Qbar) ;
nand n2(Qbar,Rbar,Q) ;

// 關鍵字 endmodule
endmodule
```

SR_latch 模擬方塊

```
// 模組名稱與輸出入埠的列表
// 觸發模組
module Top;
// 宣告 wire reg 與其他變數
wire   q,qbar;
reg    set,reset;

// 取用低階層次模組的別名
// 在本例中引用的是 SR_latch
// 將 set 與 reset 的反向訊號輸入到 SR_latch 的別名中
SR_latch m1(q,qbar,et,eset) ;

// 行為處理模式 initial
initial
begin
          $monitor($time,"set = %b,reset = %b,q = %b\n",
                    set,reset,q) ;
          set = 0 ; reset = 0 ;
          #5 reset = 1 ;
          #5 reset = 0 ;
          #5 set = 1 ;
end
// 關鍵字 endmodule
endmodule
```

4.2 輸出入埠(Ports)

- 輸出入埠提供一個模組與外界溝通的介面，外界僅能由輸出入埠來與模組傳遞訊號。
- **4.2.1 輸出入埠的列表**
  - 一個模組通常經由一系列的輸出入埠來與外界溝通，若模組不需要與外界溝通，自然就沒有輸出入埠。
  - **範例4-2 輸出入埠的列表**

```
module fulladd4(sum,c_out, a,b,c_in) ; // 有輸出入埠列表
                                       // 的模組
odule   Top; // 沒有輸出入埠的列表的模組，通常用在模擬區塊。
```

  - 4-bit加法器有 3 個輸入與 2 個輸出埠
  - Top模組是用來模擬，不需要與外界溝通

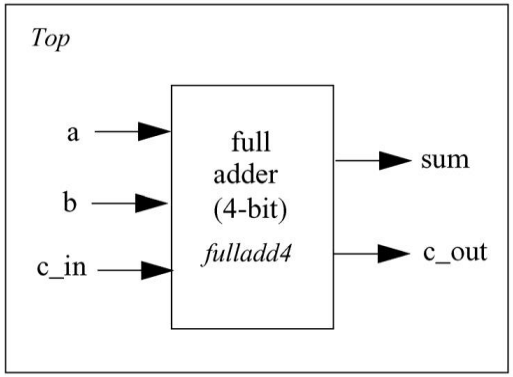

圖4-3　Top與全加器模組別名的輸出與輸入埠

4.2 輸出入埠(Ports)

- **4.2.2 埠的宣告**
  - 輸出入埠列表中的埠都必須要在模組中宣告，埠的宣告有以下幾種類別

Verilog 關鍵字	埠的類別(依方向來分別)
input	輸入埠
ouput	輸出埠
inout	雙向輸出入埠

- **範例4-3 埠的宣告**

```
module   fulladd4(sum,c_out,a,b,c_in) ;
// 開始宣告埠
output [ 3: 0]     sum ;
output c_cout ;

input   [3:0]   a,   b;
input   c_in ;
// 結束埠的宣告
...
<module internals>
...
endmodule
```

4.2 輸出入埠(Ports)

- **4.2.2 埠的宣告**
  - 在Verilog中內定的輸出入埠的宣告種類為**wire**，若在埠的宣告中只有宣告output、input 或是 inout，則皆將其資料型態視為接線型態(wire)。
  - 假如需要將訊號的值儲存起來，就要將輸出入埠的種類宣告為**reg**，
  - 注意input與inout型態的埠不能被宣告為reg，因為reg主要在儲存訊號，但input只是代表外來訊號的情況。
  - **範例4-4 DFF埠的宣告**

```
module   DFF(q,d,clk,reset) ;
output   q ;
reg   q ;   // 輸出埠 q 需要儲存資料，所以宣告成 reg 型態的變數。
input   d,   clk,   reset ;
...
...
endmodule
```

4.2 輸出入埠(Ports)

- **4.2.2 埠的宣告**
  - 以下範例說明如何利用程式語言ANSI C的習慣來宣告一個模組。
  - 範例4-5宣告將輸出埠的定義與reg/wire的定義，直接結合在一起了。
  - 沒有特別指定的型態狀況下，系統都會設定為wire型態。
  - **範例4-5 ANSI C的宣告使用習慣**

```
module fulladd4(output reg [3:0] sum,
                output reg c_out,
                input [3:0] a, b, //寫入一個預設值
                input c_in); //預設值線路
...
<module internals>
...
endmodule
```

4.2 輸出入埠(Ports)

- **4.2.3 輸出入埠的連接規定**
  - 我們可以將一個輸出入埠視為在模組內外相互連接的兩部份。
  - 在Verilog中輸出入埠的內部與外部連接，必須要遵守某些規定，若是違反規定，Verilog模擬器將會發出錯誤的訊息。

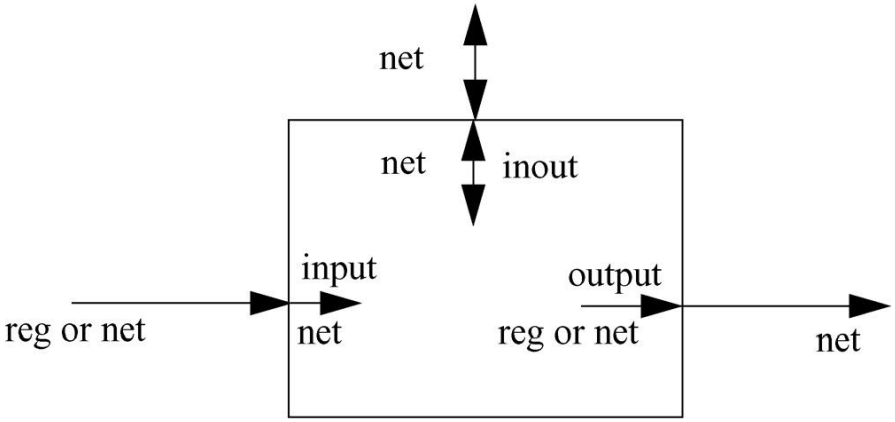

圖4-4　輸出入埠的相連規定

4.2 輸出入埠(Ports)

- **4.2.3 輸出入埠的連接規定**

輸入

- 在模組的內部，輸入埠永遠只是一個接線(net)，由外部來看，輸入埠的訊號可以接到暫存器(reg)，或是一個接線(net)的訊號。

輸出

- 在模組的內部輸出訊號，可以宣告為暫存器或是接線(net)的型態。從外部來看，必須接到一個接線(net)，不可以連接到暫存器型態的訊號。

雙向

- 雙向埠不管是在模組內或外部，都必須連接到接線的型態。

輸出入埠的寬度

- 在Verilog中，允許輸出入埠的內外連接寬度不同的訊號，但模擬器應該發出警告的訊號。

4.2 輸出入埠(Ports)

- **4.2.3 輸出入埠的連接規定**

浮接的輸出入埠

- Verilog允許輸出入埠可以浮接，例如使用來除錯的時候，若不需要接到任何訊號，就可以將它浮接。

```
fulladd4 fa0(SUM,  , A, B, C_IN);    // 輸出埠 c_out 浮接
```

非法的輸出入埠連接範例

- 範例4-6 非法的輸出入埠連接

```
module   Top;

// 宣告連接用的變數
reg   [3:0] A,B ;
reg   C_IN ;
reg   [3:0] SUM ;
wire  C_OUT ;

// 引用模組 fulladd4 並取別名為 fa0
fulladd4 fa0(SUM,C_OUT,A,B,C_IN);

// 這是一個非法的連結，因為模組 fulladd4 的輸出埠連接到一個
// 型態為 reg 的 SUM 變數上。
  .
  .
  < stimulus >
  .
  .
endmodule
```

4.2 輸出入埠(Ports)

- **4.2.4 輸出入埠與外部訊號連接的方法**
  - 將一個模組的輸出入埠與外部訊號連接的方法有兩種，這兩種方法需要分開使用，而且不可以在同一個引用模組的別名中使用。

依照定義模組時輸出入埠的列表順序來連接

- 範例4-7 依照輸出入埠的列表順序連接

```
module   Top;
// 宣告用來連接的變數
reg  [3:0] A,B;
reg  C_IN;
wire [3:0] SUM;
wire C_OUT;

   // 引用模組 fulladd4 並取別名為 fa_ordered/
   // 訊號依照輸出入埠的列表宣告的順序連接
   fulladd4 fa_ordered(SUM,C_OUT,A,B,C_IN);
   ...
   < stimulus >
   ...
endmodule
```

```
module fulladd4(sum, c_out, a, b, c_in);
output [3:0] sum;
output c_cout;
input [3:0] a, b;
input c_in;
   ...
   < module internals >
   ...
endmodule
```

4.2 輸出入埠(Ports)

- **4.2.4 輸出入埠與外部訊號連接的方法**

用指定名稱的方法(Connecting ports by name)

- 對一個大的設計電路而言，一個模組有超過50個輸出入埠是常見的。此時若要詳記輸出入埠列表的順序，就容易出錯。
- Verilog提供了一個可藉由指定輸出入埠的名稱，將外界訊號連接到輸出入埠的的方法。

```
// 訊號依照指定埠的名稱之方連接
fulladd4 fa_byname(.c_out(C_OUT), .sum(SUM), .b(B), .c_in(C_IN), .a(A), );
```

- **注意：** 只有需要與外部連接的輸出入埠，才被指定名稱連接，其餘不需要連接的埠名稱，就可以不用寫出。

```
// 訊號依照指定埠的名稱之方連接
fulladd4 fa_byname(.sum(SUM), .b(B), .c_in(C_IN), .a(A), );
```

4.3 階層化的取名方法

- Verilog支持階層化的設計方法，對於每個模組的別名、變數、訊號都賦予一個名字來做區別。
- 階層化的名稱結構是由一連串所屬階層的名字，加上本身的名字所組成，名字之間用點(“.”)分開。
- 首先我們將最上層沒有被其他模組引用的模組稱為root模組，從此模組當起始點，依照整個架構的樹狀圖來搜尋。

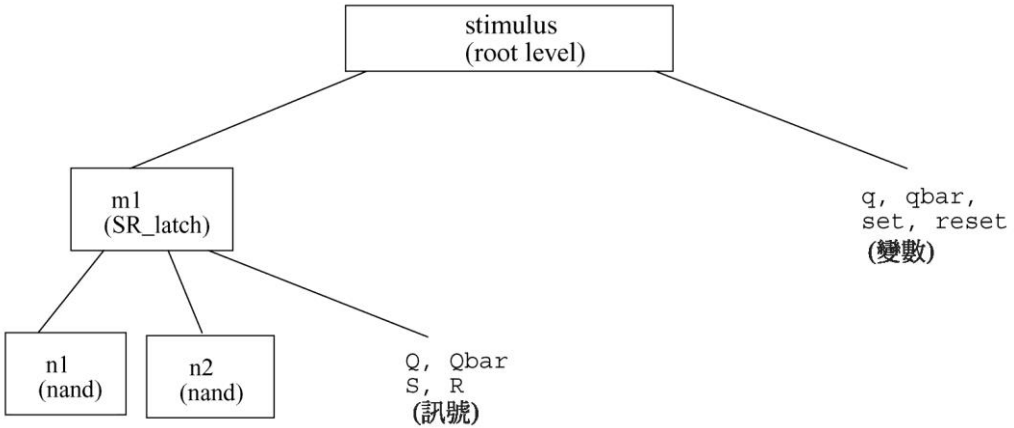

圖4-5　SR閂模擬模組的階層化架構圖

4.3 階層化的取名方法

- 在此例中的觸發(stimulus)模組，因為沒被其他模組所引用，所以稱為root模組，在此模組中的識別物件有q、qbar、set和reset。
- 在root模組中有一個SR_latch的模組別名稱做m1，m1中包含有兩個nand閘n1與n2，以及Q、Qbar、S與R的訊號名稱，由此搜尋下來並在每個名稱中以點(“.”)分開就可得如範例4-8的結果。
- 要顯示階層的階數，只需要在 $display 中加入 %m 即可(表3-4)。
- **範例4-8 階層化架構名稱**

```
stimulus                    stimulus.q
stimulus.qbar               stimulus.set
stimulus.reset              stimulus.m1
stimulus.m1.Q               stimulus.m1.Qbar
stimulus.m1.S               stimulus.m1.R
stimulus.n1                 stimulus.n2
```