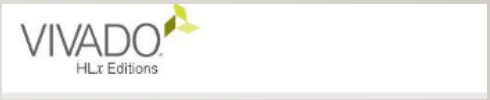
數位邏輯設計 I

VERILOG-1

2021.05.07

什麼是VERILOG?



- Verilog是一種硬體描述語言
- 透過**寫程式**的方式來描述硬體的行為
- 讓EDA tool(Electronic Design Automation)來幫你完成**電路設計**
  - 過程包含把你的描述的程式轉成gate-level的電路
  - 各個大小module的擺放
  - 繞線
  - 最後的驗證

VERILOG 基本宣告

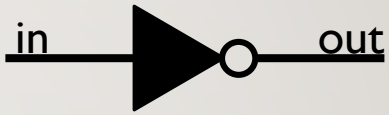
- Module :起始宣告的關鍵字，結尾有Endmodule
- Input
- Output
- 宣告訊號線
  - Wire
  - Reg
- Assign:要求指定的訊號線做運算，指定的對象必須宣告為wire的形式
- Always:宣告的形式必須為reg的形式

模組基本架構

- module 模組名稱(輸入輸出埠清單);
  - 輸入輸出埠的宣告 // input, output, inout
  - 變數資料型態宣告 // wire, reg, …
  - 引用較低階的模組
  - 邏輯閘階層之描述
  - 資料流階層之描述
- endmodule

模組基本架構(範例)

- module not_gate(in, out);
  - input in;
  - output out;
  - assign out =~in;
  - initial begin
    …
  - end
  - always @(posedge in) begin
    …
  - end
- endmodule



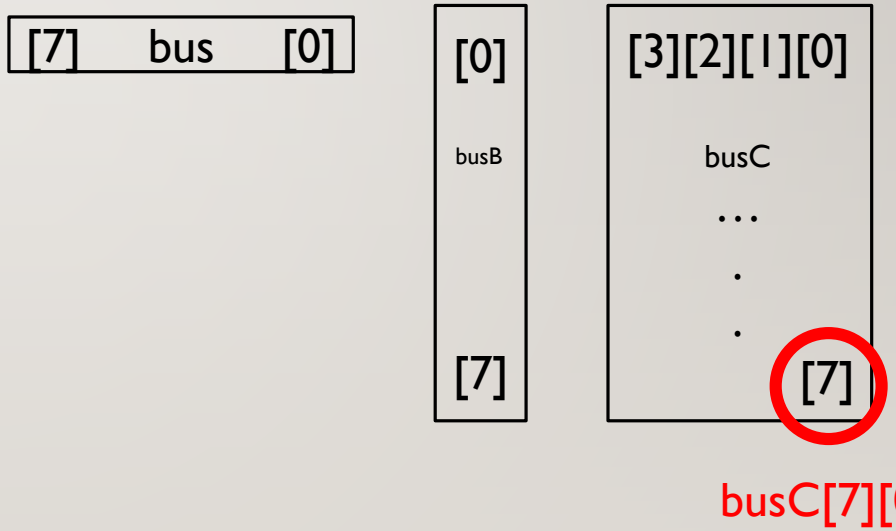
輸入	輸出
in	out
0	1
1	0

四種數值準位

值	意義	說明
0	低電位	布林代數中的假值
1	高電位	布林代數中的真值
Z	高阻抗	高阻抗斷線
X	未定值	像是線路未初始化之前，以及有0,1 兩者衝突的線路值，或者是輸入為 Z 的輸出值

向量與陣列

- 接線與暫存器皆可以訂定義為向量，如無定義的長度，就會以一個位元(純量)計算。
- Example
  - wire [7:0] bus;
  - reg clock;
  - reg [0:8] busA;
- 陣列
- Example
  - reg busB[0:7];
  - reg [3:0] busC[0:7];



語法協定-數字規格

- <size>'<base><number>
- <size>:bits大小
- <base>:
  - 十進位: d or D
  - 二進位: b or B
  - 八進位: o or O
  - 十六進位: h or H
- Example
  - 12'b0100_1100_0000;(底線可增加可讀性)
  - 8'HAF;
  - 16'd256;
- 負數，負號放在<size>
- 若沒有設定，Default是32位元的Decimal

語法協定-註解

- // 單行註解
- /* 多行註解*/

運算子

運算子種類	符號	運算功能	所需的運算元數目
算術運算符號	*	乘法	2
	/	除法	2
	+	加法	2
	–	減法	2
	%	取餘數	2
	**	指數	2
邏輯運算符號	!	邏輯上的 “NOT”	1
	&&	邏輯上的 “AND”	2
	||	邏輯上的 “OR”	2
比較運算符號	>	大於	2
	<	小於	2
	>=	大於或等於	2
	<=	小於或等於	2
相等運算符號	==	等於	2
	!=	不等於	2
	===	事件上的等於	2
	!==	事件上的不等於	2
位元運算符號	~	逐位元反相(取 1 的補數)	1
	&	對相對位元作 "AND"	2
	|	對相對位元作 "OR"	2
	^	對相對位元作 "XOR"	2
	^~或~^	對相對位元作 "XNOR"	2

運算子

運算子種類	符號	運算功能	所需的運算元數目
化簡的位元運算符號	&	化簡的 "AND"	1
	~&	化簡的 "NAND"	1
	|	化簡的 "OR"	1
	~|	化簡的 "NOR"	1
	^	化簡的 "XOR"	1
	^~或~^	化簡的 "XNOR"	1
移位運算符號	>>	向右移位	2
	<<	向左移位	2
	>>>	向右算術移位	2
	<<<	向左算術移位	2
連結運算符號	{}	連結	任意數目皆可
	{{}}	複製	任意數目皆可
條件運算符號	?:	做條件運算	3

運算子

運算式	說明	可能傳回值
a==b	a跟b相等，若是a、b中有x或是z的值的話則傳回x。	0，1，x
a!=b	a跟b不相等，若是a、b中有x或是z的值的話則傳回x。	0，1，x
a===b	a跟b相等，包括x和z的比較。	0，1
a!==b	a跟b不相等，包括x和z的比較。	0，1

運算子

種類	運算子	優先順序
一元運算 乘法、除法、取餘數	+ − ! ~ * / %	最高
加法、減法 移位	+ − << >>	
比較運算 相等運算	< <= > >= == != === !==	
化簡運算	&，~& ^ ^~ |, ~|	
邏輯運算	&& ||	
條件運算	?:	最低

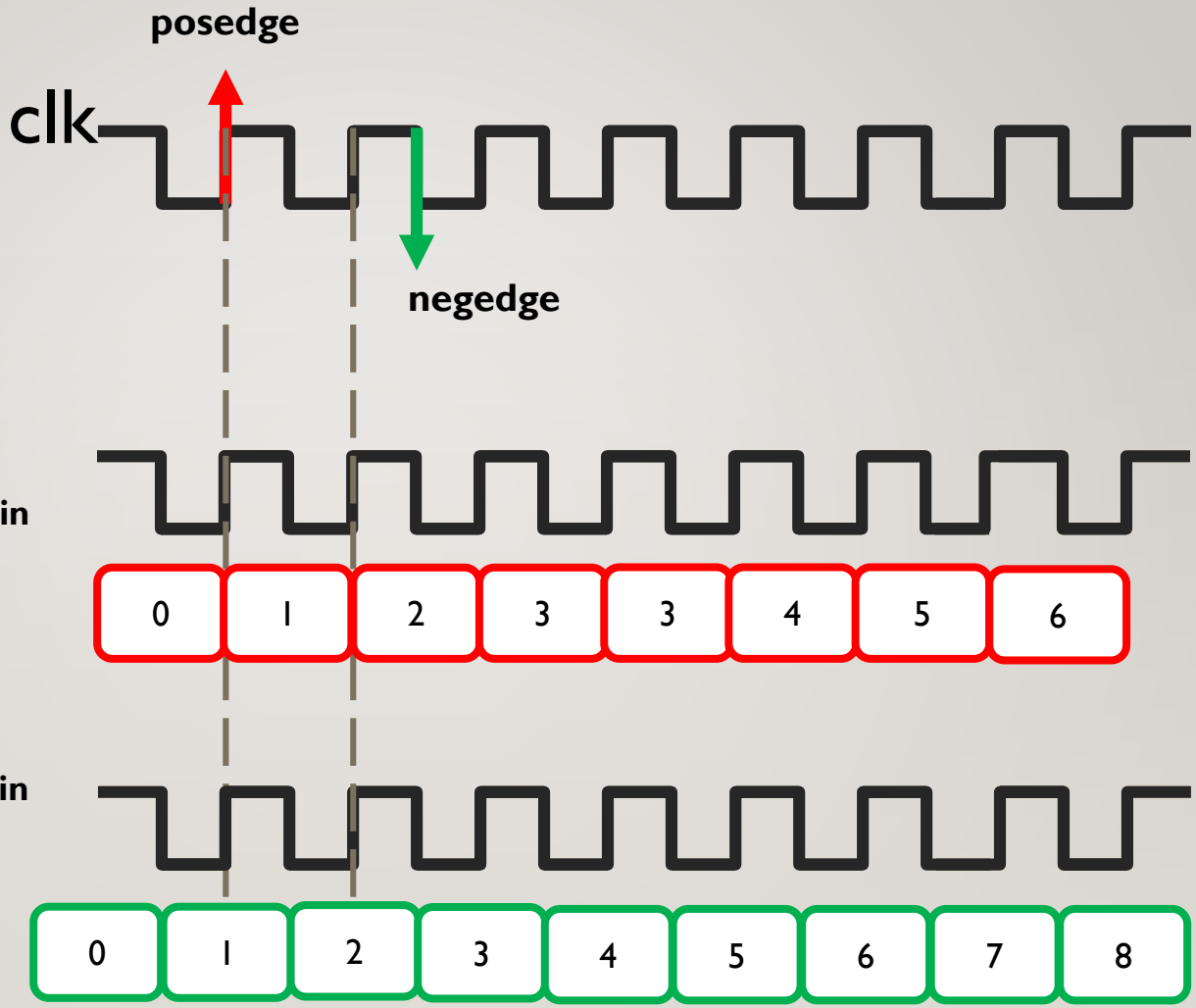
阻塞賦值(=)與非阻塞賦值(<=)

```
always@(a ,b)
begin
 b = a;
 c = b;
end
```

```
always@(posedge clk)
begin
  b <= a;
  c <= b;
end
```

	a	b	c
初始值	1	2	3
阻塞賦值	1	1	1
非阻塞賦值	1	1	2

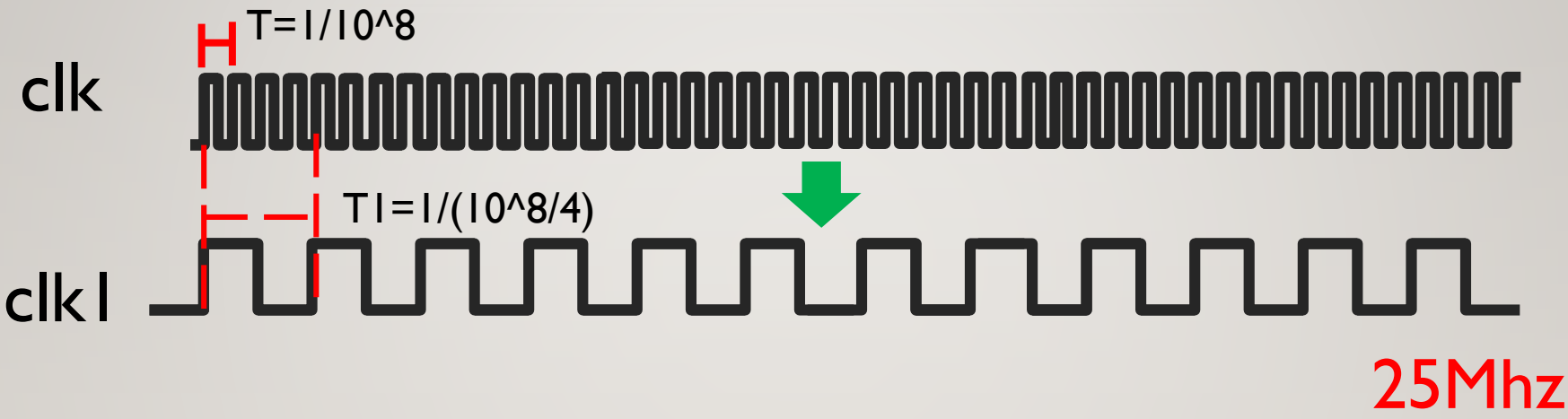
```
Input clk,
Output reg [7:0]ck;

always @ (posedge clk)begin
ck=ck+1;
end

always @ (negedge clk)begin
ck=ck+1;
end
```



除頻

透過一個頻率較快的clk去產生一個頻率慢的clk

100Mhz



25Mhz

Hint:利用counter的概念

Hint:透過

1. always @
2. if..... else

QUESTION

- If clk=40Mhz, how to generate clk2=10 hz?

- If clk=100Mhz, how to generate clk2=50M hz?

ANSWER

- If clk=40Mhz, how to generate clk2=10 hz? (40,000,000/10=4,000,000)(4,000,000/2=2,000,000)
  - module Q1(clk,clk2);

```
input clk;
output reg clk2;
reg [20:0] cnt;
Initial begin
        clk2 = 0;
        cnt = 0;
end
always @(posedge clk) begin
    if (cnt ==1,999,999) begin
            cnt <= 0;
             clk2 = ~clk2;
      end
      else cnt <=cnt+1;
   end
endmodule
```

ANSWER

- If clk=100Mhz, how to generate clk2=50M hz? (50M*2=100M)
  - module Q2(clk,clk2);

```
input clk;
output clk2;
initial begin
        clk2=0;
end
always @(posedge clk) begin
        clk2 <= ~clk2;
end
endmodule
```

參考書:

Samir Palnitkar, Verilog 硬體描述語言

ISBN:978-986-15-4104-4