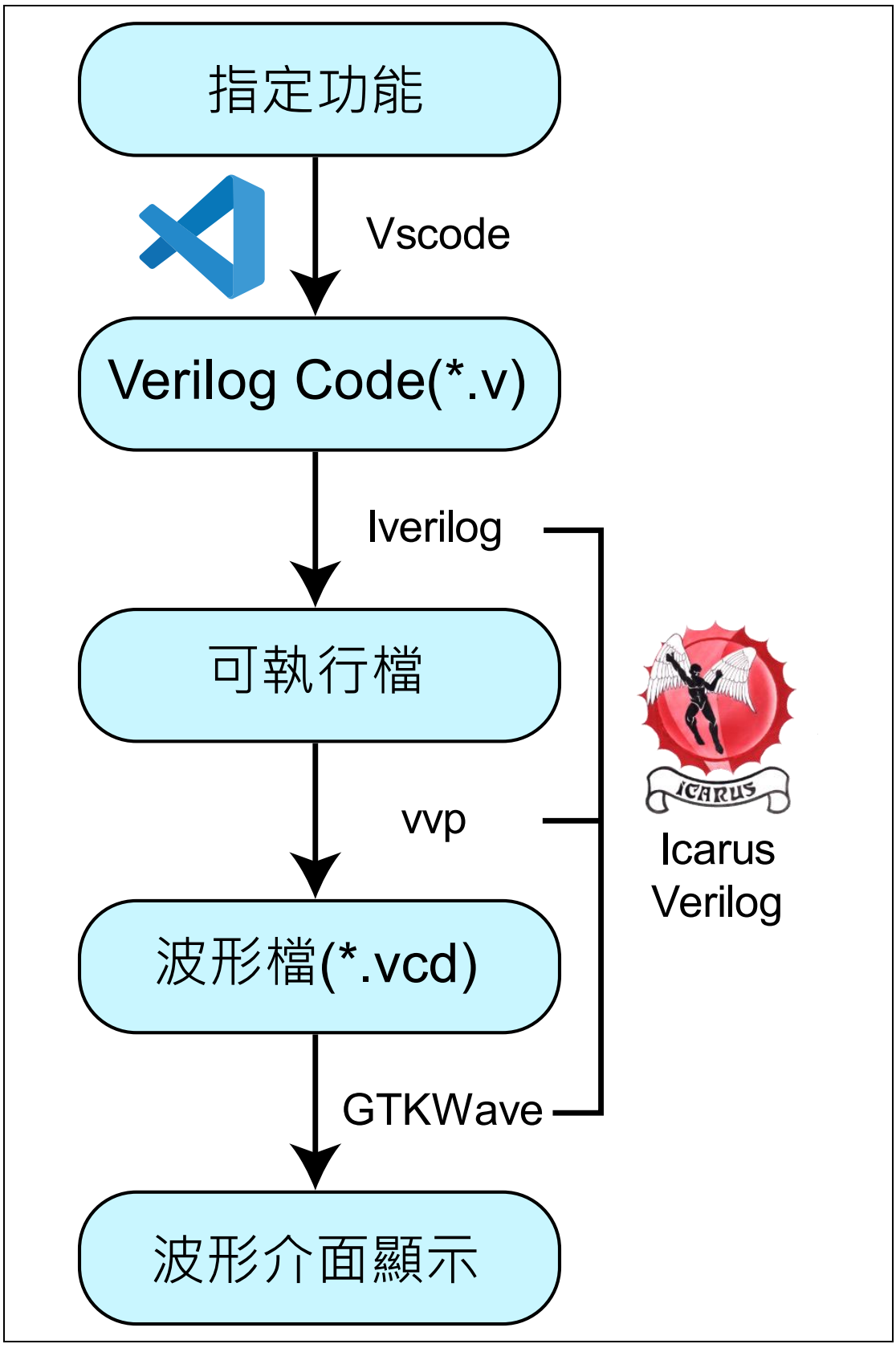
邏輯系統實驗

Lab 12

2021/05/27(四)

第 1 組	
組員姓名	學號
陳旭祺	E24099059
張振杰	E24085034
何啟造	E34085337

- **實驗流程圖(同實驗 Lab11)**



- 實作題(一): Blocking

1. Verilog Code

```
module cnt_blk(clk, rst, a_in, b_in, a_out, b_out);

    input clk, rst, a_in, b_in;
    output a_out, b_out;
    reg a, b;

    assign a_out = a;
    assign b_out = b;

    always@(posedge clk or negedge rst)begin
        if(~rst) begin
            a = a_in;
            b = b_in;
        end
        else begin
           a = b;
           b = a;
        end
    end

endmodule
```

2. Testbench for Verilog Code

```
`timescale  100ms/1ms
module testbench();
    reg clk;
    reg rst_n;

    reg a_in, b_in;
    wire blk_a_out, blk_b_out;
    wire  non_blk_a_out, non_blk_b_out;

    cnt_blk blk1(clk, rst_n, a_in, b_in, blk_a_out,
blk_b_out);
    cnt_nonblk nonblk1(clk, rst_n, a_in, b_in, non_blk_a_out,
non_blk_b_out);
    localparam CLK_PERIOD = 10;
    always #(CLK_PERIOD/2) clk=~clk;

    initial begin
        $dumpfile("testbench.vcd");
        $dumpvars(0, testbench);
    end

    initial begin
        #1 rst_n<=1'bx;clk<=1'bx; a_in =1'bx; b_in = 1'bx;
        #(CLK_PERIOD*3) rst_n<=1;
        #(CLK_PERIOD*3) rst_n<=0;clk<=0;a_in = 0; b_in = 1;
        repeat(5) @(posedge clk);
        rst_n<=1;
        #(CLK_PERIOD*6)
        #(CLK_PERIOD/2) rst_n<=0;a_in = 1; b_in = 0;
        #(CLK_PERIOD/4) rst_n<=1;
        #(CLK_PERIOD*5) $finish;
    end

endmodule
```

- 實作題(二): Non-Blocking

1. Verilog Code

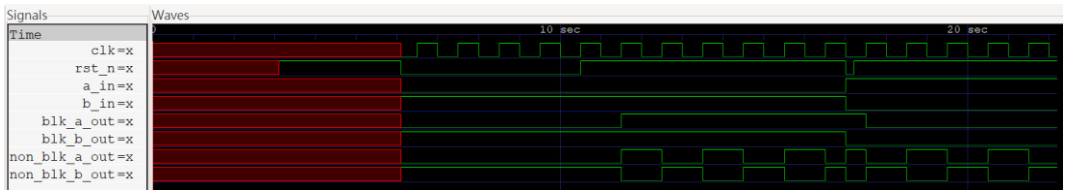
```
module cnt_nonblk(clk, rst, a_in, b_in, a_out, b_out);

    input clk, rst, a_in, b_in;
    output a_out, b_out;
    reg a, b;

    assign a_out = a;
    assign b_out = b;

    always@(posedge clk or negedge rst)begin
        if(~rst) begin
            a <= a_in;
            b <= b_in;
        end
        else begin
           a <= b;
           b <= a;
        end
    end

endmodule
```

2. Testbench for Verilog Code

同實作(一)

● 結果

Waveform



Explanation of Waveform

Blocking	一行一行依序執行 (與軟體語言觀念一樣)
Non-Blocking	併行執行

一開始$rst_n = 0$，a_in 給 a 值(0)、b_in 給 b 值(1)。

直到$rst_n = 1$，Blocking 是 b 的值(1)先給 a；a=b=1 時 a 值再給 b，，並穩定在 a=b=1。
Non-Blocking 是 a 與 b 做**同時**做互換，因此如波形圖所示，每過一個 CLK 正緣觸發就會 0、1 互換。

再到$rst_n = 0$且負緣觸發，此時與原先初始條件相反，a_in 給 a 值(1)、b_in 給 b 值(0)。也可觀察電路為**非同步 reset**，因為rst_n訊號會直接觸發，而不需要等 clk 正緣觸發才驅動。

Blocking 由於是 b 的值(0)先給 a；a=b=0 時 a 值再給 b，並穩定在 a=b=0。
Non-Blocking 一樣是 a 與 b 做**同時**做互換，不受影響。

● 心得

此次實驗介紹 Blocking 與 Non-Blocking 差異，內容相對簡單，我自己也參考了這篇文章(原創)深入探討 blocking 與 nonblocking (SOC) (Verilog)，對於兩者之間的差異有了更深入的了解，其中包括 **event queue** 的概念: 由於電腦軟體本身是依序執行，但硬體電路卻可併行執行，simulator 是軟體寫的，卻要能夠模擬出硬體電路的的並行執行，同時有很多信號被處理，因此需要將同一個 time step 要處理的信號放在一個 event queue，simulator 再依序處理，處理完後再處理下一個 time step，這樣就能使依序執行的 simulator 可以模擬出並行執行的硬體電路。