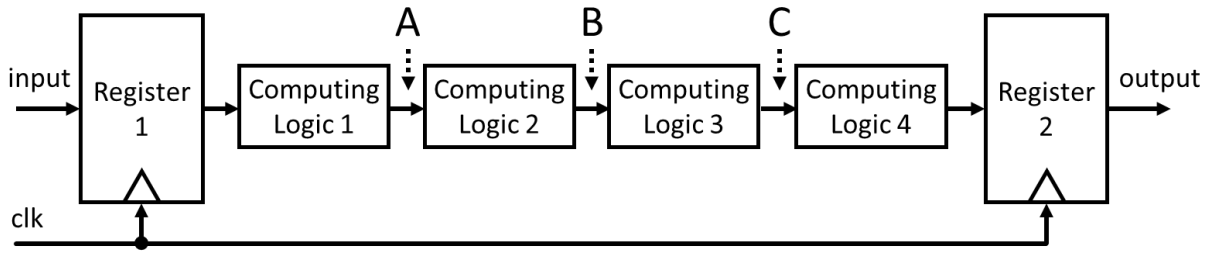
NYCU – 2022 DCS 數位電路與系統 期末考

I. 實作觀念。(共 46 分)

1. Pipeline (8%)



上圖的電路圖中，Computing Logic 1~4 都是單純的組合電路，Register1, 2 皆為正緣觸發 DFF，已知 Computing Logic 1~4 的 Delay 分別為 3ns, 2ns, 2ns, 6ns。
(假設不考慮其他 Delay。)

(a) 若要加入一組 Pipeline register 在 A, B 或 C 其中一處，何處為較好的選擇? 請寫出原因 (2%)

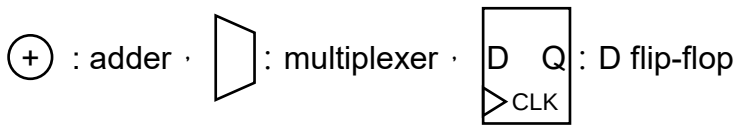
(b) 只考慮 Computing Logic 1~4 的 Delay，不考慮其他 Delay，若在 A, C 兩處加入 pipeline register，clock cycle time 最短可以設定為多少不會產生 timing violation? 請寫出原因 (2%)

(c) 假設 Computing Logic 1~4 的計算量分別為 45, 40, 50, 60 operations，請算出第(b)小題的 throughput。(2)

(d) 請依照(a)(b)小題所述的架構，加入一組或兩組 pipeline register 何者有較高的 throughput? 請寫出原因 (2%)

2. Block diagram & Timing Report (20%)

(a) 請使用 adder, multiplexer, comparator, D flip-flop 的圖例，
畫出以下電路的 Block diagram。(10%)



※ multiplexer 之選擇訊號順序請標示清楚。

※ 所有訊號不論 multi-bit bus 或 1-bit signal 都用一條線表示即可。

※ rst_n 可以不用畫，要標出除了 rst_n 以外的所有 input 和 output 訊號。

```
module Design(
    clk,
    rst_n,
    in_data,
    in_sel,
    out_data
);

input clk, rst_n;
input [3:0] in_data;
input in_sel;
output logic [3:0] out_data;

logic [3:0] data_0, data_0_nxt;
logic [3:0] data_1, data_1_nxt;
logic [3:0] data_sel, data_add;

always_ff@(posedge clk or negedge rst_n) begin
    if (!rst_n) begin
        data_0 <= 0;
        data_1 <= 0;
    end
    else begin
        data_0 <= data_0_nxt;
        data_1 <= data_1_nxt;
    end
end

always_comb begin
    if (in_sel)
        data_sel = data_1;
    else
        data_sel = data_0;
end

assign data_add = data_sel + in_data;

always_comb begin
    if (in_sel) begin
        data_0_nxt = data_0;
        data_1_nxt = data_add;
    end
    else begin
        data_0_nxt = data_add;
        data_1_nxt = data_1;
    end
end

assign out_data = data_add;

endmodule
```

(b)

下圖為(a)電路合成的 timing report，

請依照此 timing report，

回答 critical path 的起點和終點 (2%)

並在(a)的電路圖上畫出此 critical path。(3%)

```
Startpoint: in_sel (input port clocked by clk)
Endpoint: out_data[2]
          (output port clocked by clk)
Path Group: clk
Path Type: max

Point                                    Incr       Path
-----------------------------------------------------------
clock clk (rise edge)                    0.00       0.00
clock network delay (ideal)              0.00       0.00
input external delay                     2.50       2.50 f
in_sel (in)                              0.00       2.50 f
U50/Y (MXI2X4)                           0.12       2.62 f
U74/Y (OAI31X2)                          0.26       2.88 r
U68/Y (XNOR3X4)                          0.43       3.32 r
out_data[2] (out)                        0.00       3.32 r
data arrival time                                   3.32

clock clk (rise edge)                    5.00       5.00
clock network delay (ideal)              0.00       5.00
output external delay                   -2.50       2.50
data required time                                  2.50
-----------------------------------------------------------
data required time                                  2.50
data arrival time                                  -3.32
-----------------------------------------------------------
slack (VIOLATED)                                   -0.82
```

(c) 請依照(b)的 timing Report 回答，請問電路發生了什麼問題?

如果有問題發生，請解釋發生原因並寫出如何修改電路解決此問題。(5%)

3. Waveform (18%)

請參照以下 Pattern 與 Design 的 code，畫出相對應的波型圖。

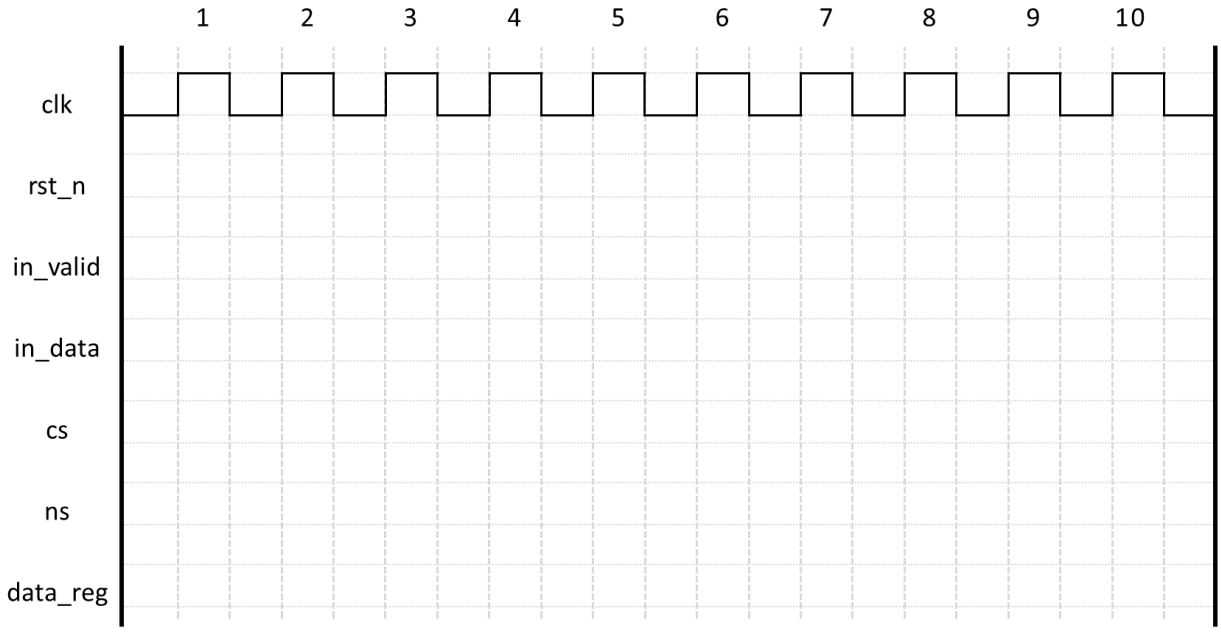
in_data, data_nxt, data_reg 皆為有號數且所有訊號都不會發生 overflow。

```
//--------------------------------
//  Pattern
//--------------------------------
logic clk, rst_n;
logic in_valid;
logic signed [4:0] in_data;

parameter CYCLE = 5;

always  #(CYCLE/2) clk = ~clk;

initial begin
    clk = 0;
    in_valid = 0;
    in_data = 0;
    rst_n = 1;
    @(posedge clk);
    rst_n = 0;
    #(CYCLE);
    rst_n = 1;
    @(negedge clk);
    in_valid = 1;
    in_data = -4;
    @(negedge clk);
    in_valid = 0;
    in_data = 0;
    repeat(2)@(negedge clk);
    in_valid = 1;
    in_data = 2;
    @(negedge clk);
    in_valid = 1;
    in_data = 4;
    @(negedge clk);
    in_valid = 0;
    in_data = 0;
    repeat(3)@(negedge clk);
    $finish;
end
```

```
//--------------------------------------------
//  Design
//--------------------------------------------
logic in_valid;
logic signed [4:0] in_data, data_reg, data_nxt;
logic [1:0] cs, ns;

parameter IDLE = 0,
          DECODE = 1,
          ALU1 = 2,
          ALU2 = 3;

always_ff@(posedge clk or negedge rst_n) begin
    if (!rst_n)
        cs <= IDLE;
    else
        cs <= ns;
end

always_comb begin
    case(cs)
        IDLE:
            if (in_valid)
                ns = DECODE;
            else
                ns = IDLE;
        DECODE:
            if (data_reg<0)
                ns = ALU2;
            else
                ns = ALU1;
        ALU1:
            ns = IDLE;
        ALU2:
            ns = ALU1;
    endcase
end

always_ff@(posedge clk or negedge rst_n) begin
    if (!rst_n)
        data_reg <= 0;
    else
        data_reg <= data_nxt;
end

always_comb begin
    case(cs)
        IDLE:
            if (in_valid)
                data_nxt = in_data;
            else
                data_nxt = data_reg;
        DECODE:
            data_nxt = data_reg;
        ALU1:
            data_nxt = data_reg * 3;
        ALU2:
            data_nxt = data_reg + 3;
    endcase
end
```

1-bit signal 與 multi-bit bus 的畫法請參照下圖範例：

請畫出 rst_n、in_valid、in_data、cs、ns、data_reg (各 3%)的波型圖。



II. 基本觀念。(每題 3 分，除最後兩題各 4 分。共 23 分)

4. FPGA 和 ASIC 是常見的兩種實現方法。請舉出兩點這兩種方法使用上的差異。(3%)

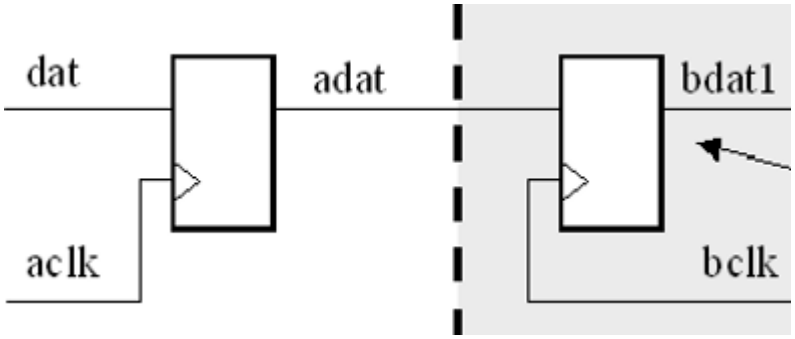
5. 模組間資料傳遞經常使用 valid-ready 方式控制。請以波形圖解釋 valid-ready 運作方式(hint: 列出以下訊號 clock, data, valid, ready) (3%)

6. 如果要簡化 valid-ready 控制，請劃出並說明簡化後的方法。(3%)

7. 假設目前有一 16K words x 4 bits SRAM。請畫圖設計可支援一次 64 bit 輸出的 SRAM 的讀取。(3%)

8. 接續上面，請畫圖設計可支援可同時支援 3 個同時讀寫的 SRAM 電路，每個讀寫四 bit。(3%)

9. 針對以下 adat 訊號在兩個 clock domain aclk, bclk 間傳遞，請劃出 adat 傳遞可能失敗的波形圖 (hint: 列出以下訊號 aclk, adat, bclk, bdat1) (4%)



10. microcode programming 是一種常見的 FSM 實現方法。請針對以下 FSM table，畫出完整使用 microcode 方法的 FSM 實現電路。請列出 microcode 中 memory 所儲存的資料與對應位址。(4%)

	State	Next State		Output
		!carew	carew	
gns	00	00	01	100001
yns	01	11	11	010001
gew	11	10	10	001100
yew	10	00	00	001010

III. Design problem

11. (18%) (Pipelining and parallelism, 3 points for each question)

You have been given the task of designing a new media chip for HD applications. Specifications call for the rendering of a 2000x1000 pixel image at 100Hz, With a single module, it takes 10us to process one pixel, and t_{reg} is 0ps.

(a) What is the throughput needed, in pixels per second, for this task?

(b) Your coworker suggests making one long pipeline. Can you do this and still meet your throughput goal? If so, how many stages does it require? If not, why not?

(c) Why could (a) be a bad idea?

(d) Another coworker suggest just replicating the processing module; how many do you need?

(e) Why this be a bad idea?

(f) After talking to the logic designers, you decide to use replicated ten-stage pipelined. How many do you need?

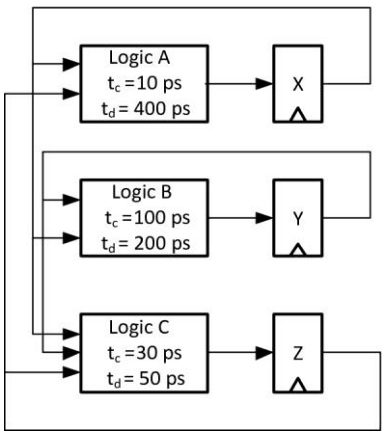
12. (13%) (setup time and hold time violation check)

For Flip-Flop A in Table 1, and a 2GHz clock, check the following figure for timing violations. If there is a hold time violation, indicate

(a) (4%) Where delay must be added, and

(b) (4%) the necessary delay .

(c) (5%) Recheck for setup time violation. Is it possible to run faster than 2GHz clock? Derive the cycle time for this maximum clock frequency?



Parameter (ps)	DFF A
Setup time: t_s	20
Hold time: t_h	30
Contamination delay, clock to q, T_{ccq}	10
Propagation delay, clock to q, t_{dcq}	20

(Hint: setup time check, tcy ≥ tdCQ + tdMax + ts

Hold time check, th ≤ tcCQ + tcMin)