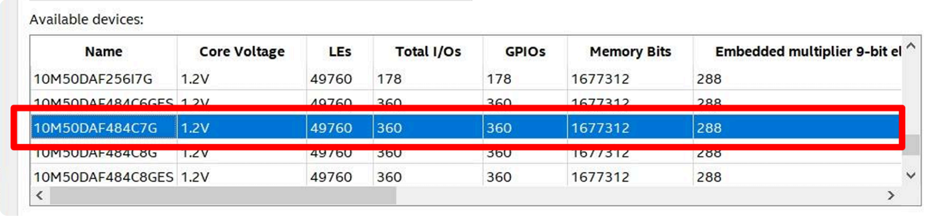
Week11 Up Down Counter

Basic Setting

1. **Device Family:** MAX 10(DA/DF/DC/SA/SC)
2. **Available Devices:** 10M50DAF484C7G



Available devices:

Name	Core Voltage	LEs	Total I/Os	GPIOs	Memory Bits	Embedded multiplier 9-bit el
10M50DAF256I7G	1.2V	49760	178	178	1677312	288
10M50DAF484C6GES	1.2V	49760	360	360	1677312	288
10M50DAF484C7G	1.2V	49760	360	360	1677312	288
10M50DAF484C8G	1.2V	49760	360	360	1677312	288
10M50DAF484C8GES	1.2V	49760	360	360	1677312	288

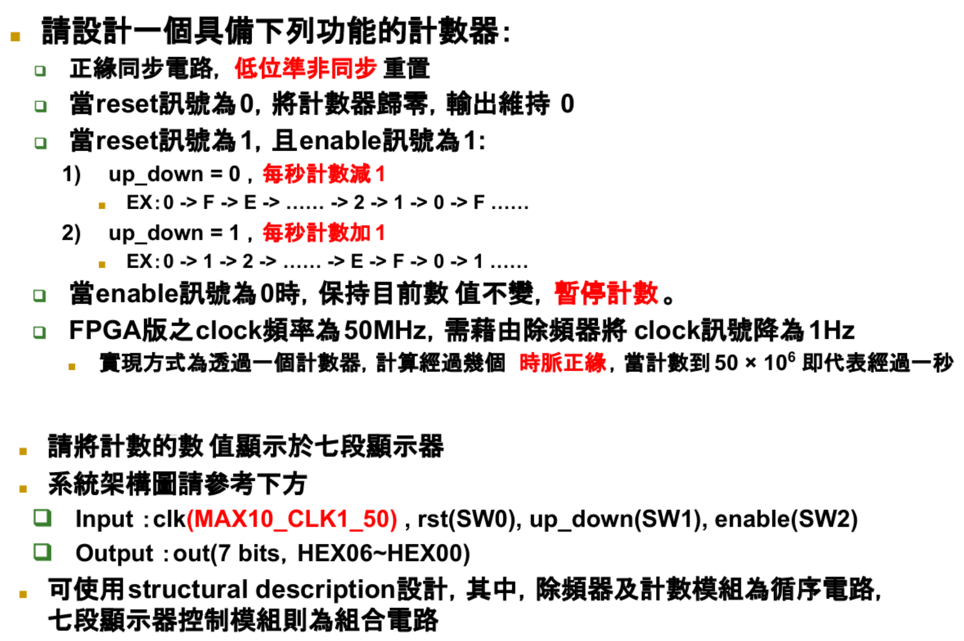
3. **EDA Tool Settings, Simultation:** Modelsim-Altera, Verilog HDL
4. Click 「Assignments > Pin Planner > Location」分配 Pin
5. 到裝置管理員，「其他裝置 > USB-Blaster」，右鍵「更新驅動程式」
6. 瀏覽電腦上的驅動程式
7. 路徑 `C:\intelFPGA_lite\16.1\quartus\drivers\usb-blaster` （16.1 或 18.1）
8. 到裝置管理員，「通用序列匯流排控制器 > Altera USB-Blaster」，有就代表成功
9. Click 「Tools > Programmer」
10. Click 「Hardware Setup > USB Blaster」，然後 Currently selected hardware 選「USB-Blaster [USB0]」
11. Click 「Add file > ~.sof > Open」（注意是在 `outputfiles` 裡面）
12. Click 「Start」顯示 Success 成功

Seven Display

- 0~15 七段顯示器對應
  - 0: 亮
  - 1: 不亮

```
4'd0: out = 7'b1000000;
4'd1: out = 7'b1111001;
4'd2: out = 7'b0100100;
4'd3: out = 7'b0110000;
4'd4: out = 7'b0011001;
4'd5: out = 7'b0010010;
4'd6: out = 7'b0000010;
4'd7: out = 7'b1111000;
4'd8: out = 7'b0000000;
4'd9: out = 7'b0010000;
4'd10: out = 7'b0001000;
4'd11: out = 7'b0000011;
4'd12: out = 7'b1000110;
4'd13: out = 7'b0100001;
4'd14: out = 7'b0000110;
4'd15: out = 7'b0001110;
```

Request



- 請設計一個具備下列功能的計數器:
  - ❑ 正緣同步電路, 低位準非同步 重置
  - ❑ 當reset訊號為0, 將計數器歸零, 輸出維持 0
  - ❑ 當reset訊號為1, 且enable訊號為1:
    1) up_down = 0，每秒計數減1
       - EX:0 -> F -> E -> ...... -> 2 -> 1 -> 0 -> F ......
    2) up_down = 1，每秒計數加1
       - EX:0 -> 1 -> 2 -> ...... -> E -> F -> 0 -> 1 ......
  - ❑ 當enable訊號為0時, 保持目前數 值不變, 暫停計數。
  - ❑ FPGA版之clock頻率為50MHz, 需藉由除頻器將 clock訊號降為1Hz
    - 實現方式為透過一個計數器, 計算經過幾個 時脈正緣, 當計數到 50 × 10^6 即代表經過一秒

- 請將計數的數 值顯示於七段顯示器
- 系統架構圖請參考下方
  - ❑ Input : clk(MAX10_CLK1_50) , rst(SW0), up_down(SW1), enable(SW2)
  - ❑ Output : out(7 bits, HEX06~HEX00)
- 可使用structural description設計, 其中, 除頻器及計數模組為循序電路, 七段顯示器控制模組則為組合電路

Logic Block

- Input: clk, rst ,enable, up_down
  - rst == 0 means 將值設為 0
  - rst == 1 means 可進行倒數
  - enable == 0 means 暫停數數，停留在該數字
  - enable == 1 means 可進行倒數
  - up_down == 0 means 每秒減 1
  - up_down == 1 means 每秒加 1
- Output: out

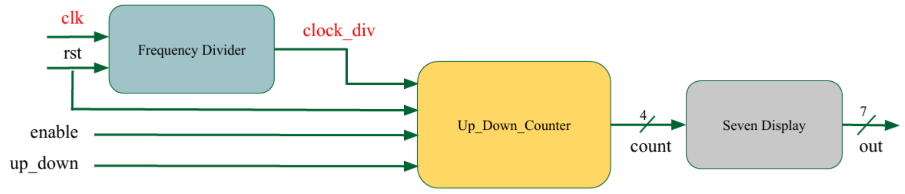
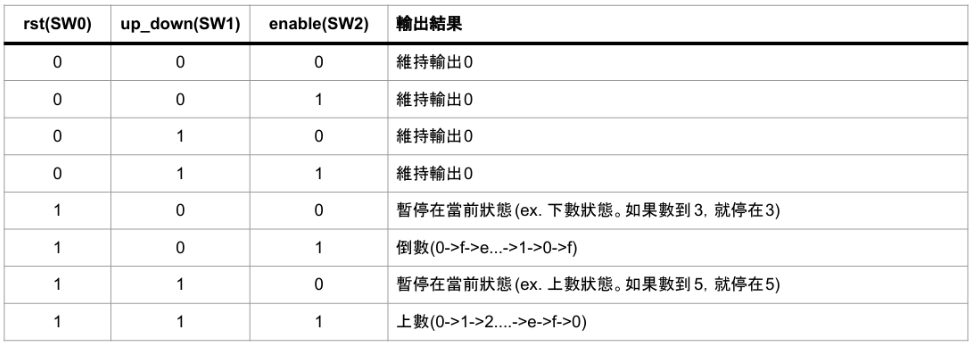


Table Information



rst(SW0)	up_down(SW1)	enable(SW2)	輸出結果
0	0	0	維持輸出0
0	0	1	維持輸出0
0	1	0	維持輸出0
0	1	1	維持輸出0
1	0	0	暫停在當前狀態(ex. 下數狀態。如果數到3, 就停在3)
1	0	1	倒數(0->f->e...->1->0->f)
1	1	0	暫停在當前狀態(ex. 上數狀態。如果數到5, 就停在5)
1	1	1	上數(0->1->2....->e->f->0)

Verilog

frequency_driver.v

一般未做轉換的頻率為 50MHz

$$\text{Frequency} = 50\,\text{MHz} = 50 \times 10^6\,\text{次}/\text{秒}$$

然後我們的目標是要將 50MHz 變成 1Hz

$$\text{Frequency} = 1\,\text{Hz} = 1\text{次}/\text{秒}$$

而 `TimeExpire` 是半個週期，故為 0.5 秒中總共的週期數，故設定為 25×10^6

```verilog
`define TimeExpire 32'd25000000
module frequency_driver(clk, rst, div_clk);
    input clk, rst;
    output div_clk;
    reg div_clk;
    reg [31:0] count;

    always@(posedge clk)
    begin
        if (!rst)
        begin
            count <= 32'd0;
            div_clk <= 1'b0;
        end
        else
        begin
            if (count == `TimeExpire)
            begin
                count <= 32'd0;
                div_clk <= ~div_clk;
            end
            else
            begin
                count <= count + 32'd1;
            end
        end
    end
endmodule
```

counter.v

```verilog
module counter(clock_div, rst, enable, up_down, count);
input clock_div, rst, enable, up_down;
output reg [3:0] count;

always@(posedge clock_div or negedge rst)
begin
        if(!rst)
        begin
                count <= 4'b0000;
        end
        else if(enable)
        begin
                if(up_down == 0)
                begin
                        count <= count-1;
                end
                else
                begin
                        count <= count+1;
                end
        end
end
endmodule
```

seven_display.v

```
module seven_display(in, out);

input [3:0] in;
output [6:0] out;
reg [6:0] out;

always @(in)
begin
    case(in)
        4'd0: out = 7'b1000000;
        4'd1: out = 7'b1111001;
        4'd2: out = 7'b0100100;
        4'd3: out = 7'b0110000;
        4'd4: out = 7'b0011001;
        4'd5: out = 7'b0010010;
        4'd6: out = 7'b0000010;
        4'd7: out = 7'b1111000;
        4'd8: out = 7'b0000000;
        4'd9: out = 7'b0010000;
        4'd10: out = 7'b0001000;
        4'd11: out = 7'b0000011;
        4'd12: out = 7'b1000110;
        4'd13: out = 7'b0100001;
        4'd14: out = 7'b0000110;
        default : out = 7'b0001110;
    endcase
end
endmodule
```

main.v

```
module main(clk, rst, enable, up_down, out);
input clk, rst, enable, up_down;
output [6:0] out; // 不需要使用 reg 因為沒有要在 always 中使用

wire clock_div;
wire [3:0] count;

frequency_driver f1(
        .clk(clk),
        .rst(rst),
        .div_clk(clock_div));

counter counter1(
        .clock_div(clock_div), // 非用 fpga 請將 clock_div 換成 clk
        .rst(rst),
        .enable(enable),
        .up_down(up_down),
        .count(count)); // 記得要多輸出端也要連接

seven_display display1(
        .in(count),
        .out(out));

endmodule
```