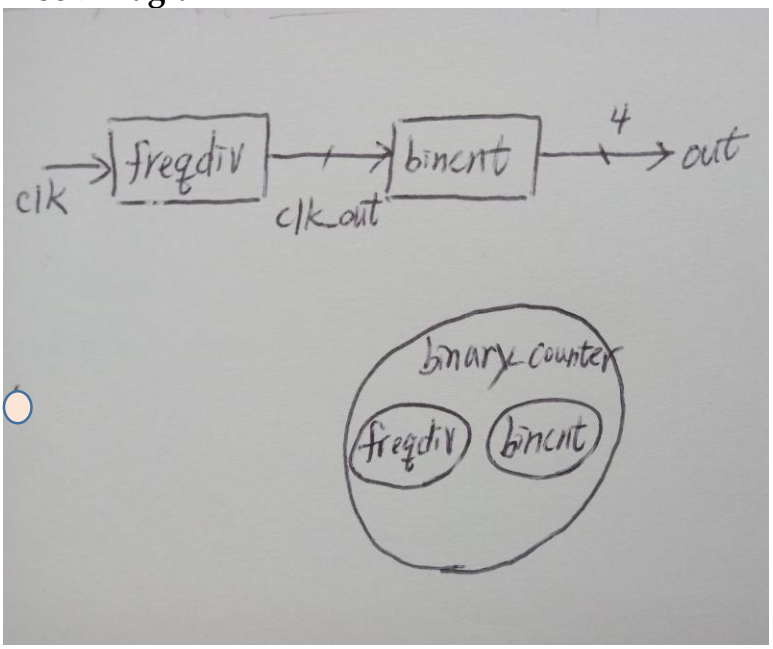
1.
Design Specification

✓ **Input / Output**

For a 4 bit binary up counter:

Input: clk, rst_n.
Output: out[3:0].

✓ **Block Diagram**



Design Implementation

✓ **Function Descriptions**

此功能為從 0 到 16 的 4 bit binary up counter, 做法為先作一個除頻器，輸出 1hz 到 binary up counter 當作控制的 clock.而 counter 做法為 一個 4 bit 加法器，每一個 clock 週期加 1 送到 DFF, 最後 out 輸出。

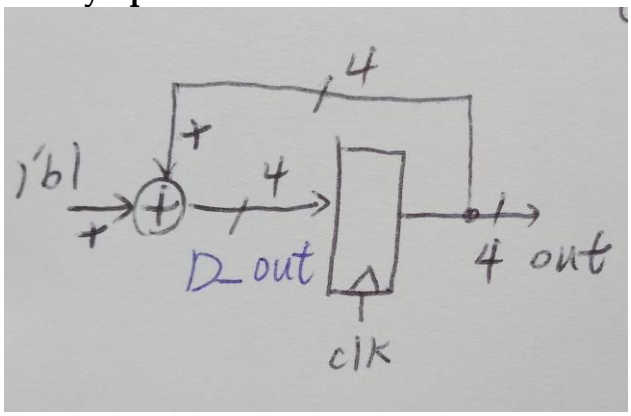
✓ **Logic Functions**

(freqdiv 已在前實驗題及)

- **binary up counter**
  D_out = out + 1;

✓ **Logic Diagram**

- **Binary up counter**



✓ **I/O pin**

I/O	clk	rst_n	out[3]	out[2]	out[1]	out[0]
LOC	W15	V17	V18	U19	E19	U16

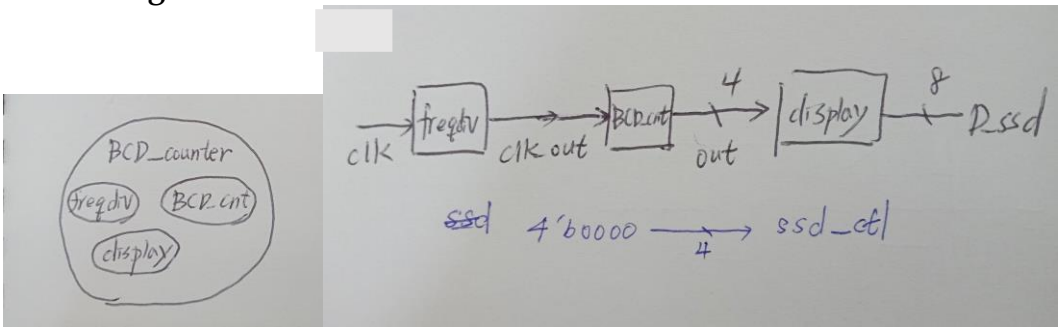
2.
Design Specification

✓ **Input / Output**

For a BCD up counter:

Input: clk, rst_n.
Output: D_ssd[7:0], ssd_ctl[3:0].

✓ **Block Diagram**



Design Implementation

✓ **Function Descriptions**

此功能為一個 BCD up counter, 數 0-9 輸出到七段顯示器。

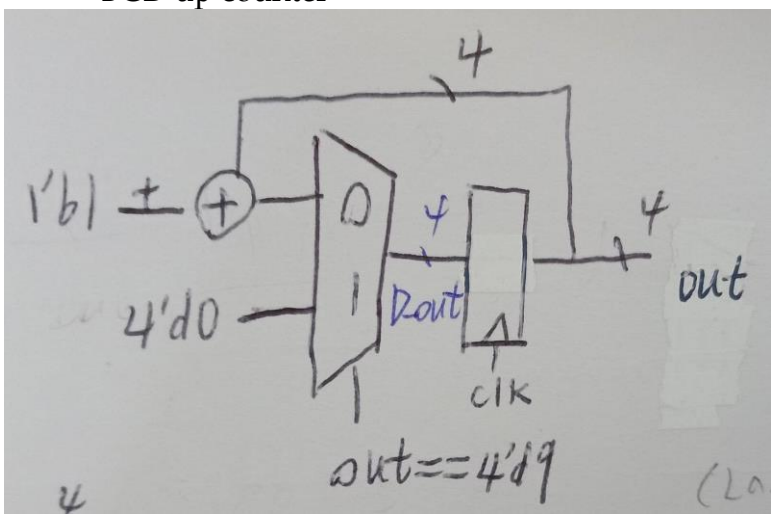
我的做法為先做一個除頻器輸出 1hz 作 BCD counter 的 clock 控制, BCD counter 中有一個 mux 做選擇，若現在的值等於 9 則選 0 輸入，否則繼續把上一個值加 1.此 counter 輸出的 4 bit 結果再輸進 display 中，功能為做 BCD 轉 7-segment display decoder 而此功能的作法在之前的實驗已經提及。
另外，在此讓七段顯示器四個同時亮。

✓ **Logic Functions**

- BCD up counter
  if (out==4'd9) D_out = 4'd0;
  else D_out = out + 1;

✓ **Logic Diagram**

- BCD up counter



✓ **I/O pin**

I/O	clk	rst_n	ssd_ctl[3]	ssd_ctl[2]	ssd_ctl[1]	ssd_ctl[0]		
LOC	W15	V17	U2	U4	V4	W4		
I/O	D_ssd[7]	D_ssd[6]	D_ssd[5]	D_ssd[4]	D_ssd[3]	D_ssd[2]	D_ssd[1]	D_ssd[0]
LOC	W7	W6	U8	V8	U5	V5	U7	V7

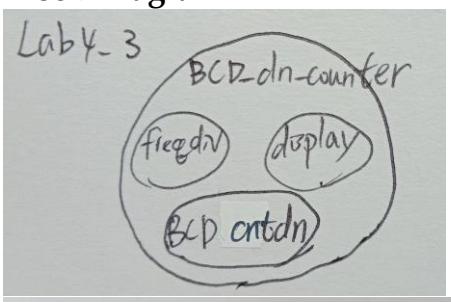
3.
Design Specification

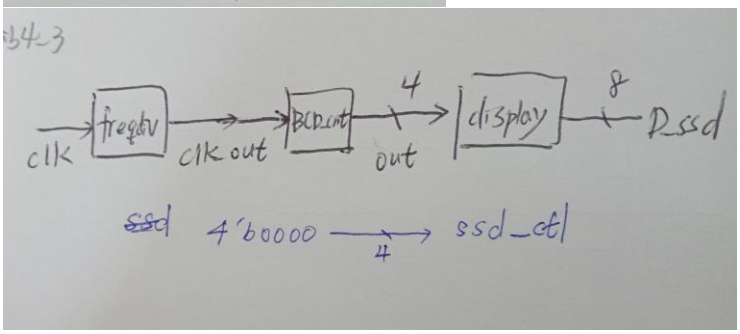
- ✓ **Input / Output**

  For a BCD down counter:

  Input: clk, rst_n.
  Output: D_ssd[7:0], ssd_ctl[3:0].

- ✓ **Block Diagram**





Design Implementation

- ✓ **Function Descriptions**

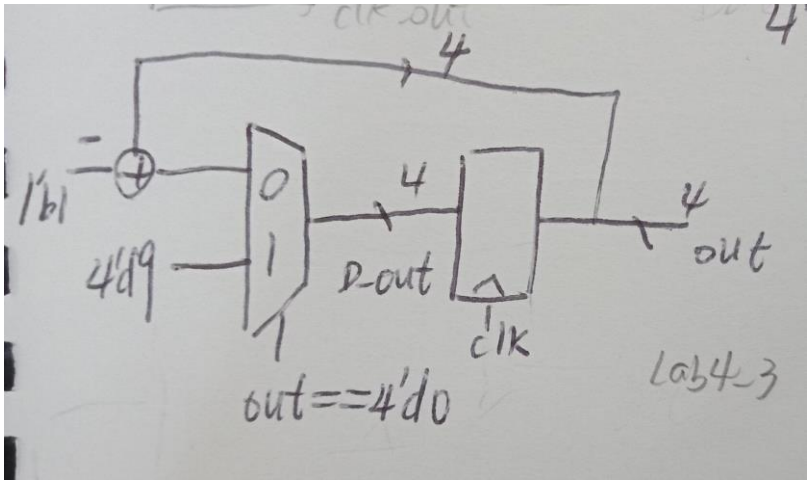
  此功能為一個 BCD down counter, 數 9~0 輸出到七段顯示器。
  先做一個除頻器輸出 1hz 作 BCD down counter 的 clock 控制, BCD counter 中有一個 mux 做選擇，若現在的值等於 0 則選 9 輸入，否則繼續把上一個值減 1. 此 counter 輸出的 4 bit 結果再輸進 display 中，功能為做 BCD 轉 7-segment display decoder 而此功能的作法在之前的實驗已經提及。
  另外，在此讓七段顯示器四個同時亮。

- ✓ **Logic Functions**

  - BCD down counter
    if (out==4'd0) D_out = 4'd9;
    else D_out = out - 1;

✓ **Logic Diagram**

- BCD down counter



✓ **I/O pin**

I/O	clk	rst_n	ssd_ctl[3]	ssd_ctl[2]	ssd_ctl[1]	ssd_ctl[0]		
LOC	W15	V17	U2	U4	V4	W4		
I/O	D_ssd[7]	D_ssd[6]	D_ssd[5]	D_ssd[4]	D_ssd[3]	D_ssd[2]	D_ssd[1]	D_ssd[0]
LOC	W7	W6	U8	V8	U5	V5	U7	V7

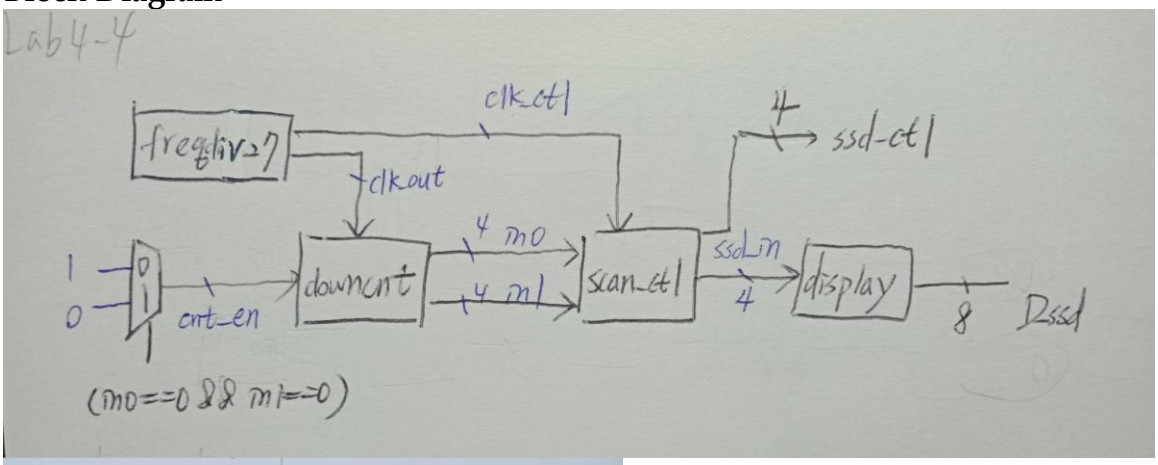
4.
Design Specification

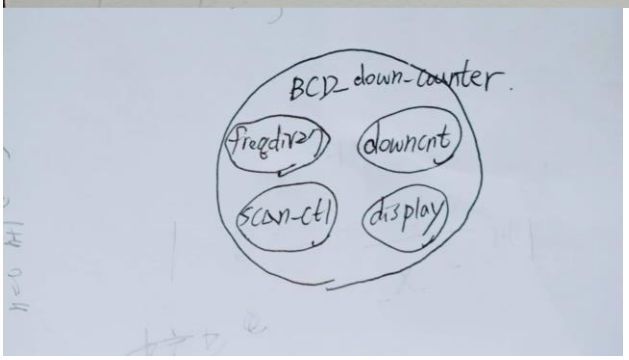
✓ **Input / Output**

For a 30 seconds count down timer:

Input: clk, rst_n.
Output: D_ssd[7:0], ssd_ctl[3:0].

✓ **Block Diagram**





Design Implementation

✓ **Function Descriptions**

此功能為作一個 30 秒倒數計時。

我的做法為除頻器輸出兩種頻率，第一種頻率接近 1hz, 輸進 down counter 中，做兩個 down counter 的 clock 輸入。第二種頻率取中間段的訊號，頻率大概可以使人類的視覺產生暫留，當作 scan 的 clock 輸入。

Down counter 的做法為輸入 decrease(要不要減)，counter reset 時的初始值，limit(counter 數到最大)，然後輸出 val(值)與 borrow(是否要上一位數借位)。在 counter 中有一個 mux, 若 decrease 為 1, val 為 0，則選他的 limit 進來，若 decrease 為 1, val 不等於 0, 則把原本的值減 1. 若 decrease 為 0 則不做改變，維持原來的值。另外一個 counter 外的 mux 選擇 borrow 的值。若 decrease 為 1, val=0 則選 1 為 borrow, 否則選 0 為 borrow 輸出。

至於兩個 down counter 的 decrease 輸入，第一個 down counter 的輸入為 mux 做選擇:判斷現在是否兩個 counter 輸出的值都為 0，若是，把 decrease = 0，這樣就能停在 00.若否，則 decrease = 1. 第二個 down counter 的輸入為第一個 counter 輸出的 borrow.

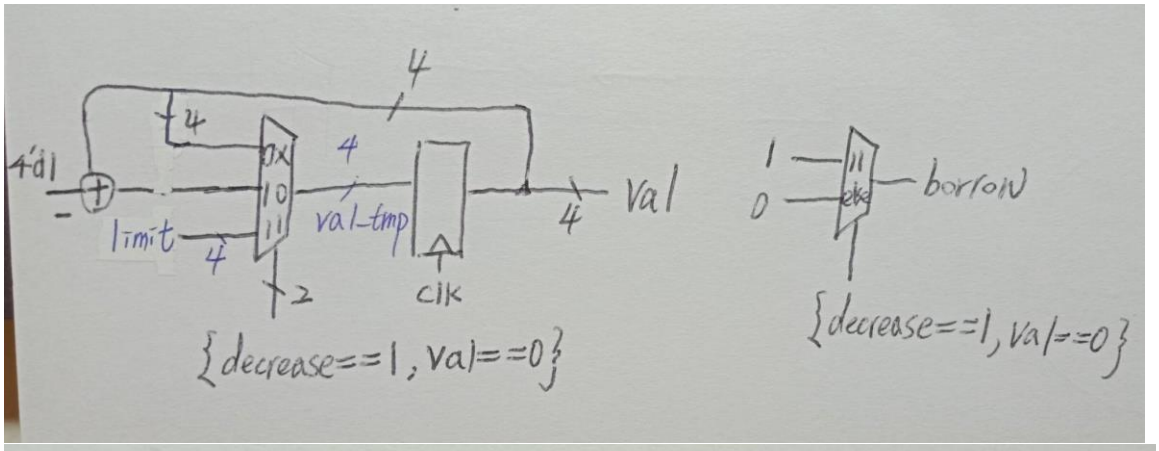
然後兩個 down counter 輸出的值輸進 scan 裡，scan 由除頻器輸進的頻率做選擇現在顯示誰，藉以產生視覺暫留的效果。scan 輸出的 4 bit 值又輸進 display 做 decode 到七段顯示器。(此兩個功能之前的實驗已提及)

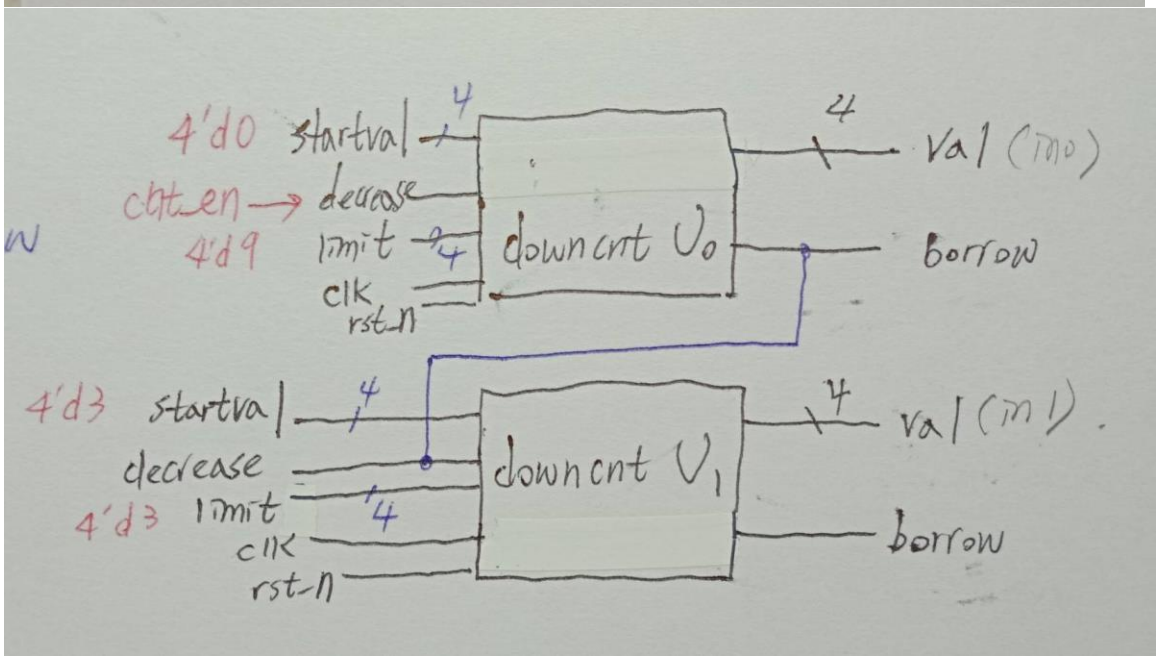
✓ **Logic Functions**

- **Down counter**
  if (decrease && val== 4'd0), val_tmp = limit, borrow = 1
  else if (decrease && val!=4'd0), val_tmp = val-1, borrow =0
  else val_tmp = val, borrow = 0
- **mux for the first down counter input**
  if (in0==0&&in1==0) cnt_en = 1'b0
  else cnt_en = 1'b1

✓ **Logic Diagram**

- **Down counter**





✓ **I/O pin**

I/O	clk	rst_n	ssd_ctl[3]	ssd_ctl[2]	ssd_ctl[1]	ssd_ctl[0]		
LOC	W15	V17	U2	U4	V4	W4		
I/O	D_ssd[7]	D_ssd[6]	D_ssd[5]	D_ssd[4]	D_ssd[3]	D_ssd[2]	D_ssd[1]	D_ssd[0]
LOC	W7	W6	U8	V8	U5	V5	U7	V7

Discussion

這次的實驗主要在做 counter，不管是 up counter 跟 down counter，概念都很像。這兩個最大的差別在於判斷當值等於甚麼時要輸入甚麼值。
因為上次有做過除頻器了，因此這次的實驗比較輕鬆，只要引入之前的除頻器，自己在寫一個 counter，若要顯示出來則再引入 scan 跟 display module.
從開學到現在的每一次實驗，越來越能感覺到 module 功能連結在一起的感覺。要甚麼功能就寫甚麼功能，先畫出方塊圖，把每個方塊都寫成一個 module，再把中間的輸入輸出連接好，就沒有問題了。

Conclusion

這次實驗的核心功能為 counter，但 counter 需要 clock 來控制因此需要除頻器。而 counter 要顯示在七段顯示器上面，因此需要顯示模組(scan, display)，而這次的最後一題要停在 00 又需要一個能控制 counter 的功能。把這些功能都完成，就能看到成品了!