**Lab3.1** Construct a 27-bit synchronous binary counter

1. Design specification

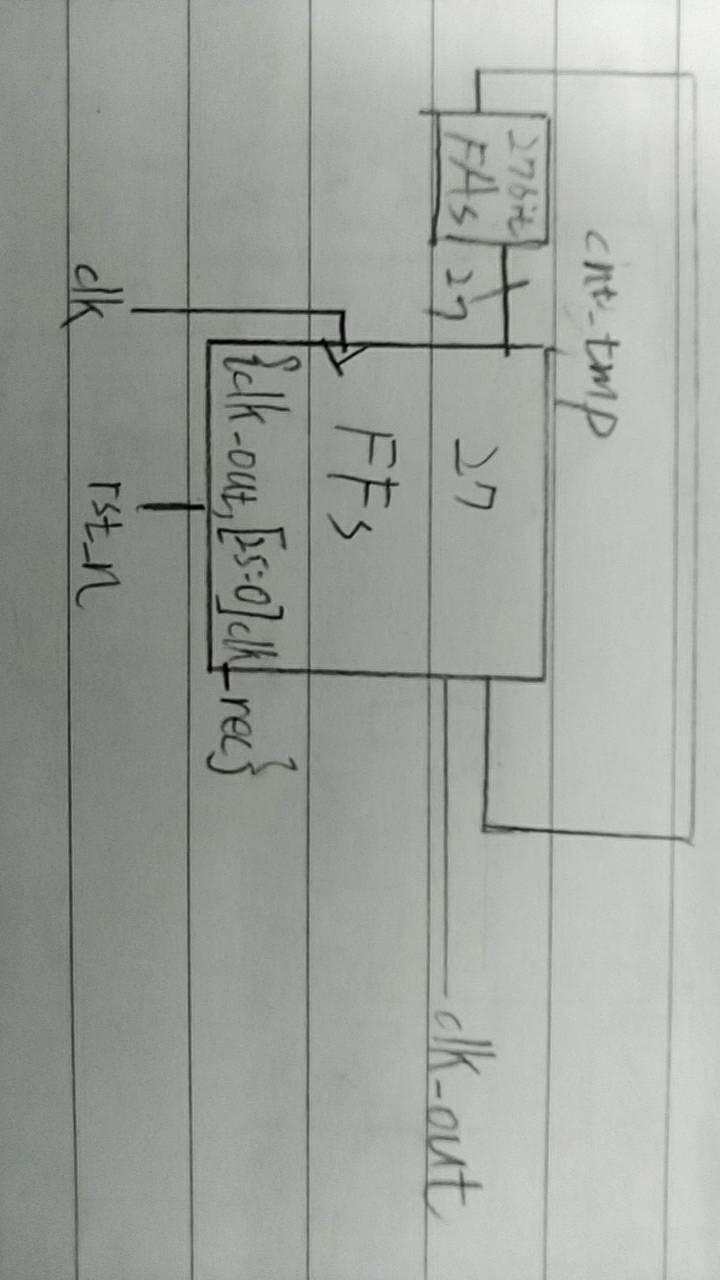
功能: 將時脈頻率變為原本的1/倍。

輸入: clk (原石英震盪器頻率)

rst\_n(重設除頻器)

輸出: clk\_out (輸出除頻後時脈)

2. Design implementation

二的二十七次方除頻器是將原本的頻率縮小二的二十七倍，也就是說原本的clk每過0.5個頻率就會由低電位到高電位，或由高電位到低電位。而現在需要過二的二十六次方才會變一次。所以至少需要27個暫存器，最前面的用來輸出新clk訊號，後面的用來進位累積用。然後再一個由原本訊號正緣觸發的加法器，就可以使暫存器最末位產生出想要的時脈了。 

3.I/O pin assignment

輸入: clk 接至 W5

rst\_n 接至 V17

輸出: clk\_out 接至 U16

**Lab3.2** Use a count-for-50M counter and some glue logics to construct a 1 Hz clock frequency.

1. Design specification

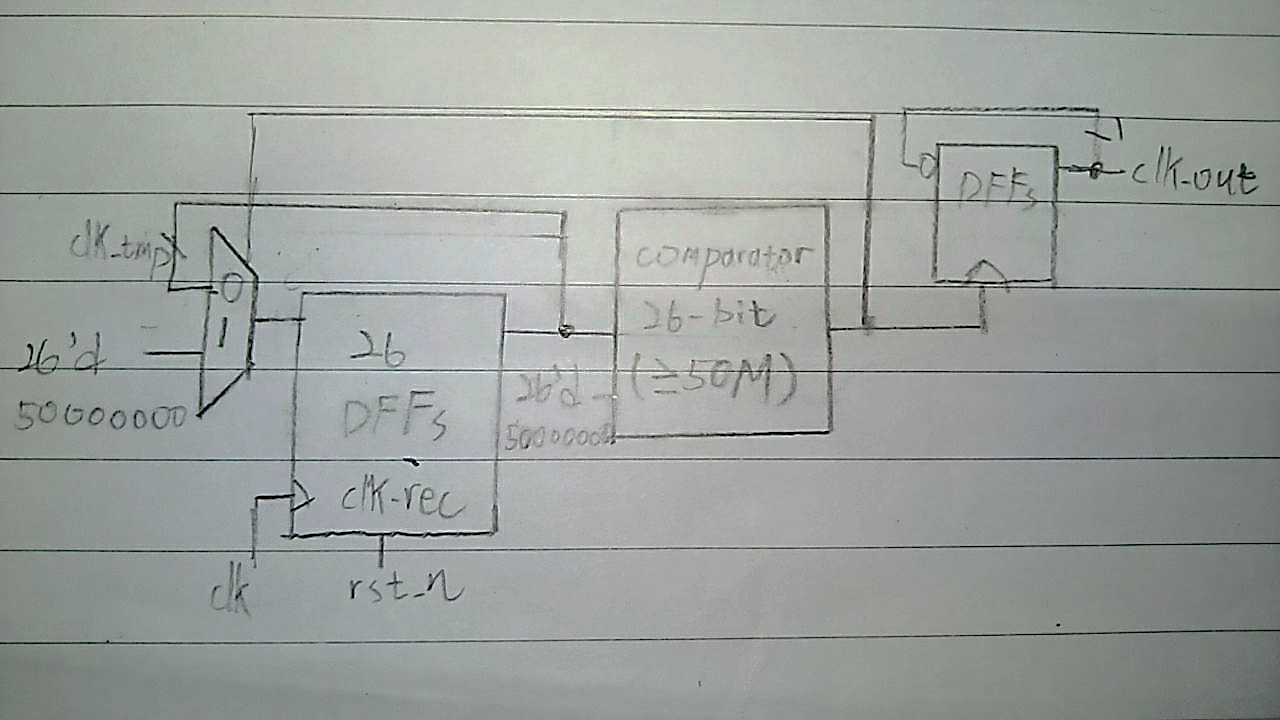
功能: 將原本一億赫茲的頻率除至一赫茲。

輸入: clk (原石英震盪器頻率)

rst\_n(重設除頻器)

輸出: clk\_out (輸出除頻後時脈)

2. Design implementation

第二題由於需要的頻率不是二的倍數，所以需要使用其他方式。我們可以嘗試讓計數器們加到一億時，使用比較器，使其輸出高訊號。在輸出的同時，可以同時將暫存器存值零，使其重新開始數。但是這樣產生的訊號不會是高低各佔一半，而是只有一個原clk單位。所以還需要再加一個toggle暫存器，而當暫存器的值到達五千萬時，會同時觸發Toggle暫存器和歸零計數器，產生我們想要的訊號。

3.I/O pin assignment

輸入: clk 接至 W5

rst\_n 接至 V17

輸出: clk\_out 接至 U16

**Lab3.3** Implement pre-lab2 with clock frequency of 1 Hz

1. Design specification

功能:每過一秒鐘把值傳到下一個暫存器。

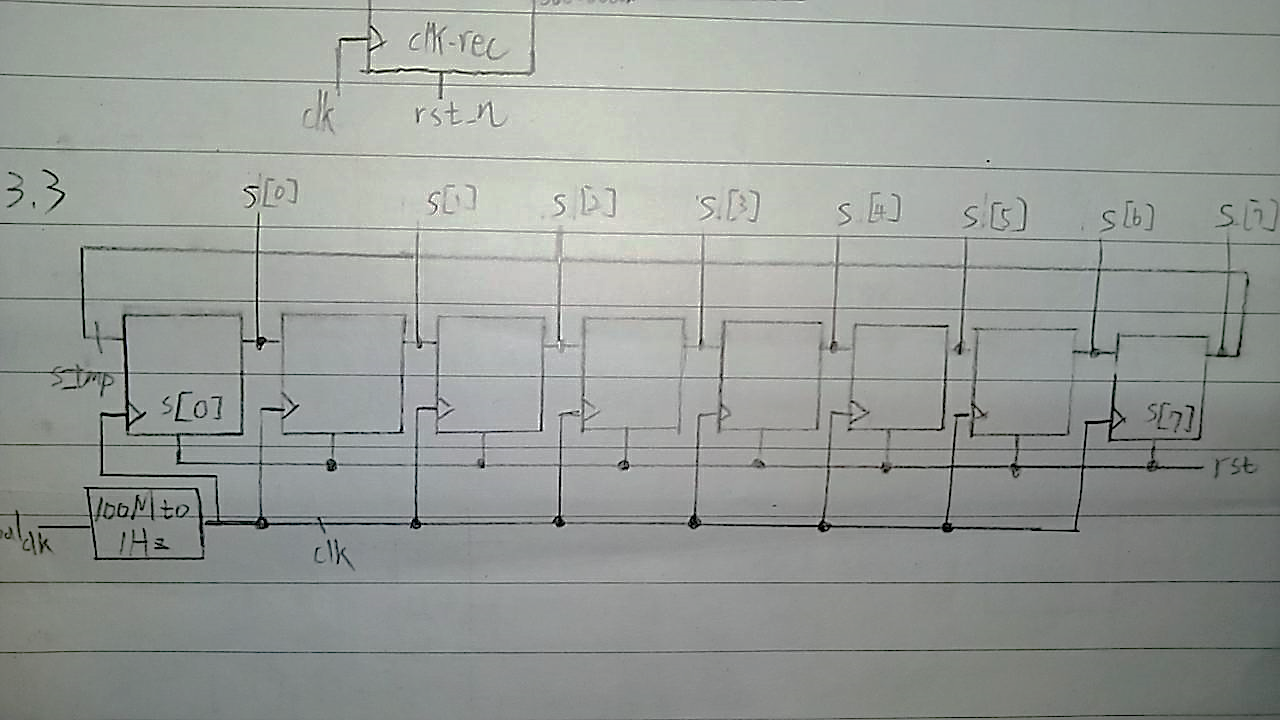
輸入: globalclk (原石英震盪器頻率)

rst (重設機器)

輸出: [7:0]led (輸出相對應燈泡)

2. Design Implementation

將pre-lab2的ringcounter接上剛剛Lab3.2的除頻器即可。



3. I/O pin assignment

|  |  |  |
| --- | --- | --- |
|  | Verilog變數名稱 | 輸入/輸出腳位 |
| 輸入 | globalclk | W5 |
| rst | V17 |
| 輸出 | [0]s | U16 |
| [1]s | E19 |
| [2]s | U19 |
| [3]s | V19 |
| [4]s | W18 |
| [5]s | U15 |
| [6]s | U14 |
| [7]s | V14 |

**Lab3.4** Assume we have the pattern of E, H, N, T, U for seven-segment display as shown below. Try to implement the scrolling pre-stored pattern NTHUEE with the four seven-segment displays.

1. Design specification

功能: 顯示NTHUEE於七段顯示器上，並且如卷軸般移動。

輸入: globalclk (原本內建石英震盪器)

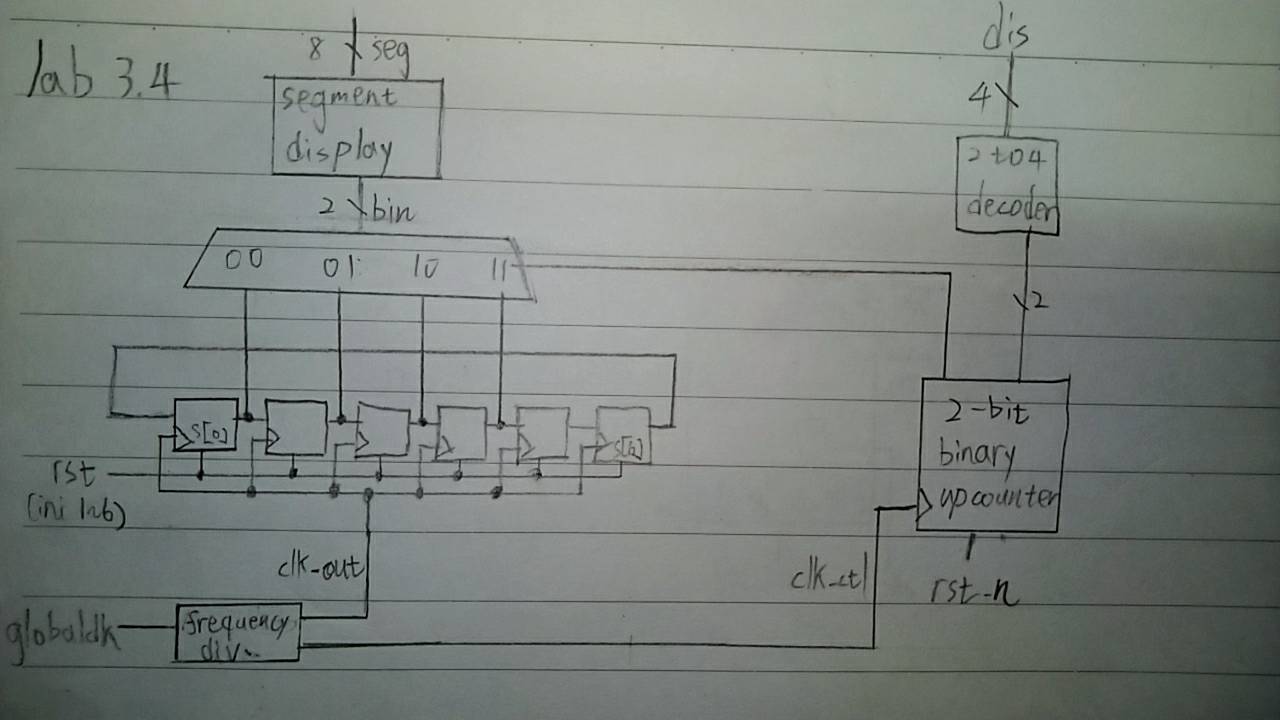
rst\_n (將儀器重置到初始狀態)

輸出: [3:0]dis (控制四個七段顯示器的顯示順序)

[7:0]seg (控制七段顯示器那些地方發亮)

2. Design implementation

首先先設置6\*3個暫存器，這些暫存器各存放從1~6六個數字，這些數字各對應NTHUEE六個字母。當clk\_out(1hz)的訊號傳送到暫存器們後，暫存器的數字就會被傳到下一個暫存器。其中只有前面四個暫存器的數字會被傳到多工器中。而旁邊會有一個計數器，由clk\_ctl(約1us)控制，每過約1us就會向上加一次，輪流選擇四個暫存器中一個數字輸出時，同時選擇對應的四個七段顯示器中其中一個。暫存器數字會被傳輸到display decoder中，而累加器的數字則直接輸出，控制哪個英文字母要配到哪個顯示器上顯示。



3. I/O pin assignment

|  |  |  |
| --- | --- | --- |
|  | Verilog變數名稱 | 輸入/輸出腳位 |
| 輸入 | globalclk | W5 |
| rst\_n | V17 |
| 輸出(display) | [0]seg | V7 |
| [1]seg | U7 |
| [2]seg | V5 |
| [3]seg | U5 |
| [4]seg | V8 |
| [5]seg | U8 |
| [6]seg | W6 |
| [7]seg | W7 |
| 輸出 | [0]dis | U2 |
| [1]dis | U4 |
| [2]dis | V4 |
| [3]dis | W4 |

Discussion & Conclusion:

從這一次的Lab開始，就已經開始感覺到難度的增加，也開始體會到了.v檔要分開來打，module不能放在一起，然後.v檔要記得複製到source裡面，而不是單單只add到source裡面，因為沒有複製過來，只要更動原文件，其他的project也會影響到。然後這一次開始lab會比較複雜，我也開始體會到軟體語言和硬體語言的區別，有些在C語言上不受限制，會因為硬體設計上的限制，使得verilog出現error，而且error時常講的不清不楚，找到bug都會用掉很長時間。也因此先畫圖很重要。