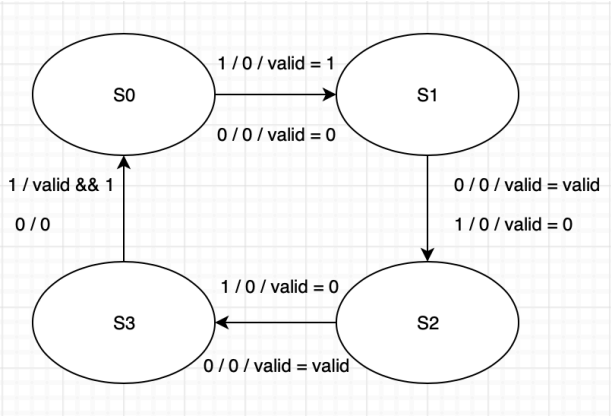
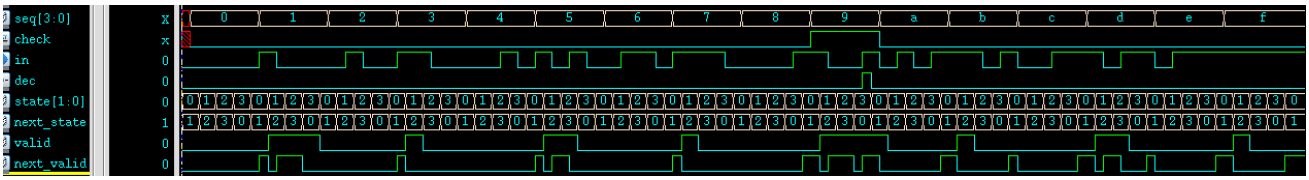
Lab4 Team6 Report

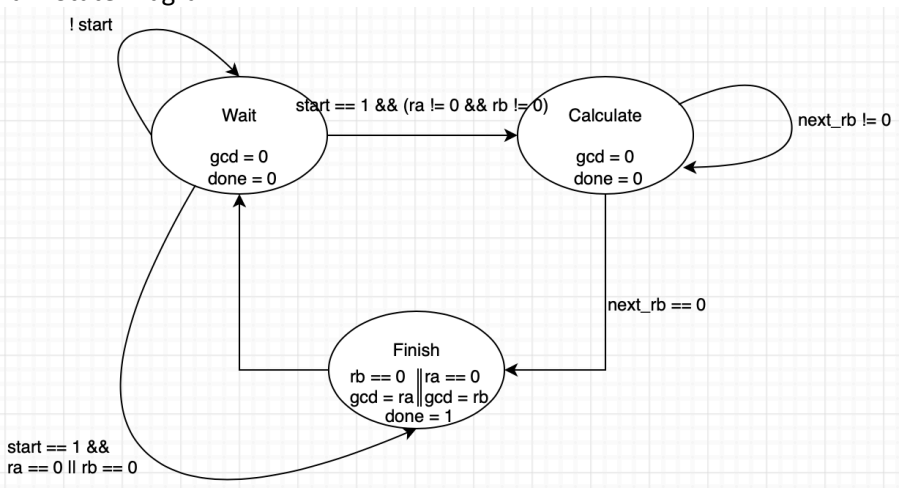
1. Mealy Machine Sequence Detector
    a. State Diagram:



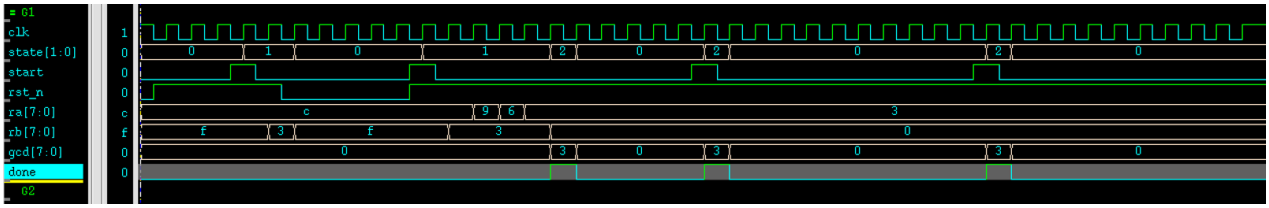
    b. Design Explanation:
        i. 用一個 D-flip-flop 儲存 state 和 valid 信號的變化。
        ii. 信號 state 記錄檢查到第幾個 bit，valid 記錄目前 sequence 的正確性。
        iii. 在 state S3 時，如果 valid 是 1，則輸出 1；其他時候均輸出 0。
    c. Testing;
        i. 我將所有 4 bit 字串的組合輸入測試，並檢查是否只有一組答案是 1。
        ii. Waveform:



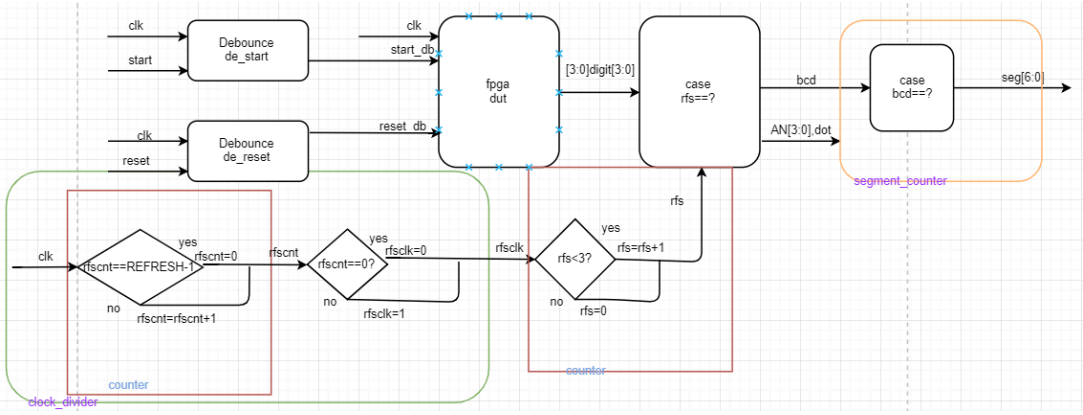
2. Greatest Common Divisor:
    a. State Diagram:



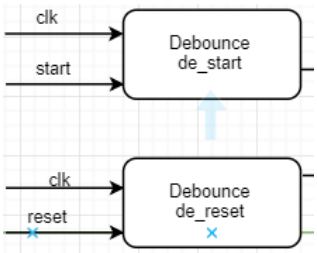
b. Design Explanation:
   i. 用 register ‘state’儲存現在 state，用 ra 和 rb 儲存每次計算的結果。
   ii. State WAIT:
      1. 如果還沒開始（start == 0），則停留在 state WAIT。
      2. 如果開始了，但其中一個數等於零，則跳到 state FINISH。
      3. 如果開始了，又非上述情況，則跳到 state CAL。
   iii. State CAL:
      1. 如果 ra 大於 rb，則更新 ra 為 ra-rb；反之則更新 rb 為 rb-ra。
      2. 如果計算結果其一為零，則跳到 state FINISH。
      3. 反之則停留在 state CAL，繼續計算。
   iv. State FINISH:
      1. 信號 gcd 的輸出為兩數中非零的值，若皆為零則輸出其一。
      2. 因為只維持一個 cycle，所以下個 state 是 state WAIT。
      3. 信號 done 只在這個時候輸出 1。
c. Testing:
   i. 我檢查了五種狀況：
      1. 信號 start 未拉起，是否會停留在原 state。
      2. 信號 rst_n 中途拉起，是否會強制回到 state WAIT。
      3. 一般的測資計算是否正確。
      4. 極端測資：(0, 0)計算是否正確。
      5. 極端測資：(1, x)（此處用 8）計算是否正確。
   ii. Waveform:



3. stopwatch FPGA
   a. Block Diagram + Explanation
      i. Top module



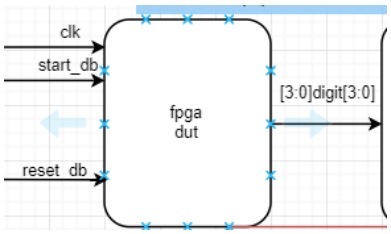
ii. Debounce：

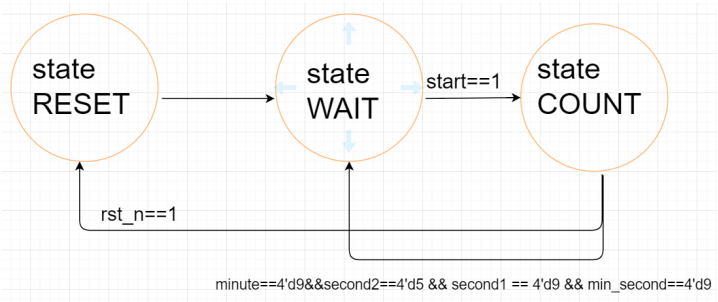


將輸入信號放進連續四個 flip-flop 裡面,待信號夠穩定（即四個 flip-flop 都儲存 1）再輸出 1.使用目的是為了過濾掉 pushbutton 的 bounce.

分別將兩個按鍵(start,reset)做 debounce,上次的 lab 是要 debounce 和 one_pulse 合起來做,這次並不要求做 one_pulse,因此就直接實作 debounce 做出 debounce 後的 reset 和 start.

iii. Fpga dut：



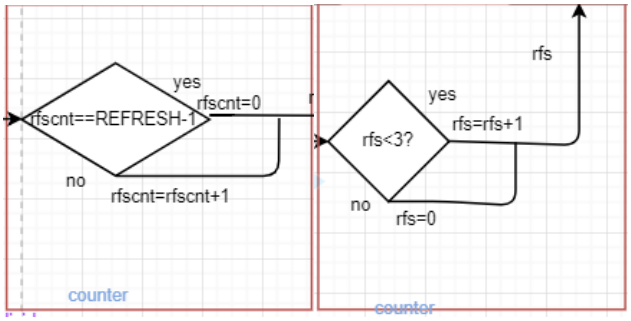


首先先處理 clk.因為我們的做法是將最小位數的時間(也就是 0.1 秒)用進位的方式來做出秒和分鐘,所以我們需要做個 counter,數 0.1 * (10*10^-9),也就是慢 10^7 倍才會在 fpga 板能夠準確的跳出秒數.我們將四個位數分開,分成 min_second,second1,second2,minute,second1 要在 min_second 為 9 才會進位,而 second2 要在 min_second 為 9 且 second1 為 9 才會進位,以此類推.我們用兩個 combinational always block 來處理 state 和給值,用 sequential always block 來做 dff .

State 的部分,分成 RESET,WAIT,COUNT,三個部分.一開始為 RESET,並維持一個 cycle,進到 WAIT.在 state WAIT,等到按下往上的按鈕(start==1)

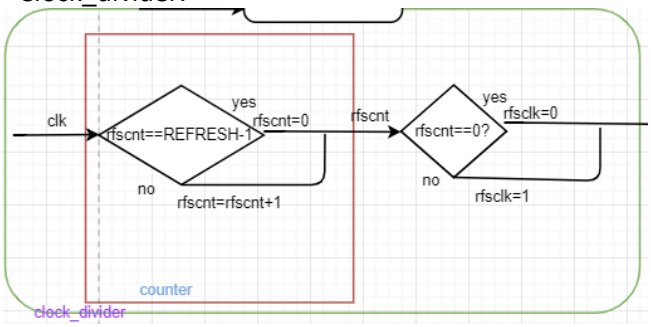
才進到 state COUNT.在 COUNT 的部分,如果按到 reset 的話,則進入 RESET,若秒數讀到 959.9,則回到 WAIT 等帶 start 按下重新開始.

iv. Counter:



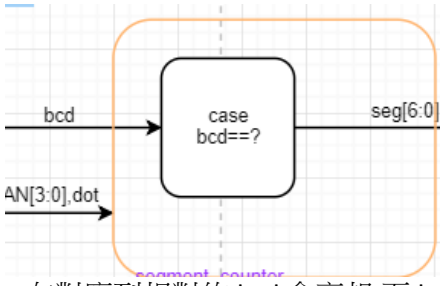
在 top module 有兩個 Counter,第一個是拿來調整 fpga 顯示頻率的(每 2^17 次 rfscnt 才為 1),第二個 counter 是拿已經調頻過的 rfsclk 來觀察現在的 rfs (An)為多少.

v. Clock_divider:



由 counter 和一個判斷條件組合,這樣出來的 rfsclk 會符合 我們要的結果:慢 2^17(REFRESH).

vi. Segment Converter：



segment 在對應到相對的 bcd 會亮起,而 bcd 的值是要先看現在的 rfs 是哪個值,再來把對應的 2 維陣列 digit(從前面的 fpga dut)給 bcd,最後用 bcd 的值來看現在是要對應哪個 segment.這次我們 segment 用數值的方式給值而不是用陣列的方式給值.

b. Testing:

一開始我們還沒有寫 debounce,就直接來試,用手機測的時間是不對的,我們以為是在 10^7 那邊出了問題,後來一直調數值都不對,最後抱著試一試的心態去寫了 debounce 再測一次就對了.我們才知道原來 debounce 會影響跳動頻率.

4. 心得:

- 學到了如何讓 verilog 上面跳動的頻率和實際上的時間對上,另外也更了解 fpga 的操作.另外也對 debounce 有更深入的了解.
- 學會在工作站上的操作，包括 ncverilog 指令，使用$display 顯示 debug message，sftp 指令。
- 學會不將除頻後的信號作為 clock 使用，但達到除頻效果的做法。