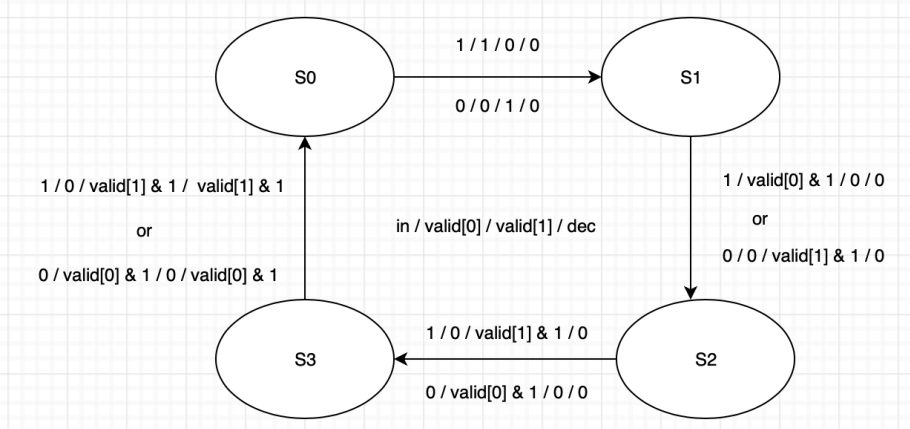
Lab5_Team6_Report

1. Mealy Sequence Detecter：（陳騰鴻）
    a. Design+Explanation
        i. State transition Diagram:

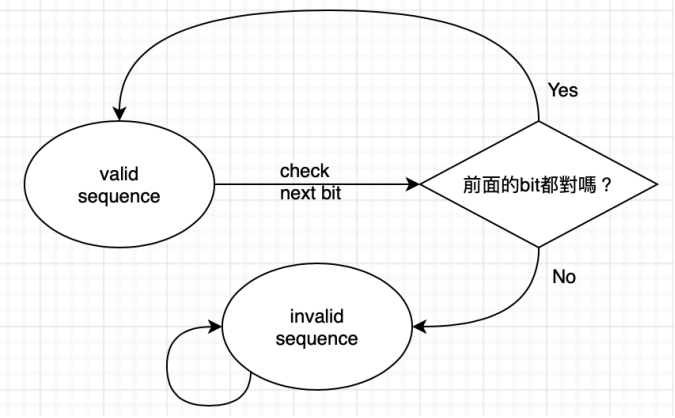


(1) 暫存器 valid[0] 和 valid[1] 分別記錄先前輸入是否可能為 1100 和 0011。

(2) 記錄方法：將先前輸入之對錯以 0 或 1 記錄下，再用布林邏輯ＡＮＤ加入每次結果。

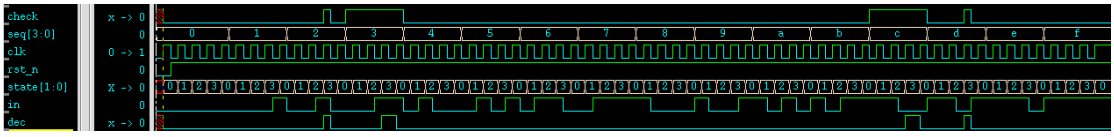
(3) 最後一個 bit 將兩組結果以布林邏輯ＯＲ取是否有其一為正確組合，再輸出。

        ii. Block Diagram:



        iii. Design Testing:

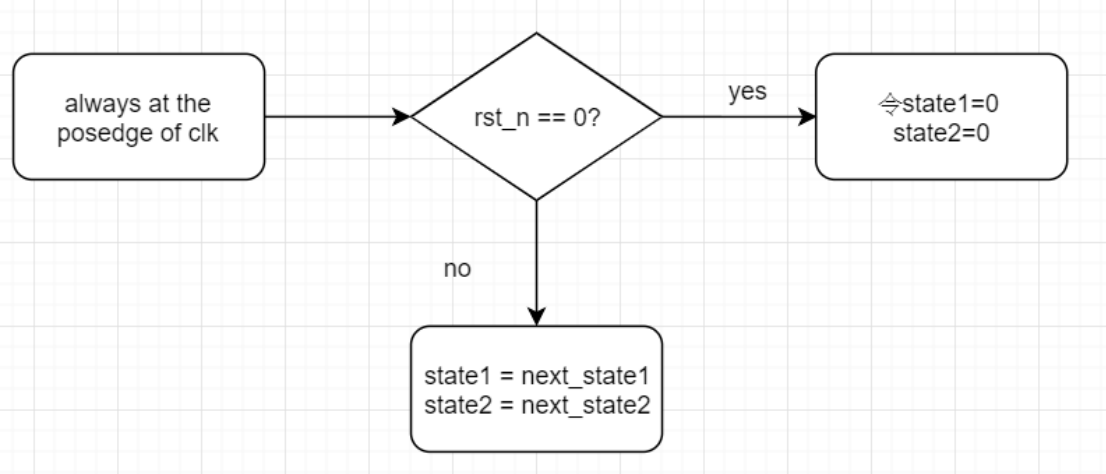
將所有可能組合輸入，並設置信號 check，使其在信號 dec 等於 1 或輸入為正確組合的時候拉起，方便觀看。



2. Sliding Window sequence detector (mealy machine)（王駿）

a. Diagram+ Explanation

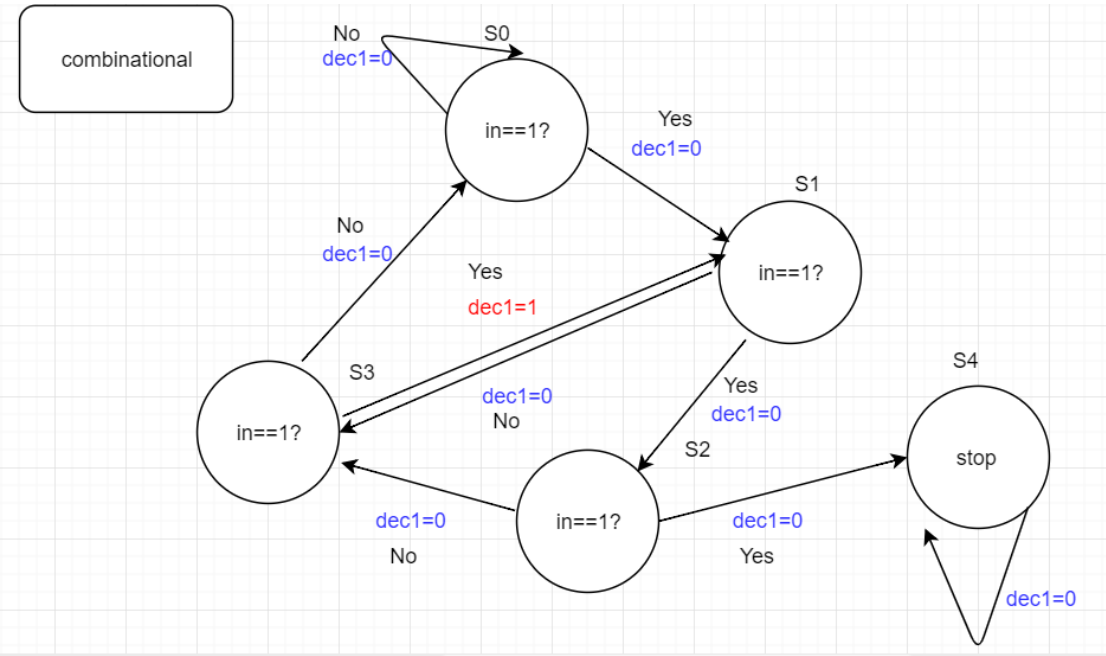
i. Sequential Circuit:



這邊的 State1 是拿來判斷 dec1,state2 拿來判斷 dec2.利用 sequential circuit 決定 state1 和 state2 什麼時候要 reset,什麼時候要被 next_state 給值.

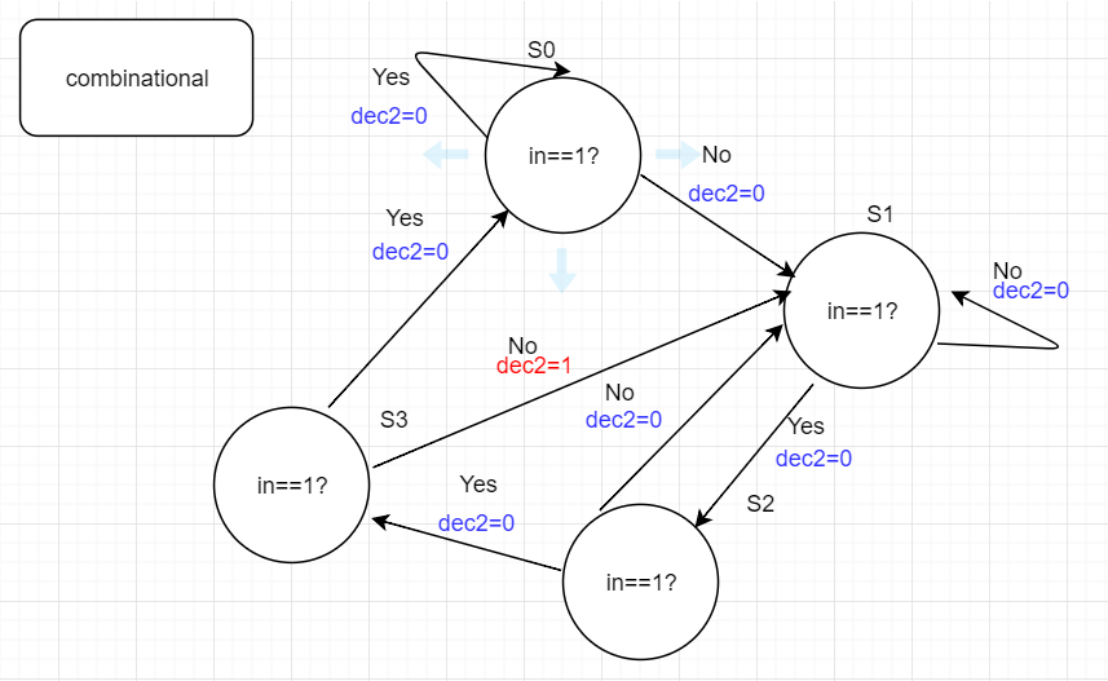
ii. State transition diagram:

這邊是state1的 transition diagram:



這邊需要特別解釋的是 S4.題目要求說當讀到連續三個 1,也就是依序經過 S0,S1,S2,S4.當進入 S4 時,接下來不管 in 為多少都不會影響 state 和 dec1.

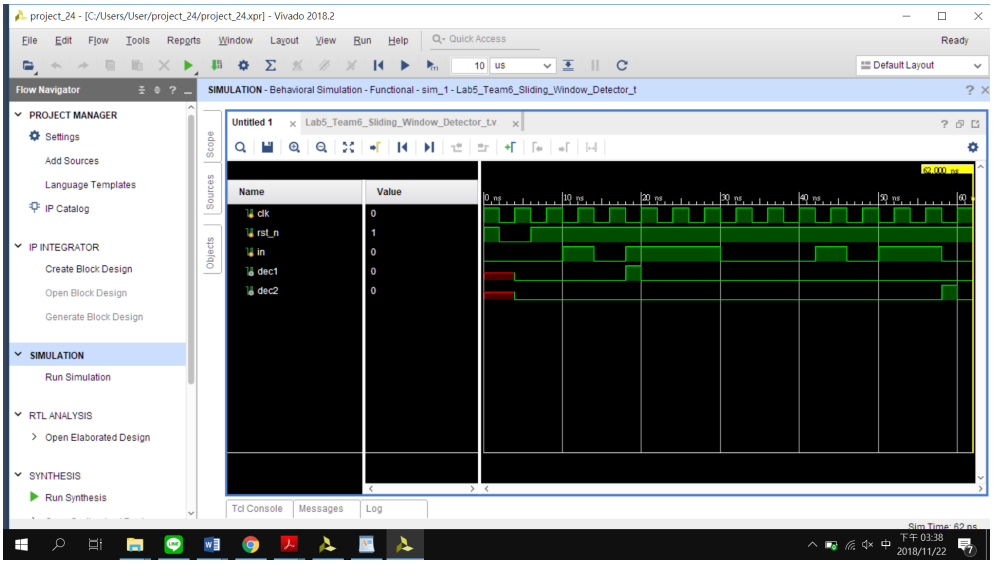
State2 的 transition diagram:



這邊就沒有任何限制,只要吃到 0110 字串,也就是依序經過 S0,S1,S2,S3,S1,dec2 就會直接變 1.那會接到 S1 是因為最後一個吃到的 0 可以當作下一個 0110 的第一個 0,所以才會進到 S1 而非 S0.

b.testing:

waveform:



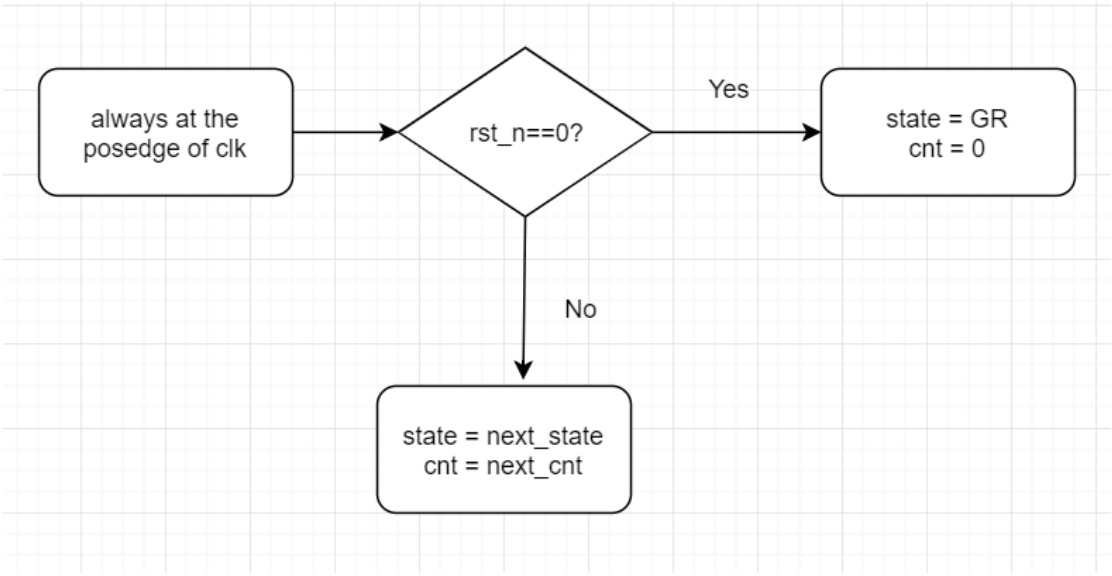
觀察這個圖,在 18 到 20s dec1 是立起來的,原因是吃到了 101 且前面沒有 111.後面 48 到 50s 有一個一樣的情況,但 dec1 沒有立起來,原因是在 30 秒他讀到了 111, state1 進到 S4,之後就會一直待在 S4,後來讀進來的 in 完全不會影響 dec1,而是讓 dec1 一直維持 0.

Dec2 是要偵測 0110,前半段的 010 和 01110 都沒有讓 dec2 立起來,直到後面的 0110 dec2 才順利立起.

3. Traffic light controller for highway (HW) and local road (LR) intersection（王駿）

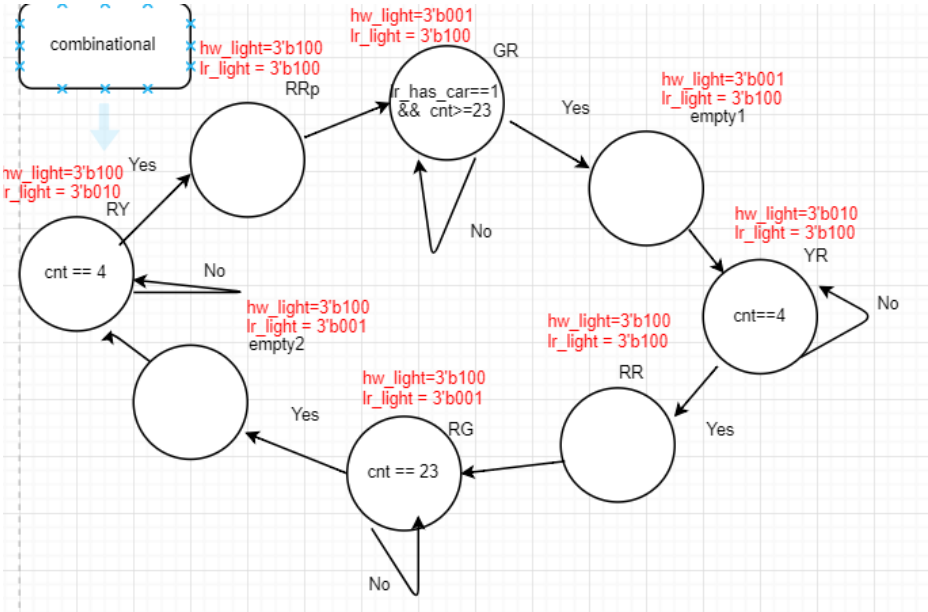
a. Diagram+ Explanation

i. Sequential Circuit:



利用 sequential circuit,決定當 rst_n==0 時,state 要初始為 GR,若開始則將 next_state 給 state.而 cnt 則是拿來當計數器,題目要求有些情況要 25cycle 或 5cycle 才能進到下一個 cycle,而 cnt 就是拿來判斷是否已經達到判斷 cycle 值

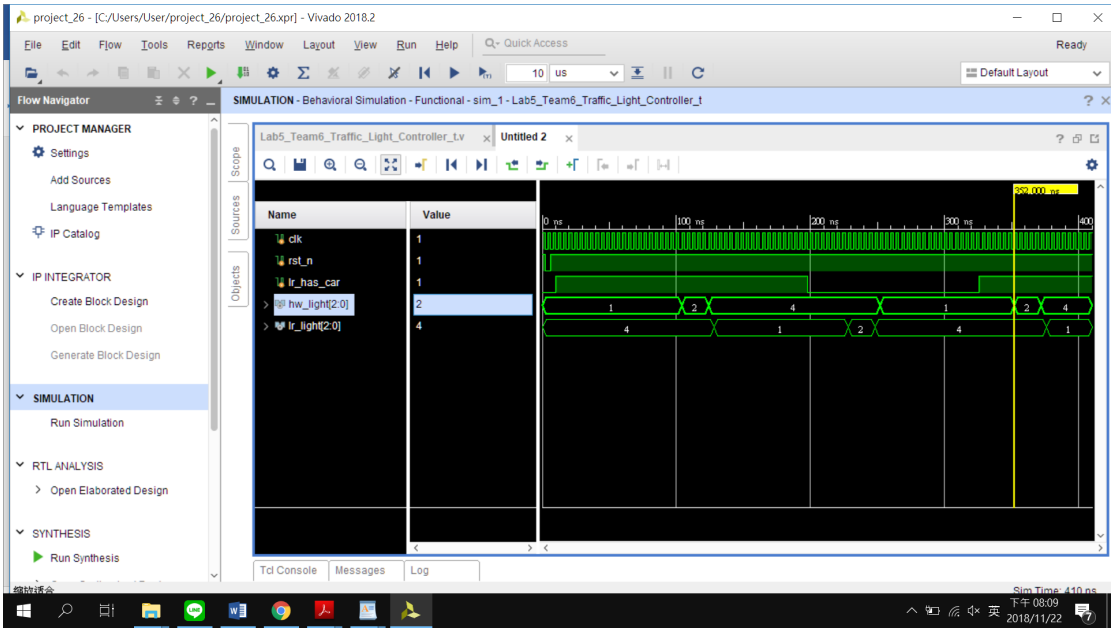
ii. State transition diagram:



在這次的 state transition,除了原本的六個 state,我還另外加了兩個 state,分別是 empty1 和 empty2 .加上這兩個 state 的原因是在 GR 接 YR 的地方還有 RG 接 RY 應該要停一個 cycle,而我的作法是直接跳進一個空的 state 讓他能多維持一個 cycle.至於為什麼要用 23 而不是 24,我自己也不太清楚,不過用 waveform 去對是要 23 沒錯,還麻煩助教幫忙解惑.

b.testing:

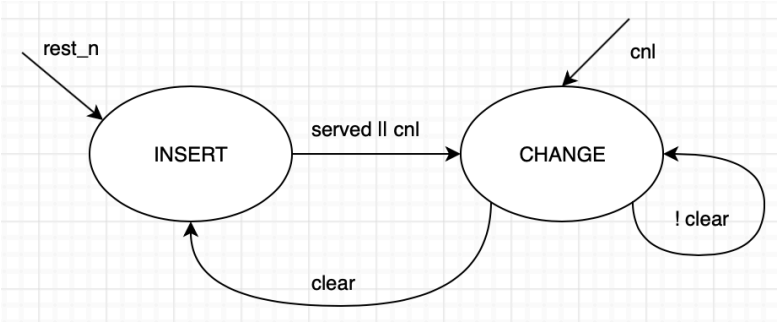
waveform:



在一開始一直有車子等待(lr_has_car==1),不過要等到 25 cycles 之後才能轉黃燈.而後半段車子又來的時候,不管是什麼時候開始有車子開進來,都是 25 cycles 就會亮黃燈.

4. Vending Machine

i. State transition Diagram:



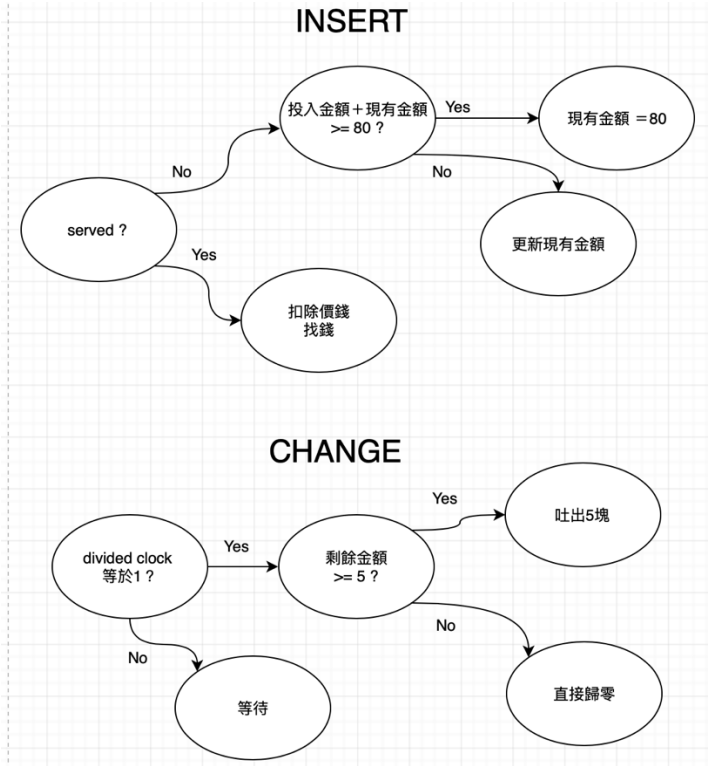
(1) INSERT 為處理投幣和等待飲料選擇，CHANGE 則找錢或退錢。

(2) 信號 cnl 即 spec 中的 cancel 按鍵，cnl 拉起即到 CHANGE 狀態。

(3) 信號 served 拉起表示成功買到飲料，接著便要找錢。

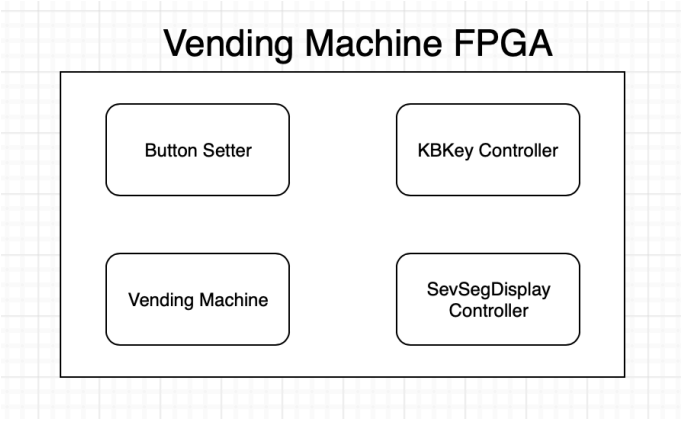
(4) 信號 clear 指示現在機器中是否有剩下錢，有錢則繼續吐錢，沒錢了就回到 INSERT。

ii. Block Diagram:

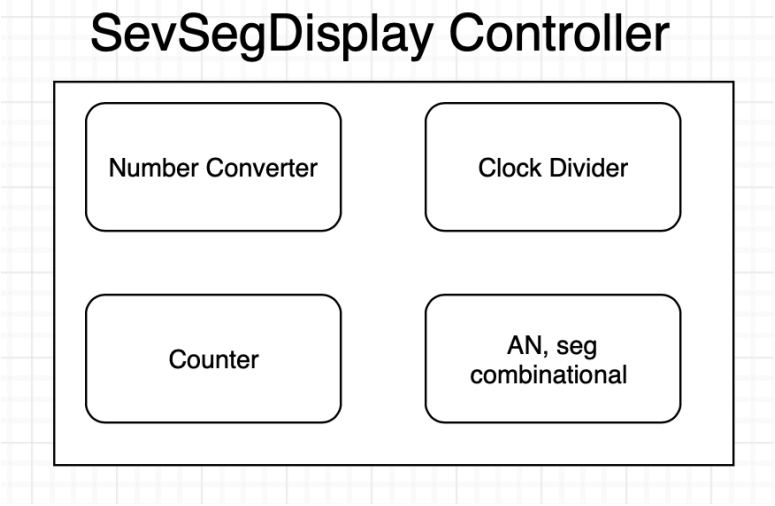


- Divided clock 利用一個 counter 製造出比板子上慢 10^8 倍的 clock，已達到題目每秒吐錢的要求。

iii. Design+Explanation:



- Button Setter: 將輸入的五個按鍵信號導入五組 debounce 和 onepulse，製造穩定信號輸入。
- KBKey Controller: 從模組 Keyboard decoder 取出需要的五個鍵盤按鍵信號作為輸入。
- Vending Machine: 控制輸入和輸出結果。
- SevSegDisplay Controller: 從 Vending Machine 拿到輸出結果，將金額轉換成七段顯示器顯示數字所需之信號。



- Number Converter: 將輸入的數字轉為個位數和十位數。作法：用 8 個 if-else 判斷數字在哪兩個十的倍數之間，扣掉十位數部分便得個位數。
- Clock Divider 和 Counter: 製造一個以慢 system clock 2^17 倍的刷新頻率，用以刷新七段顯示器的兩位數。
- AN 和 seg 的 assignment: 用 if-else 或 case 製造一對一對應，例如當刷新到個位數，則輸入個位數信號；當個位數信號為 3，其相對應的 seg 應該量哪些等等。

心得:（王駿）

1. 對 state transition 更了解
2. 更熟悉用 waveform debug

心得：（陳騰鴻）

1. 學會用 compiler 的 optimization 的結果猜測自己哪些東西沒有寫好。
2. 更熟悉 Clock Divider 信號的產生與效果。
3. 學會 coding 小技巧：bitwise operation 和 module array instantiation。
4. 知道下次要提早開始寫作業。
5. 了解到信號在 initial block 在 waveform 中的長相。