Lab5 Report

Team25

組長：蔡登瑞、組員：蔡政諺

Contents

- **Designs**

1. Mealy Sequence Detector

這題跟上次 hw4 的第一題類似，所以我們用相同的想法去寫這題，不過這次的 Detector 要是判斷兩個 sequence。這一次，我們使用了 9 個 state，並且依照題目需求，每四個 clock cycle 一個循環，並在每個循環檢驗 in 是否為 1100 或是 0011。

S0：初始的 state，作為每一次循環的第一個 state。

若是偵測到的 sequence 為 110X，則 state transition 便是 S0->S1->S2->S3，並在 S3 的時候，若是 in 為 0，dec 輸出 1；in 為 1，dec 輸出 0。

若是偵測到的 sequence 為 001X，則 state transition 便是 S0->S4->S5->S6，並在 S6 的時候，若是 in 為 1，dec 輸出 1；in 為 0，dec 輸出 0。

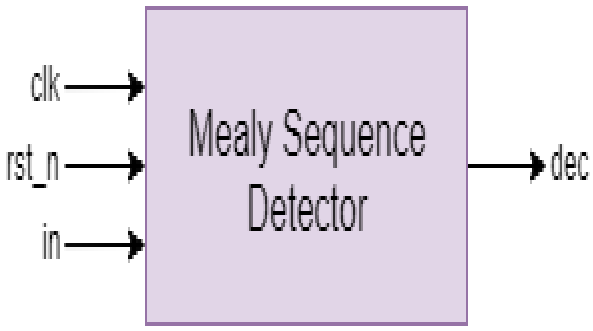
然而，N2 跟 N3 是用來當偵測到的 sequence 非 1100 或是 0011 時的 default，跑完這一次的循環。

若是在 S1 的時候 in 為 0(在 S4 的時候 in 為 1)，也就是說 sequence 若是 10XX(01XX)，則 state transition 會變成 S0->S1(S4)->N2->N3。

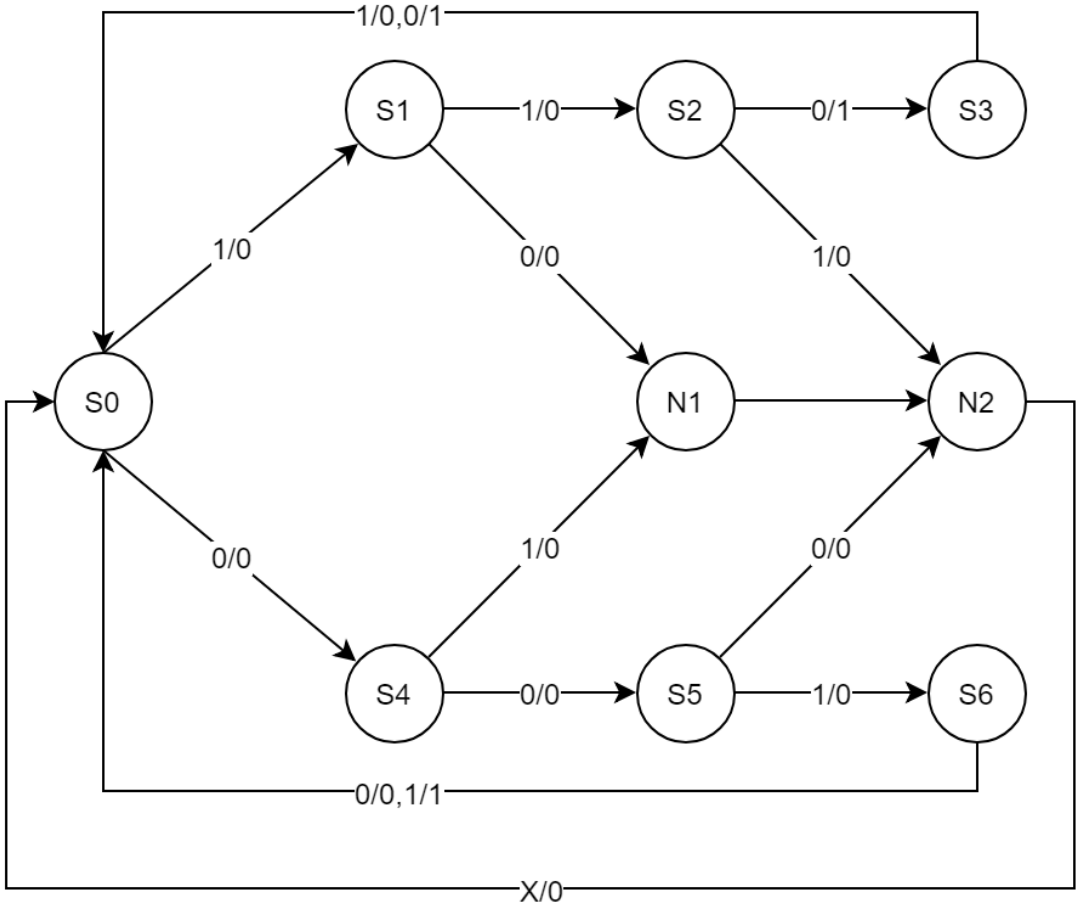
若是在 S2 的時候 in 為 1(在 S5 的時候 in 為 0)，也就是說 sequence 若是 111X(000X)，則 state transition 會變成 S0->S1(S4)->S2(S5)->N3。

以上 9 個 state，只有在 S3 和 S6 的時候 dec 會根據 in 的值輸出 0 或是 1，其他 7 個 state 無論 in 的值為何 dec 都只會輸出 0。

Block diagram



State Transition Diagram



2. Sliding Window Detector

由於這題沒有固定幾個 clock cycle 為一個循環，也就是說，無法使用固定長度的 state 像是 S0->S1->S2->S3->S0，這個樣子去實行這題。再加上兩個 dec 根據不同長度的 sequence 來輸出，因此我們 dec1 跟 dec2 的 state 分開來寫。

在 dec1 的部分，dec1 會輸出 1 有一個前提是不能出現過連續三個 1，否則無論 sequence 為何都只會輸出 0，因此我們設計了 5 個 state。

STATE：

F0：判斷 sequence 的第一個，若是 1 進入到 F1，若是 0 則繼續待在 F0。

F1：判斷 sequence 的第二個，若是 1 進入到 F3，若是 0 則進入到 F2。

F2：判斷 sequence 的第三個，若是 1 進入到 S1(第三種)，若是 0 則進入到 S0(第一、二種)。

F3：判斷 sequence 的第三個，若是 1 進入到 S4(第五種)，若是 0 則進入到 S2(第四種)。

F4：若是進入這個 state，代表說曾經有出現過連續三次的 111，因此無論 in 為 0 或 1，state 都會待在 F4。

以上 5 個 state，只有在 F2 跑到 F1 的時候，dec1 才會輸出 1，其他時候 dec1 只會輸出 0。

考慮到的 sequence 種類：

1、...101...，一次 101 (F0->F1->F2)

2、...101...101...101...，多次 101 並且不連續 (重複 F0->F1->F2)

3、...101010101...，多次 101 並且連續 (F0->F1->F2->F1->F2.....)

4、...0110...，連續兩次的 1 (F0->F1->F3->F2.....)

5、...01110...，連續三次的 1 (F0->F1->F3->F4->F4.....)

6、...01...10...，連續三次以上的 1 (F0->F1->F3->F4->F4.....)

在 dec2 的部分，若是 sequence 是 0110，dec2 就會輸出 1，沒有其他條件限制，因此我們設計了 4 個 state。

STATE：

S0：判斷 sequence 的第一個，若是 1 繼續待在 S0，若是 0 則進入到 S1。

S1：判斷 sequence 的第二個，若是 1 進入到 S2，若是 0 則繼續待在 S1。

S2：判斷 sequence 的第三個，若是 1 進入到 S3，若是 0 則進入到 S2。

S3：判斷 sequence 的第四個，若是 1 進入到 S0，若是 0 則進入到 S1。

以上 4 個 state，只有在 S3 跑到 S1 的時候，dec2 才會輸出 1，其他時候 dec2 只會輸出 0。

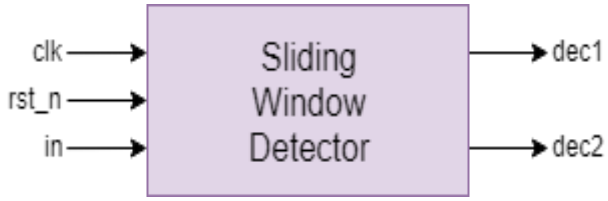
考慮到的 sequence 種類：

1、...0110...，一次 0110 (S0->S1->S2->S3)

2、...0110...0110，多次 0110 並且不連續 (重複 S0->S1->S2->S3)

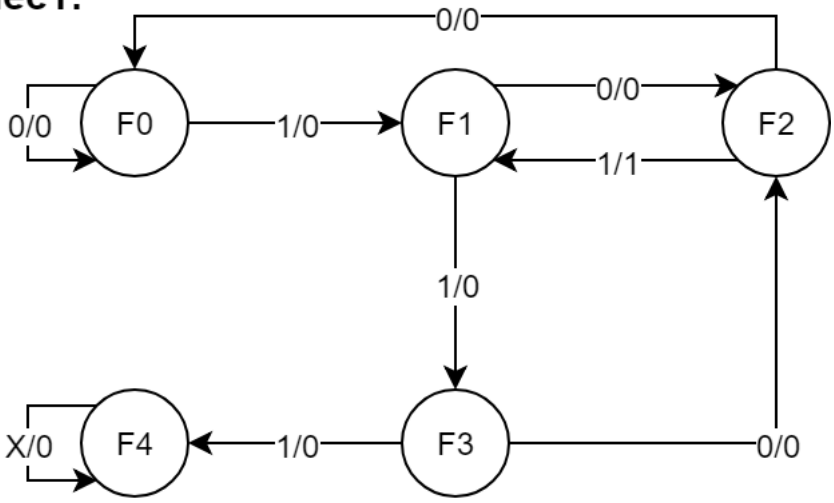
3、...0110110110...，多次 0110 並且連續 (S0->S1->S2->S3->S1->S2->S3.....)

Block diagram

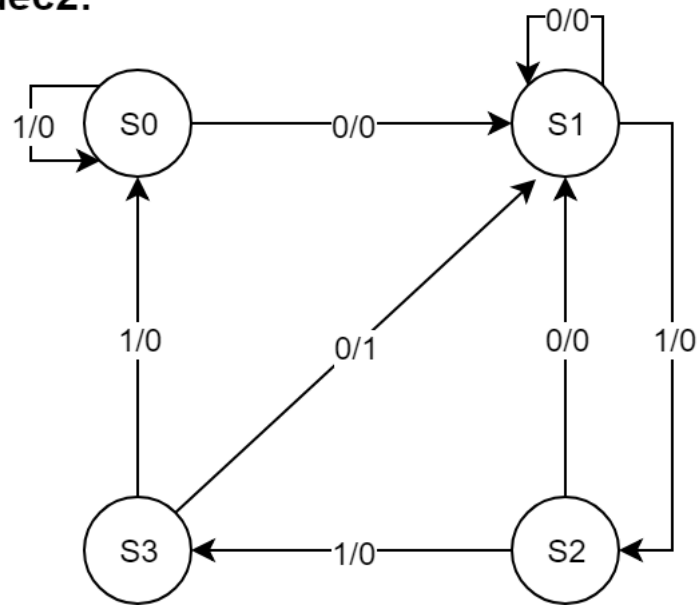


State Transition diagram

dec1:



dec2:



3. Traffic Light Controller

這題我們依照 pdf 上面提示的 state transition 去實作，共使用到了 6 個 state。並且使用 2 個 counter 分別計算綠燈跟黃燈的時間。

STATE：

HGLR：主幹道綠燈，支幹道紅燈。維持超過 25 個 clock cycles，並且支幹道上有車(lr_has_car == 1)，state 轉換到 HYLR。

HYLR：主幹道黃燈，支幹道紅燈。維持亮 5 個 clock cycles 時，state 轉換到 HRLR1。

HRLR1：主、支幹道皆紅燈。維持 1 個 clock cycle，state 轉換到 HRLG。

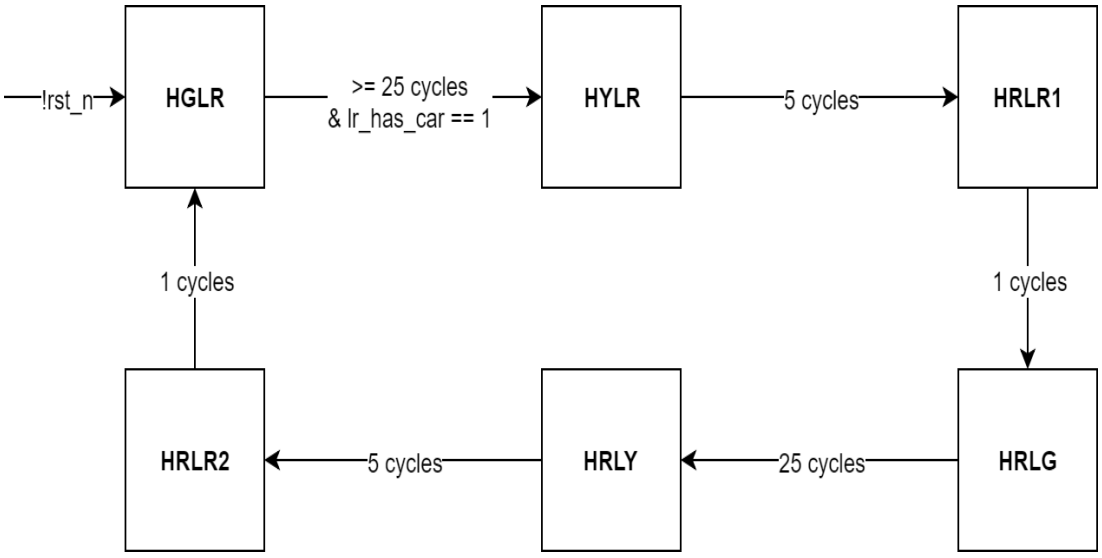
HRLG：主幹道紅燈，支幹道綠燈。維持 25 個 clock cycles 後，state 轉換到 HRLY

HRLY：主幹道紅燈，支幹道黃燈。維持亮 5 個 clock cycles 時，state 轉換到 HRLR2。

HRLR2：主、支幹道皆紅燈。維持 1 個 clock cycle，state 轉換到 HGLR。

Block diagram

State Transition diagram



4. Vending Machine

首先我們用 Basic 題中使用的 Keyboard_Decoder 處理 PS2_DATA 和 PS2_CLK，再使用 Output 的 key_down，分別算出 a、s、d、f 四個鍵的 Key Code 分別是多少，則 key_down[KEY_CODE[i]](0≦i≦3)就會是我們要的按鍵的訊號。由於這題的 Input 除了 clk 以外全都是按鍵、按鈕式的，所以我們把這些 Input 都接上 Debounce 和 Onepulse。處理完 Input 之後，就可以接到我們設計的 Finite State Machine－Vending_Machine 裡。

在 Vending_Machine 中，總共有 3 個 State：INSERT、RETURN、FINISH。

INSERT：投錢的 State。如果投入的金額會導致總額超過 80 元，我們就令金額為 80 元。如果按下 cancel(而且已經有投錢進去)，就會進入 RETURN；或是按下買飲料(而且金額足夠買選定的飲料)，就會扣錢並且進入 RETURN。

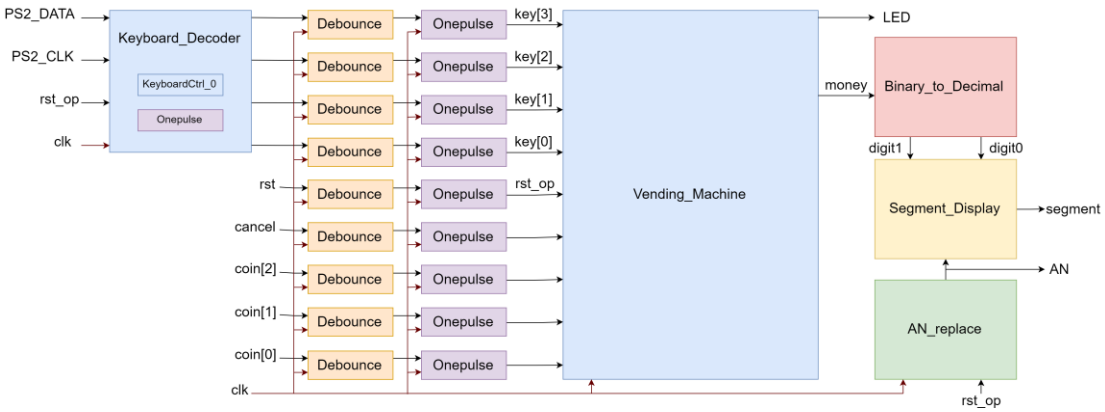
RETURN：退錢的 State。用 Counter 計算 10^8 個 Clock cycle(10ns*10^8=1sec)，每次數到 10^8 時就將 count 歸零，並且退 5 塊錢，但如果已經沒錢可以退了(money == 8'd0)，就不會再扣錢，並且下一個 State 就會進入 FINISH。

FINISH：結束的 State，經過一個 Clock cycle 後會回到 INSERT。

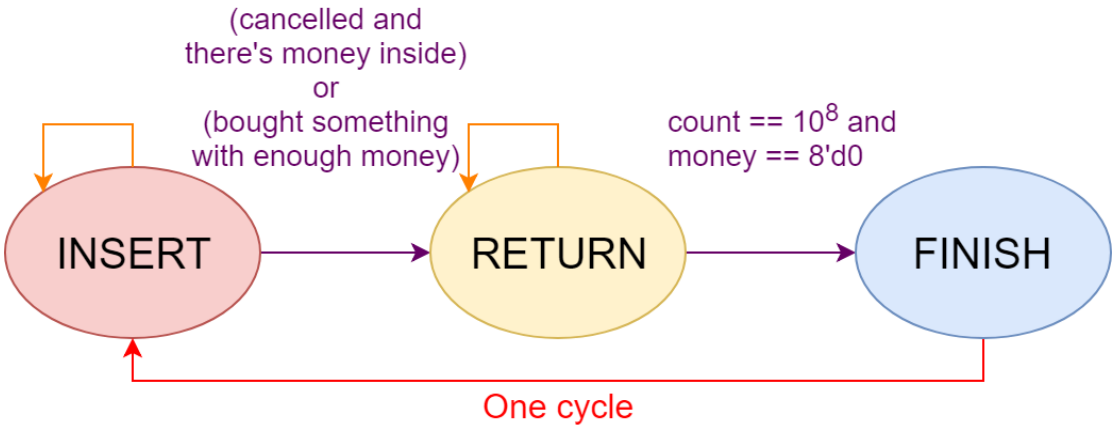
我們可以從 Vending_Machine 得到兩個 Output－LED[3:0]和 money[7:0]。LED 就是之後四種飲料分別亮的燈，money 則是我們的販賣機內現在有多少錢。因為在 7-segment 上，要把 money 分成十位數(digit1[3:0])和個位數(digit0[3:0])，所以我寫了一個 Binary_to_Decimal 的 Module，裡面就是用 case 條列 money 為多少時，digit1、digit0 分別要為多少。

算出 digit1、digit0 之後，就可以接到 Segment_Display，用 AN_replace 算出來的 AN[3:0]，配合 digit1、digit0 計算 segment[7:0]於不同 AN 時應為多少。

Block diagram



State Transition diagram



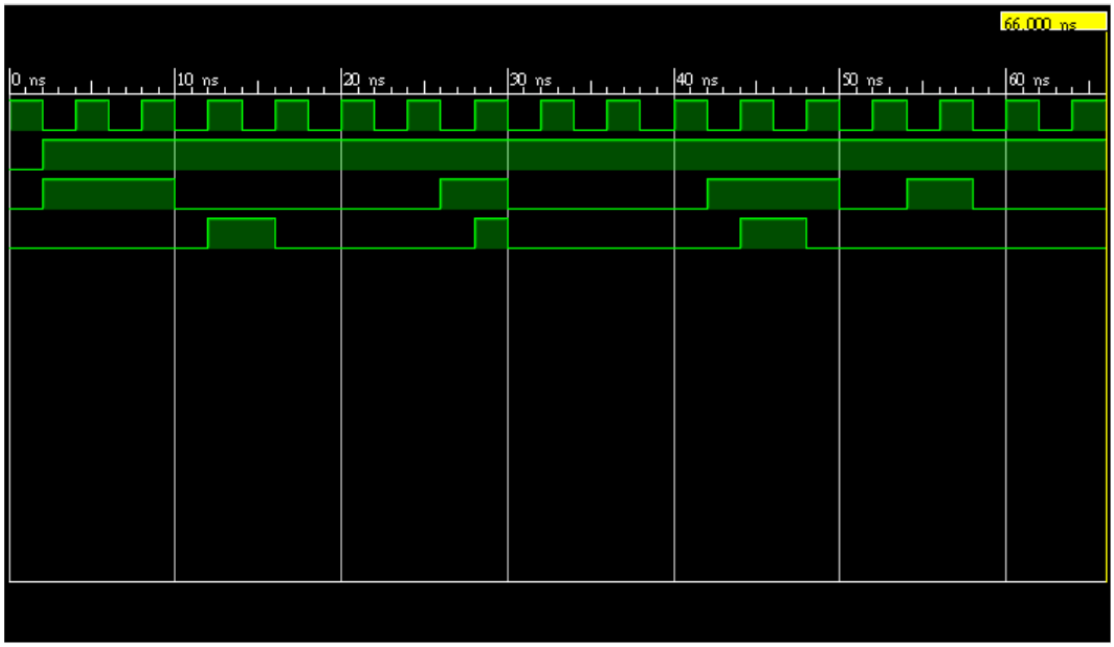
- **How we test our designs?**

1. Mealy Sequence Detector

這題我們是參照作業 pdf 所提供的波形圖去寫 testbench。若 simulation 結果與 pdf 中的圖形相同，就表示 design 無誤。

由上至下的 Signal 依序為 clk、rst_n、in、dec。

Waveform

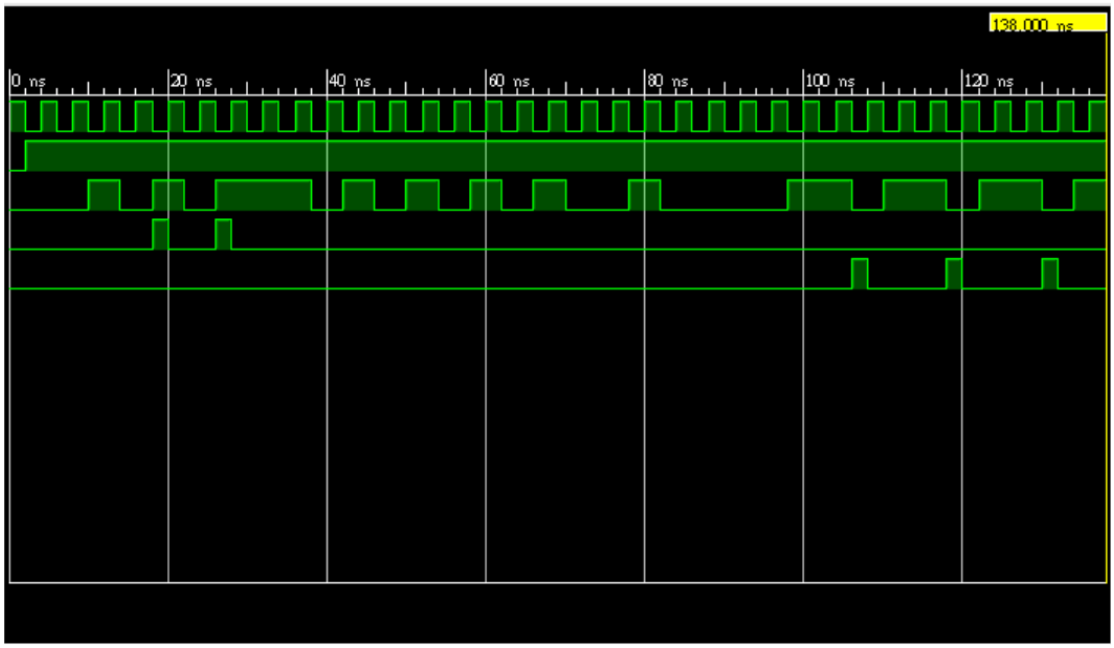


2. Sliding Window Detector

這題我們也是參照作業 pdf 所提供的波形圖去寫 testbench。Reset 後的 17 個 Clock cycle 用來測試 Dec1，再接著的 17 個 Clock cycle 用來測試 Dec2。若 simulation 結果與 pdf 中的圖形相同，就表示 design 無誤。

由上至下的 Signal 依序為 clk、rst_n、in、dec1、dec2。

Waveform

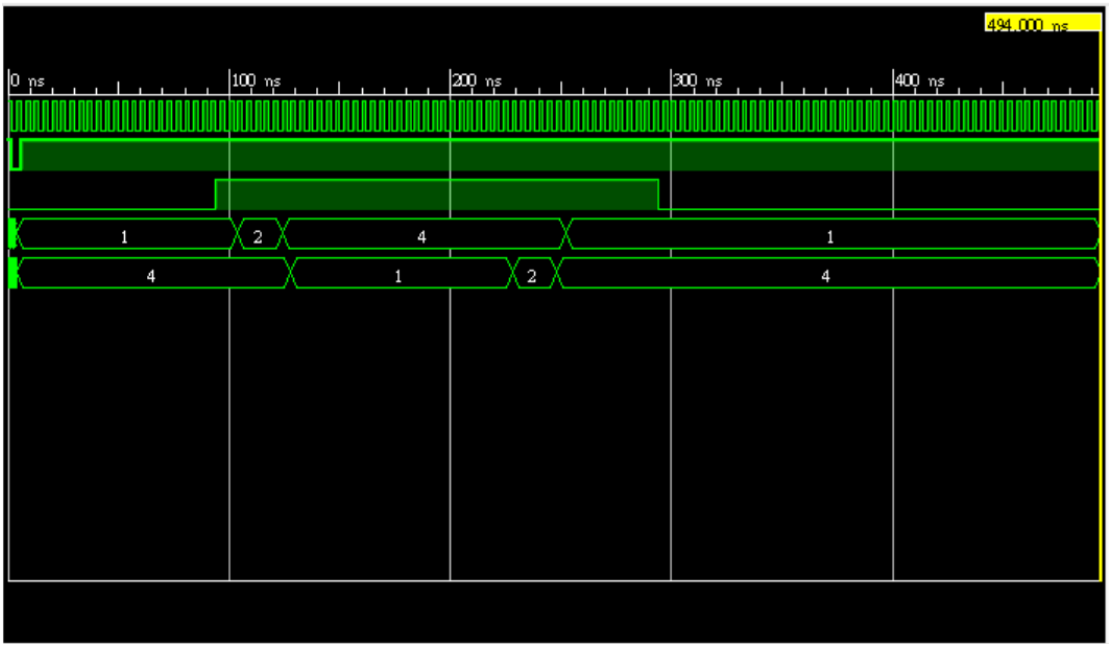


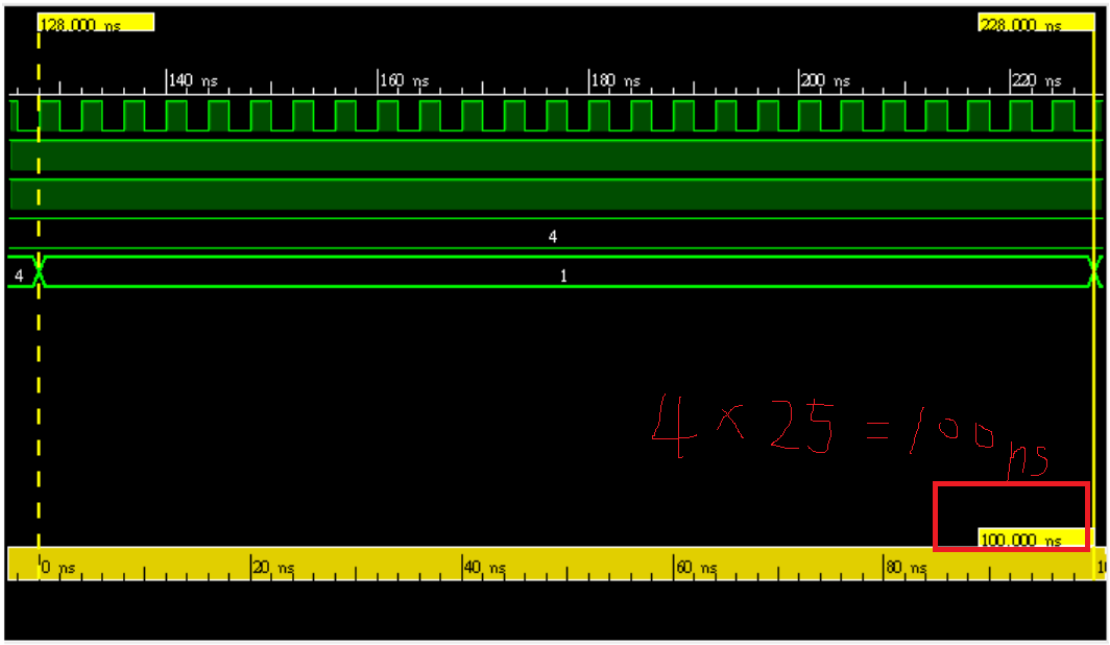
3. Traffic Light Controller

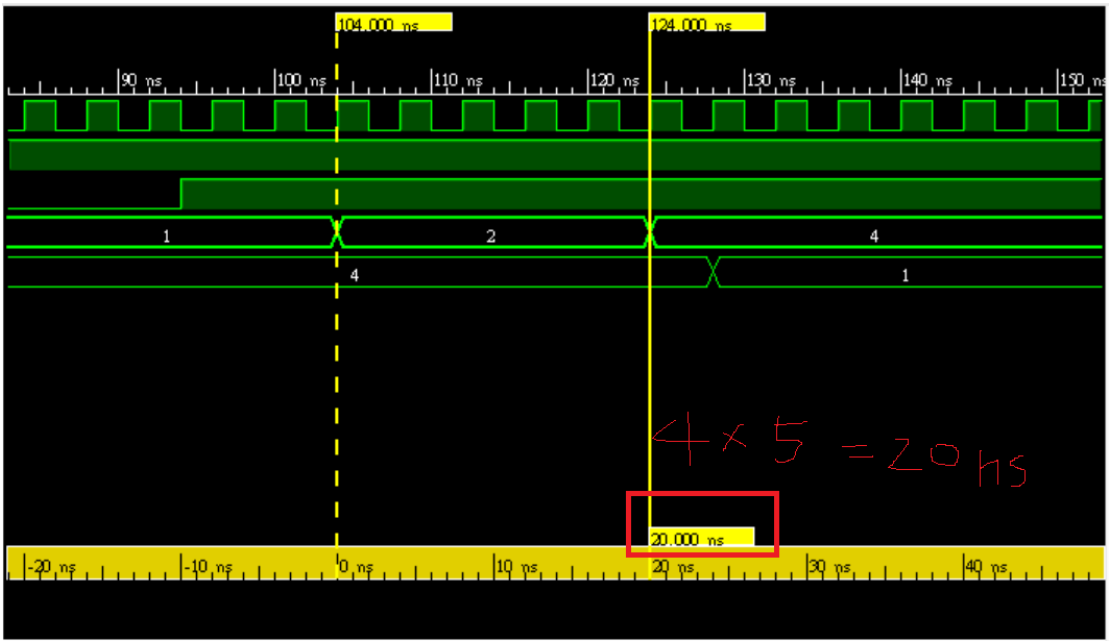
這題我們讓整個 state transition 跑過一次，也就是說主幹道綠燈、黃燈、紅燈，支幹道綠燈、黃燈、紅燈，主幹道綠燈，總共七次的 state 變換。主要是測試綠燈、黃燈、紅燈維持的 cycles 是不是題目要的(25、5、1 cycles)。再來就是測試說如果支幹道都沒有車輛，那主幹道是否會一直是綠燈。

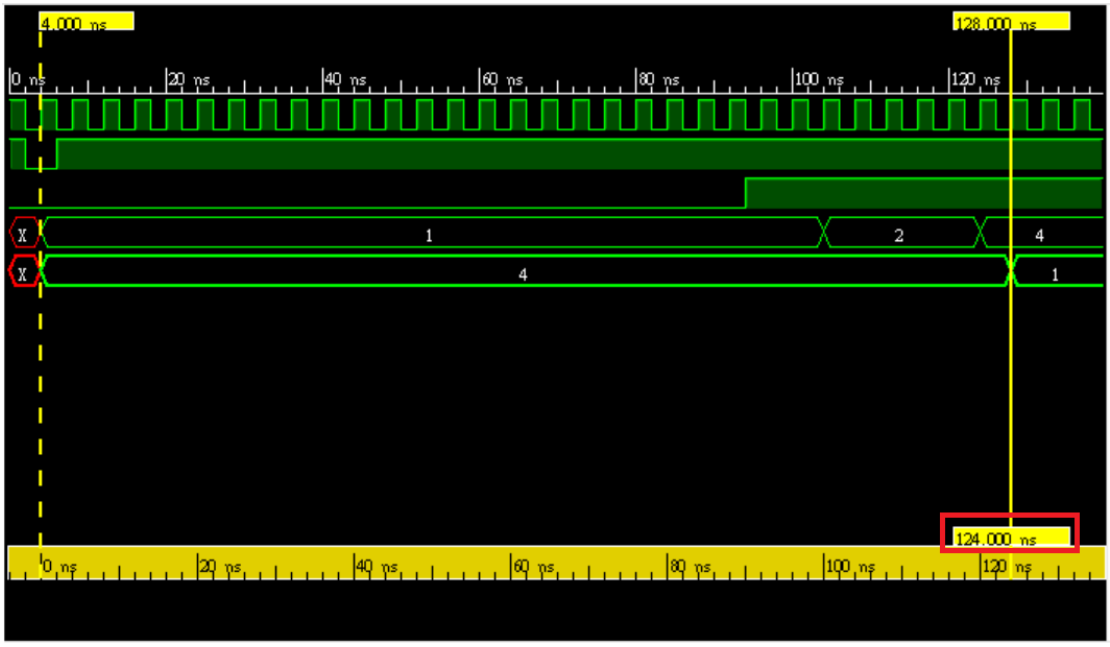
由上至下的 Signal 依序為 clk、rst_n、lr_has_car、hw_lgiht、lr_light。

Waveform






104.000 ns
124.000 ns
90 ns
100 ns
110 ns
120 ns
130 ns
140 ns
150 ns
1
2
4
4
1
4 x 5 = 20 ns
20.000 ns
-20 ns
-10 ns
0 ns
10 ns
20 ns
30 ns
40 ns


4.000 ns
128.000 ns
0 ns
20 ns
40 ns
60 ns
80 ns
100 ns
120 ns
X
1
2
4
X
4
1
124.000 ns
0 ns
20 ns
40 ns
60 ns
80 ns
100 ns
120 ns

- **What we have learned from Lab5?**

這次的 Lab 我們學到了怎麼以 Keyboard 作為 Input 使用，其中也學會怎麼設定 IP。這次我們覺得 Basic question 的難度其實比 Advanced question 的難度更高，相對地也學到很多－除了 Keyboard control 之外，我們學會使用音效模組，能讓 FPGA 發出不同音高的聲音，或是演奏一段樂句。相信這些技能對我們 Final qroject 的實作都是十分有幫助的。

Contribution List

- 蔡登瑞
  - 畫 state transition diagram
- 蔡政諺
  - 實作 FPGA
- 共同完成
  - 寫 code
  - 寫 testbench
  - 畫 block diagram
  - 寫 report