**0819823 陳子祈 HW2**

**A. BCD-to-Binary Converter (50%)**

**i. 以採BCD編碼之十進位數值 857為例，演示此演算法如何產生對應之二進位數字；在每個步驟中顯示出暫存器A與暫存器B之內容。此外，並說明為何此演算法是正確的。**

Illustrate the algorithm starting with the BCD number 857. Showing A and B register in each step. Moreover, describe the reason why this algorithm is correct.

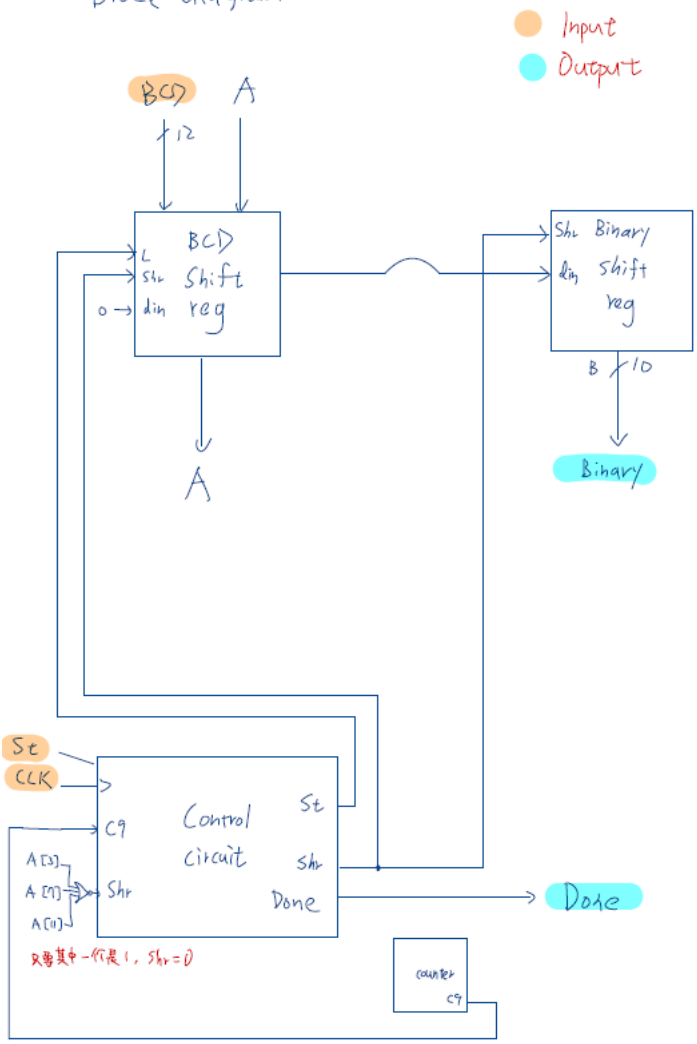

這個演算法的核心概念就是如果1右移到下一個digit，則這個digit要-3。可以反過來想，假設A進位到下一個digit變成B，本來應該是B=A*2，但是因為進位上16 base(4位2bit組成)比10 base多6，所以應該是B=A*2+6，那反過來看，B右移到下一個digit變成A，A=(B-6)/2=B/2-3，因此證明此演算法是對的。

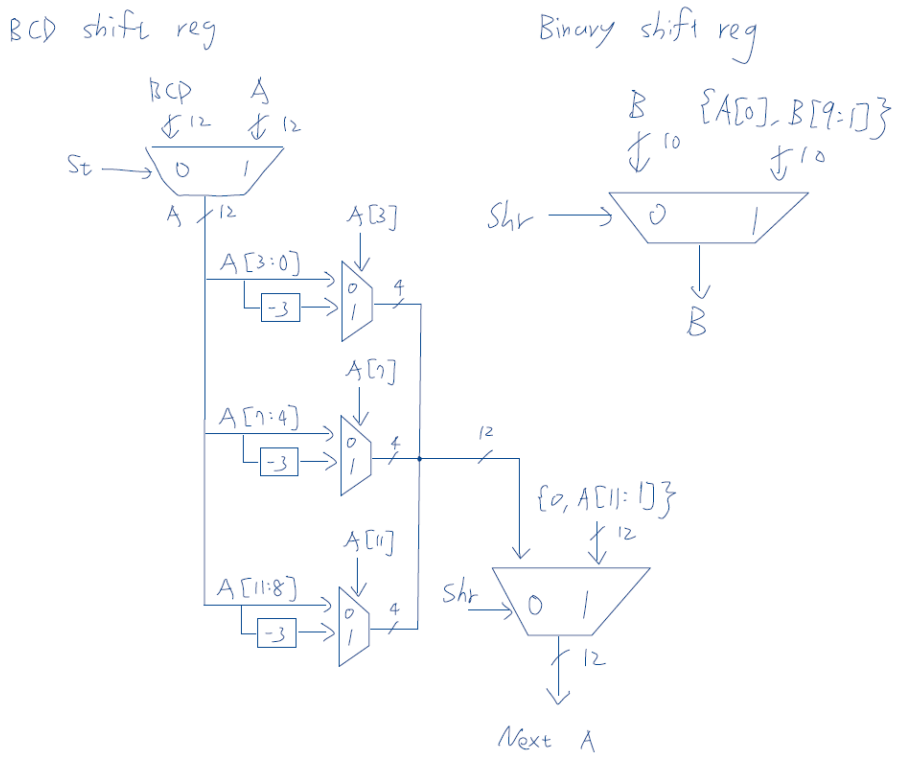
**ii. 說明你的設計，繪製A(b)的方塊圖與A(c)的狀態圖。**

**Describe your design. Draw the block diagram in A(b) and the state graph in A(c).**

**block diagram:**

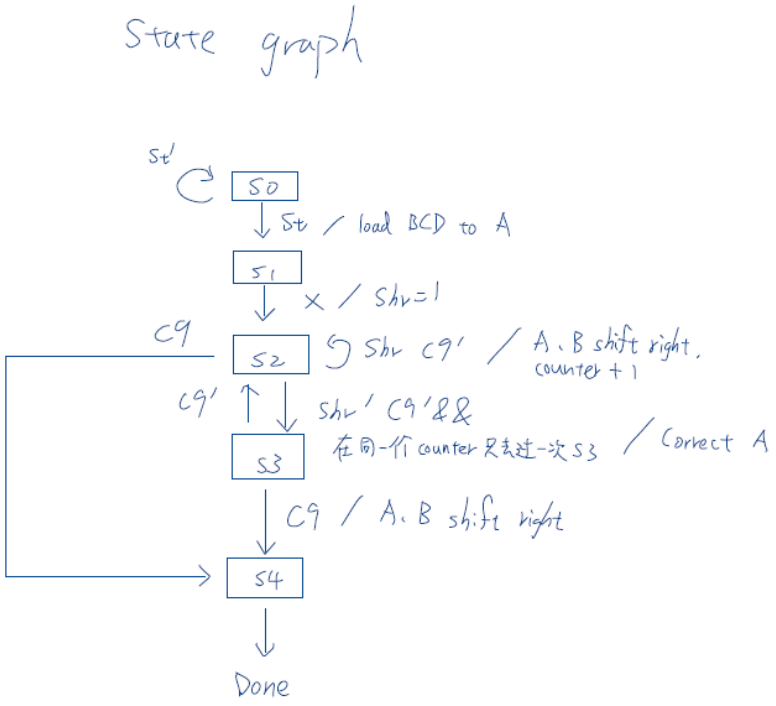
Block diagram






我的設計主要是使用St、Shr這些control signal來控制A、B、counter這些暫存器。St可以決定A要不要load進來BCD，沒有要load就沿用本來的A。Shr則可以決定要不要將A跟B做shift right還是要把A做correct，若shift right則把A的最右邊bit移到B的最左邊的bit。

**state graph:**



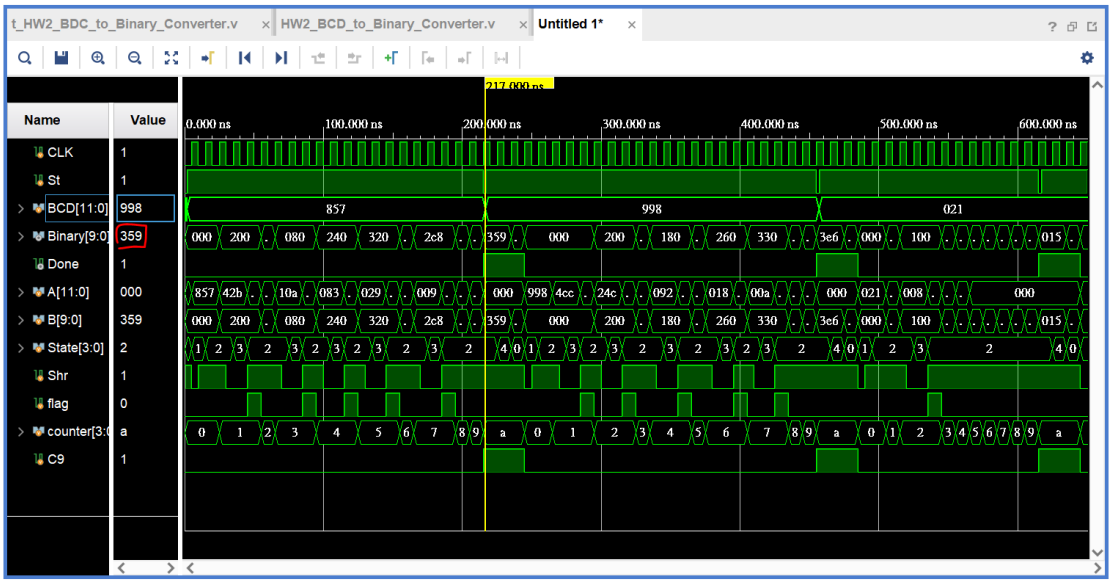
那State各設計如下: State=0直到St=1就load BCD進來並轉為State=1，然後無條件進到State=2，State=2時若發現有需要correct A則進到State=3，反之就繼續留在State=2並做Shift right。State=3除非count到9不然一定會回到State=2。不管在State=2 or 3，只要發現count到9則直接跳到State=4並發出Done的output signal。在我的設計中不需要Cor，因為Cor是Shr的反向。

**iii. 附上A(d)模擬結果之波型圖，解釋波形圖是否正確，並列出每筆測資需要幾個cycles完成。**

Show the waveform of the simulation results for the circuit module in A(d), explain whether the results are correct or not, and indicate how many cycles are required to complete the computation of each test case.

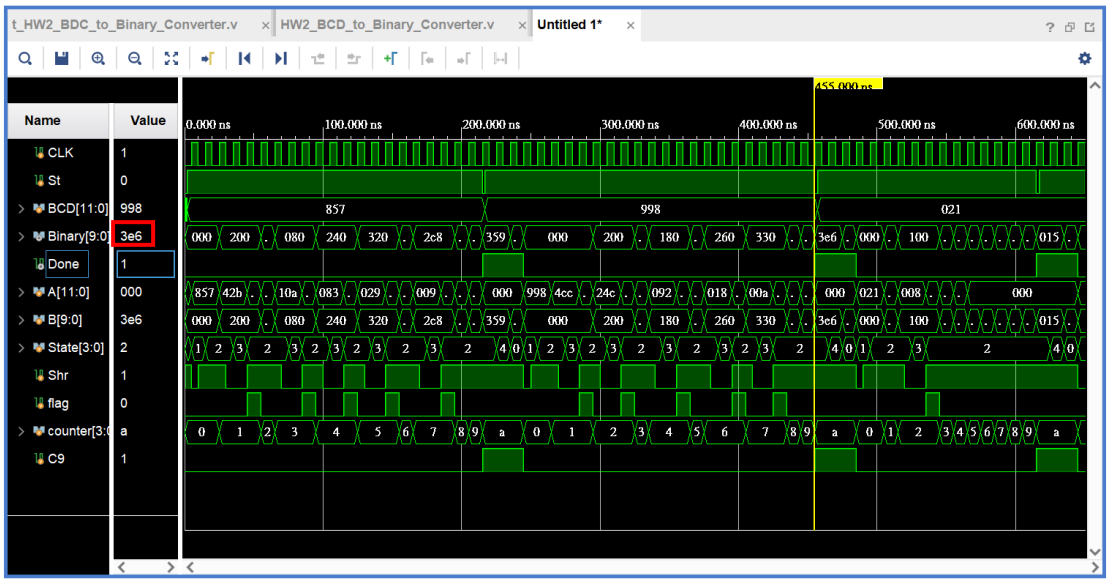

經過計算之後得到將857、998、21轉成HEX的形式為359、3E6、15。

模擬將BCD為857轉成Binary的示波圖如下:



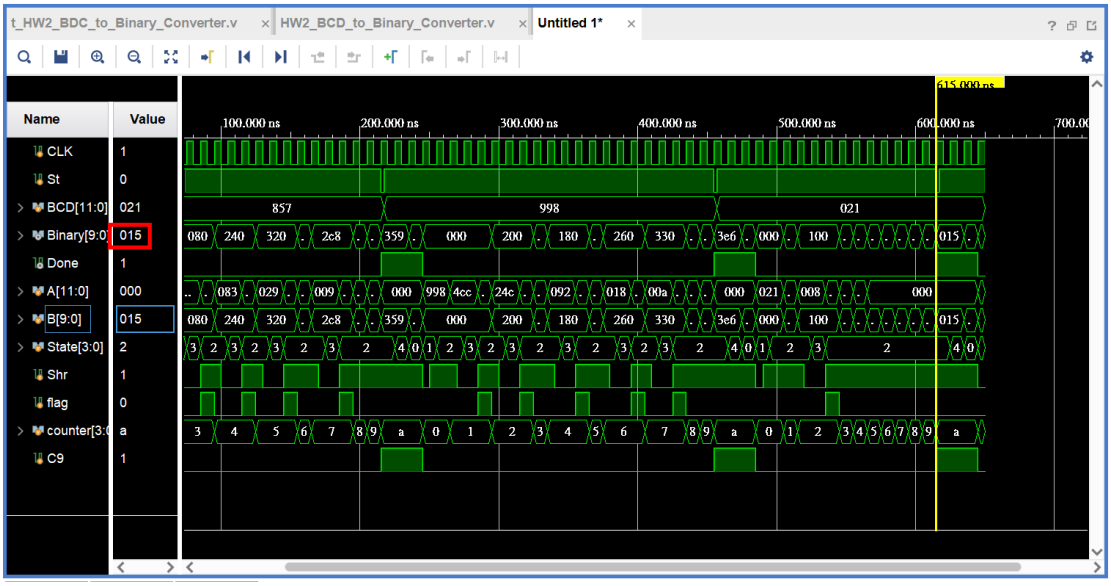

在Done=1的時候，Binary以16進位表示確實是359。

模擬將BCD為998轉成Binary的示波圖如下:



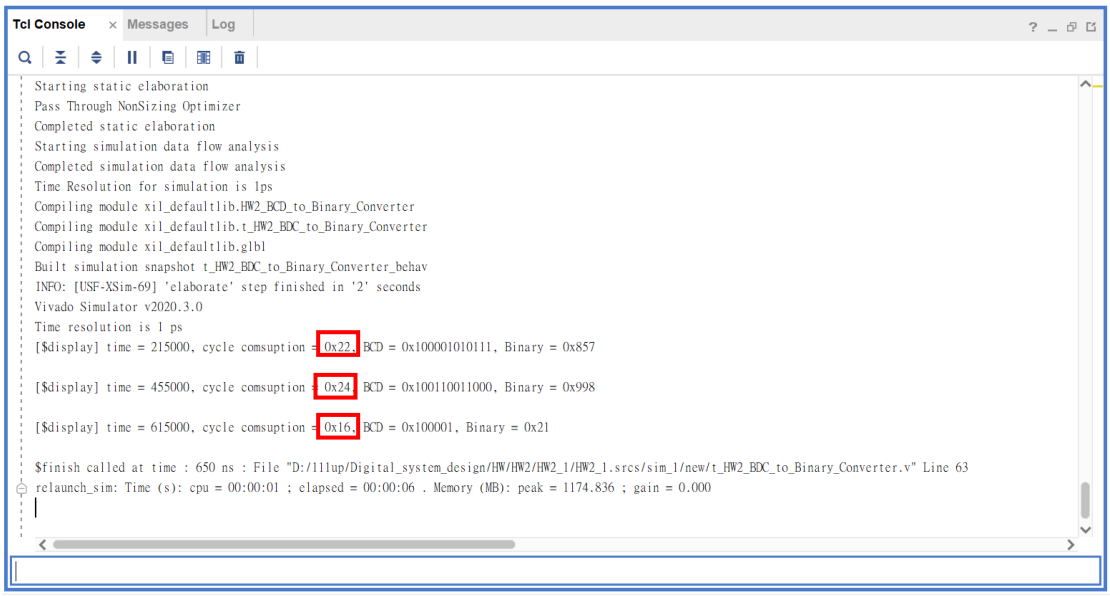

在Done=1的時候，Binary以16進位表示確實是3E6。

模擬將BCD為021轉成Binary的示波圖如下:




在Done=1的時候，Binary以16進位表示確實是015。

這三次轉換所需要的cycle數目如下:



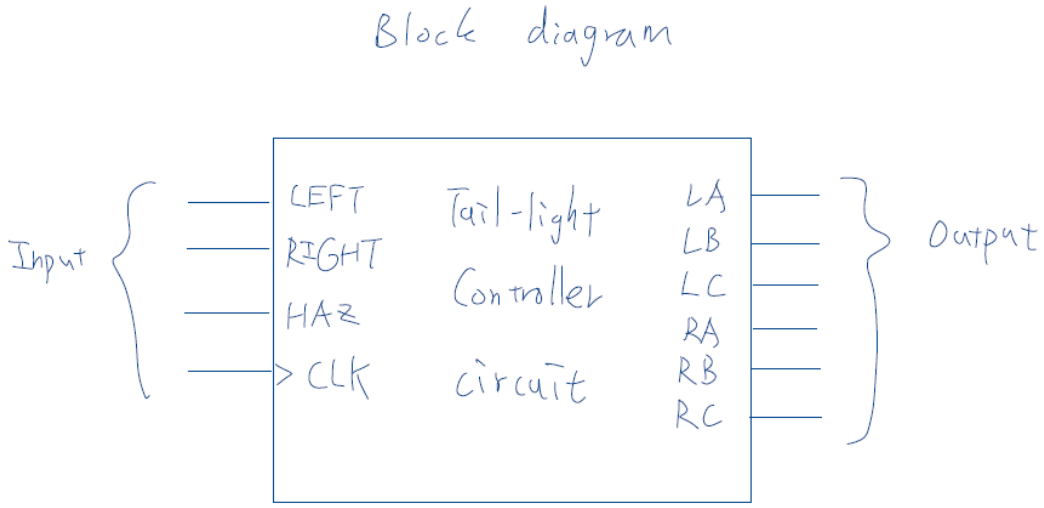

分別需要22、24、16個cycle。

B. Tail-Light Controller (40%)

i. 說明你的設計，繪製B(a)的方塊圖與B(b)的狀態圖。

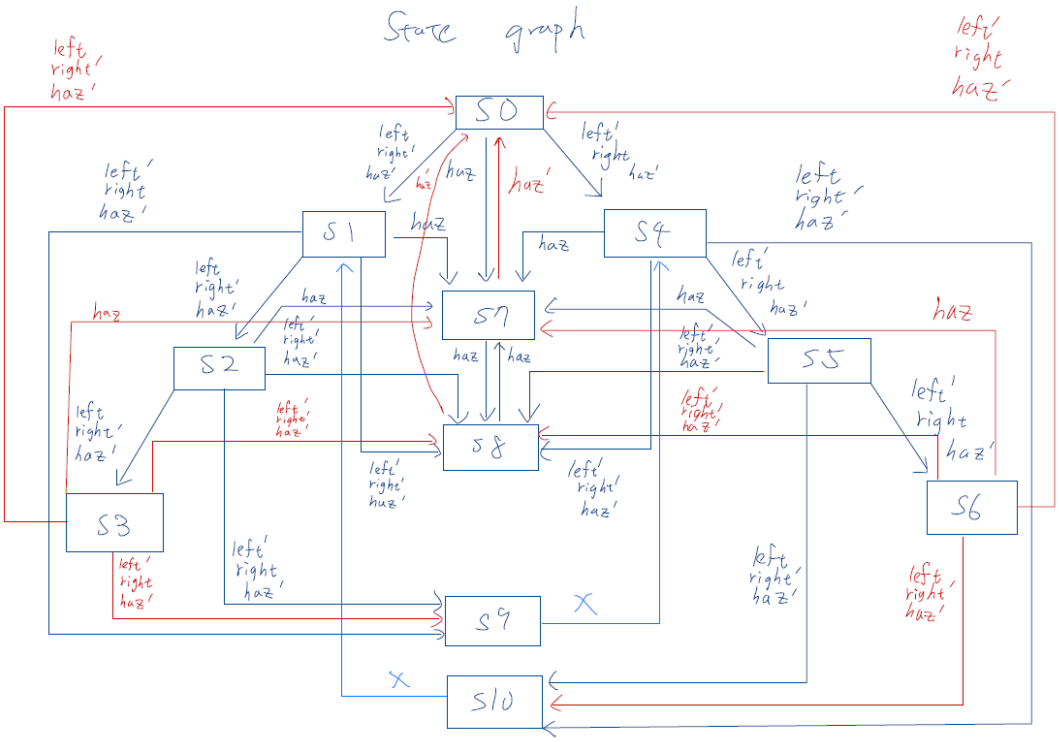
Describe your design. Draw the **block diagram** in B(a) and the **state graph** in B(b).

**block diagram:**




這個設計的輸入輸出如上圖，以Behavior的說法，當LEFT=1的時候，LA、LB、LC會照這個順序亮起來，當RIGHT=1的時候，RA、RB、RC會照這個順序亮起來，HAZ的優先權是最高的，不管LEFT及RIGHT，當HAZ=1的時候，LA、LB、LC、RA、RB、RC全部的燈都會暗下來。State graph的部分會做更詳細的說明。

**state graph:**




各State的設計如下:

S0: 初始狀態，LEFT=1則進到LEFT mode的第一個(S1)，RIGHT=1則進到RIGHT mode的第一個(S4)，輸出為全暗。

S1: LEFT mode的第一個，輸出LA亮。

S2: LEFT mode的第二個，輸出LA、LB亮。

S3: LEFT mode的第三個，LEFT、RIGHT沒變則預設回到S0，輸出LA、LB、LC亮。

S4: RIGHT mode的第一個，輸出只有RA會亮。

S5: RIGHT mode的第二個，輸出RA、RB會亮。

S6: RIGHT mode的第三個，LEFT、RIGHT沒變則預設回到S0，輸出RA、RB、RC亮。

要注意的是，在以上的狀況中，若HAZ=1(熄火)就直接中斷進入S7，若LEFT跟RIGHT都=0(方向燈按掉)則進入S8，若從LEFT=1轉成RIGHT=1(方向燈直接打另一邊)則進入S9，若從RIGHT=1轉成LEFT=1(也是方向燈直接打另一邊)則進入S10。

S7: 待機狀態，輸出為全暗。

S8: 待機狀態，輸出為全暗。

另外要注意的是，以上的狀況中，當HAZ=0(打開電屏、發動引擎)，會回到初始狀態S0，否則會在S7、S8繞圈圈等待。

S9: 準備直接從LEFT mode換到RIGHT mode的狀態，輸出為全暗。

S10: 準備直接從RIGHT mode換到LEFT mode的狀態，輸出為全暗。

ii. 說明你的測資，附上B(d)模擬結果之波型圖，並解釋波形圖是否正確。
Describe your test data. Show the waveform of the simulation results for the circuit module in B(d), and explain whether the results are correct or not.

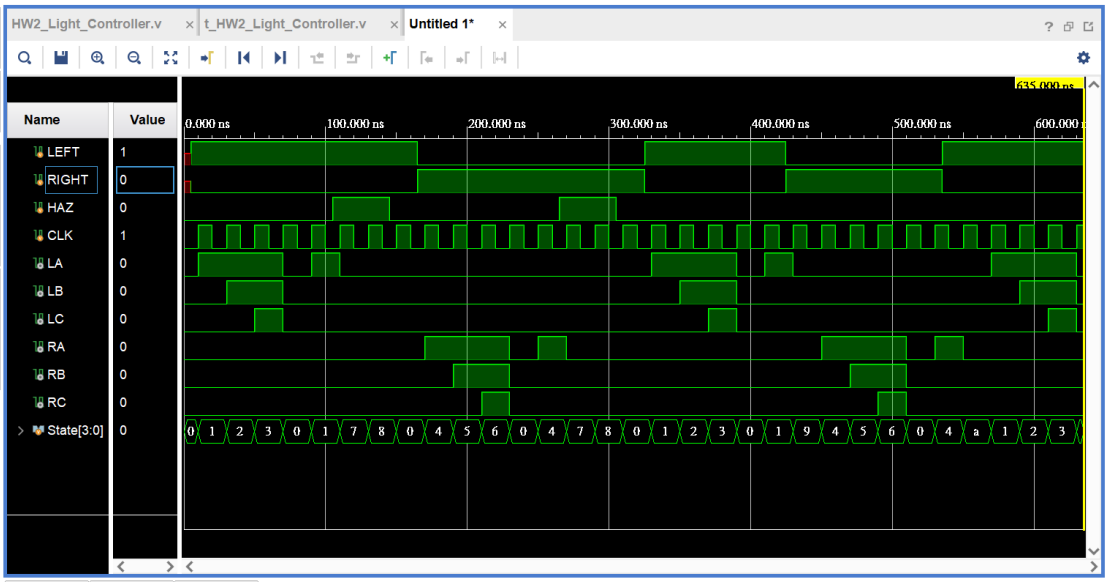
我的測資如下:

```
30  initial begin
31      CLK = 1'b0;
32
33      HAZ = 1'b0;
34
35      #5; LEFT = 1'b1; RIGHT = 1'b0;
36      #100; HAZ = 1'b1;
37      #40; HAZ = 1'b0;
38
39      #20; LEFT = 1'b0; RIGHT = 1'b1;
40      #100; HAZ = 1'b1;
41      #40; HAZ = 1'b0;
42
43      #20; LEFT = 1'b1; RIGHT = 1'b0;
44      #100; LEFT = 1'b0; RIGHT = 1'b1;
45      #110; LEFT = 1'b1; RIGHT = 1'b0;
46
47      #100 $finish;
48  end
```

一開始先進入LEFT mode，過一段時間之後HAZ=1中斷，再回到初始狀態。
然後進入RIGHT mode，過一段時間之後HAZ=1中斷，再回到初始狀態。

然後再進入LEFT mode，過一段時間之後RIGHT=1中斷，進入RIGHT mode。
過一段時間之後LEFT=1中斷，進入LEFT mode。

示波圖結果如下:



可觀察到LEFT=1從初始狀態進入LEFT mode，輸出LA、LB、LC照順序亮起來再全按然後重複，State則是從0->1->2->3->0->1，直到HAZ=1中斷，輸出全暗，State直接變成7，HAZ變回來0之後，照一樣的方法測試RIGHT mode，都很正常。

接著再進入LEFT mode，然後RIGHT=1進入RIGHT mode，State則1->9->4。
接著LEFT=1進入RIGHT mode，State則4->10->1。

**C. 心得感想、結論、及討論 (10%)**
**Conclusions and Discussions**

我覺得第一題的演算法很容易懂，但是要怎麼定義每一個state並想控制訊號要怎麼控制next state與output實在很困難，經過一番努力之後還好最後有成功做出來。第二題的題目比較好定義state，而且因為每個state就固定有自己的output，對programmer來講非常好設計。希望這次作業經驗有幫助到期末project。