EECS2070 Logic Design Lab 1

lab5 Report

lab4\_4

Design Specification

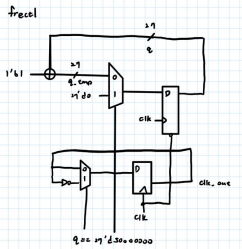
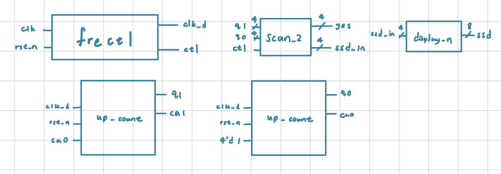
For a 2-digit BCD up counter display on the seven-segment display

Input: clk, rst\_n

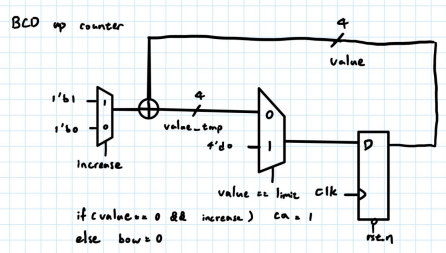
Output: [7:0] ssd, [3:0] yes.

Design Implementation

Block diagram



© Ma, Hsi-Pin, Lab for Reliable Computing (LaRC), EE, NTHU. Spring, 2021

EECS2070 Logic Design Lab 2 

| I/O | fcrystal | yes 3 | yes 2 | yes 1 | yes 0 |  |  | ssd7 ssd6 ssd5 ssd4 ssd3 ssd2 ssd1 ssd0 rst\_ |  |  |  |  |  | n |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Site | W5 | V19 | U19 | E19 | U16 |  |  | V14 U14 U15 W18 V19 U19 E19 U16 R2 |  |  |  |  |  |  |

Discussion

這個題目其實就是利用類似 lab4\_3 所做出的 BCD down counter 而此題則是修改為 up counter 並改成 2digit 將個位數的 carry 連接到十位數的 count\_enable，也就是當個位數有 carry 時，十位數才會往上數一，就可以達到題目所要求之效果。而因為在 lab4\_3 裡面因為 題目不需用到太複雜的功能，我只寫了沒有停止功能，也沒有紀錄 borrow，所以在做這個 子題的時候，多花了一些時間理解兩者之間的連接。

Lab5\_1

Design Specification

For a 40sec down counter

Input: clk, rst\_n, restart, stop

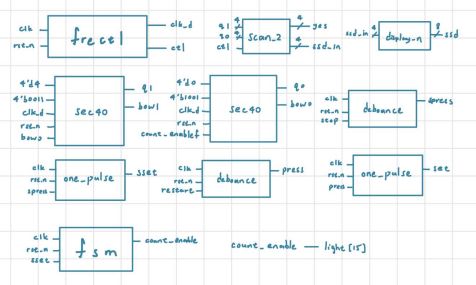
Output: [7:0] ssd, [3:0] yes, [15:0] light

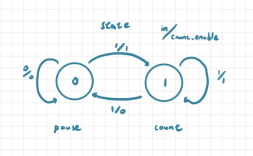
Design Implementation

Block diagram

© Ma, Hsi-Pin, Lab for Reliable Computing (LaRC), EE, NTHU. Spring, 2021

EECS2070 Logic Design Lab 3

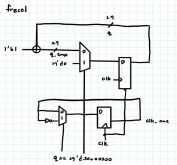
 fsm



If state = 0 (pause) and signal is 0, then keep the state.

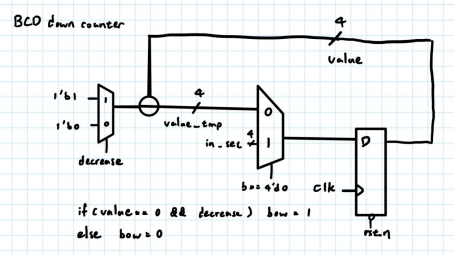
If state = 0 (pause) and signal is 1, then jump to next state (state = 1) (start). If state = 1 (start) and signal is 0, then keep the state.

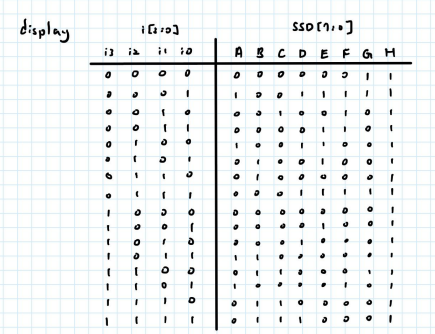
If state = 1 (start) and signal is 1, then jump to next state (state = 0) (pause) frectl



© Ma, Hsi-Pin, Lab for Reliable Computing (LaRC), EE, NTHU. Spring, 2021

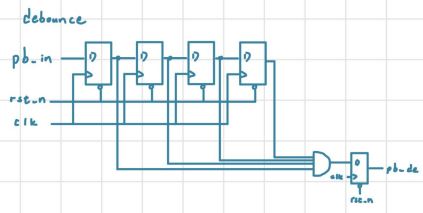
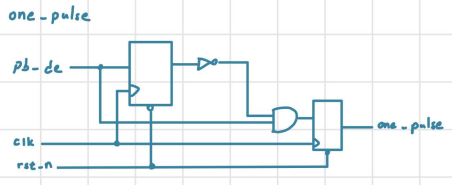
EECS2070 Logic Design Lab 4





© Ma, Hsi-Pin, Lab for Reliable Computing (LaRC), EE, NTHU. Spring, 2021

EECS2070 Logic Design Lab 5

| I/O | fcrystal | yes 3 | yes 2 | yes 1 | yes 0 |  |  | ssd7 ssd6 ssd5 ssd4 ssd3 ssd2 ssd1 ssd0 rst\_ |  |  |  |  |  | n |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Site | W5 | V19 | U19 | E19 | U16 |  |  | V14 U14 U15 W18 V19 U19 E19 U16 R2 |  |  |  |  |  |  |

| light [15] | light [14] | light [13] | light [12] | light [11] | light [10] | light [9] | light [8] |
| --- | --- | --- | --- | --- | --- | --- | --- |
| L1 | P1 | N3 | P3 | U3 | W3 | V3 | V13 |

| light [7] | light [6] | light [5] | light [4] | light [3] | light [2] | light [1] | light [0] |
| --- | --- | --- | --- | --- | --- | --- | --- |
| V14 | U14 | U15 | W18 | V19 | U19 | E19 | U16 |

Discussion

本實驗要做出一個 40 秒的倒數計時器，就會直接想到 lab4\_3 的 BCD down counter，仿照 lab4\_4 的接線方式將十位數的初始值設為 4 個位數為 0，最困難的步驟應該是 Fsm 的處 理，第一次使用上卡頓了一下，不過還好只有兩個 state 相對輕鬆，設定按下 stop 時，轉

© Ma, Hsi-Pin, Lab for Reliable Computing (LaRC), EE, NTHU. Spring, 2021

EECS2070 Logic Design Lab 6

換 state 使 count\_enable 為 0 讓倒數計時暫停

Lab5\_2

Design Specification

For a stopwatch

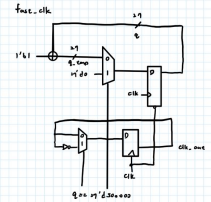
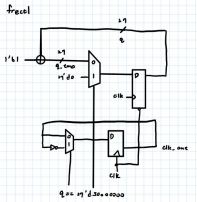
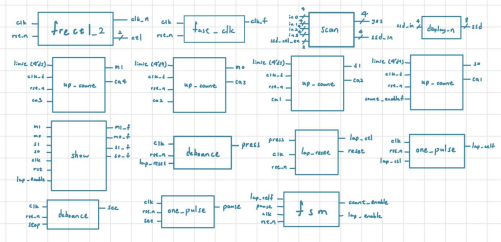
Input: clk, rst\_n, lap\_reset, stop, clk\_ctl

Output: [7:0] ssd, [3:0] yes,

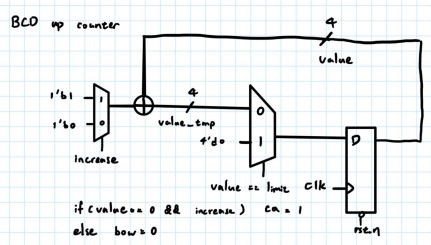
[1:0] light\_state, light\_lap, light\_press(for checking)

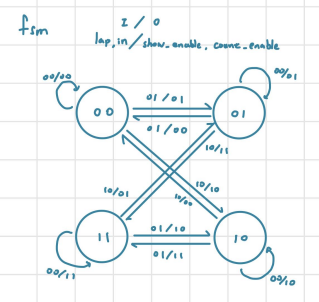
Design Implementation

Block diagram



© Ma, Hsi-Pin, Lab for Reliable Computing (LaRC), EE, NTHU. Spring, 2021

EECS2070 Logic Design Lab 7 



因為 in 跟 lap 不會同時為 1，所以沒有 11 的這種 input

Show

是利用 if else 選擇要輸出 counter 的值還是 lap 的值，為了顯示即時的數字所以使用 100M 的 clk

© Ma, Hsi-Pin, Lab for Reliable Computing (LaRC), EE, NTHU. Spring, 2021

EECS2070 Logic Design Lab 8 

| I/O | fcrystal | yes 3 | yes 2 | yes 1 | yes 0 |  |  | ssd7 ssd6 ssd5 ssd4 ssd3 ssd2 ssd1 ssd0 rst\_ |  |  |  |  |  | n |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Site | W5 | V19 | U19 | E19 | U16 |  |  | V14 U14 U15 W18 V19 U19 E19 U16 R2 |  |  |  |  |  |  |

| light\_state[1] | light\_state[0] | light\_press | light\_lap | lap\_reset | stop |
| --- | --- | --- | --- | --- | --- |
| L1 | P1 | P3 | N3 | T18 | U18 |

Discussion

想法是先接好 4 個 BCD up counter，使 FPGA 可以順利地從 00:00 數到 59:59，然後因為多 了 lap 所以有 fsm 出現了 4 個 state，讓難度提高了一些，關於 lap 的功能，想法是利用 fsm 輸 出一個 show\_enable 來決定要表現出哪一組數字，我的想法是在做出一個 register 去儲存按下 lap 當下的數字，最後再用選擇器去選擇要輸出的數字，而暫停的功能則可以沿襲 lab5\_1 的做 法，最後一部分則是設計長按是 reset 短按是 lap，想法是在把 lap\_reset 接到另外一組 up counter 的 count\_enable 使得按下去的時候 counter 開始數，數到一個特定的數字時則輸出 reset 如果 counter 小於特定的數且不等於 0 時則輸出 lap。

© Ma, Hsi-Pin, Lab for Reliable Computing (LaRC), EE, NTHU. Spring, 2021

EECS2070 Logic Design Lab 9

Lab5\_3

Design Specification

For a timer

Input: clk, rst\_n, clk\_ctl, mode, pause\_resume, start\_stop, add\_min, add\_sec Output: [7:0] ssd, [3:0] yes, [15:0] light

Design Implementation

Block diagram

按鈕皆有接 one pulse 跟 debounce 只是省略

因為 in 跟 mode 不會同時為 1，所以沒有 

11 的這種 input

© Ma, Hsi-Pin, Lab for Reliable Computing (LaRC), EE, NTHU. Spring, 2021

EECS2070 Logic Design Lab 10

Timer

裡面包四個 BCD down counter，rst 接到 start/stop 使得每一次按下 start/stop 的時候 timer 的值都會跑回 set 所給定的值，而當 count\_enable 等於 1 時就正常的 down count，並在 數到 0 的時候停下來且發光

| I/O | fcrystal | yes 3 | yes 2 | yes 1 | yes 0 |  |  | ssd7 ssd6 ssd5 ssd4 ssd3 ssd2 ssd1 ssd0 rst\_ |  |  |  |  |  | n |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Site | W5 | V19 | U19 | E19 | U16 |  |  | V14 U14 U15 W18 V19 U19 E19 U16 R2 |  |  |  |  |  |  |

| light [15] | light [14] | light [13] | light [12] | light [11] | light [10] | light [9] | light [8] |
| --- | --- | --- | --- | --- | --- | --- | --- |
| L1 | P1 | N3 | P3 | U3 | W3 | V3 | V13 |
| light [7] | light [6] | light [5] | light [4] | light [3] | light [2] | light [1] | light [0] |
| V14 | U14 | U15 | W18 | V19 | U19 | E19 | U16 |

| mode | pause\_resume | start\_stop | add\_min | add\_sec |
| --- | --- | --- | --- | --- |
| V17 | T17 | W19 | T18 | U17 |

Discussion

這個 lab 與 lab5\_1 較相同只是多了設定時間的功能，所以第一個想法就是在開一個 register 儲存設定的值，並利用上數器去控制設定值，設定完畢以後再透過按鈕將值送進 timer 也就是 down counter 裡面，使他倒數，數到 0 時則停下來且 led 全亮。

Conclusion

這次的 lab 難度值接大大的提升了一個階級，因為 module 的數量變多，所以接線時的難度 就更高了，所以到 lab5\_3 時，選擇再利用一個 module 分別把四個 up counter 與 down counter 包起來，使版面更為簡潔。且因為狀態較為多，寫 fsm 的時候要更加小心，但完成 block diagram 及 fsm 整個 lab 都會變得流暢許多。

References

老師的ppt

© Ma, Hsi-Pin, Lab for Reliable Computing (LaRC), EE, NTHU. Spring, 2021