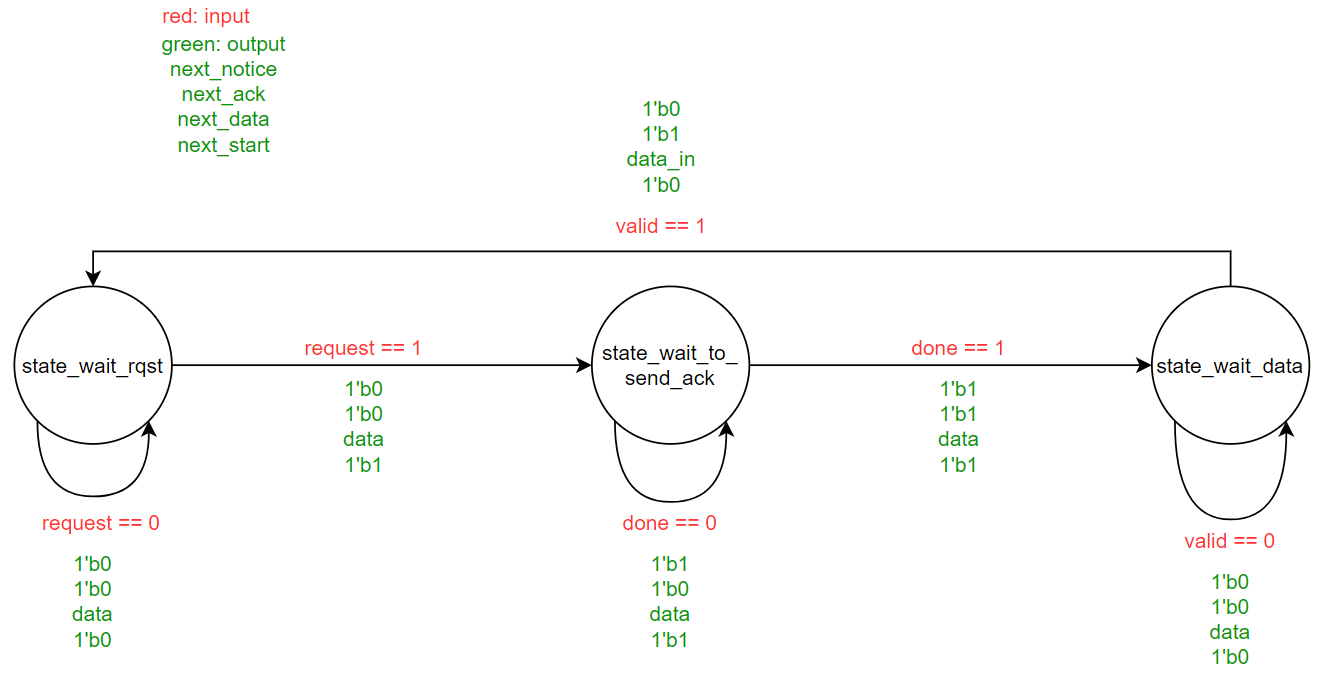
Lab 6 report

組員：110062221李品萱

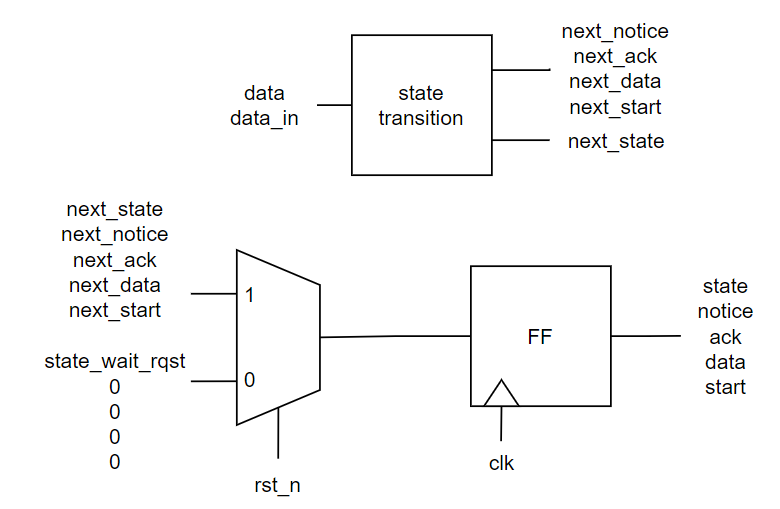
110062213唐翊雯

1. **Dual FPGA communication**

這題要設計一個FPGA-to-FPGA的communication protocol，基本的步驟與state spec都已經給了，template也完成了大部分功能，我們需要做的部分只有將slave FPGA有關state的code完成，這部分也只要寫出每個state的output以及next state為何即可，其state diagram如下：



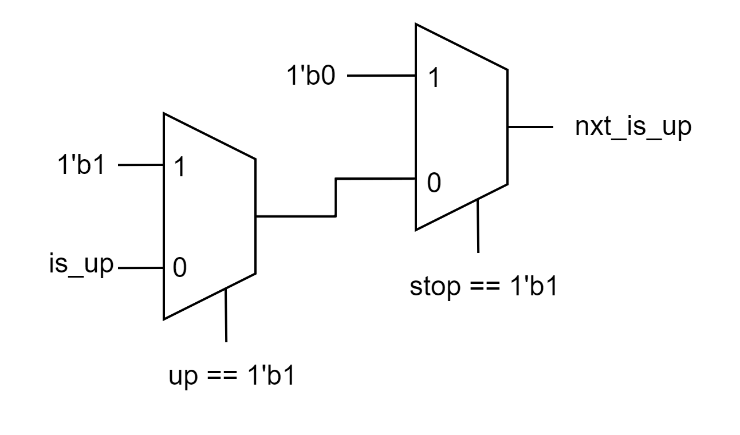
這個module的block diagram如下：



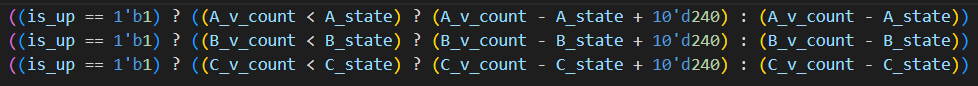
1. **The slot machine**

這題我們需要實作一個拉霸機，在sample code的部分已經幫我們寫好大部分的功能，我們只需要修該code使其作出往上跑的部分。Trace了一下sample code之後，我們先將控制向上的按鈕做debounce及onepulse，然後將處理過的up\_op接到state control的module內，在這個module內，我們用一個is\_up的reg去記現在的方向（1為up，0為down），如下圖。

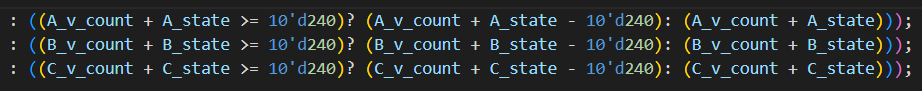




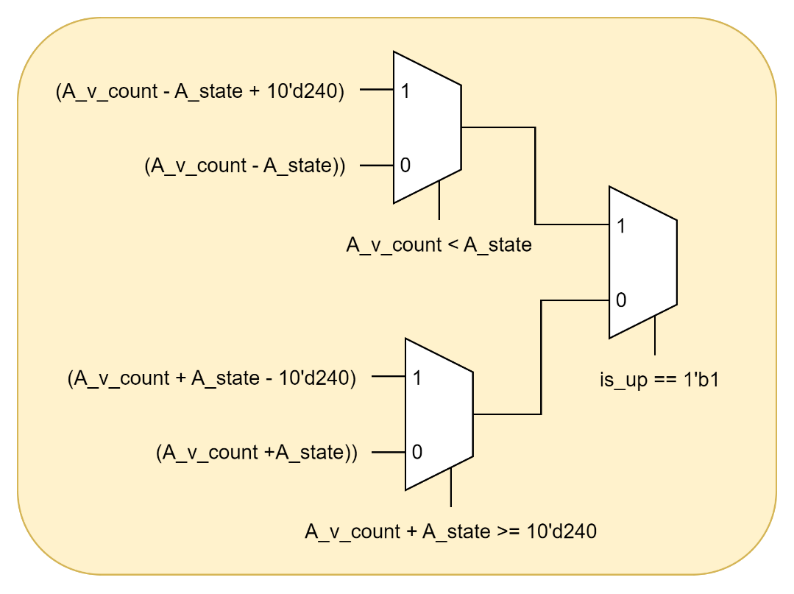
接下來我們需要修改配合給定方向做顯示的部分，也就是next\_A\_v\_count、next\_B\_v\_count、next\_C\_v\_count。首先我們用is\_up判斷現在的方向，如果向上的話接下來需要判斷邊界情況，由於是以向下為正、向上為負，因此向上的部分必須判斷他是正的，維護他在顯示的合法範圍。這部分在sample code做的向下的操作也有類似的處理，而我們對像上的操作如下圖。

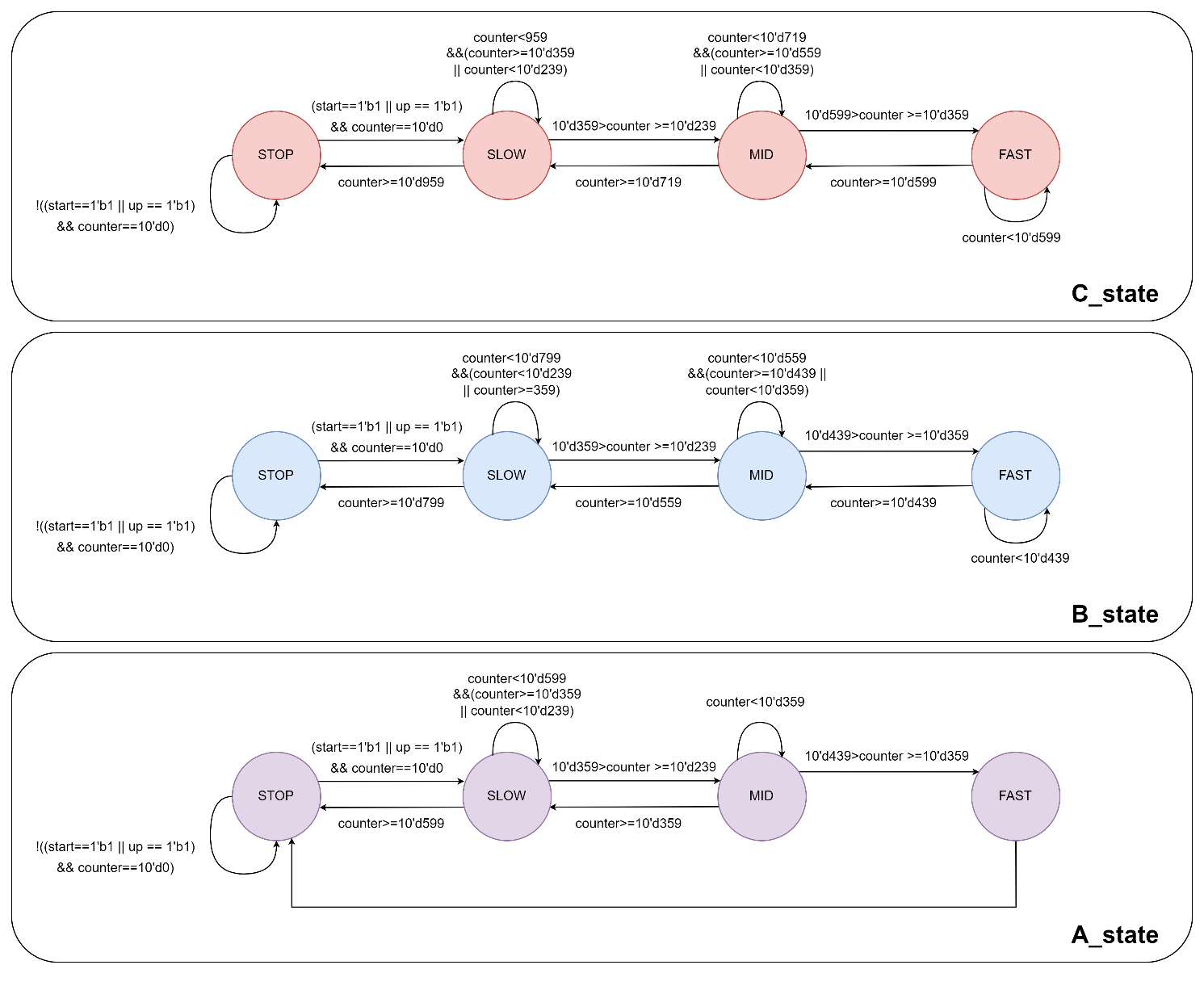


而sample code內的操作如下圖，因為我們的v方向最大值是240，因此需要維護他在向下加的過程中是在合法的顯示範圍內。

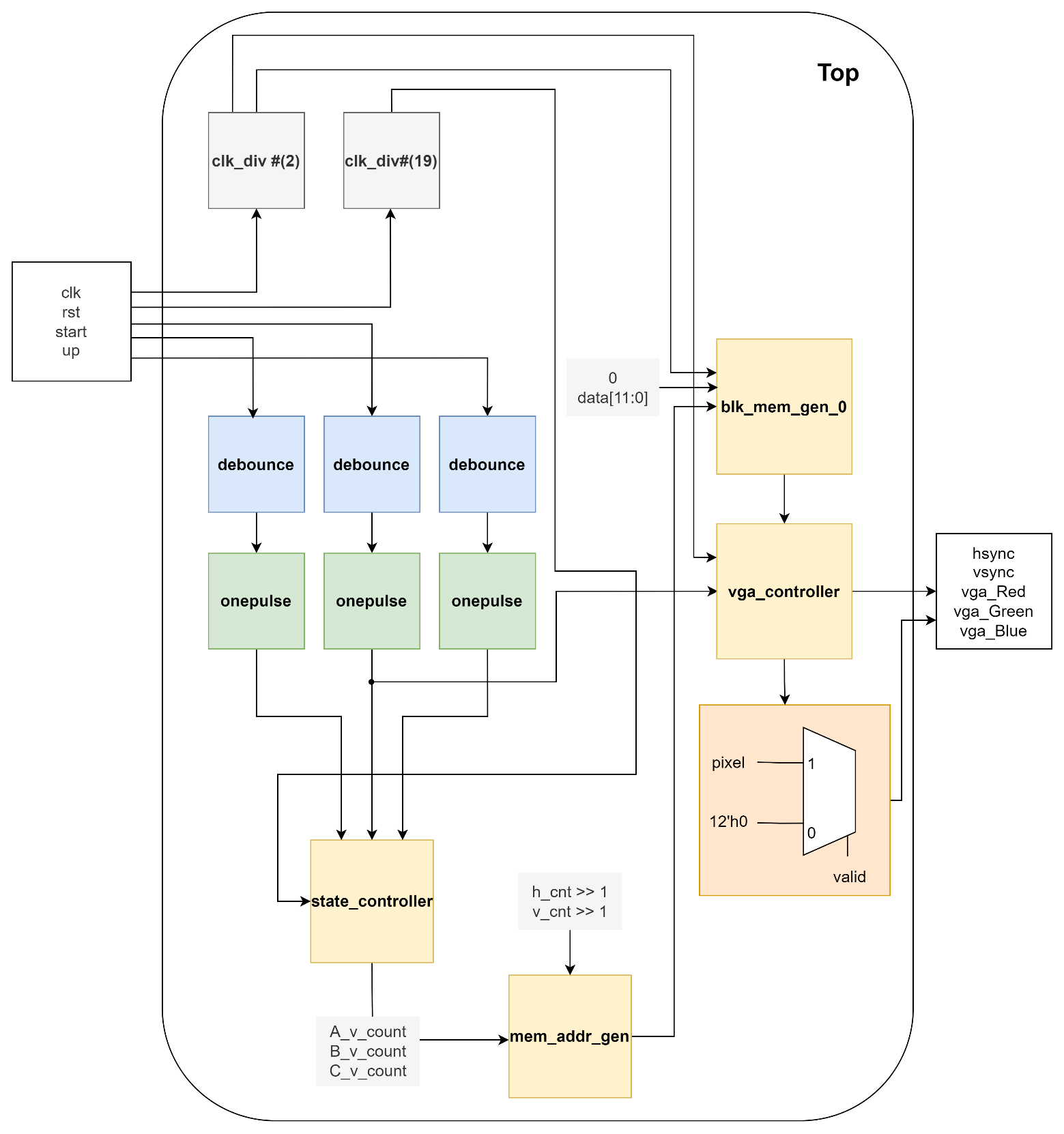


next\_A\_v\_count（next\_B\_v\_count、next\_C\_v\_count同理）的block diagram如下圖。



在顯示速度的部分，我們只對A、B、C的stop state加上up == 1’b1的判斷條件，這邊的state diagram如下圖。

而為了讓slot machine可以連續觸發，不須按下reset才能跑下一次，我們修改了counter的部分，讓他在符合條件時歸零，如下圖。

本題的block diagram如下。

1. **The car**
2. **Summary**
3. **Contributions**

* **Code**

Dual FPGA communication by 李品萱

The slot machine by唐翊雯

The car

* **Report**

各自描述負責的題目