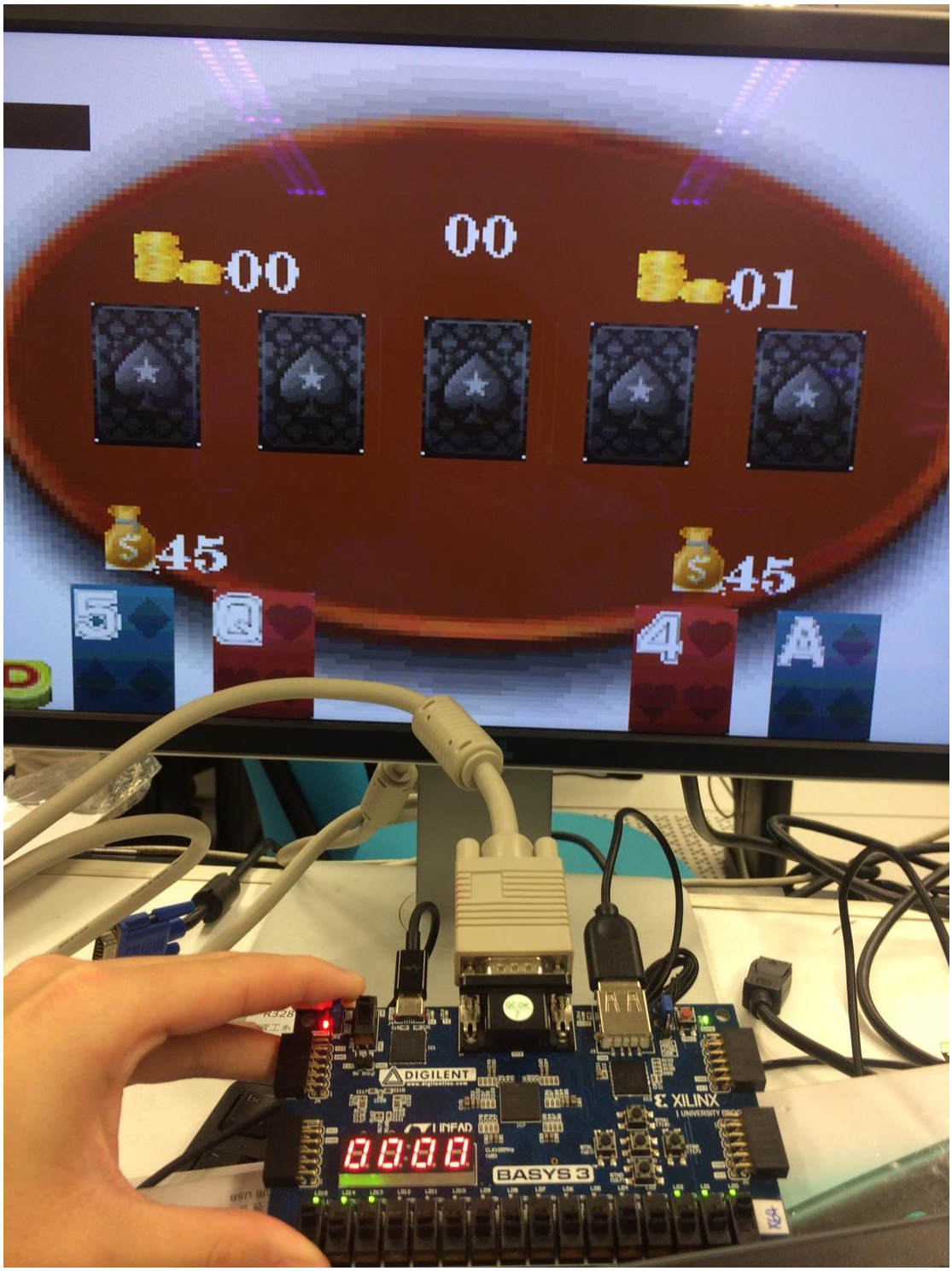
硬體設計與實驗期末 Final：雙人德州撲克

第 48 組 105062224 鐘浩瑋 105062115 曾鼎睿



一、前言

我們組所做的 final project 是能夠兩人進行遊玩的德州撲克遊戲，因此在此簡略一下此遊戲的規則，一開始會發給雙方一人兩張的手牌，接著進入下注流程，每位玩家都有 4 種指令可以使用，分別為：call or check(跟注

或檢查)、raise+數字(加注)、fold(棄牌)以及 all in(梭哈)，接過一輪的下注流程後會進入發牌流程，接著下注流程與發牌流程互相交替直到桌上有 5 張公牌，而發牌流程會以發 3、1、1 張分成 3 個回合進行，最後贏家可以將下注的所有錢拿回，接著遊戲再回到一開始繼續遊玩。

二、code 解析

我們大致將我們的 code 分成五大部分，分別為 random、VGA、牌型判斷、金錢運算以及 FSM，以下將依序介紹：

1、 random：

為了將撲克牌轉為 code 進行使用，我們使用了 6 個 bit 的變數儲存一個數字，最左邊兩位數控制花色，右邊的 4 個 bit 控制數字，因此我們的 random 將這個變數拆成兩個部份，最左邊兩位在 0~3 之間變化，而讓右邊四位在 1~13 之間進行變化，從 random 輸出合併後的 6-bit 變數產生卡牌，我們特意留了 6'd0 這組數字做為初始化使用。在能夠產生隨機卡牌後，接下來就遇到了不能產生相同卡片的問題，於是我們又使用了一個 CompareCard 的 module，將使用到的 9 組變數傳入後，當讀到變數香同時讓其中一個在跑一次 random 達到讓每組數字都不相同的效果，因此當我們用於印出卡片時，是以 CompareCard 的 output 為依據進行顯示。

2、 VGA：

由於我們的螢幕輸出以靜態為主，因此我們最大的課題是如何在相同位置上印出相對應的卡片。首先，我們先將要印出卡片及圖案的位置找出來，接下來我們將控制花色、卡片數字、金錢數字等的變數分別傳入選擇哪個圖案的 module 裡，每個 module 裡將所需要的圖案的 pixel 傳入，使用 mux 將與使用變數相應的 pixel 當作 output 使用，因此 VGA 用到了非常多的 mux，如果設計不當會發生燒不下的問題，經過數個 mux 後，我們便取得了需要印出來的 pixel，接下來在 assign {Red,Green,Blue}的地方進行選擇判斷印出哪張圖片，此處要注意的是由於我們的卡片將數字與花色分開印，因此必須先判斷數字，再判斷花色，這樣才不會發生被覆蓋的問題，另外在處理去背的問題，我們再判斷式上多了一行(XXX_pixel!=12'h0)用於判斷當遇到黑色像素時省略，這樣就能達到美化螢幕輸出的效果。

3、 牌型判斷

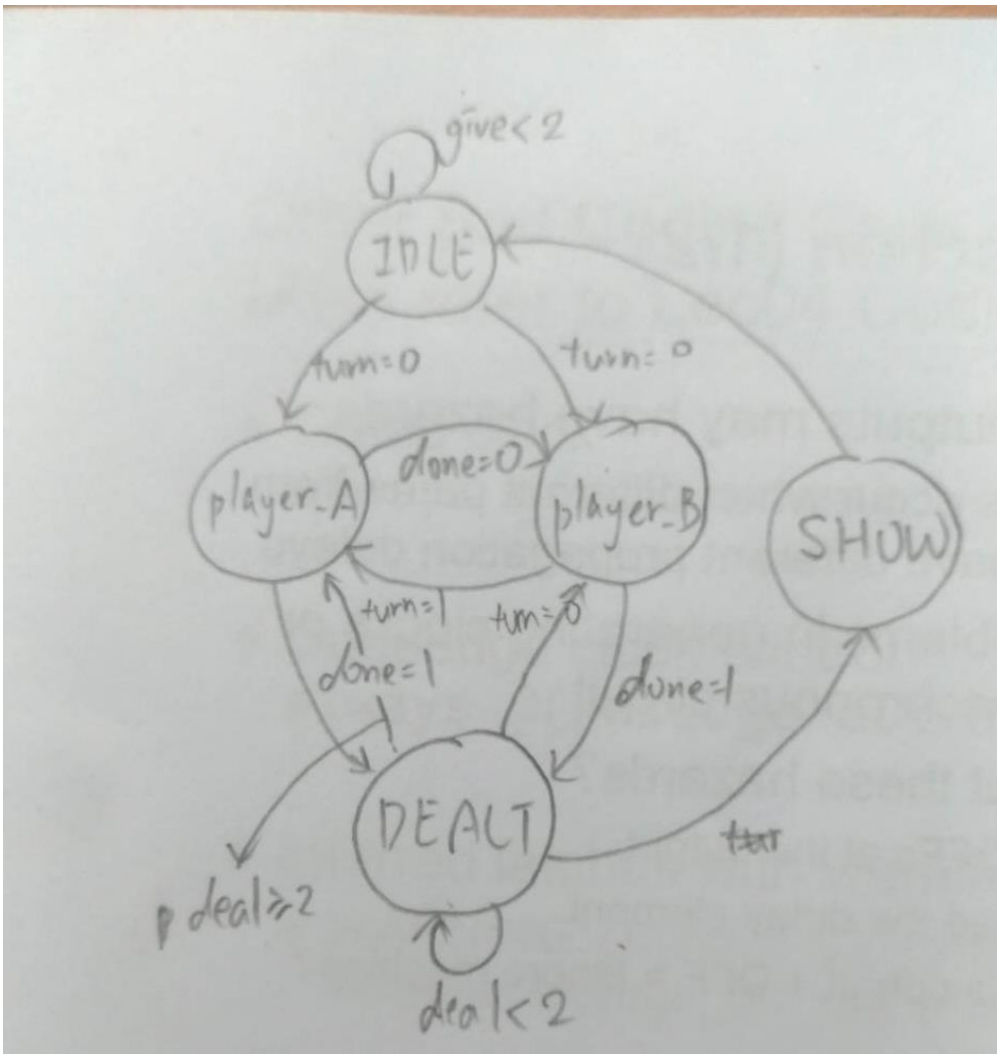
我在牌型判斷遇到的第一個問題是判斷的邏輯本深很複雜，時十種牌型再加上每種牌型又有大小或平手的狀況，要想通其中的邏輯如何判定本身就很困難。接著的問題還有硬體並沒有 sort 可以用。有許多牌型例如同花順、順子在軟體利用 sort 可以輕鬆辨識出來，但硬體若是模擬最基本的 insertion sort 或 bubble sort 需要 O(n^2)的時間，依照我們牌的數量也就是 7*7=49 個 cycle，而我認爲這個時間消耗太過昂貴，畢竟牌型判斷是要接上主 module 的，我希望盡量減低時間消耗。於是我直接全部展開，遍歷

所有的牌，用 reg 存取每個數字與花色大小，（這裡的花色存取是為了同花比大小時可以用）並同時用一個小 module 算有無同花，期許能在 7 個 cycle 內拿到所有所需資料，再做同樣牌型的判斷

->最多 13cycle，總共 20cycle。但後來發現 two pair 一定必需再花 13 個 cycle 檢查，無法避免，也無法同時判定。其實這中間想了很多辦法，但後來還是覺得這樣是目前的我能寫出的最好版本了。除此之外測試 20 個 sample 測資的時候我也 debug 的很辛苦。因爲 code 長達 1500 行，光要判斷第一輪 or 第二輪哪個出錯就耗費了許多心力。但也因爲花了整整三個全天修正所有 bug，測完測資後的試玩與後來合併 module 後牌型判斷就從來沒出錯過。甚至到後來這個 module 還比我肉眼判斷的準一點。雖然牌型判斷在呈現上就只是一個 win 的字樣，但卻是最難以實做的部份，也是完成這個真正德州撲克的重要拼圖。

4、 金錢運算

金錢判斷的難處在於要搭配上 FSM 的設計讓盲注、底池、兩個人的金錢能夠在正確的時間減少/增加。除此之外，All in 的部份也需要特別處理。



5、 FSM&TOP

當我們完成以上四個部份的 code 後基本上遊戲已經有個大概的樣子了，剩下的就是處理遊戲流程以及金錢和其餘設備的處理了，LED、7-SEG 以及鍵盤如同前幾次的 lab 將所需的部分引用即可，並沒有太大的問題，接下來則是繪製 Finite state machine，這次我使用了 IDLE、player_A、player_B、DEALT、SHOW，IDLE 為發手牌及準備階段；player_A 與 player_B 為玩家下注階段；DEALT 為發公牌的階段；SHOW 為顯示結果的階段。

- IDLE：

  在 IDLE 裡設置了一個 give 的變數進行發牌，在 give=0~2 時將 state 停在 IDLE，每輪取一個 random 並讓 give+1，在 give=3 時讓下一輪進入 player_A 或 player_B，而選擇哪個是靠著 turn 這個變數決定誰是莊家(進入哪個)

- player_A & player_B：

  在 player_A 及 player_B 時，玩家可以進行遊戲下注，玩家能夠利用鍵盤的 C(call/check)、F(fold)、Enter(raise)、backspace(重新輸入)及數字鍵進行控制，詳細功能在下面的鍵盤會有詳細的介紹，此處的 state 轉換與德州撲克的規則相關，頗為複雜，簡單來說，我們使用了一個 done 控制 state 的轉換，當 done=1 時按下 C 進入 DEALT，其餘的則跑到對方玩家的 state 裡。

- DEALT：

  在 DEALT 我們使用了 deal 這個變數進行發公牌的行為，由於第一輪必須發三張牌，因此當 deal=0、1 時讓 state 回到 DEALT，當 2<=deal<5 時則回到 player_A 或 player_B，而回到哪一方則依樣式由 turn 控制，state 會回到不是莊家的那位玩家的回合，接著當 deal=5 時，公牌已發完 5 張，接下來要按下 get_win 的 button 進行勝利判斷並讓 deal=6，當 deal=6 時則讓下個階段到 SHOW。

- SHOW：

  SHOW 這個 state 極為簡單，單純將結果顯示在螢幕上並停留不動，這樣的話在該輪結束後玩家可以打開雙方的牌進行討論後在回到 IDLE 進入下一輪，使得遊戲的進行有了更多的彈性，此處要注意當 SHOW 回到 IDLE 時要將變數回到初始化的階段。

三、功能介紹

我們的遊戲裡，使用了螢幕、鍵盤、FPGA 板這三項，以下將依序介紹各項設備的功能。

1、 螢幕

螢幕與 VGA 是一體兩面的，因此在上面 VGA 所提到的部分大致也等於是在介紹螢幕的功能了，這裡重新強調螢幕的重點也就是在固定的位置不停的轉換不同的圖片，這次螢幕的挑戰在於使用 mux 且不會讓板子燒不進去。

2、 鍵盤

德州撲克所需的行為算是頗多，因此在鍵盤上使用了不少的按鍵，以下將一一介紹各鍵的功能。C 鍵為 call or check，在德州撲克上為跟注與看牌，由於兩件事不會同時出現解性質相同，因此和為同一鍵使用，在 call 時會付出與對方相同的金錢並進入 DEALT，而 check 則是不進行下注選擇看牌；F 鍵為 Fold，即德州撲克的棄牌，使用 Fold 鍵後會將已下注的金錢全數交給對方並重新下一輪；使用數字鍵、Enter 鍵及 backspace 進行 raise 的動作，數字鍵 1、5、0 讓加注金額加 1、5、10 元，增加的結果會顯示在七段顯示器，若不小心超過想要的數字，則使用 backspace 重新輸入加注金額，另外數字 9 提供了 all-in 的功能，按下後在七段顯示器上顯示 ALL 並按 Enter 完成 all-in，在 raise 時下一輪會到對方玩家的回合。

3、 FPGA

FPGA 上我們用到了 switch、led、button 等功能，我們的遊戲是一款雙人遊戲但螢幕只有一個，因此我們使用了最左邊與最右邊的 switch 控制手牌，當 switch 拉起時將手牌顯示卡背，達到遮蔽的效果，另外，由於只有 switch 常常會忽視 switch 是否被拉起，因此當 switch 拉起時，讓 switch 上方的 3 個 led 亮起達到提醒的作用；另外 button 使用了三個，中心的那一個是 rst，按下 rst 後將遊戲回到最一開始，也就是雙方初始 45 塊錢的時候，由 A 玩家為莊家開始；上面那一件事 start 鍵，此鍵有與多功用，基本上所有遊戲的執行都是看這一鍵進行，不論是取隨機、進入下一輪、發牌等功能都是看這一鍵進行，接著是左邊 get_win 這一鍵，由於我們設計上要給入一個訊號才能得到勝負，因此我們特別設立的一個 button 用來獲得勝負，在 deal=5 的時候使用，可以判斷出該局勝負。

四、實作完成度及難易度說明

經過了很多個禮拜的工作加上最後幾天的衝刺，這次的完成度算是十分完整了，由於組員其中一位是德撲的愛好者，在規則上決不讓步，因此努力不懈的修改及訂正後遊戲規則幾乎與真正的德州撲克一樣，在難易度上，最難的顯然就是勝負判定了，除了必須熟知規則外，由於牌型十分複雜又難以判定，負責的組員花了 2 多禮拜思考+打 code，打出了 1500 行 code 成功處理勝負判定，另外在 FSM 上也必須熟知規則才能寫出完整流程，幸虧在規則上的巧合使得 state 的移動簡單的多，但對於不熟之規則的人來說還是不太容易，整體上我們覺得不算太簡單，而且螢幕處理上也需要很多的功夫，花費的時間絕不算少。

五、遇到的困難與解決辦法

我們在這次的 final 中遇到滿多次的危機，在 VGA 方面我們遇到了 2 次稍不進去的問題，第一次我將 52 張卡牌一張一張燒進去，雖然每張圖不大，但

在 mux 時卻需要做到 52 選 1 的電路，再加上有 9 個地方用到，加上其他東西容量就爆炸了，解決它的方式就是將花色與卡片分開，這樣的話 mux 就能分成 13 跟 4，大大減少了 mux 的電路所需的大小，接著在 final 後期加入金錢計算的時候，又一次容量爆炸了，而這次也沒有辦法減少 mux，最後將背景圖的解析度從 360*240->180*120，減少 3/4 的圖容量，還好有成功燒進去，而在 random 方面，在一開始想辦法不讓卡牌重複時遇到了很多困難，在思考如何達到目的時花了非常多的時間在牌型判斷如何節省 cycle 與判斷出正確的答案也耗費了許多時間與心力。，最後，在將兩人 code 合併時又是一大問題，由於我們兩人的 coding style 差異偏大，在合併時花了一點功夫，在分工時盡量將兩人的工作分配到 code 互不相關，讓雙方的 code 可以獨立進行，雖然盡量避免還是會遇到，在這方面花了頗多的時間。

六、心得討論

這一次的 final 都是我們兩個花最多時間的一次 project，在我們討論出德州撲克這遊戲的可行性時就大大提高了我們的興致，在早早規劃完怎麼做、要有哪些功能後，我們便撥空開始做，在前幾個禮拜，其實一位組員一直覺得太過繁瑣及困難想要換主題，但仍被另外一位組員堅持繼續而沒有放棄，在經過好幾次崩潰後在最後幾天集中修補及 debug，在 demo 前一天還有 bug 沒 de 完，在 demo 當天凌晨才成功修完，在趕工的這幾個禮拜，我們過得算是十分水深火熱，不過在 demo 完後看到很不錯得回饋後，我們也覺得十分滿足了，這次的 final 與大一的程設不同，從零開始都由我們自己規劃、實作，完全靠著自己成功做出令人滿意的作品，這樣的經驗對我們來說也是十分難得開心的了！

七、分工

由於這款遊戲需要強大的遊戲知識因此我們的分工將較需要遊戲知識的 code 交給懂的人打，而另外一位則做剩餘部分，雖然分工如下，但仍有不少共同作業或互相幫忙的狀況。

VGA、random、FSM：鐘浩瑋

牌型判斷，金錢計算：曾鼎睿