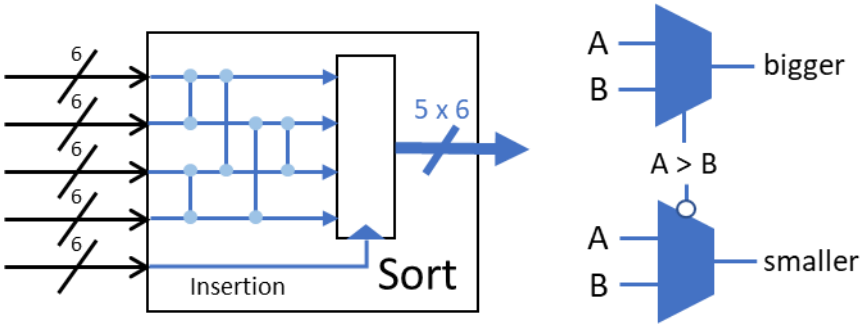
數位電路與系統 HW1 report

110511233 李承宗

由於現在才學期初，沒有考試的壓力，因此這次作業我花了許多時間在研究並嘗試各種組合電路的寫法以及優化方式。

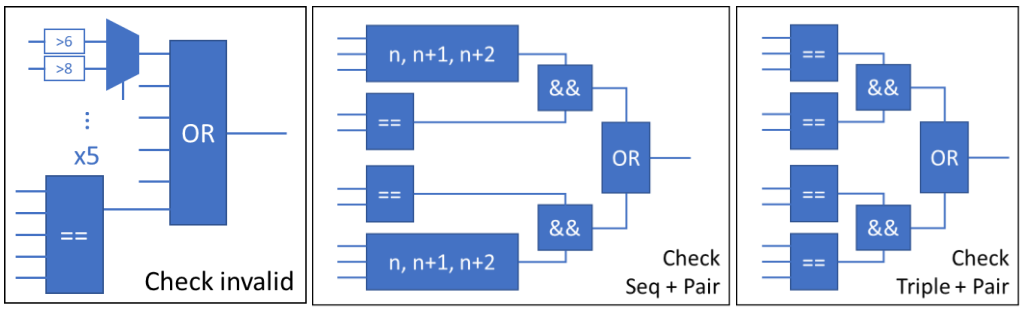
對於這次作業，我最一開始的想法是：兜出三個小模組，分別是檢查輸入是否為 sequence + pair、檢查輸入是否為 triplets + pair 以及檢查是否為有效輸入的電路，再使用編碼/解碼器來將剛剛三個小電路得到的結果轉換成作業要求的輸出格式。

由於輸入的部分可能會有很多種情況，因此我做了一個排序電路來減少做檢查電路時會出現的狀況。



如左圖所示，我做了一個可以將 5 個輸入做排序並且照順序輸出的電路。考慮到數位電路都是使用比較器來達成 2 個值的比比較，並使用選擇器來達成輸出的調換，因此排序單元的設計就如同右圖的電路，能夠做 2 個輸入的排序。由於一個排序單元只能用來做 2 輸入的排序，所以我就把對 5 輸入的排序電路拆成 4 個輸入的比較+排序，再使用得到的結果來判斷剩下的輸入要被排在哪一個位置。

相較於 Lab2 的 Merge Sort 排序方式會使用到 10 個比較器， 4 輸入排序+插入的排序會少用一個比較器 (插入會使用到 4 個比較器來讓最後一個輸入跟其他 4 個排序好的值做比較，因此只需要 9 個)，因此佔用的資源會比較少。

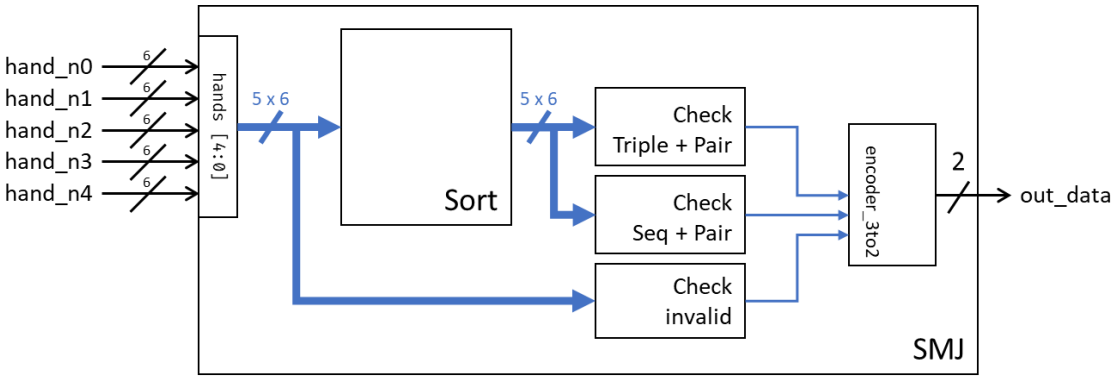


做完排序之後，檢查電路就變得簡單許多。檢查輸入可以分成兩種情況：是字牌、不是字牌。如果是字牌，就檢查號碼是否大於 6；如果不是字牌，就檢查號碼是否大於 8。因此檢查單元就可以用一個與常數比較的比較器（應該會被優化成 encoder？）以及選擇器，然後只要一張牌無效，整組輸入就會被視為無效，因此最後會把五個輸入的結果做 OR，加上輸入全部一樣的情況。

由於值都排序好了，因此檢查是否為 sequence + pair 的邏輯就可以簡化很多，只需要判斷有沒有一串連續的數字加上兩個相等的牌就可以做出來（實際上還有五個輸入皆為萬子的情況，因此 OR 前面需要多一組判斷電路）。

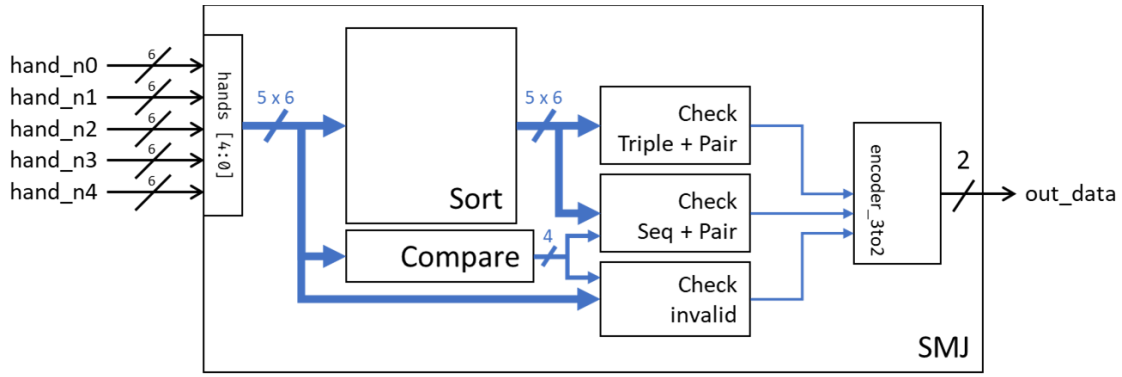
檢查 triplets + pair 就更簡單了，只需要檢查輸入是否為 3+2 或 2+3 的情況即可。

電路整體的架構如下：

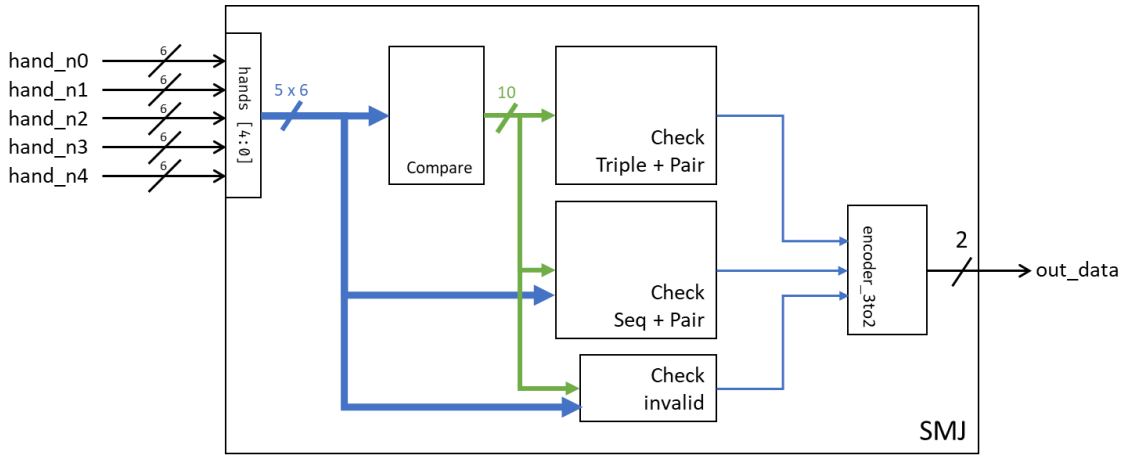


整個電路做出來之後的面積大概在 5500 左右，途中我想到了比較的部分會重複使用到（在檢查無效輸入跟 triplets + pair 的部分），因此我將比較的電路拉出來共用。另外我也有考慮到比較器在輸入 bit 數比較高的時候會變得很複雜，因此我嘗試著使用一些方式來兜出二元比較的電路，最後想到可以用 bitwise XOR 來判斷兩數是否相等（若兩數相等，則每 bit 做 XOR 出來的結果全為 0），因此我使用 ~|(A^B) 的方式來替代 A == B。出來的結果跟我想像中的差不多，在判斷的 bits 數比較高的時候面積會稍微下降，但如果 bits 數少的話可能就都會被 EDA 工具優化成 encoder，所以沒有影響。

最終優化出來的電路架構如下，合成出來的面積大約在 5100 左右，相較一開始寫的算是進步一些。



但是，在 Lab2 的時候我發現，其實比較跟排序會使用到非常多的資源來合成。因此，我開始想有沒有可以不使用排序，就能得到結果的電路。經過一番分析，我發現在沒有排序的情況下，會出現 triplets + pair 的輸入組合其實最多也只有十種 case 需要檢查（C_2^5種排列組合），sequence + pair 如果在不考慮 sequence 的順序也只有十種輸入組合，好像也沒有比排序完後再做判斷多出很多 case。因此，我就有了一個大膽的想法：把每種 case 都排列出來。



雖然程式上的邏輯判斷會變得很多，但是大部分都只是在比較相等，也就是在判斷 A==B，加上這些部分我在上一次做好的電路就已經獨立出來了，使用~|(A^B)這種奇怪的方法來取代 A==B 甚至可以讓比較單元變得更小，因此我就開始驗證我的假設。

把各種情況都列出來之後（判斷方式跟一開始的設計一樣，只有多考慮判斷電路的輸入組合會有不同排列的情況。），我發現出來的面積降到了 4853，看來排序電路確實浪費了不少資源。但我覺得只是因為這次需要檢查的條件（贏的條件）不多，而且輸入也只有五組，因此就算採用窮舉法，需要做判斷的 case 也不多。如果輸入數量、贏的情況變多的話，排序之後再判斷才是比較好的做法。

在這次作業中，由於對之前邏輯設計還有點印象，所以我還不至於兜出很奇怪的電路；但在 Verilog 上我可能還不夠熟悉，總是將 task 跟 function 搞混，不然就是少打 begin...end 而導致 EDA Tool 產生奇怪的 error。總而言之這次作業雖然難度不高，但還是有讓我開始回想起之前的邏輯設計，思考怎麼設計能讓使用到的資源變少（雖然跟別人比起來我的設計應該還是很浪費資源 XD），而且也讓我對 Verilog 越來越熟悉，coding style 應該也相比之前好了一些，想必下次作業能寫的更得心應手。