|  |  |
| --- | --- |
| **Lab 6** | |
| 學號: 108032053 | 姓名: 陳凱揚 |

1. 實作過程

在此題中，我使用原始的100MHz作為clock，以接收來自鍵盤的訊號，並及時以LED更新B1、B2車站等待的人數和目前車上的人數，再使用數clock cycle的方式，將其他flip-flop，如state、SevenSegment等能夠以100/(2^27)MHz的頻率更新，以下圖1的next\_state為例，只有在cnt == CNT時，才會更新next\_state（cnt為數clock的filp-flop，CNT為{27{1’b1}}）。

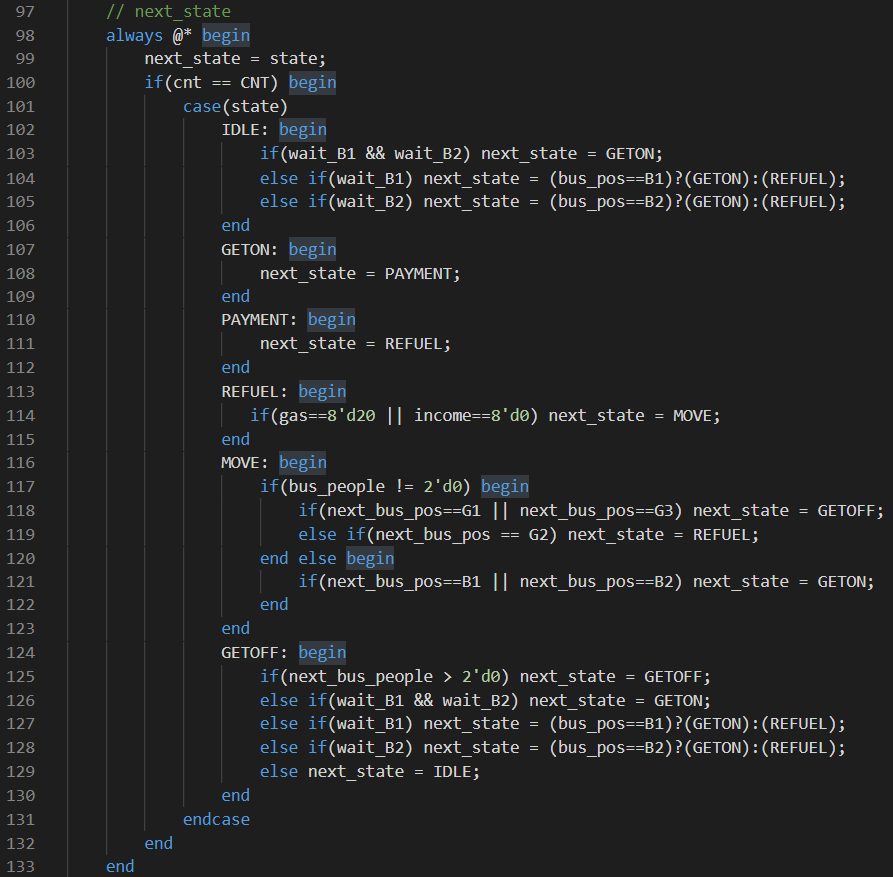
下圖2為我建立的BCD module，用來轉換以binary表示的gas和income至BCD，由於這些數字最大只有2位數，所以只要簡單判斷十位數字，就能得知個位數，並輸出此兩位數的BCD。

下圖3為我判斷鍵盤輸入的方式，當been\_ready和key\_down[last\_change] == 1’b1時，代表目前有個按鍵被按下了，接著判斷這個鍵是數字1（KEY\_CODE\_1）或數字2（KEY\_CODE\_2），並對應改變B1和B2車站的等待人數，其他鍵按下則不會有反應。下圖4則是KeyboardDecorder的接線方式

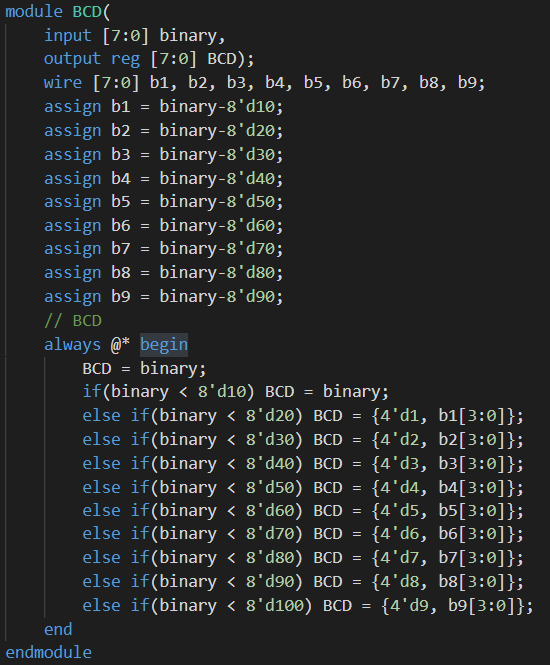
在module lab6中，我使用了11個flip-flop，功能分別如下：

1. state：記錄目前狀態，有6個狀態，state diagram如下圖5所示，分別為IDLE（閒置）、GETON（上車）、PAYMENT（付錢）、REFUEL（加油）、MOVE（移動）、GETOFF（下車）。
2. bus\_pos：記錄目前車子的位置，根據dir決定移動方向。
3. wait\_B1、wait\_B2：B1、B2車站等待人數，可由鍵盤輸入增加人數，最多至2人。
4. bus\_people：記錄目前車上的人數，在上下車時增減人數，最多至2人。
5. gas：記錄目前的油量，每人經過每個加油站減5，每次加油加10，最多至20。
6. income：記錄目前的金錢，每人上山加30，下山加20，每次加油減10，最多至90。
7. nums：16bits中前8bits為gas\_BCD，後8bits為income\_BCD。
8. LED：[15:14]、[12:11]、[10:9]分別為B1等待人數、B2等待人數、車上人數，[6:0]中只有車子所在位置會亮，其他都是暗的，[13]、[8:7]則是永遠保持暗的。
9. dir：記錄目前車子方向，有L跟R兩種。
10. cnt：27bits，每經過一個clock就加一。

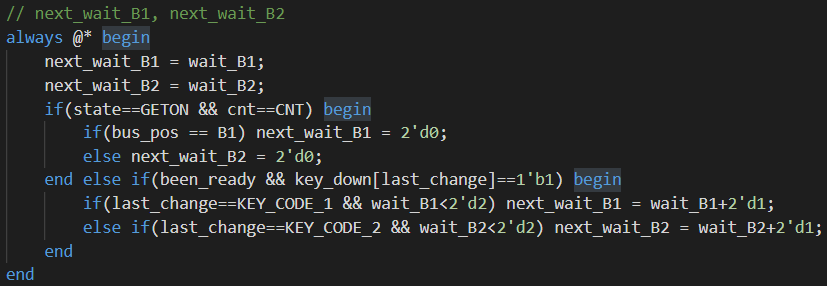
下圖5、圖6分別為state diagram和block diagram。



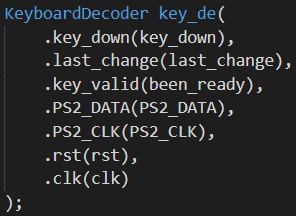
▲ 圖1



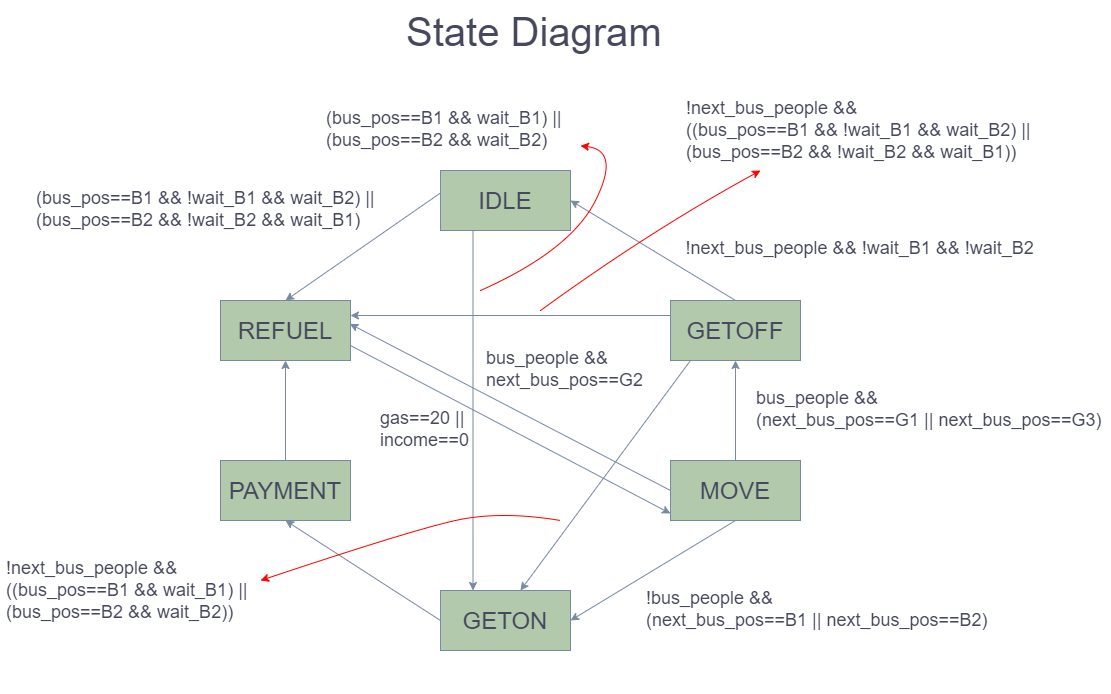
▲ 圖2



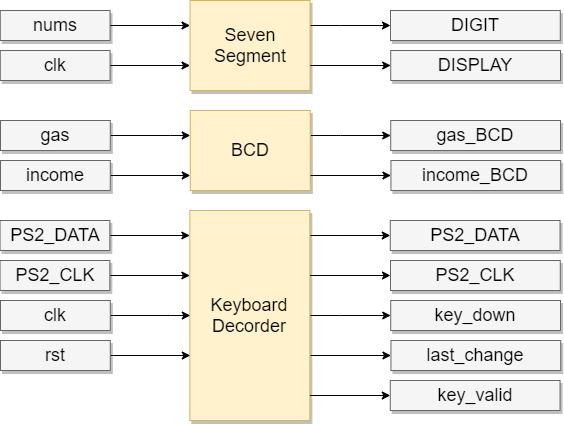
▲ 圖3

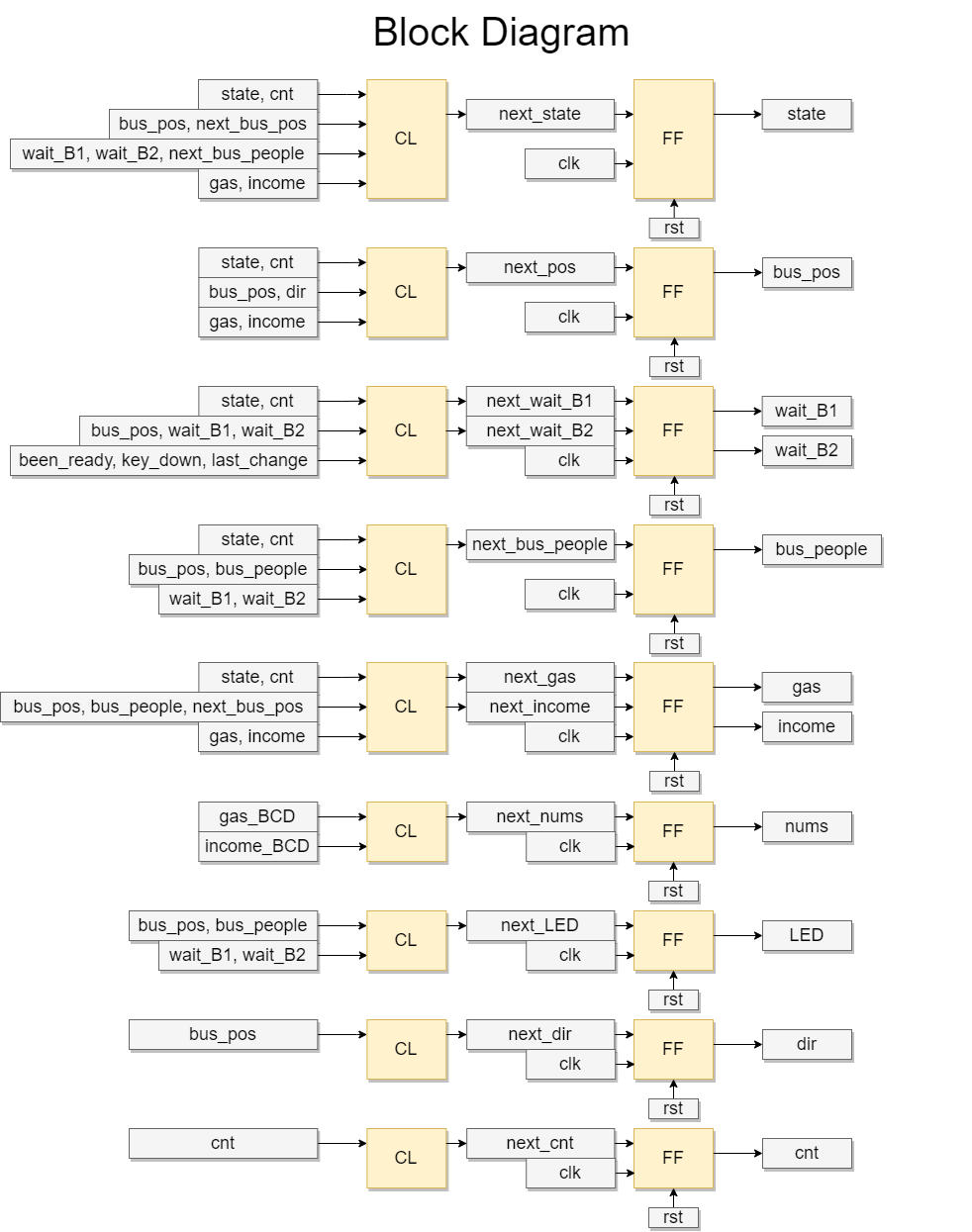


▲ 圖4



▲ 圖5





▲ 圖6

1. 學到的東西與遇到的困難

在這次lab中，我覺得FSM的複雜度與前幾次不會差太多，但多了不少條件判斷，而且需要自己完整從零設計出來，不像前幾次都有既定流程，我覺得這讓我們可以對設計流程更加熟悉，畢竟通常要求的是功能，而不會有人幫我們設定好完整流程。而我覺得這次最大的困難在於不同clock的整合，因為處理鍵盤的輸入，需要100MHz的clock，而車站的等待人數不僅可以使用鍵盤馬上增加人數，也需要除頻過的clock，在上車時每隔一段時間減少等待人數。一開始我使用兩種clock的混用，但處理相對複雜，也出現了不少bug無法解決，接著我改採只使用100MHz的clock，加上數clock cycle的方式，才順利的完成所需的功能，因此我認為整個design還是只使用一個clock還是會比較好，不應該偷懶使用多個clock，造成許多問題。

1. 想對老師或助教說的話

