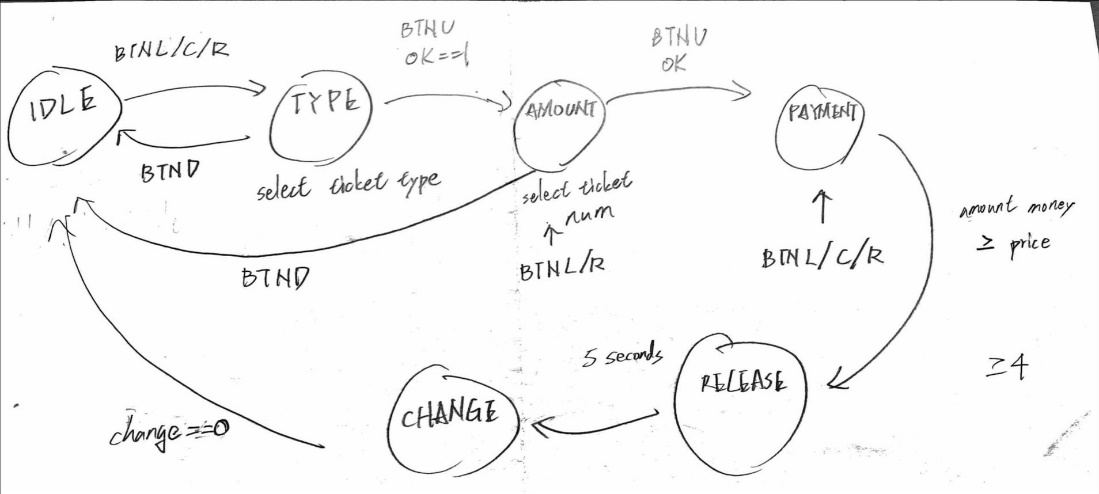
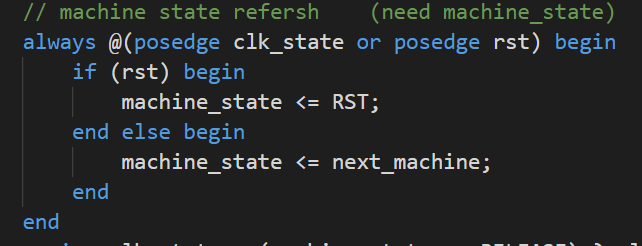
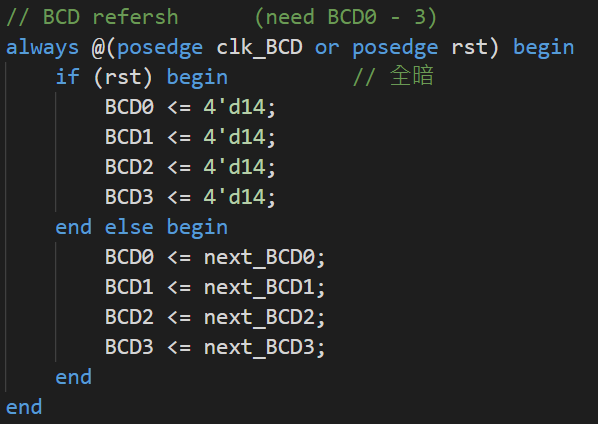
|  |  |
| --- | --- |
| **Lab 5** | |
| 學號: 109062202 | 姓名: 陳禹辰 |

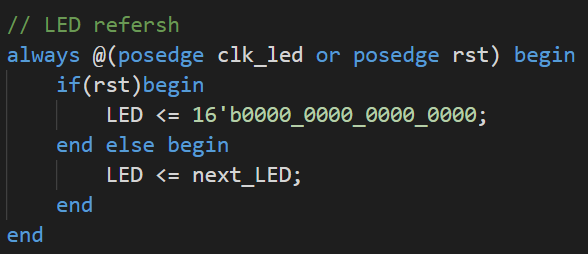
1. 實作過程

因為這次lab裡面所包含的state比之前得多，所以我就在開始寫之前先畫了FSM圖來幫助我了解整個machine在不同state要做的事情，以及在各個state內遇到什麼情況時要進到的下一個state會是什麼。



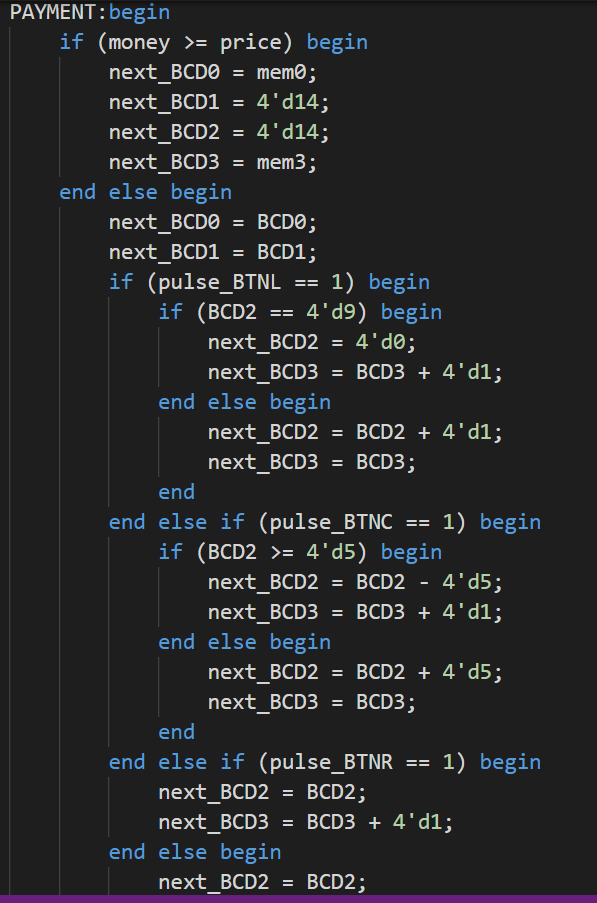
總共包含6個state，分別是IDLE, TYPE, AMOUNT, PAYMENT, RELEASE, CHANGE然後在不同的state會有某些條件使得他跑到下一個state，而且在不同的state內7-segment或LED的更新時間所需要用到的clk也是會也不同，所以也要根據所在的state來更改clk。因此，我把整個架構分成四個 machine : 負責處理machine state的改變，BCD : 負責處理7-segment的變化，LED : 負責處理LED的變化，tool : 負責更新個個state內所需要用到的東西(紀錄所需要付的錢, 紀錄總共的票數, etc )，然後根據machine所在的state來決定每個state內BDC與LED該顯示什麼或做什麼樣的的事情。接下來我會分別解釋我在不同state內的實際操作情形。





首先是IDLE state，在這個state內7-segment與LED的燈需要以0.5hz來閃爍，也就是一秒亮一秒暗，所以在這個state內7-segment與LED的block我會將clk換成0.5hz的clk，然後BCD的處理就是如果現在的BCD是顯示亮的話next\_BCD就顯示暗的，反之則顯示中間有亮的。而LED也是一樣與現在的顯示相反，也就是next\_LED = ~LED。而machine的clk則需要用比較快的clk，因為他要在按下BTNL, BTNC, BTNR時更新到TYPE state內，如果clk不夠快的話有可能按下去之後，還要過一段時間sequential block才會將next\_machine\_state assign to machine\_state或是按了之後沒有反映之類的情況。

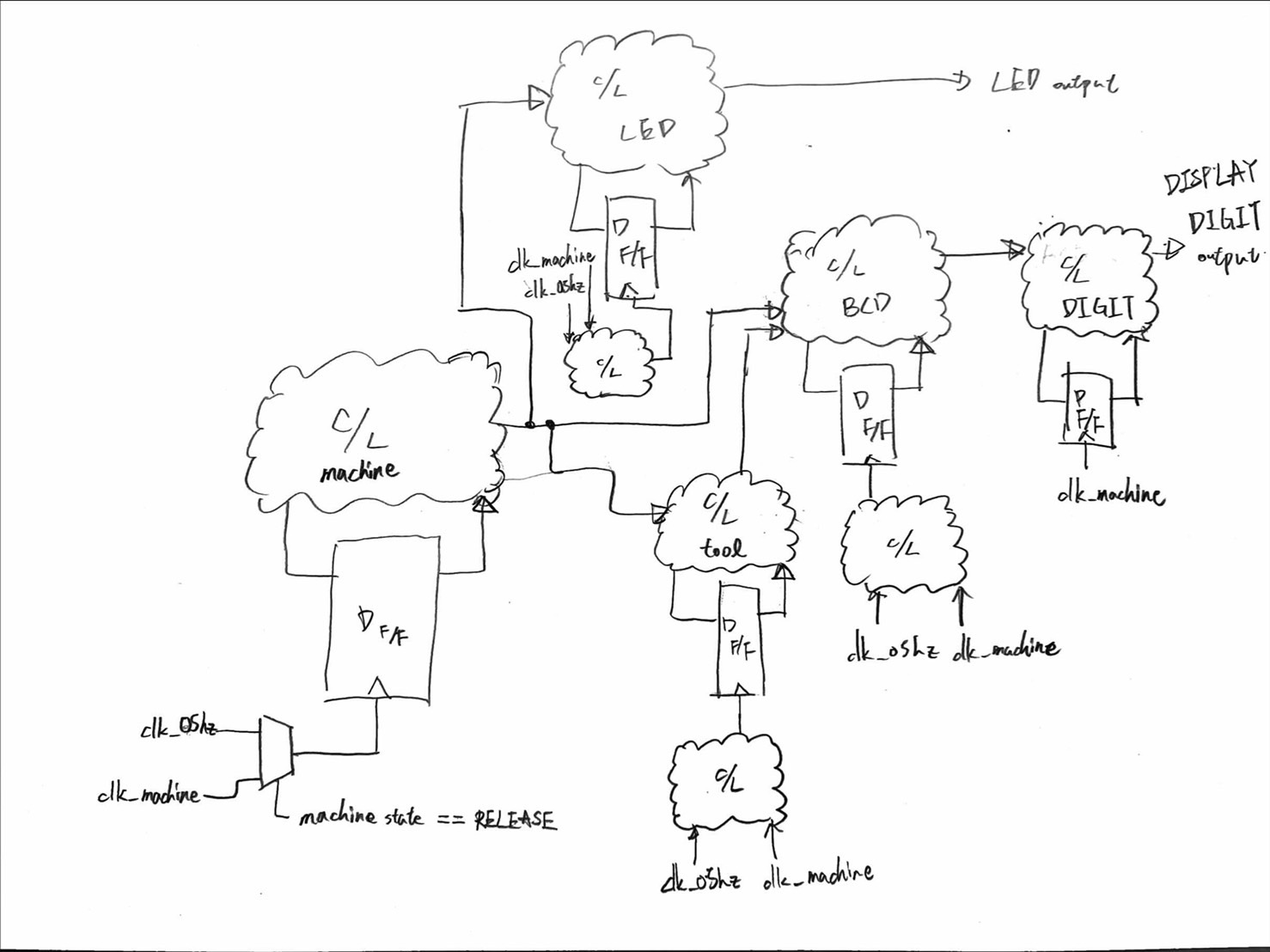
再來是TYPE state，在這個state內如果按下BTNL, BTNC, BTNR會改變7-segement所要顯示的東西，所以我就讓他判斷如果按下了哪一個按鈕，他就會顯示先對應的，例如按下BTNL則要顯示C 05那就是next\_BCD3 顯示C, next\_BCD2 顯示暗的, next\_BCD1 顯示 0, next\_BCD0顯示1，結著再由sequential block assign next\_BCD to BCD，然後因為這個state是按下按鈕就要有反應，所以我就讓BCD的sequential block’s clk 會跟machine 所用的clk 是相等的。接著，如果按下ok(BTNU)則會進到AMOUNT，如果按下cancel則會回到IDLE。

接著是AMOUNT state，需要留著TYPE state內所選擇的票的種類，所以我在一進來這個state時next\_BCD3 = BCD3 這樣即使state有改變我最左邊所顯示的東西還是會一樣，接著就是按下BTNR時數量會增加，按下BTNL時數量會減少並且票的數量只能在1到3之間，我是讓他按下按鈕時判斷現在的數量是多少如果是在1到3之間才能做運算，因為一樣是一按下按鈕就要更新顯示，所以clk 一樣是用跟machine 一樣的。然後我會用一個price在AMOUNT state去儲存總共的票價，就是各個票種的價錢 \* 票的數量，這樣才能在之後的state時當作使用。接著，如果按下ok(BTNU)則會進到PAYMENT，如果按下cancel則會回到IDLE。

接著是PAYMENT state，在這個state時要進行的操作是付錢，按下BTNL時付1塊錢，按下BTNC時付5塊錢，按下BTNR時付10塊錢，就是把money(已付的金額)的值往上加然後個位數表示在BCD2十位數則是表示在BCD3，如果付的錢超過price(總共的票價)則進入到RELEASE，如果是按下cancel(BTND)則是直接進入到CHANGE。因為一樣是一按下按鈕就要更新顯示，所以clk 一樣是用跟machine 一樣的。

接著是RELEASE state，這個state要顯示選擇的票種跟相對應的價錢，我是用在TYPE state所記錄的來讓這邊顯示前面所選擇的票種，然後LED燈要閃五秒，我就是用跟在IDLE一樣的做法讓他閃爍，不同的是我有用一個cycle來記錄過了多少時間，如果cycle超過5秒，就進到CHANGE state

接著是CHANGE state，在這個state會將錢(付出的金額 – 票的總價 or 付的金額)找出，如果是大於5時則每次都-5，不是的話則一次扣一，直到退錢到0為止，我的作法是先判斷是要退多少錢，因為有可能是直接從PAYMENT進來的，所以先看money有沒有>=price如果有的話就讓money = money – price沒有的話就直接把money丟過來，然後再慢慢扣，扣完之後就回到IDLE。因為是要讓她一秒退一次錢所以這個state我一樣是用0.5hz的clk來進行。



我是讓machine 去輸出現在所在的state然後讓BCD, tool, LED根據machine state去做相對應的動作，不斷更新裡面的值，然後每個block有可能因為所在的state不同而有不同的clk。

1. 學到的東西與遇到的困難

這次因為大部分的東西就是之前幾次lab的內容，只是把全部放在一起操作，所以實作上並沒有遇到太大的困難，大多數是一些條件沒考慮好。

1. 一開始在IDLE state內LED跟7-segement閃爍不同步，我後來是讓他在進到IDLE時在同時有一個初始值，然後根據同樣的clk\_05hz更新就可以了。
2. 在IDLE跑到TYPE時有時候會需要按兩次按鈕才會跳過去，原因是我原本在BCD內也是用0.5hz的clk所以就會造成按鈕按了，machine state更新了，但是BCD還沒有即時更新到(因為還沒到BCD clock的posedge)，後來就是讓BCD在IDLE時是用machine 一樣的clk就沒有問題了。
3. RELEASE在第五秒時會還沒數完第五秒就結束(第五秒時LED閃了一下就暗掉)，這邊我是用來記錄屬幾次的cycle判斷上有問題，後來改成是從1開始數他如果一數到6就結束，這樣就會剛好是數完完整的5秒鐘。
4. CHANGE扣錢的問題，原本在某些錢的狀況下，他會變成無限回圈，就可能46, 41, 36, 31, 26, 21, 16, 11, 46, 41 ….之類的，後來發現是因為我要讓他在這個state先過一秒在扣，然後我就寫如果cycle > 0 才開始，但cycle也會一直數，我原本cycle只有開reg[2:0]，但這樣他如果數到111的話，下一個cycle又會變回0，然後我是在cycle給定初始值的，所以就扣錢扣一扣又會回到初始值，後來是把cycle開大一點就可以解決這個問題了。

剩下的可能就一些小小的問題，比較快就找到了。

1. 想對老師或助教說的話

謝謝助教辛苦看了那麼多份report，聽朋友說看report的助教很喜歡貓貓狗狗，所以就付上幾張可愛的貓貓照片。然後雖然離期中考已經有一段時間了，但還是小小抱怨一下，好像寫的時間不太夠的樣子…，來不及把第三題打完阿阿阿阿阿QQ。

