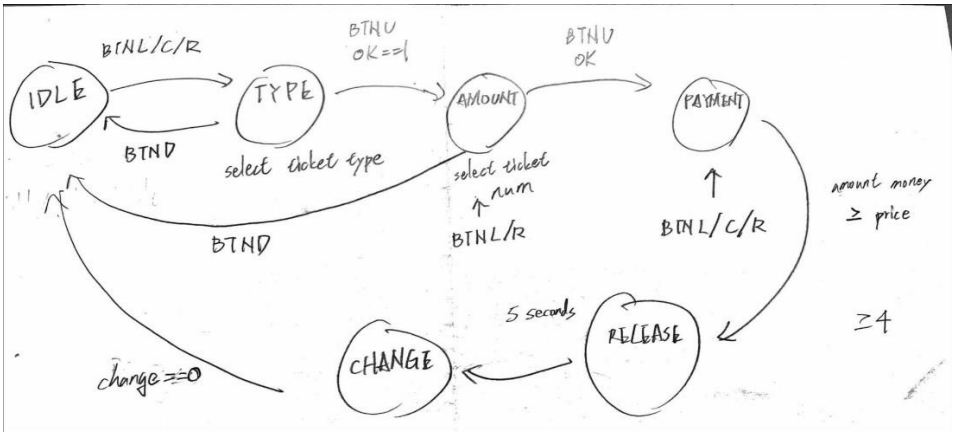
Lab 5

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。

```
// machine state refersh    (need machine_state)
always @(posedge clk_state or posedge rst) begin
    if (rst) begin
        machine_state <= RST;
    end else begin
        machine_state <= next_machine;
    end
end
// LED refersh
always @(posedge clk_led or posedge rst) begin
    if(rst)begin
        LED <= 16'b0000_0000_0000_0000;
    end else begin
        LED <= next_LED;
    end
end
```

```
// BCD refersh      (need BCD0 - 3)
always @(posedge clk_BCD or posedge rst) begin
    if (rst) begin          // 全暗
        BCD0 <= 4'd14;
        BCD1 <= 4'd14;
        BCD2 <= 4'd14;
        BCD3 <= 4'd14;
    end else begin
        BCD0 <= next_BCD0;
        BCD1 <= next_BCD1;
        BCD2 <= next_BCD2;
        BCD3 <= next_BCD3;
    end
end
```

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。

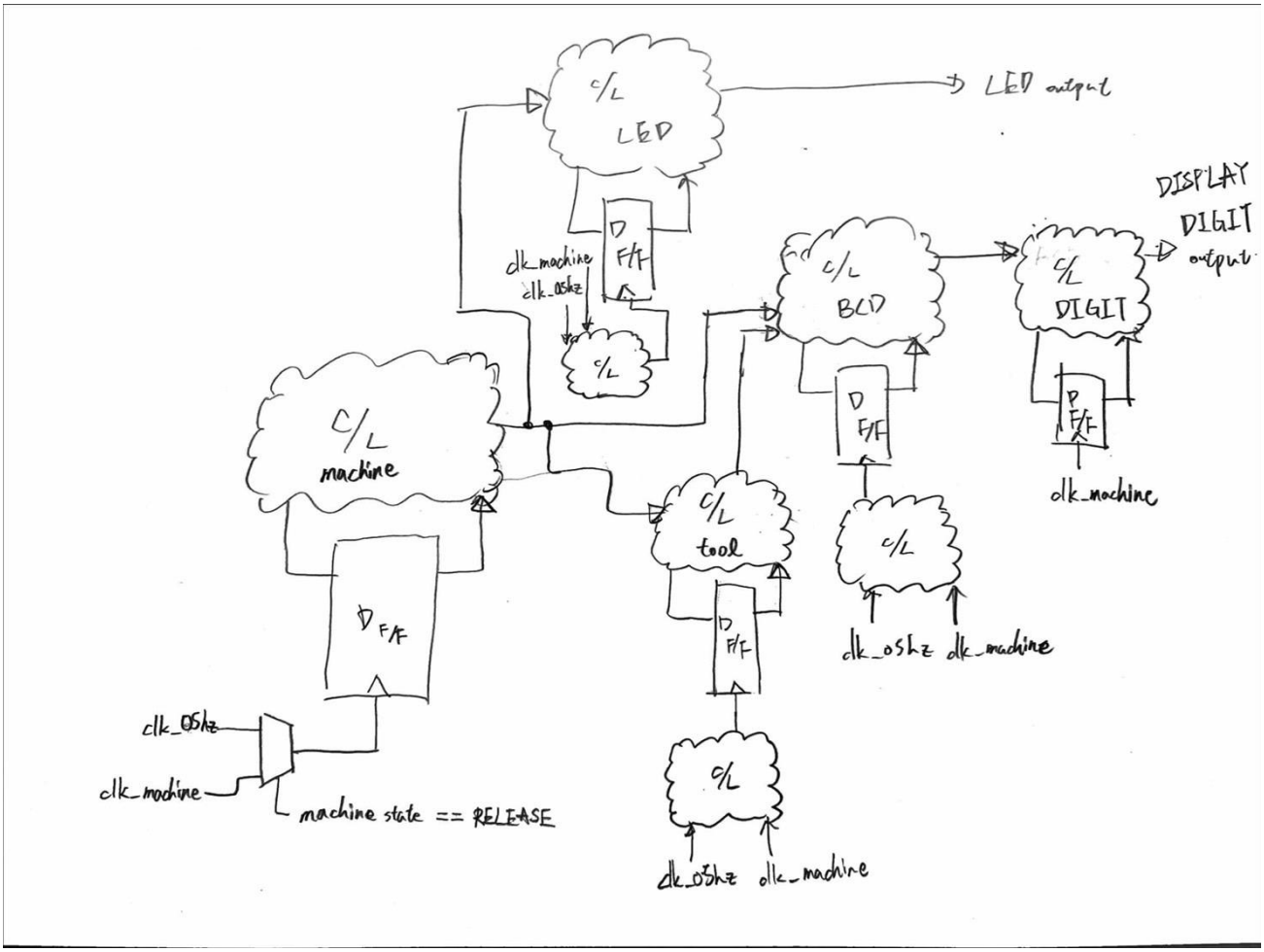
接著是 AMOUNT state，需要留著 TYPE state 內所選擇的票的種類，所以我在一進來這個 state 時 next_BCD3 = BCD3 這樣即使 state 有改變我最左邊所顯示的東西還是會一樣，接著就是按下 BTNR 時數量會增加，按下 BTNL 時數量會減少並且票的數量只能在 1 到 3 之間，我是讓他按下按鈕時判斷現在的數量是多少如果是在 1 到 3 之間才能做運算，因為一樣是一按下按鈕就要更新顯示，所以 clk 一樣是用跟 machine 一樣的。然後我會用一個 price 在 AMOUNT state 去儲存總共的票價，就是各個票種的價錢 * 票的數量，這樣才能在之後的 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。

```
PAYMENT:begin
    if (money >= price) begin
        next_BCD0 = mem0;
        next_BCD1 = 4'd14;
        next_BCD2 = 4'd14;
        next_BCD3 = mem3;
    end else begin
        next_BCD0 = BCD0;
        next_BCD1 = BCD1;
        if (pulse_BTNL == 1) begin
            if (BCD2 == 4'd9) begin
                next_BCD2 = 4'd0;
                next_BCD3 = BCD3 + 4'd1;
            end else begin
                next_BCD2 = BCD2 + 4'd1;
                next_BCD3 = BCD3;
            end
        end else if (pulse_BTNC == 1) begin
            if (BCD2 >= 4'd5) begin
                next_BCD2 = BCD2 - 4'd5;
                next_BCD3 = BCD3 + 4'd1;
            end else begin
                next_BCD2 = BCD2 + 4'd5;
                next_BCD3 = BCD3;
            end
        end else if (pulse_BTNR == 1) begin
            next_BCD2 = BCD2;
            next_BCD3 = BCD3 + 4'd1;
        end else begin
            next_BCD2 = BCD2;
```

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。

2. 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。

3. 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。