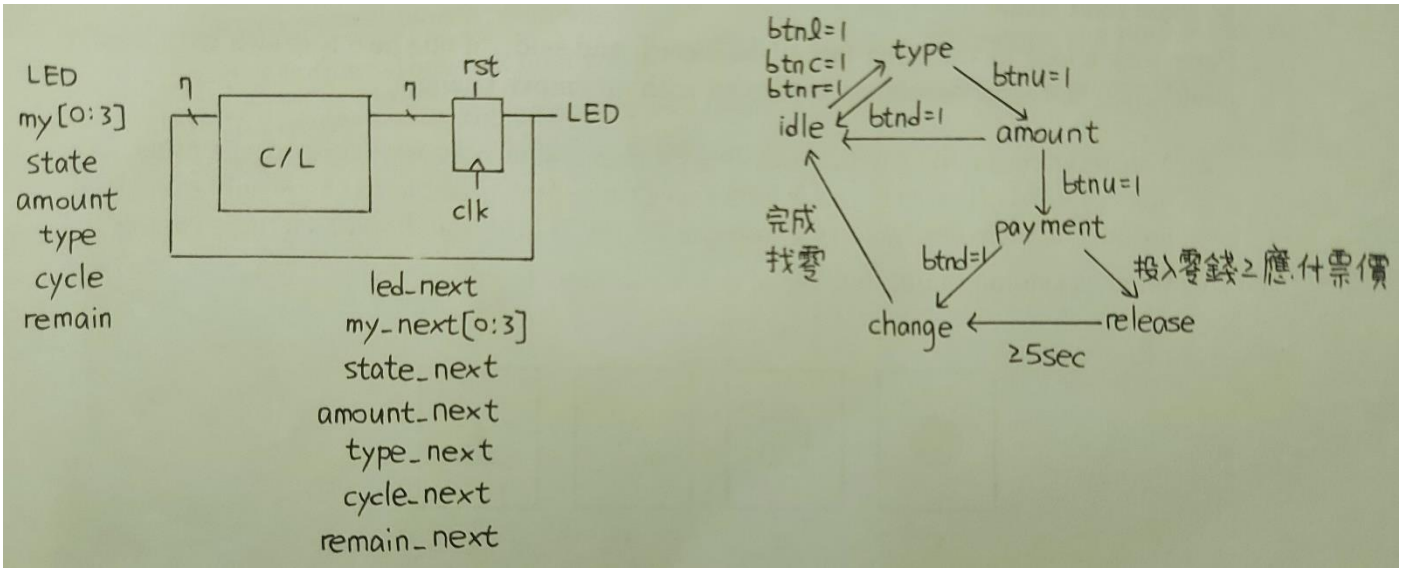
# Lab 5

學號: 109062318 姓名: 簡弘哲

1. 實作過程




在這次 lab 中的 FSM 總共有 6 個 state，分別是 IDLE,TYPE,AMOUNT,PAYMENT,RELEASE,CHANGE，我覺得這些 state 就已足夠。至於 block diagram 由於變數有點多，所以我就用斜線上面標記數字 7 的方法來表示有 7 個 input。

IDLE:

```
if(state==IDLE) begin
    if(led_clk_1p) begin
        for(i=0;i<4;i=i+1) begin
            my_next[i] = (my[i]==DARK) ? 10 : DARK;     //flash
        end
        led_next = ~LED;
    end

    if(btnl_1pulse || btnc_1pulse || btnr_1pulse) begin
        state_next=TYPE;
        if(btnl_1pulse) begin
            type_next=CHILD;
            my_next[0]=CHILD;
            my_next[1]=DARK;
            my_next[2]=0;
            my_next[3]=5;
        end else if(btnc_1pulse) begin
            type_next=STUDENT;
            my_next[0]=STUDENT;
            my_next[1]=DARK;
            my_next[2]=1;
            my_next[3]=0;
        end else if(btnr_1pulse) begin
            type_next=ADULT;
            my_next[0]=ADULT;
            my_next[1]=DARK;
            my_next[2]=1;
            my_next[3]=5;
        end
    end else begin
```

上面的 if(led_clk_1p)負責 led 與 7segment 的閃爍，其中 led_clk_1p 是每秒產生”將 1 維持一個 clock cycle”的訊號。DARK 則是 DISPLAY 的設定(DISPLAY=7'b111_1111)，從右到左是 GFEDCBA，讓它都是暗的。

下面的 if 負責 state 的轉換，並且在轉換之前先把 7segment (my 這個變數負責顯示 7 segment) 該顯示的數字先設定好，以確保在進入下一個 state 的時候數字不會亂掉(其他 state 的轉換也是一樣的精神，先把 7segment 設定好再跳過去那個 state，在 report 底下說明其他 state 的時候就不贅述)。其中 ADULT,STUDENT,CHILD 都是設定 7segment 的字母 A,S,C，為了方便理解與 debug 才將它 parameterize。

TYPE:

```
else if(state==TYPE) begin
my_next[1]=DARK;
led_next=0;
type_next=type;
if(btnl_1pulse) begin   //child 5
    type_next=CHILD;
    my_next[0]=CHILD;
    my_next[2]=0;
    my_next[3]=5;
end else if(btnc_1pulse) begin  //student 10
    type_next=STUDENT;
    my_next[0]=STUDENT;
    my_next[2]=1;
    my_next[3]=0;
end else if(btnr_1pulse) begin  //adult 15
    type_next=ADULT;
    my_next[0]=ADULT;
    my_next[2]=1;
    my_next[3]=5;
end
```

在 TYPE state 就根據使用者按下哪個按鈕，7segment 就顯示相對應的字母與價錢就 ok 了。

這裡的 type_next 要存起來是因為之後 RELEASE 會用到這個資訊，然後 led 要關掉。

AMOUNT:

```
else if(state==AMOUNT) begin
my_next[0]=my[0];   //maintain the type
my_next[1]=DARK;
my_next[2]=DARK;
my_next[3]=my[3];
led_next=0;
type_next=my[0];
amount_next=my[3];

if(btnl_1pulse && my[3]>1) begin    //-1
    amount_next=my[3]-1;
    my_next[3]=my[3]-1;
end else if(btnr_1pulse && my[3]<3) begin   //+1
    amount_next=my[3]+1;
    my_next[3]=my[3]+1;
end else begin
    amount_next=my[3];
    my_next[3]=my[3];
end
```

在決定購票數量的時候，我有加上數量判斷(my[3]負責顯示 amount)以防止超過 3 或小於 1 的情況發生，這裡變數 amount 要記住買了幾張，因為在顯示應付價格時需要知道這個資訊，led 要全暗。

```
if(btnu_1pulse) begin
    state_next=PAYMENT;
    my_next[0]=0;
    my_next[1]=0;

    //show the needed money(6 situations in total)
    if(type==CHILD && my[3]==1) begin //child*1 => $5
        my_next[2]=0;
        my_next[3]=5;
    end else if((type==CHILD && my[3]==2) || (type==STUDENT && my[3]==1)) begin   //child*2, student*1 => $10
        my_next[2]=1;
        my_next[3]=0;
    end else if((type==CHILD && my[3]==3) || (type==ADULT && my[3]==1)) begin   //child*3, adult*1 => $15
        my_next[2]=1;
        my_next[3]=5;
    end else if(type==STUDENT && my[3]==2) begin   //student*2 => 20
        my_next[2]=2;
        my_next[3]=0;
    end else if((type==STUDENT && my[3]==3) || (type==ADULT && my[3]==2)) begin   //student*3, adult*2 => 30
        my_next[2]=3;
        my_next[3]=0;
    end else if(type==ADULT && my[3]==3) begin   //adult*3 => 45
        my_next[2]=4;
        my_next[3]=5;
    end
```

值得一提的地方是在從 AMOUNT 轉 PAYMENT 的時候，因為沒辦法用/,%，所以十位數跟個位數要分開處理。只好先算價錢總共有幾種可能，算出來是 6 種(5,10,15,20,30,45)，就把相同價錢的組合一起寫在一個 if 裡面，像是 child*2 跟 student*1 的價錢一樣，那就把它們寫在同一個 if 裡。另外 led 要全暗。

PAYMENT:

```
end else if(state==PAYMENT) begin
    led_next=0;

    //deal with input money
    if(btnl_1pulse) begin   //1
        if(my[1]==9) begin  //09=>10
            my_next[1]=0;
            my_next[0]=my[0]+1;
        end else begin  //01=>02
            my_next[1]=my[1]+1;
            my_next[0]=my[0];
        end
    end else if(btnc_1pulse) begin  //5
        if(my[1]>=5) begin  //16=>21
            my_next[1]=my[1]+5-10;
            my_next[0]=my[0]+1;
        end else begin  //14=>19
            my_next[1]=my[1]+5;
            my_next[0]=my[0];
        end
    end else if(btnr_1pulse) begin  //10
        my_next[1]=my[1];
        my_next[0]=my[0]+1;
    end
```

在付錢的時候須注意進位的情形，+1$要檢查個位數(my[1])是不是 9，+5$的時候則要檢查個位數

>=5，+10$什麼都不用檢查(因為價錢最多 45$，不會再更高了)，直接十位數+1 即可，led 保持全暗。

RELEASE:

```
end else if(state==RELEASE) begin
    my_next[0]=type;
    my_next[1]=DARK;
    my_next[2]=DARK;
    my_next[3]=amount;
    if(led_clk_1p) begin
        led_next = ~LED;
    end

    if(onesec_clk) begin
        cycle_next=cycle+1;
    end
    if(cycle>5) begin
        my_next[0]=DARK;
        my_next[1]=DARK;
        my_next[2]=0;   //the max
        my_next[3]=remain;
        state_next=CHANGE;
    end else begin
        state_next=state;
    end
```

這個 state 需要 flashing( ~LED )以及數 5 秒再進到 CHANGE，所以 led_clk_1p 所產生的訊號負責反轉 LED 燈、變數 cycle 負責數秒，每秒+1。其中 onesec_clk 是一個每秒產生”持續一個 clock cycle 的 1”之訊號。當 cycle 數超過 5 秒時，就可以進入 CHANGE state 了，反之就維持在 RELEASE。

CHANGE:

```
if(onesec_clk) begin    //slow down
    if(my[0]==DARK && my[1]==DARK && my[2]==0 && my[3]==0) begin
        //go to IDLE state,reset everything
        for(i=0;i<4;i=i+1) begin
            my_next[i]=10;
        end
        led_next=65535;
        state_next=IDLE;
        amount_next=0;
        type_next=ADULT;
        cycle_next=0;
        remain_next=0;
    end else begin
        if(10*my[2] + my[3] >= 5) begin //if the total>=5,then de
            if(my[3]<5) begin
                if(my[2]>0) begin   //14-5=09
                    my_next[3]=my[3]+10-5;
                    my_next[2]=my[2]-1;
                end else begin  //09-5=04
                    my_next[3]=my[3]-5;
                    my_next[2]=my[2];
                end
            end else begin  //18-5=13
                my_next[3]=my[3]-5;
                my_next[2]=my[2];
            end
        end else begin  //decrease by 1
            my_next[3]=my[3]-1;
            my_next[2]=my[2];
        end
```

紅色框框是找完零錢的時候要準備進入 IDLE 的前置工作，將 led,7segment 都亮起來，其他變數都重置(圖中的 amount_next 應改為 1，因 amount 初始值是 1)。

藍色框框則是找零的 logic，先判斷要 return 的錢有沒有>=5，如果有就可以先 return 5 元再 return 1 元，只是這邊也需要注意退位的情況，找 5 塊的時候要注意個位數<5 ?如果是就要退位，退位的時候也要檢查十位數是否>0 ?如果十位數是 0 就繼續維持 0，否則就-1。

return 1 元的時候就不用檢查退位了，因為要 return 1$只有 4 種情況(01,02,03,04)，所以可以直接個位數-1。

2. 學到的東西與遇到的困難

這次很幸運地幾乎沒什麼遇到困難，一天內就搞定了，只有在設計的時候花了一些時間想該怎麼處理票的數量、種類(child,student,adult)以及找零，其中算出找零該找多少是我花最多時間去思考的。

```
if(10*my[0]+my[1] >= 10*my[2]+my[3]) begin
    my_next[0]=type;
    my_next[1]=DARK;
    my_next[2]=DARK;
    my_next[3]=amount;
    remain_next = (10*my[0]+my[1]) - (10*my[2]+my[3]);
    state_next=RELEASE;
end else if(btnd_1pulse) begin
    my_next[0]=DARK;
    my_next[1]=DARK;
    my_next[2]=my[0];
    my_next[3]=my[1];
    remain_next = 10*my[0] + my[1]; //if cancelled,return the deposited money
    state_next=CHANGE;
end else begin
    state_next=state;
end
```

(Index 0 是 7-segment 最左邊的數字、3 則是最右邊。reg [3:0] my[0:3]是用來顯示在 7-segment 上長度為 4 的陣列。remain 則是記錄要找多少錢)

第一個 if 是投入的零錢比應付價格多的情況，如果是這樣就先設定好 7-segment 要顯示什麼數字然後再進到 release state。else if 則是使用者按下 cancel 的情形，那就要將最右邊的兩位數字設成使用者所投入的零錢量，因為該找零的錢就是使用者所投入的金額，最後再進入到 change state。

3. 想對老師或助教說的話

   為什麼傳入 debounce 跟 onepulse 的 clk 要跟用來操作 fsm 的 clk 一樣?

   如果不一樣的話可能會有甚麼問題?

   Ex. debounce 跟 onepulse module 傳入 100MHz/2^13，但 fsm 運作在 100MHz 底下。