Lab 5

學號: 109000168 姓名: 許媄香

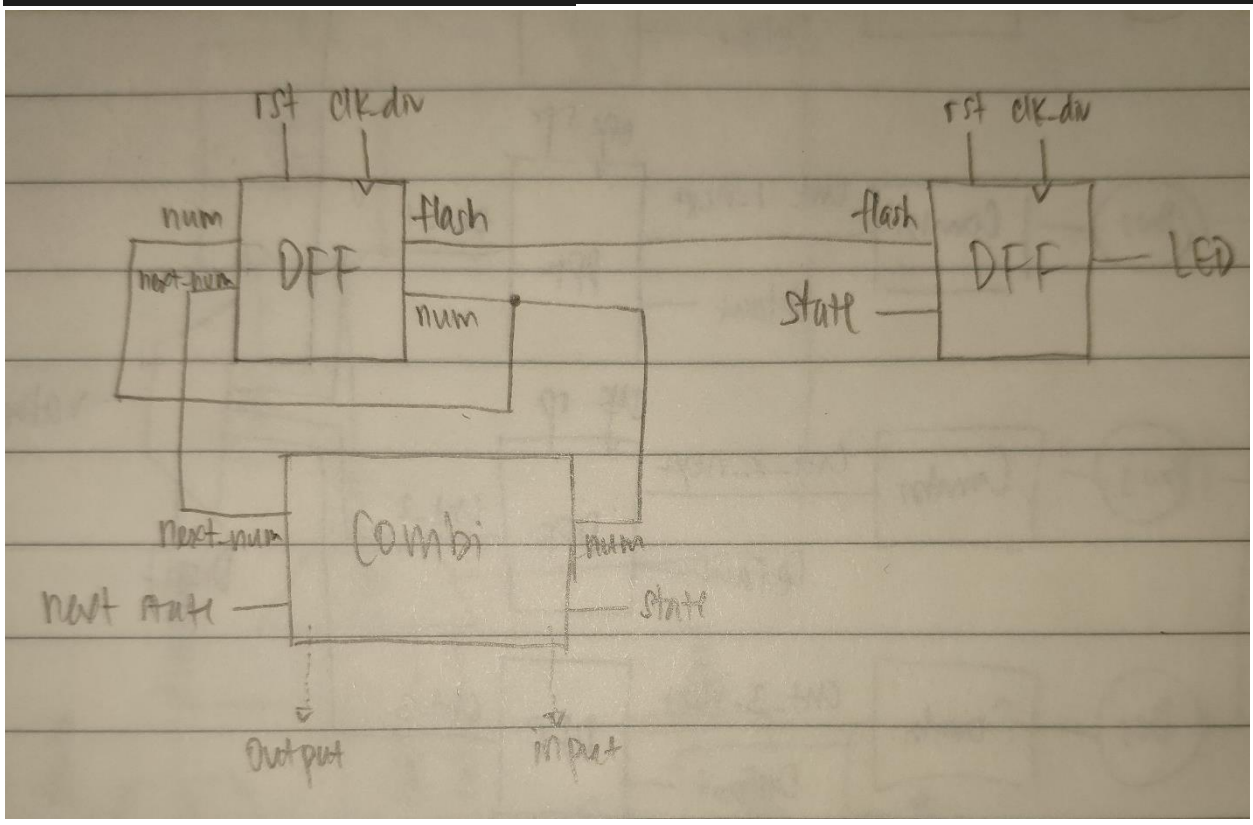
1. 實作過程

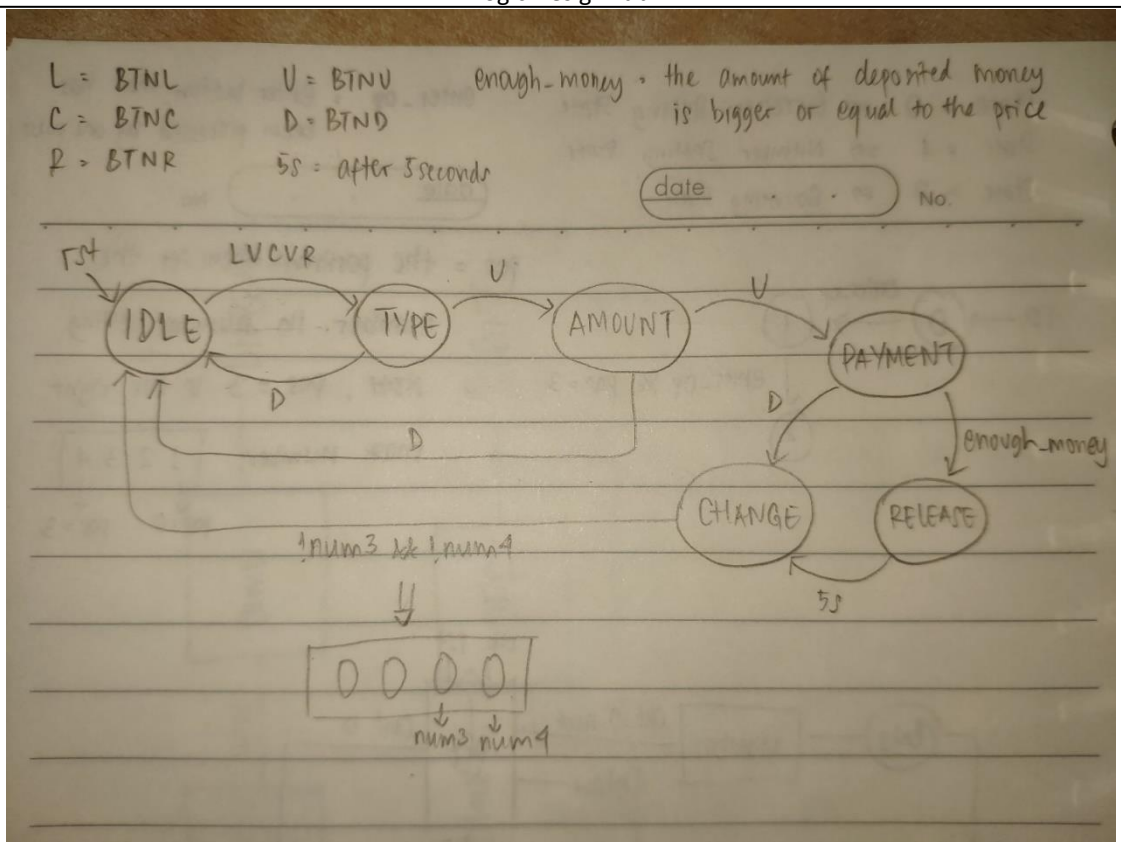
- LED Flashing
  I use 10kHz speed for the refresh rate (**clk_div**), and create 1s **flash** signal (flag) by divided **cnt_div** by 10000. I reset **next_num** to 0 when exchanging some state to make sure the timing is stable, if not, sometimes the first **LED** flash won't light for one second and will be off immediately.

  In the **IDLE** and **RELEASE** state, **LED** will keep flashing. I design a sequential block to change the **LED** light only when the flash signal is 1 and also the state is either **IDLE** or **RELEASE**, otherwise, **LED** light is off.

```
always@(posedge clk_div, posedge rst) begin
    if(rst) begin
        num <= 15'd10000;
        flash <= 0;
    end else begin
        if(num == 15'd10000) begin
            num <= 0;
            flash <= 1;
        end else begin
            num <= next_num;
            flash <= 0;
        end
    end
end
```

```
always@(posedge clk_div, posedge rst) begin
    if(rst) LED <= ON;
    else begin
        if(flash && (state == IDLE || state == RELEASE)) begin
            if(LED == ON) LED <= ~ON;
            else LED <= ON;
        end else if(state != IDLE && state != RELEASE) begin
            LED <= ~ON;
        end
    end
end
```





- IDLE
  7 segment display should be flashing together with the **LED**. Therefore, if the current LED is on, then the next display should be off and vice versa. (4'd10 will display dash, 4'd15 will make it off)

```
if(flash && (state == IDLE)) begin
    if(LED == ON) begin
        num1 <= 4'd15;
        num2 <= 4'd15;
        num3 <= 4'd15;
        num4 <= 4'd15;
    end else begin
        num1 <= 4'd10;
        num2 <= 4'd10;
        num3 <= 4'd10;
        num4 <= 4'd10;
    end
```

When the **BTNL**, **BTNC** and **BTNR** button is pressed, it will go to **TYPE** state and display the ticket type and price with **num_2_next** off.

num1_next is the left most display, and **num4_next** is the right most, consecutively.

```
IDLE : begin
    if(BTNL_op) begin //child
        next_state = TYPE;
        num1_next = C;
        num2_next = 4'd15;
        num3_next = 4'd0;
        num4_next = 4'd5;
    end else if(BTNC_op) begin //student
        next_state = TYPE;
        num1_next = S;
        num2_next = 4'd15;
        num3_next = 4'd1;
        num4_next = 4'd0;
    end else if(BTNR_op) begin //adult
        next_state = TYPE;
        num1_next = A;
        num2_next = 4'd15;
        num3_next = 4'd1;
        num4_next = 4'd5;
    end
end
```

- TYPE
  If any button among the five buttons (except **BTNU** and **BTND**) is pressed, it will display the ticket price and type. But if **BTNU** (**ok_op**) button is pressed, then it will go to **AMOUNT** state and the display of **num3_next** will be off, and **num4_next** will be changed as the display of **AMOUNT** state which is '1' (the minimum amount of ticket to purchase). However, if the **BTND** (**cancel_op**) is pressed, then it will be canceled or reset to **IDLE** state. Also need to set **next_num**

```
TYPE : begin
    num2_next = 4'd15;
    if(BTNL_op) begin //child
        num1_next = C;
        num3_next = 4'd0;
        num4_next = 4'd5;
    end else if(BTNC_op) begin //student
        num1_next = S;
        num3_next = 4'd1;
        num4_next = 4'd0;
    end else if(BTNR_op) begin //adult
        num1_next = A;
        num3_next = 4'd1;
        num4_next = 4'd5;
    end else if(ok_op) begin
        next_state = AMOUNT;
        num3_next = 4'd15;
        num4_next = 4'd1;
    end else if(cancel_op) begin
        next_state = IDLE;
        next_num = 15'd10000;
    end
end
```

to 10000, so that the LED will be light up immediately when it get into **IDLE** state. (explained at the LED flashing code above)

- AMOUNT

If we pressed **BTNL**, the amount of the ticket will be decrease by one if only if **num4** is not smaller or equal to '1'. And if **BTNR** is pressed, then it will increase by one if **num4** is not is not bigger or equal to '3'. It will be the same as previous state, if **BTND** (**cancel_op**) is pressed, then **next_state** will become **IDLE** and **next_num** is '10000'.

Money	Money	Price	Price

However, if **BTNU** (**ok_op**) is pressed, **next_state** will be **PAYMENT** state and display the deposit money (left two digits, initial amount is '00') and total price of the ticket (right two digits). I use case statement to assign the value of **num1_next** until **num4_next**.

```
case (num1)
    A : begin
        case (num4)
            4'd1 : begin
                num3_next = 4'd1;
                num4_next = 4'd5;
            end
            4'd2 : begin
                num3_next = 4'd3;
                num4_next = 4'd0;
            end
            4'd3 : begin
                num3_next = 4'd4;
                num4_next = 4'd5;
            end
        endcase
    end
```

```
C : begin
    case (num4)
        4'd1 : begin
            num3_next = 4'd0;
            num4_next = 4'd5;
        end
        4'd2 : begin
            num3_next = 4'd1;
            num4_next = 4'd0;
        end
        4'd3 : begin
            num3_next = 4'd1;
            num4_next = 4'd5;
        end
    endcase
end
```

```
    S : begin
        case (num4)
            4'd1 : begin
                num3_next = 4'd1;
                num4_next = 4'd0;
            end
            4'd2 : begin
                num3_next = 4'd2;
                num4_next = 4'd0;
            end
            4'd3 : begin
                num3_next = 4'd3;
                num4_next = 4'd0;
            end
        endcase
    end
endcase
```

Besides, I keep updating the ticket type and amount in sequential block (with posedge **rst** and posedge **clk_div**) every clock cycle because it will be used in **RELEASE** state. I store it in **ticket** (ticket type) and **ticket_amount** (amount of ticket to buy) reg.

```
if(state == AMOUNT) begin
    ticket <= num1;
    ticket_amount <= num4;
    enough_money <= 1'b0;
```

- PAYMENT

If deposit $1, the digit will change according to the second digit. If it's not '9', then just add the second digit by '1'. But if the second digit is 9, check the first digit, is it 9 or not (actually it's not possible to be 9, but I just make that condition). If the first digit is not 9, then the first digit will be added by one, but the second digit become 0. Otherwise, the digits will be the same.

a. When BTNL ($1), BTNC ($5), or BTNR ($10) is pressed, the corresponding amount of money will be deposited.

When deposit $5, need to check whether the second digit is less then 5 or not. If yes, then just add the second digit by 5. Otherwise check the first digit is it '9'. And if it's not '9', same as depositing 1 dollar, add first digit by one, but the second digit will become **num2** – 5.

The last one is $10. This one is simple. If the first digit is not 9, then add it by one.

```
end else if(BTNL_op) begin //$1
    if(num2 != 4'd9) num2_next = num2 + 4'd1;
    else if(num1 != 4'd9 && num2 == 4'd9) begin
        num1_next = num1 + 4'd1;
        num2_next = 4'd0;
    end
end else if(BTNC_op) begin //$5
    if(num2 < 4'd5) num2_next = num2 + 4'd5;
    else if(num1 != 4'd9 && num2 >= 4'd5) begin
        num1_next = num1 + 4'd1;
        num2_next = num2 - 4'd5;
    end
end else if(BTNR_op) begin //$10
    if(num1 != 4'd9) num1_next = num1 + 4'd1;
end
```

When the amount of money is equal to or greater than the price of tickets, the machine goes to RELEASE state.

Here I also keep updating a flag (**enough_money**) in sequential block to check if the money is enough to pay the ticket. If the money not enough, I update change money (**num3_change** and **num4_change**) equal to the amount of deposit money. Otherwise, it will be the deposit money subtract by the ticket price. Also if the money is enough to pay, it will go to **RELEASE** state and display ticket type on the first digit and ticket amount on the last digit.

```
end else if (state == PAYMENT) begin
    if((num1 > num3) || (num1 == num3 && num2 >= num4)) begin
        enough_money <= 1'b1;
        if(num2 < num4) begin
            num3_change <= (num1 - 4'd1) - num3;
            num4_change <= (4'd10 + num2) - num4;
        end else begin
            num3_change <= num1 - num3;
            num4_change <= num2 - num4;
        end
    end else begin
        enough_money <= 1'b0;
        num3_change = num1;
        num4_change = num2;
        cnt <= 4'd0;
    end
```

```
if(enough_money) begin
    next_state = RELEASE;
    num1_next = ticket;
    num2_next = 4'd15;
    num3_next = 4'd15;
    num4_next = ticket_amount;
    next_num = 15'd10000;
end else if(cancel_op) begin
    next_state = CHANGE;
    num1_next = 4'd15;
    num2_next = 4'd15;
    num3_next = num3_change;
    num4_next = num4_change;
```

When BTND (cancel) is pressed, the machine goes to the CHANGE state.

When it change to **CHANGE** state, **num1_next** and **num2_next** digit will be off and **num3_next** and **num4_next** display the amount of deposited money just now. (the code is above ↑↑↑)

- RELEASE

  Here LED will keep flashing for 5 seconds. So I use a 3 bit reg to count it in a sequential block.

```
end else if(flash && state == RELEASE) begin
    cnt <= cnt + 4'd1;
end
```

  State will change to **CHANGE** state when the **cnt** already 6.

```
RELEASE : begin
    num2_next = 4'd15;
    num3_next = 4'd15;
    if(cnt == 4'd6) begin
        next_state = CHANGE;
        num1_next = 4'd15;
        num2_next = 4'd15;
        num3_next = num3_change;
        num4_next = num4_change;
    end
end
```

- CHANGE

  b. If there are $5 or more to return, the change will be decreased by $5 (to simulate dropping a $5 coin) every second. When there is less than $5 to return, the change will be decreased by $1 (to simulate dropping a $1 coin).

  c. After all the change is returned, the machine will return to the IDLE state.

  If **num3** is not 0, it will be decreased by one. Otherwise, check if **num_4** is bigger or equal to 5. If it is, then subtract it by 5 otherwise subtract it by 1.
  When both digit, **num3** and **num4** are 0, it will go to **IDLE** state and **next_num** is 0.

```
CHANGE : begin
    num1_next = 4'd15;
    num2_next = 4'd15;
    if(num3 == 4'd0 && num4 == 4'd0) begin
        next_state = IDLE;
        next_num = 15'd0;
    end else begin
        if(num3 != 4'd0) num3_next = num3 - 4'd1;
        else begin
            if(num4 >= 4'd5) num4_next = num4 - 4'd5;
            else if(num4 != 4'd0) num4_next = num4 - 4'd1;
        end
    end
end
```

2. 學到的東西與遇到的困難

The problem I face in lab5 is doing the report :). It is kind of hard for me to explain something so that other's can understand it easily. I always confuse about which part I need to explain it in detail because I always think that people might understand it without having me to explain. However, when I tried to understand other's writing or report, I found out that it is hard to understand what they (writer) mean when they didn't explain it in more detail. That's why I want to try to explain everything clearly but on the other hand, I also afraid that the readers would feel their time is wasted because I explain something so basic. To be honest, I still don't know how to write a report.

3. 想對老師或助教說的話