

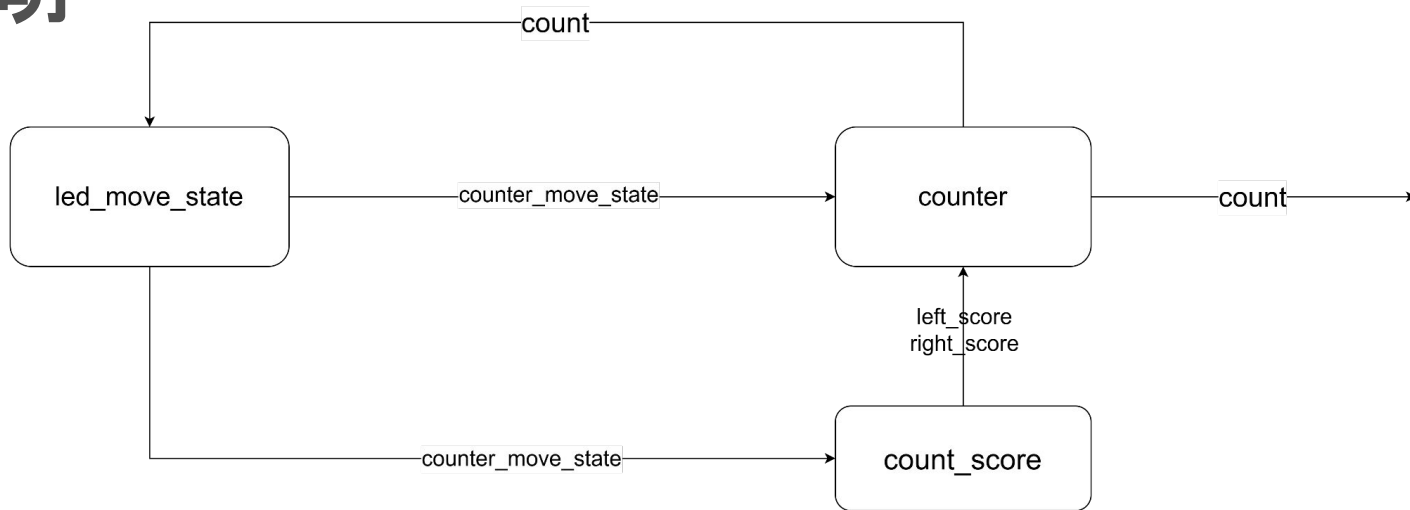


# pingpong

---

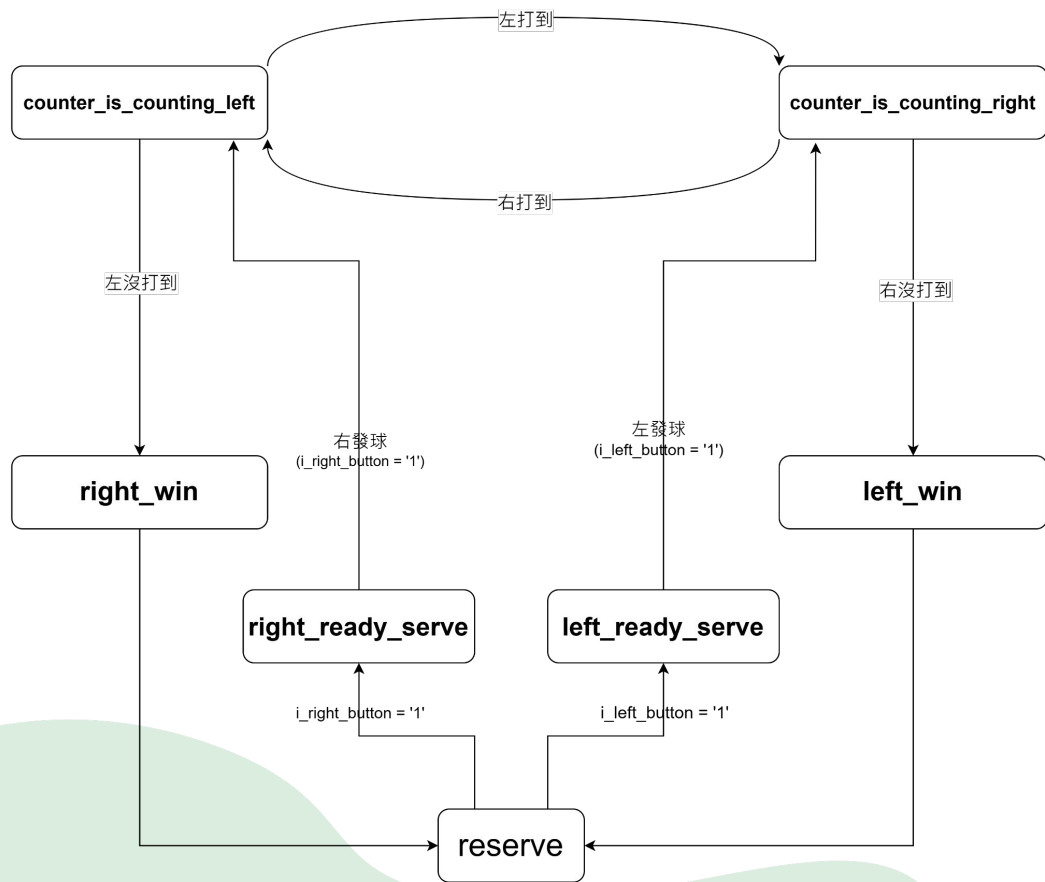
C111112132 蕭詠釗

# 說明



- ❑ Led\_move\_state從led的狀態, 更新 counter\_move\_state傳給 counter和count\_score
- ❑ counter是用來位移 led
- ❑ counter負責計分假如左邊獲勝左邊就會加分再傳給 counter來顯示

## led\_move\_state



## Led\_move\_state 內部架構圖說明

剛開始狀態從 `reserve` 開始，如果 `i_left_button` 按下進入左發球狀態，如果 `i_right_button` 按下進入右發球狀態，當打出勝負並顯示完分數後會再進入 `reserve`，以此循環

# 說明

```
led_move_state :process (i_clk , i_rst , i_left_button , i_right_button)
begin
```

```
if i_rst = '0' then
```

```
    counter_move_state <= reserve;
```

```
elsif i_clk' event and i_clk = '1' then
```

```
    prestate <= counter_move_state;
```

```
    left_button <= i_left_button;
```

```
    right_button <= i_right_button;
```

```
    case counter_move_state is
```

```
        when counter_is_counting_left =>
```

```
            if (count = "10000000") and (i_left_button = '1') then
```

```
                counter_move_state <= counter_is_counting_right;
```

```
            elsif (i_left_button = '0' and count = "00000000") or (count < "10000000" and i_left_button = '1') then
```

```
                counter_move_state <= right_win;
```

```
            end if;
```

```
        when counter_is_counting_right =>
```

```
            if (count = "00000001") and (i_right_button = '1') then
```

```
                counter_move_state <= counter_is_counting_left;
```

```
            elsif (i_right_button = '0' and count = "00000000") or (i_right_button = '1' and count > "00000001") then
```

```
                counter_move_state <= left_win;
```

```
            end if;
```

```
        when right_win =>
```

```
            if count = (left_score(0)&left_score(1)&left_score(2)&left_score(3)) & right_score then
```

```
                counter_move_state <= reserve;
```

```
            end if;
```

```
        when left_win =>
```

```
            if count = (left_score(0)&left_score(1)&left_score(2)&left_score(3)) & right_score then
```

```
                counter_move_state <= reserve;
```

```
            end if;
```

```
        when left_ready_serve =>
```

```
            if count = "10000000" then
```

```
                counter_move_state <= counter_is_counting_right;
```

```
            end if;
```

```
        when right_ready_serve =>
```

```
            if count = "00000001" then
```

```
                counter_move_state <= counter_is_counting_left;
```

```
            end if;
```

```
        when reserve =>
```

reset為0時進入決定由誰發球的狀態

```
if i_left_button = '1' and left_button = '0' then
    counter_move_state <= left_ready_serve;
elsif i_right_button = '1' and right_button = '0' then
    counter_move_state <= right_ready_serve;
else
    counter_move_state <= reserve;
end if;
```

```
when others =>
```

```
    null;
```

```
end case;
```

```
end if;
```

```
end process;
```

按鈕按下那邊發球

搶拍或是漏接都算輸

顯示分數和實際分數相同時進入決定誰發球狀態 (PS)最小位元的再最邊邊

Mealey machine  
確保不會因為按太久導致球直接發出去,需要放再按才能進下一個狀態

# 說明

顯示led的方塊給  
除頻clk才看的到

```
counter :process (i_clk , i_rst)
```

```
begin
```

```
if i_rst = '0' then
```

```
count <= "00000000";--count初始值
```

reset為0時預設led不亮

```
elsif led_clk' event and led_clk = '1' then
```

```
case counter_move_state is
```

```
when counter_is_counting_left =>
```

```
count <= count(6 downto 0) & '0'; --左移
```

led左右移動一格

```
when counter_is_counting_right =>
```

```
count <= '0' & count(7 downto 1); --右移
```

```
when right_win =>
```

```
count <= (left_score(0)&left_score(1)&left_score(2)&left_score(3)) & right_score; --看分數
```

顯示分數

```
when left_win =>
```

```
count <= (left_score(0)&left_score(1)&left_score(2)&left_score(3)) & right_score; --看分數
```

```
when left_ready_serve =>
```

```
count <= "10000000"; --左初始值
```

左右發球預備狀態

```
when right_ready_serve =>
```

```
count <= "00000001"; --右初始值
```

```
when others =>
```

```
null;
```

```
end case;
```

```
end if;
```

```
end process;
```

# 說明

```
count_score : process (i_clk, i_rst)
begin
    if i_rst = '0' then
        right_score <= "0000"; -- 初始化 score
        left_score <= "0000"; -- 初始化 score
    elsif i_clk' event and i_clk = '1' then
        case counter_move_state is
            when counter_is_counting_left =>
                null;
            when counter_is_counting_right =>
                null;
            when right_win =>
                if prestate = counter_is_counting_left then
                    right_score <= right_score + '1'; --right_win
                else
                    right_score <= right_score;
                end if;
            when left_win =>
                if prestate = counter_is_counting_right then
                    left_score <= left_score + '1'; --left_win
                else
                    left_score <= left_score;
                end if;
            when left_ready_serve =>
                null;
            when right_ready_serve =>
                null;
            when others =>
                null;
        end case;
    end if;
end process;
```

Mealey machine  
確保分數不會一直  
加

## 管理分數

右贏右加 1分  
左贏左加 1分

# 說明

```
fd:process(i_clk ,i_rst)
begin
  if i_rst = '0' then
    divclk <= (others => '0');
  elsif rising_edge(i_clk) then
    divclk <= divclk +1 ;
  end if;
end process fd;
led_clk <= divclk(24);
```

除頻

**説明**

**Thanks**