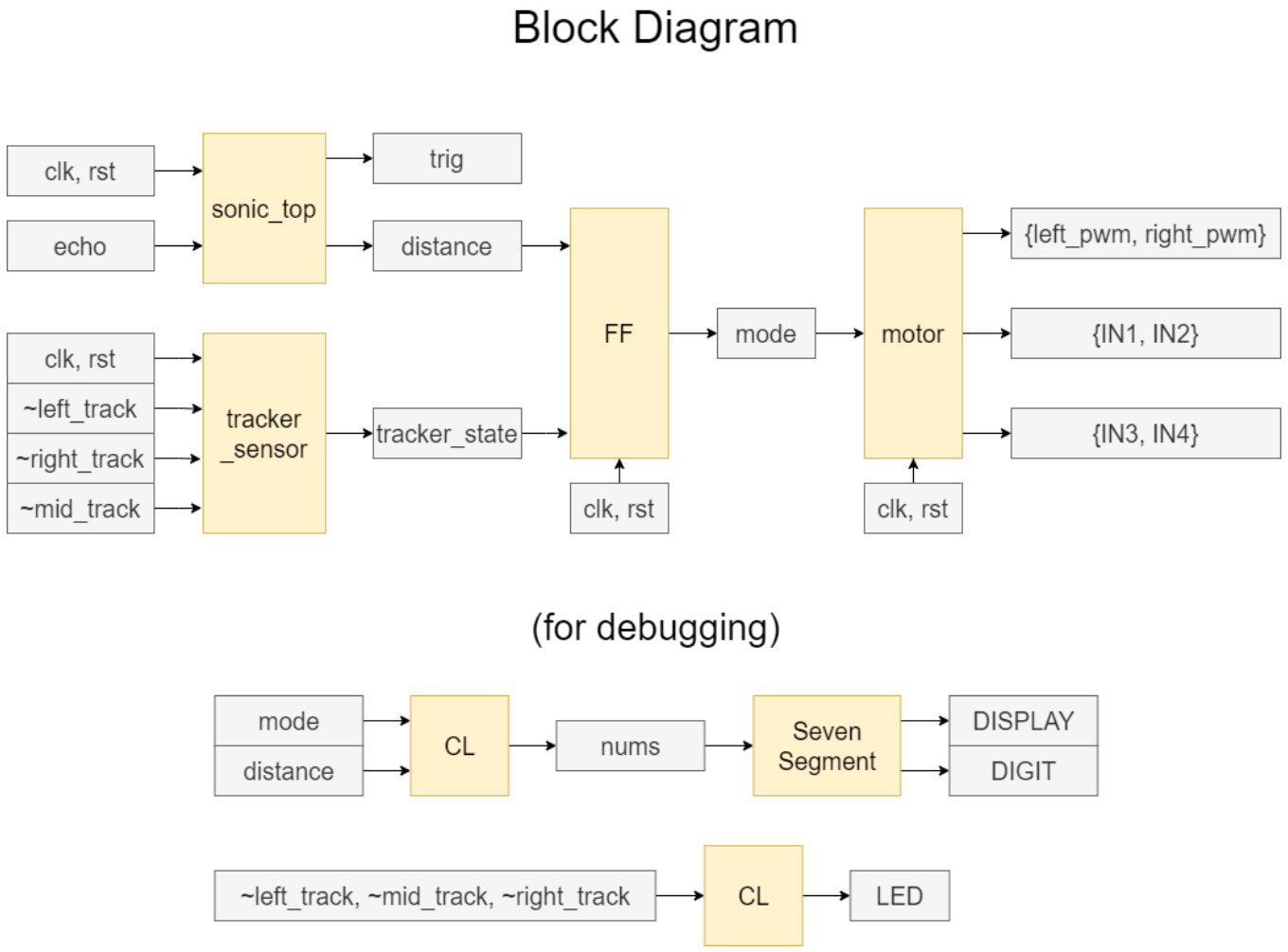
Lab 9

學號: 108032053、109062174 姓名: 陳凱揚、謝承恩

1. 實作過程

下圖 1 為 module lab9 的 block diagram，首先將 module sonic_top 取得的 distance 和 module tracker_sensor 取得的 tracker_state 做運算輸出 mode，如下圖 2，在 rst 或是 distance 小於 15 公分時，mode 設為停止，其他時候則直接設為 tracker_state。接著便將 mode 接至 module motor 中，控制馬達的轉速及轉向。此外，我們也另外加上 SevenSegment 和 LED 來幫助 debug，其中第 1 位數字用來顯示 mode 狀態，有 0~3 等 4 種，第 3~4 位數字用來顯示 distance，大於 100 公分時則不顯示；並用 LED[2:0]分別顯示~right_track、~mid_track、~left_track 的偵測狀態，亮的時候代表黑色。



▲ 圖 1

```
// mode
always @(posedge clk, posedge rst) begin
    if(rst) mode <= `STOP;
    else mode <= (distance<20'd15)?(`STOP):(tracker_state);
end
```

▲ 圖 2

如下圖 3，在 module motor 中，mode 會有 4 種狀態，在 FRONT 時，左右輪的轉速都設為 FAST(800)；在 LEFT 時，左輪設為 SLOW(600)，右輪設為 FAST；在 RIGHT 時則相反；在 STOP 時設為 STOP(0)。而 l_IN 和 r_IN 除了停止時都設為向前轉，因為並不會有倒車的情況發生。

```
// next_left_motor, next_right_motor, r_IN, l_IN
always @* begin
    case(mode)
        `FRONT: begin
            next_left_motor = FAST;
            next_right_motor = FAST;
            l_IN = 2'b10;
            r_IN = 2'b10;
        end
        `LEFT: begin
            next_left_motor = SLOW;
            next_right_motor = FAST;
            l_IN = 2'b10;
            r_IN = 2'b10;
        end
        `RIGHT: begin
            next_left_motor = FAST;
            next_right_motor = SLOW;
            l_IN = 2'b10;
            r_IN = 2'b10;
        end
        `STOP: begin
            next_left_motor = STOP;
            next_right_motor = STOP;
            l_IN = 2'b00;
            r_IN = 2'b00;
        end
    endcase
end
```

▲ 圖 3

在 module sonic_top 下，會有個 module PosCounter 負責輸出距離（單位為公分），我們所做的只是將已經數好的 cycle 數，根據 cycle 的週期和音速轉換成單位為公分的距離，如下圖 4 所示。

```
// distance_count
assign distance_count = distance_register*34/2000;
```

▲ 圖 4

在 module tracker_sensor 中，我們會根據 3 個 track 所偵測的結果組合來決定 state（下圖 5）。

(1) 當只有右邊或只有中間和右邊偵測到黑色時，設為向右。

(2) 當只有左邊或只有中間和左邊偵測到黑色時，設為向左。

(3) 當只有中間偵測到黑色時（通常在直線時偶爾發生），設為向前。

(4) 當全部都是黑色時（通常發生在車子拿在空中時，在跑道時不太可能發生，因為跑道黑線較細），設為停止。

(5) 當只有中間是白色時，延續前一次的狀態，因為此狀態不應該發生在跑道上，可能只有極少數時候意外發生，因此忽略此偵測結果。

(6) 當全部都是白色時，延續前一次狀態，因為在實際測試時發現在大多數情況下，偵測結果皆為此種，可能是因為跑道線較細，因此讓車子延續前一次的狀態。

```
// state
always @(posedge clk, posedge reset) begin
    if(reset) begin
        state <= `STOP;
    end else begin
        case({left_track, mid_track, right_track})
            // 3'b000: state <= state;
            3'b001: state <= `RIGHT;
            3'b010: state <= `FRONT;
            3'b011: state <= `RIGHT;
            3'b100: state <= `LEFT;
            // 3'b101: state <= state;
            3'b110: state <= `LEFT;
            3'b111: state <= `STOP;
        endcase
    end
end
```

▲ 圖 5

2. 學到的東西與遇到的困難

學到的東西：

(1) 紅外線感測器的使用：如果感應到黑色，回傳 0，感應到白色，回傳 1，不過 template code 把它們顛倒過來了。

(2) 馬達的控制：

方向：

(IN1, IN2) = (0, 0) 或 (IN1, IN2) = (1, 1)：馬達停止

(IN1, IN2) = (1, 0) 馬達正轉

(IN1, IN2) = (0, 1) 馬達反轉

*正反轉的部分，要實際用車子測試再修改 code 比較準確

速度：

原理是藉由控制 PWM 訊號(in module PWM_gen)拉成 1 的時間佔整個 cycle 的比例，來調整馬達的速度，簡單來說，我們只需要設定不同的 duty 值給 PWM_gen 即可，duty 的值越大(0~1023)，則電壓越高，馬達轉速越快

- PWM_gen module

```
module PWM_gen (
    input wire clk,
    input wire reset,
    input [31:0] freq,
    input [9:0] duty,
    output reg PWM
);
    wire [31:0] count_max = 100_000_000 / freq;
    wire [31:0] count_duty = count_max * duty / 1024;
    reg [31:0] count;

    always @(posedge clk, posedge reset) begin
        if (reset) begin
            count <= 0;
            PWM <= 0;
        end else if (count < count_max) begin
            count <= count + 1;
            if(count < count_duty)
                PWM <= 1;
            else
                PWM <= 0;
        end else begin
            count <= 0;
            PWM <= 0;
        end
    end
```

(3) 超音波模組的使用：

- PosCounter module

  初始狀態狀態為 S0: 此時 count 的值維持 0。當 start 訊號為 1 時，進入到 S1: 此時開始計算經過的 cycle 數。當 finish 訊號為 1 時，進入到 S2: 此時將經過的 cycle 數存到 distance_register 裡，並將 count 歸 0，回到 S0。

```
else begin
    echo_reg1 <= echo;
    echo_reg2 <= echo_reg1;
    case(curr_state)
        S0:begin
            if (start) curr_state <= next_state;
            else count <= 0;
        end
        S1:begin
            if (finish) curr_state <= next_state;
            else count <= count + 1;
        end
        S2:begin
            distance_register <= count;
            count <= 0;
            curr_state <= next_state; //S0
        end
    endcase
end
```

start, finish 訊號的產生：

echo_reg1 為 1, echo_reg2 為 0 => echo 訊號的 posedge, 此時開始計時

echo_reg1 為 0, echo_reg2 為 1 => echo 訊號的 negedge, 此時結束計時

```
assign start = echo_reg1 & ~echo_reg2;
assign finish = ~echo_reg1 & echo_reg2;
```

- PosCounter module
  每經過 100ms 將 trig 設為 1 10us

```
    end
    always @(*) begin
        next_trig = trig;
        next_count = count + 1;
        if(count == 999)
            next_trig = 0;
        else if(count == 24'd9999999) begin
            next_trig = 1;
            next_count = 0;
        end
    end
endmodule
```

(4) 寫一個 FSM 控制車子行進的方向

遇到的困難：

(1) 車子的接線：組裝車子的時候超音波模組和感測器有接錯線，導致 fpga 用外部電源供電的時候吃不到電，一開始應該就要照著 xdc 檔接線，我貪圖方便照著 pdf 檔的照片去接線(用顏色去對)，最後反而浪費更多時間。

(2) 沒注意到 track 的值被反轉：因為 template code 有將 `left_track, mid_track, right_track` 的值進行反轉，我們一開始沒注意到，花了不少時間 debug

(3) 馬達轉向的控制：一開始我們的車子會在原地打轉(一顆輪子往前轉一顆輪子往後轉)，但我們的程式並沒有這樣的操作，後來就將往後轉的輪子改成往前轉(用程式控制)，發現車子變成倒著走(兩顆輪子都往後走)，我們才發現左右輪跟我們想的剛好相反。

(4) 馬達速度的控制：我們一開始將 duty 設為 500 發現輪子不會動，花了蠻多時間檢查程式結果最後發現只是 duty 設太小。

(5) 車子方向控制：我們原本想針對 `{left_track, mid_track, right_track}` 所有的可能值(000~111)描述車子的行進方向，但是車子在跑的時候感測器的值很不穩定，當出現非預期的數值時(EX：101, 兩邊黑中間白)，我們沒辦法確定車子要往哪邊走，因此

我們改成若出現未定義的結果時，就讓它維持先前的狀態，我們只定義 100, 110 要左轉，001, 011 要右轉, 010 要直走，經過測試後發現使用這種寫法車子就能順利跑完跑道。

(6) 轉彎幅度過大：車子在轉彎的時候迴轉半徑很大，雖然最後都轉的過去，但在轉彎的時候會有點偏離軌道，後來發現是我們在轉彎的時候輪子速度太慢，如果在轉彎的時候將車輪速度調快，轉彎幅度就會比較大，車子就可以貼著軌道跑。

(7) debug 困難：一開始我們沒有使用七段顯示器和 LED，因此完全不知道是哪邊出了問題，後來將車子的狀態以及感測器的值輸分別輸出到七段顯示器和 LED 以後，debug 速度就變快很多。

(8) 超音波感測器偵測到不存在的東西：有發生前方明明沒有障礙物但超音波感測器的距離顯示小於 15，於是車子停下來的情況，後來重新寫了一份 code 就沒有遇到這個問題，事後猜測當初可能是 distance 發生溢位所導致，印象中我有修改過 distance 的 bit width。

3. 想對老師或助教說的話

想問的問題：

(1) 我們有自己去便利商店買 9V 的電池，結果裝上去發現沒辦法供電(馬達的燈有亮，但 fpga 的 power 燈沒亮)，不太清楚為什麼會這樣。

(2) 想請問這次 lab 的分數會如何計算，因為 lab9 不在原本的課程大綱內。

笑話：

4. 分工

陳凱揚	1. 程式編寫 2. 車子測試 3. Report - 實作過程
謝承恩	1. 程式編寫 2. 車子測試 3. Report - 學到的東西與遇到的困難 4. Report - 想對老師或助教說的話