Lab08_Speaker_report

106061146 陳兆廷

Experiments:

1.1 mclk : 25Mhz, lrck : (25/128)Mhz, sdin : (25/128/32)Mhz

這題要利用講義底下的 code，填入 speaker_control.v，產出四個頻率的 audio 輸出，分別接入晶片的其中四個腳，進而發出聲音。

在撰寫之前，我先了解四種 audio 輸出的作用為何。mclk 及 lrck 為固定之值，要依照晶片之分配比例來控制，而 sck 依照助教所說只要一直維持 1 即可，上述三個輸出都可以用簡易的 cnt 及 assign 解決，板子上的 clk 為 100Mhz，因此要輸出 25Mhz 要 assign 至 cnt[1]，因為兩次變化為一週期，所以不是 cnt[2]，而 lrck 為 cnt[8](2^7 倍)。

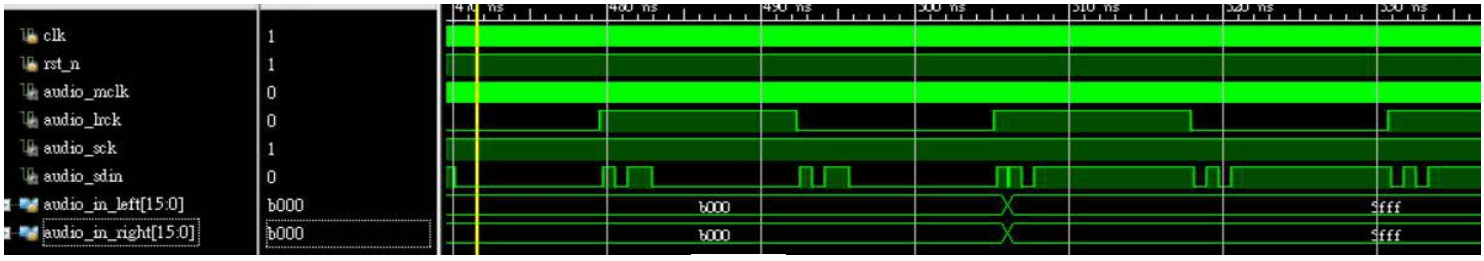
接著是 sdin，其週期為 lrck 之週期，但是要分成 32 等分，各個等分為進入的 audio_left 及 audio_right 之值，因此我新設定一個 sdincnt，當 cnt[3:0]==0000 時就加一，直到 32，剛好是 lrck 的 cnt[8]，再 case sdincnt 之值，輸出 audio_sdin 之各位值。

```
left <= audio_in_left;
right <= audio_in_right;
case(sdincnt)
        6'd0: audio_sdin <= right[0];
        6'd1: audio_sdin <= left[15];
        6'd2: audio_sdin <= left[14];
        6'd3: audio_sdin <= left[13];
        6'd4: audio_sdin <= left[12];
        6'd5: audio_sdin <= left[11];
        6'd6: audio_sdin <= left[10];
        6'd7: audio_sdin <= left[9];
        6'd8: audio_sdin <= left[8];
        6'd9: audio_sdin <= left[7];
        6'd10: audio_sdin <= left[6];
        6'd11: audio_sdin <= left[5];
        6'd12: audio_sdin <= left[4];
        6'd13: audio_sdin <= left[3];
```

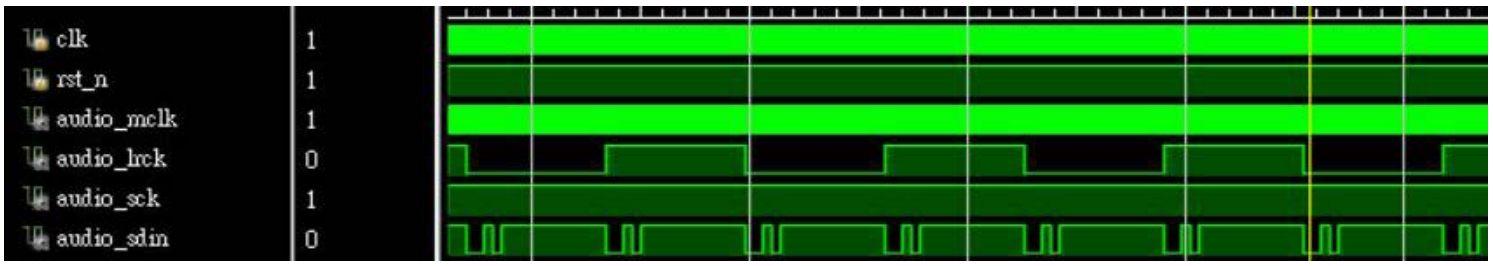
圖(一)

1.2 waveform in FPGA

在這裡遇到了一些困難，當我寫好 speaker_control.v 後，note_gem.v 照著講義打，但在最後輸出的 audio_in_left 及 right 只有 B000 的狀況，雖然，因此顯然是 b_clk 沒有反應。因此我重寫了 note_gen.v，依照先前計算至 50Mhz 之 stopwatch 一般，cnt 計算至輸入的 note_gen 後將 b_clk 變成倒數，才能用接下來的 code 完成 note_div.v。note_gen.v 之波形圖如圖(二)，而當輸入值是 Do 時，波型圖如圖(三)(頻率只有一種因為 note_div 極大)。



圖(二)



圖(三)

2.1 Do Rei Mi

這題需要寫出三種不同頻率的聲音，以三個按鈕控制，因此我多寫了一個 module(music.v)，輸入值為 clk、rst_n、[2:0]in，輸出值為[21:0]note_div，依照輸入之 in 為 100、010、001 來輸出三種音符的音色即可。

2.2 Do Rei Mi with volume control

這題除了增加兩個按鈕，分別是增加及減少音量。因此我先加入了先前的 freq_div.v，當作音量按鍵的時間標準還有後續需要的 ssd_ctl_en，再寫了一個 volume.v，輸入值為 clk_out(from freq_div)、rst_n、[1:0]invol、輸出值為一個 vol，及 vol1、vol0 用來輸出至兩位的 ssd display。再將 vol 輸入至 note_gen.v 來控制音量，如圖(四)，依照 vol 的值，藉由修改震幅的上界及下界來控制音量的大小。接著依照第一題的方式輸入至 speaker_control 輸出，最後再增加 ssd 的輸出即可。程式的 module 如圖(五)。

```
    case(vol)
        4'd0: begin volu2 = 16'h0000; volu = 16'h0000; end
        4'd1: begin volu2 = 16'h0000 - (16'd320*1); volu = 16'h0000 + (16'd384*1); end
        4'd2: begin volu2 = 16'h0000 - (16'd320*2); volu = 16'h0000 + (16'd384*2); end
        4'd3: begin volu2 = 16'h0000 - (16'd320*3); volu = 16'h0000 + (16'd384*3); end
        4'd4: begin volu2 = 16'h0000 - (16'd320*4); volu = 16'h0000 + (16'd384*4); end
        4'd5: begin volu2 = 16'h0000 - (16'd320*5); volu = 16'h0000 + (16'd384*5); end
        4'd6: begin volu2 = 16'h0000 - (16'd320*6); volu = 16'h0000 + (16'd384*6); end
        4'd7: begin volu2 = 16'h0000 - (16'd320*7); volu = 16'h0000 + (16'd384*7); end
        4'd8: begin volu2 = 16'h0000 - (16'd320*8); volu = 16'h0000 + (16'd384*8); end
        4'd9: begin volu2 = 16'h0000 - (16'd320*9); volu = 16'h0000 + (16'd384*9); end
        4'd10: begin volu2 = 16'h0000 - (16'd320*10); volu = 16'h0000 + (16'd384*10); end
        4'd11: begin volu2 = 16'h0000 - (16'd320*11); volu = 16'h0000 + (16'd384*11); end
        4'd12: begin volu2 = 16'h0000 - (16'd320*12); volu = 16'h0000 + (16'd384*12); end
        4'd13: begin volu2 = 16'h0000 - (16'd320*13); volu = 16'h0000 + (16'd384*13); end
        4'd14: begin volu2 = 16'h0000 - (16'd320*14); volu = 16'h0000 + (16'd384*14); end
        4'd15: begin volu2 = 16'h0000 - (16'd320*15); volu = 16'h0000 + (16'd384*15); end
    endcase
end
assign audio_left = (b_clk == 1'b0) ? volu2 : volu;
assign audio_right = (b_clk == 1'b0) ? volu2 : volu;
```

圖(四)

圖(五)