數位電路實驗 Lab3 Report

B07901056 張凱鈞、B07901100 林亮昕、B07901108 馬健凱

壹、實驗器材及架設

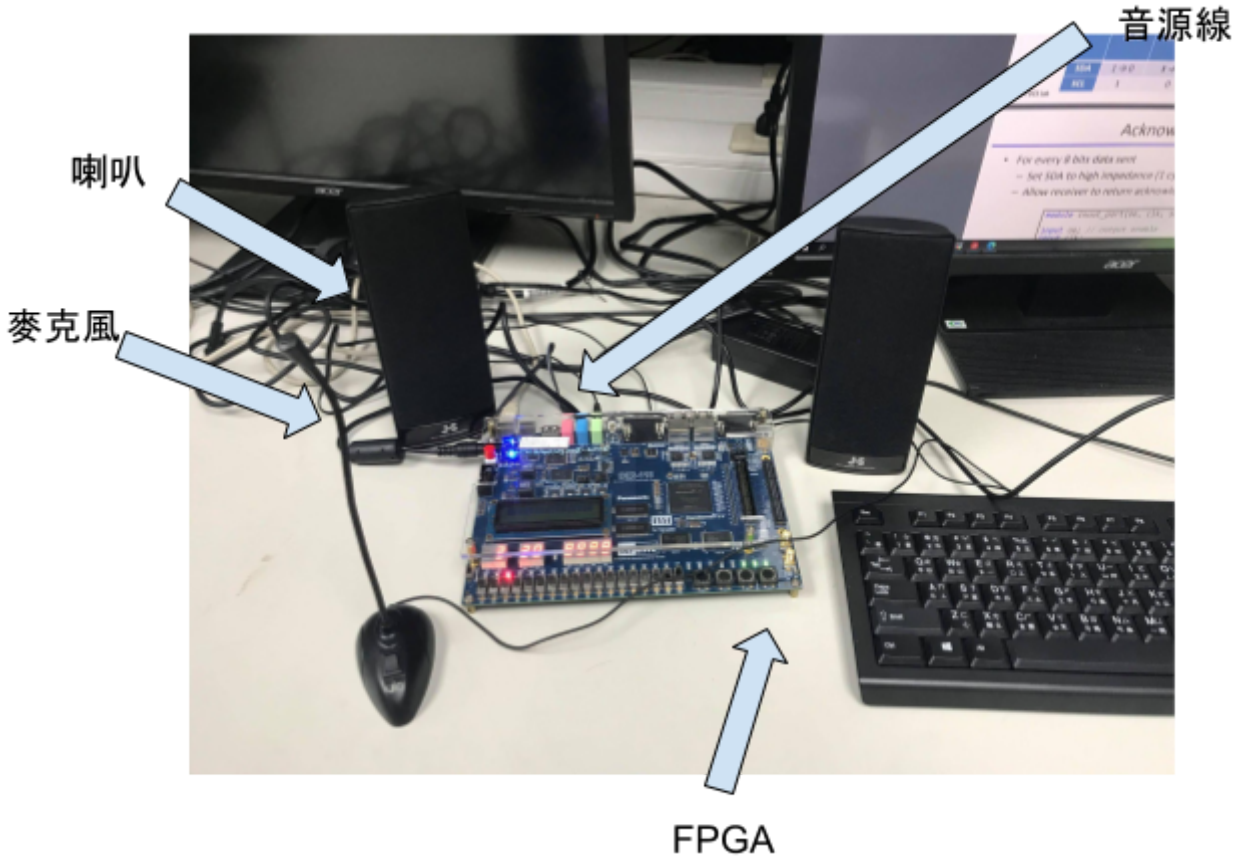

Fig.1　實驗器材架設圖

貳、使用方式與步驟

一、燒錄步驟

1、將verilog 經由Quartus compile
2、將compile後的code燒錄到FPGA板上
3、燒錄成功後，按下key3 reset即可開始使用

二、使用說明

1、基本操作：

(1)按下key 1可以開始錄音，key 0可以暫停錄音，key 2可以停止錄音。
(2)按下key 0可以開始播放，key 1可以暫停播放，key 2可以停止播放。

此外，在錄音時，若錄音超過SRAM儲存上限(32秒)，就會把前面的錄音內容蓋掉；播放時，當播完一次錄音內容，就會暫停播放，但若想重新播放或是錄音仍需要按下停止鍵(key 2)。

	開始/繼續	暫停	停止
錄音	key1	key0	key2
播放	key0	key1	key2

2、播放模式：

由 SW[6:0] 來操控，其對應如下(0代表關、1代表開、X代表don't care):

SW[6]	SW[5]	SW[4]	SW[3]	SW[2:0]	播放模式
0	0	0	0	X	正常播放
0	0	0	1	speed	以(speed+1)倍的速度快速播放
0	0	1	0	speed	以1/(speed+1)倍的速度、零次內插慢速播放
0	1	0	0	speed	以1/(speed+1)倍的速度、一次內插慢速播放
1	0	0	0	X	倒轉播放

3、其他功能：

(1) 錄音、播放時間顯示:

在錄音或播放時，會使用HEX[1:0]來顯示目前錄製或播放的秒數。其中，暫停時秒數就會暫停、停止時秒數就會歸零。此外，當快速或慢速播放時顯示的秒數也會相應地加快或放慢。

(2) 播放模式顯示:

在播放時，會使用HEX[7:6]來顯示目前的播放模式，包括0(正常)、F(加速)、S0(零次慢速)、S1(一次慢速)、R(倒轉)。HEX[5:4]則顯示目前的倍數。例如，1/3倍速零次內插播放就會顯示S0_03。

(3) 錄音音量顯示:

在錄音時，會使用LED燈來顯示目前的音量，當亮起的LED燈越多，就代表目前使用者的錄音音量越大，可以用來提醒使用者放大或縮小音量。

參、系統架構與模組分割

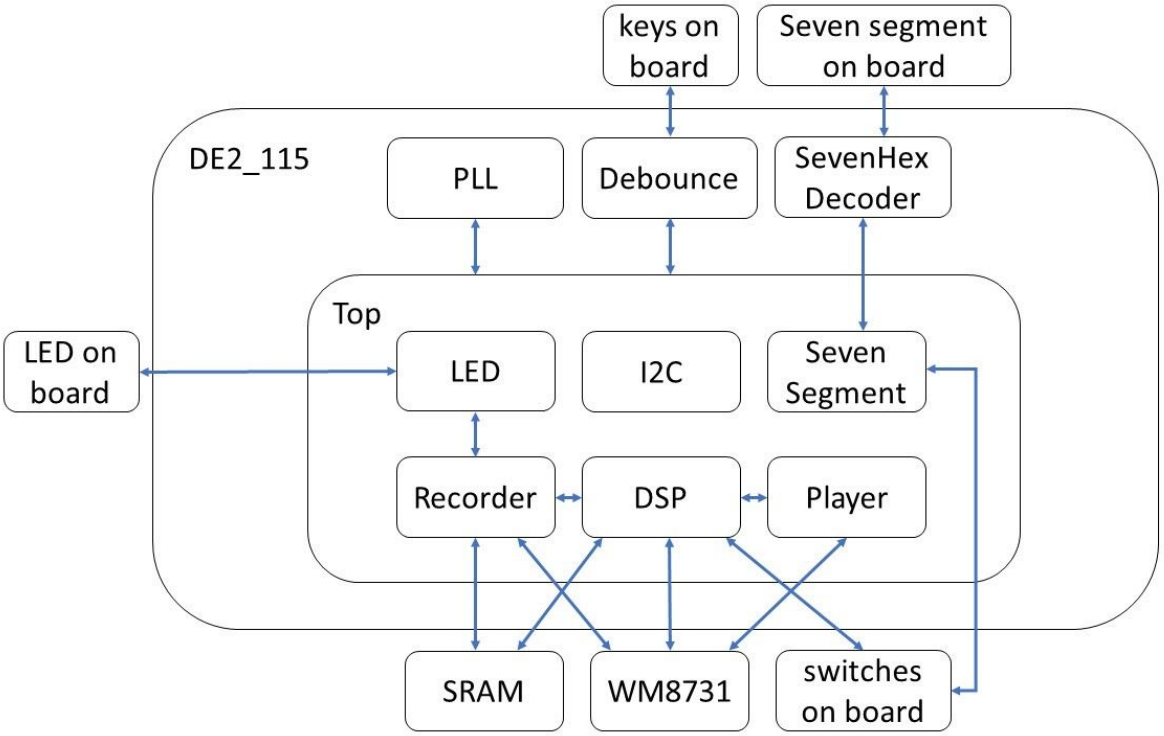

Fig.2 System Structure

我們的系統中，會使用到板子上的SRAM、WM8731，以及key, switch, LED, seven segment display等等，並透過DE2_115來傳遞。DE2_115內則包括了Top、pll、debounce、SevenHexDecoder等module。在核心的Top內則包括I2C.sv, AudRecorder.sv, AudDSP.sv, AudPlayer.sv, SevenSegmentDisplay.sv, LEDVolume.sv共六個submodule。以下將分別介紹這些submodule的實作方式。

肆、實作設計

(note: 以下 FSM 中，藍色字為執行內容，紅色字為觸發條件)

一、Top.sv

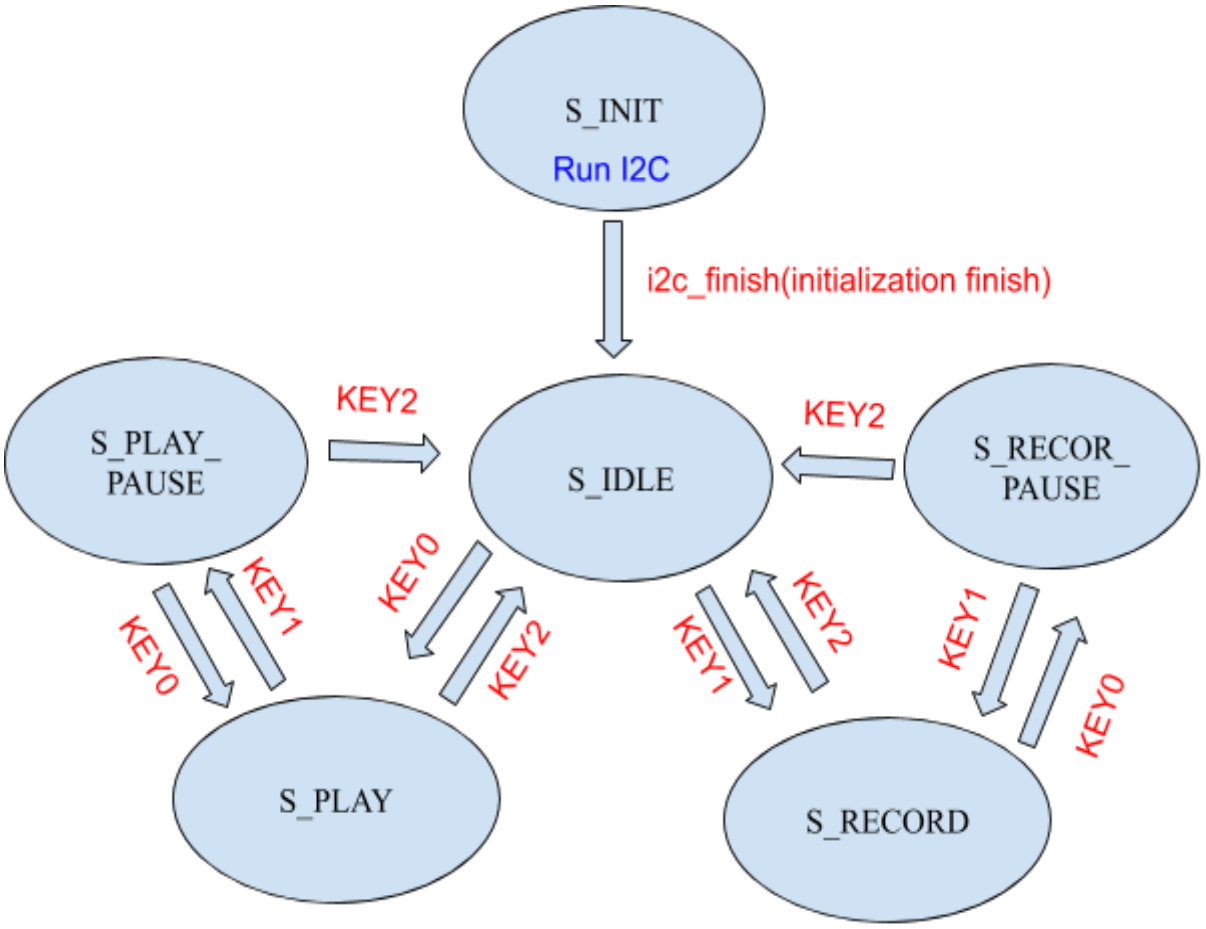

Fig.3 Top FSM

一開始按下key3 reset後，程式會進入S_INIT開始跑I2C；當I2C結束後，就會進入S_IDLE，等待使用者進行錄製或播放，同時FSM也會進入相對應的state。

二、Audrecorder.sv

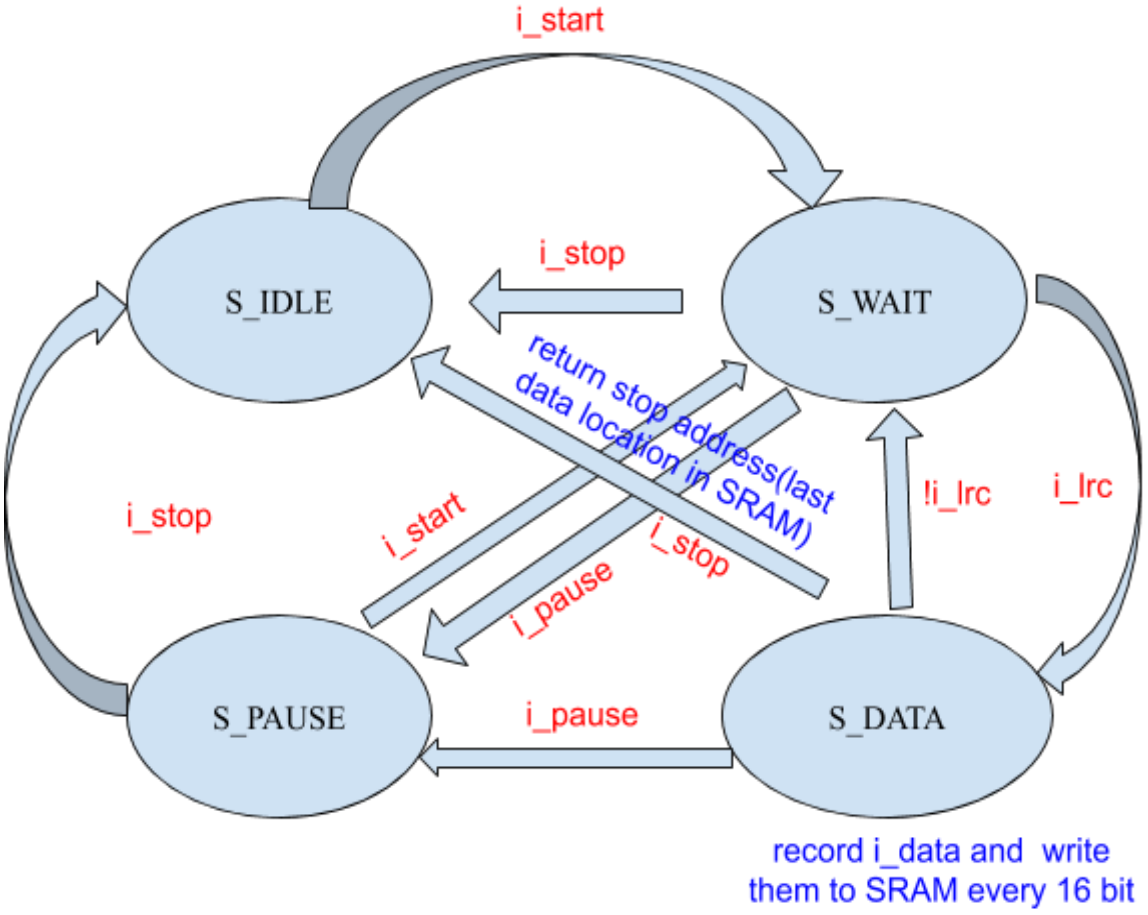

Fig. 4 Recorder FSM

在 reset 後，程式會進入 S_IDLE，當i_start(開始錄音)的訊號變成1時，程式會進入 S_WAIT，當i_lrc為1時(右聲道)進入S_DATA，並且歸零 counter。接著會開始接收 i_data(0 or 1)，當接收了16個i_data之後(counter=16)，程式將這一串資料轉成16bit的數字，在lrc切換成0時連同sram address一同輸出。輸出之後會將sram address+1，並將counter重新設成0。過程中如果收到i_pause的訊號則暫停，直到又收到i_start的訊號才會到S_WAIT準備繼續錄音，若是收到i_stop則是返回S_IDLE，終止錄音。

三、I2C.sv

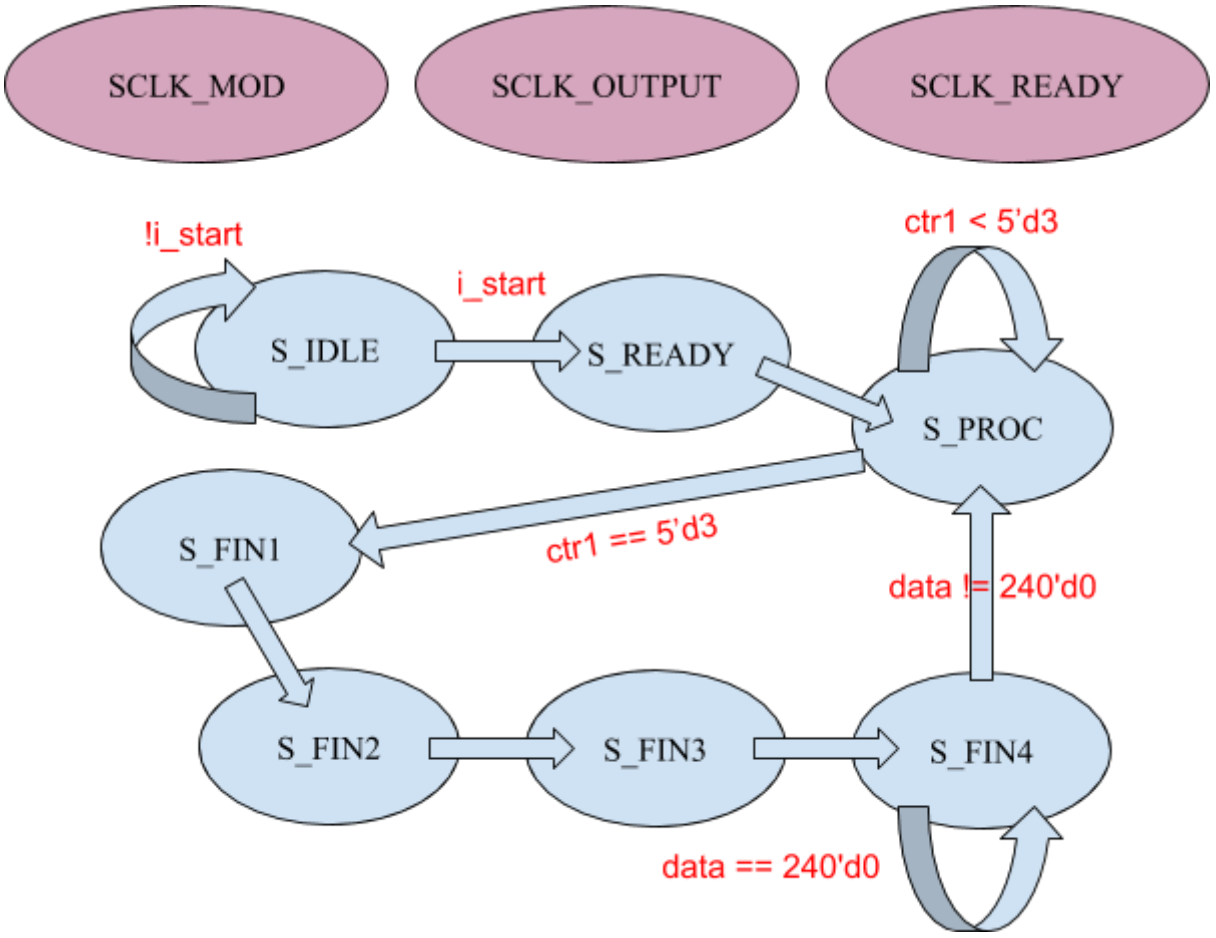

Fig. 5 I²C FSM(blue) and SCLK states(purple)

reset之後，I²C會進入S_IDLE，等待啟動訊號 i_start。接著進入 S_READY，開始用I²C設定WM8371。傳輸設定資料以及接收 acknowledgement bit 都在 S_PROC 中進行。我們將傳輸和接收都區分為 SCLK_READY, SCLK_OUTPUT, SCLK_MOD 三個sclk state，以方便程式編寫。由於一個設定有3個byte，當 ctr1 變成3的時候，會進入 S_FIN1, S_FIN2, S_FIN3 以及 S_FIN4，表示一個設定完成。我們設定WM8371的data有240個 bit，全部傳送完成會維持在 S_FIN4，o_finish設成1，代表I²C啟動完成。

四、AudDSP.sv

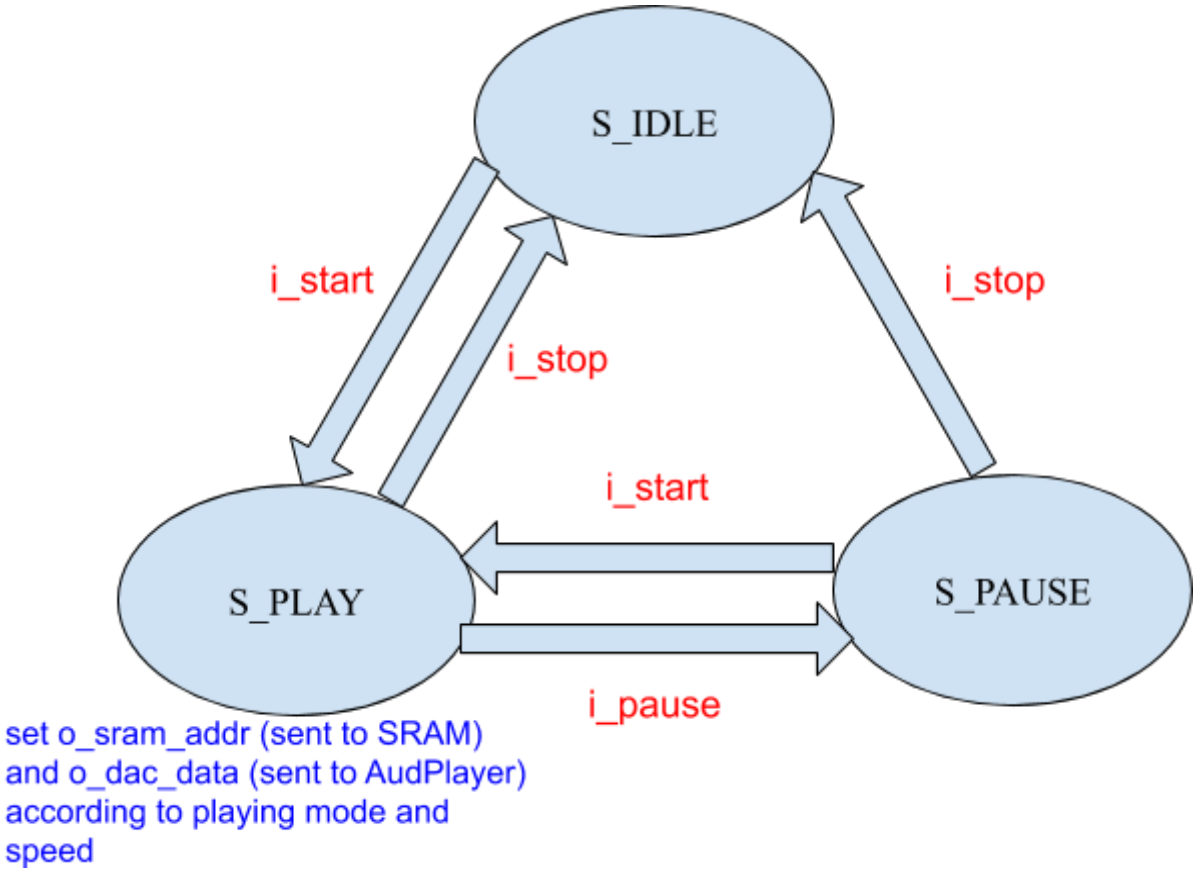

Fig. 6 DSP FSM

在reset後，會進入S_IDLE等待播放。當使用者按下播放鍵後(i_start = 1)，就會進入S_PLAY，並根據SW[6:0]輸入的播放模式及速度來「從頭」進行播放；若按下暫停鍵(i_pause = 1)，就會暫停播放，各個相關的register則會hold住當下的值，等待再次按下播放鍵後「繼續」播放；若按下停止鍵(i_stop = 1)，則會停止播放，回到S_IDLE。

在播放模式的實作上，各種模式間的不同之處主要在於o_sram_addr和o_dac_data的設定，實作方法如下：

播放模式	**o_sram_addr setting**	**o_dac_data setting**
正常播放	初始值為0，每次聲道訊號(i_daclrck)切換到我們要輸出的聲道的時候，就加1	從SRAM輸入的data (i_sram_data)
加速播放	初始值為0，每次聲道訊號(i_daclrck)切換到我們要輸出的聲道的時候，就加上(i_speed+1)	從SRAM輸入的data (i_sram_data)
零次內插慢速播放	初始值為0，每次聲道訊號(i_daclrck)切換到我們要輸出的聲道的時候，就讓counter加1；每次counter > i_speed時，o_sram_addr就加1，並歸零counter	從SRAM輸入的data (i_sram_data)
一次內插	初始值為0，每次聲道訊號(i_daclrck)切	記錄前一個i_sram_data，

慢速播放	換到我們要輸出的聲道的時候，就讓counter加1；每次counter > i_speed時，o_sram_addr就加1，並歸零counter	根據counter、i_speed數值和當前的i_sram_data線性內插得到o_dac_data
倒轉	初始值為stop_addr(錄音停止的addrress)，每次聲道訊號(i_daclrck)切換到我們要輸出的聲道的時候，就減1	從SRAM輸入的data (i_sram_data)

五、AudPlayer.sv

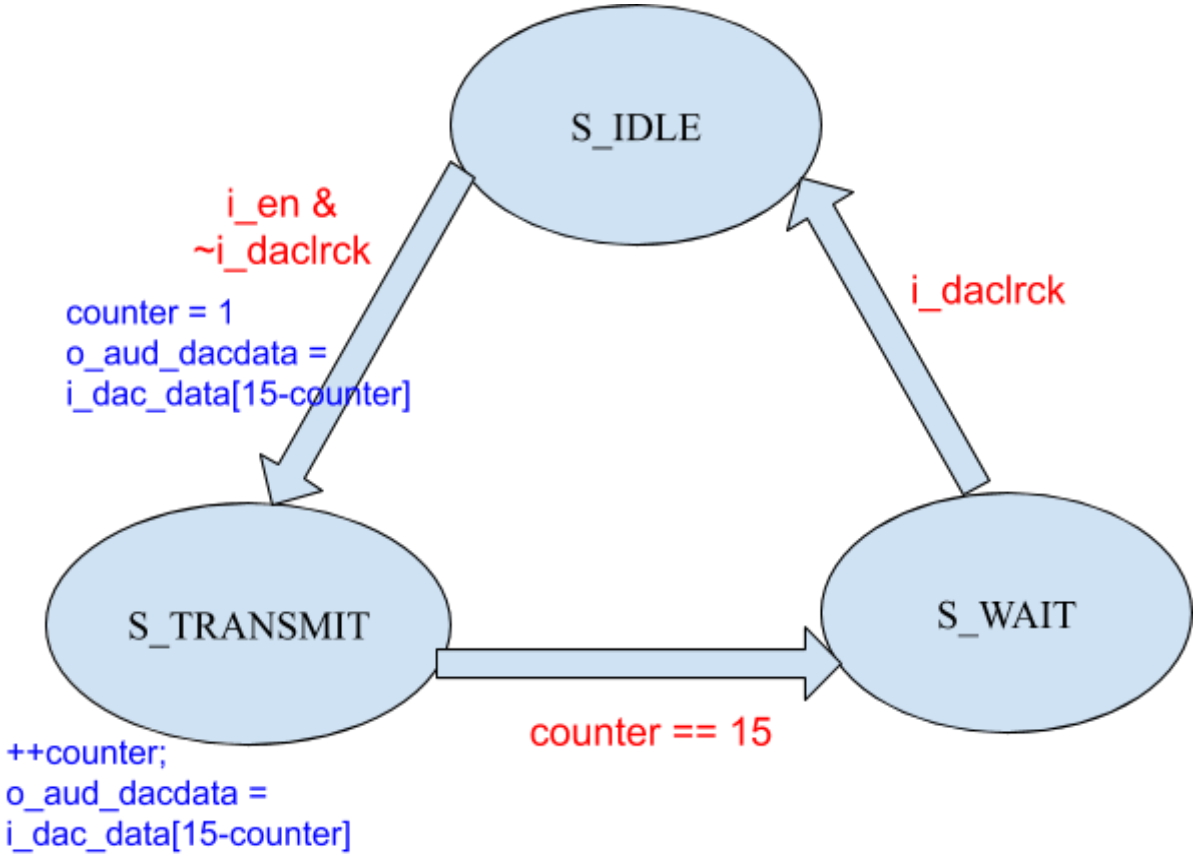

Fig. 7 Player FSM

Player在reset後進入S_IDLE，當進入播放狀態(i_en)且正處於我們要播放的聲道(idaclrck = 0)的時候，就會進入S_TRANSMIT並開始傳輸，把i_dac_data從MSB開始逐bit傳出。傳完16bit後，就會進入S_WAIT並等待聲道切換時再回到S_IDLE。

六、SevenSegmentDisplay.sv

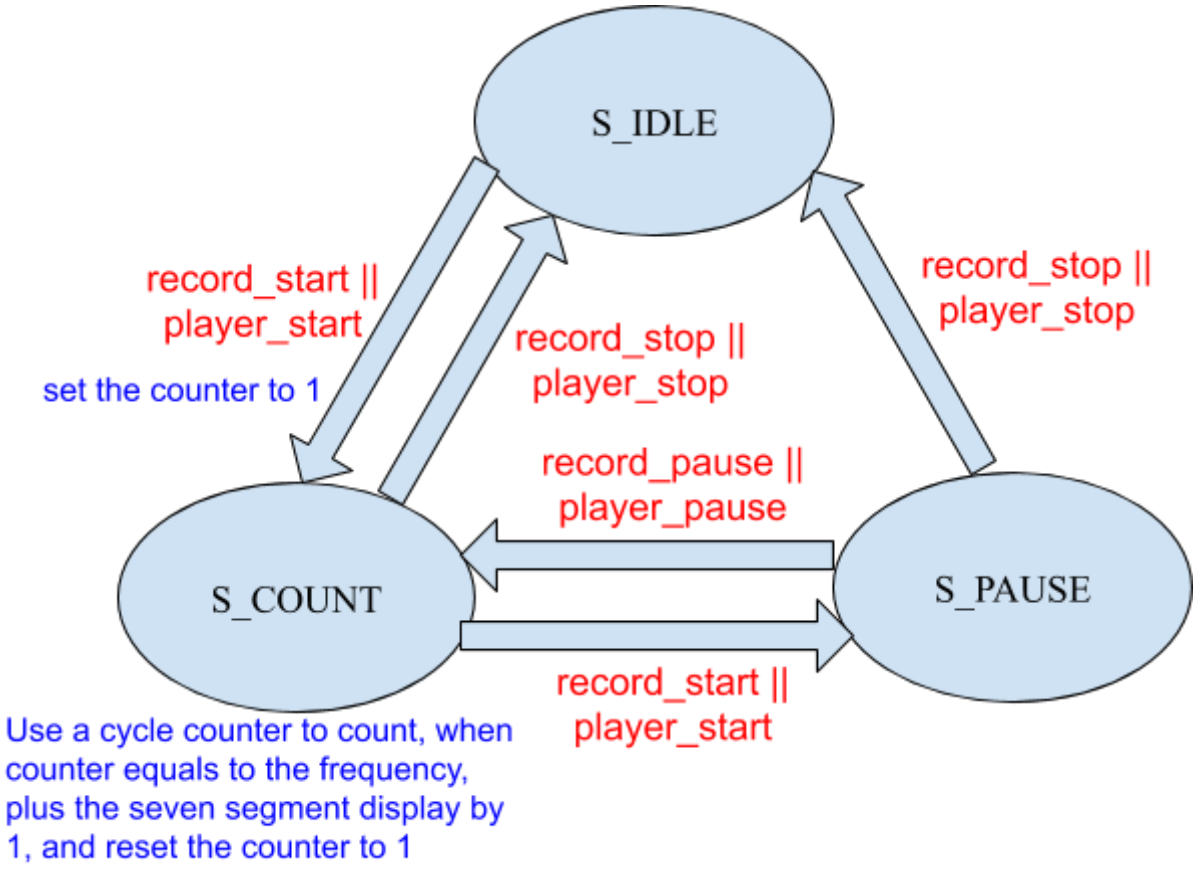

Fig. 8 SevenSegmentDisplay FSM

這個submodule負責控制七段顯示器上顯示的錄音/播放秒數，其FSM與Top的FSM類似，只是一開始就進入了S_IDLE，且沒有區分player和recorder。而顯示的秒數是透過cycle counter來控制，當cycle counter數到相當於一秒的cycle數時(例如正常模式、12MHz cycle下就是數到12000000)，就會讓顯示的秒數加一，並重置counter。

七、LEDVolume.sv及播放模式顯示

LEDVolume.sv根據recorder輸入的data，決定LED燈亮的數量，音量越大則顯示越多；播放模式顯示則是根據i_speed、i_fast、i_slow_0、i_slow_1、i_reverse等訊號決定在七段顯示器上顯示的數字或字母。這兩部分都是採用純粹的combinational circuit實作而成。

伍、問題與挑戰

我們認為這次實驗最困難的地方在於寫出來的程式，必須符合板子上的IP(例如WM8731)的規定，才能夠順利運作。有可能我們以為我們已經知道它的訊號傳輸方式，甚至也透過testbench驗證了，但事實上IP的規定並非我們想像的那樣。因此，就可能導致I2C無法順利initialize、player播出的音訊有雜音等等。我們只能透過查找資料、以及嘗試各種可能的方法(例如調整訊號輸出間隔的cycle、把FSM從Moore

Machine調整成Mealy Machine等等)，再燒到板子上確認，直到能正常運作，因此花費了不少時間。不過這也算是對於final project的暖身，因為final project很有可能會使用到更多不同的IP，要如何讀懂這些IP的SPEC、轉換成verilog code並燒到板子上，可能會是相當重要的部分。