數位電路實驗 final report

卡拉 ok 評分系統

組別：team09

組員：吳睿哲(B06901018)

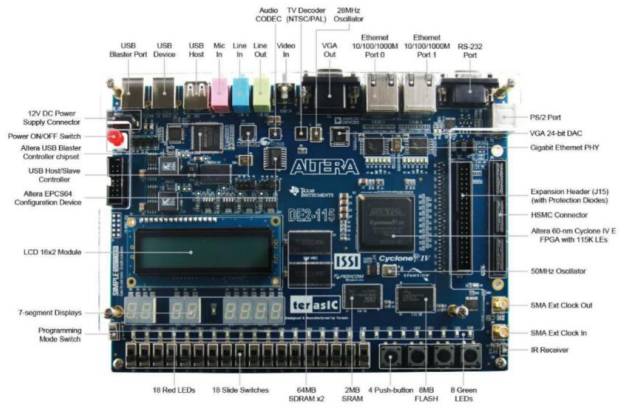
謝兆和(B06901026)

鐘民憲(B06901017)

一、實驗目標

建立一套卡拉 ok 系統，能夠從 SD 卡中讀取音檔，包括原始音檔、純人聲音檔與純背景音樂音檔，接著可以同步讀入所選音檔及麥克風的聲音訊號，即時從喇叭輸出混和的音訊，並且能夠在 LCD 上實時顯示演唱當下的得分。

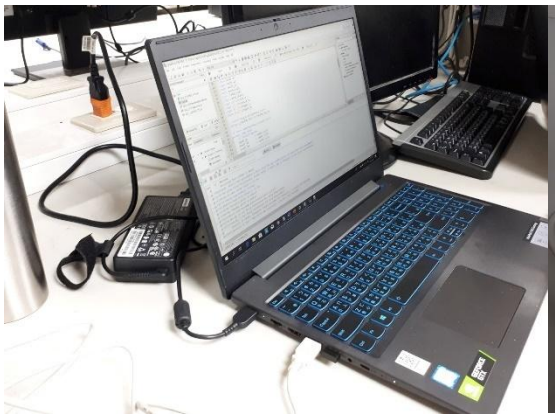
二、使用設備與器材



DE2-115SD 卡

喇叭

麥克風



筆電

傳輸線

三、操作說明

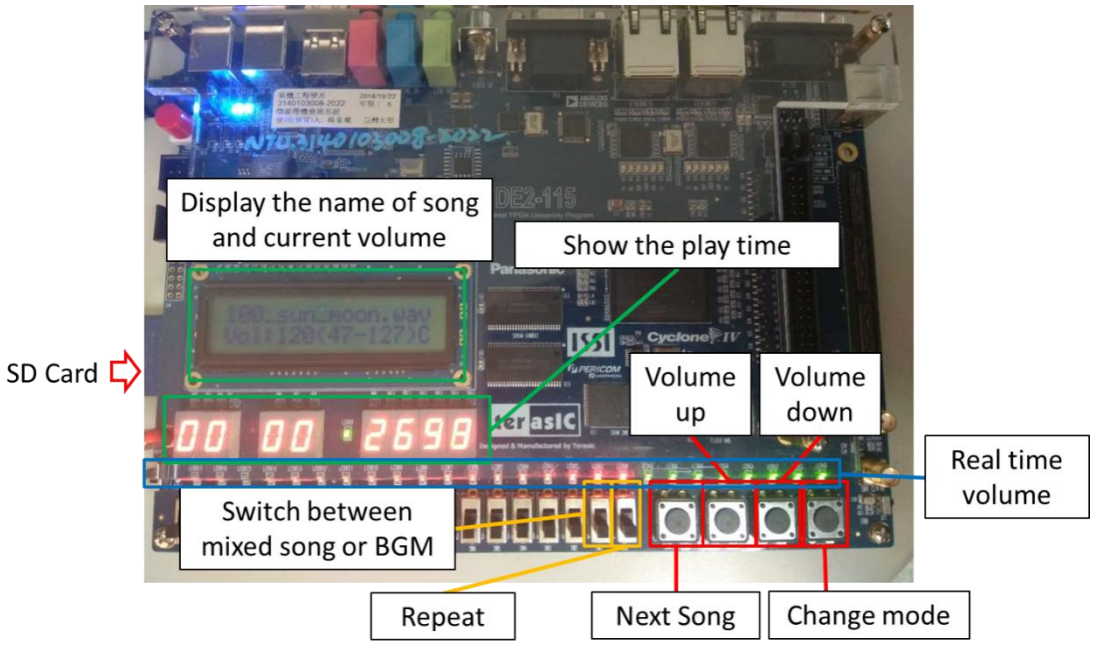
1.插入 SD 卡，系統會自動進入「聽歌模式」，按照字母排序從第一首歌曲開始播放，LED 燈會即時顯示聲音大小，七段顯示器會顯示播放時間，LCD 面板第一行為播放中的歌曲名稱，第二行為當前音量大小。

2.按壓 KEY1 可以調低音量，按壓 KEY2 可以調高音量。

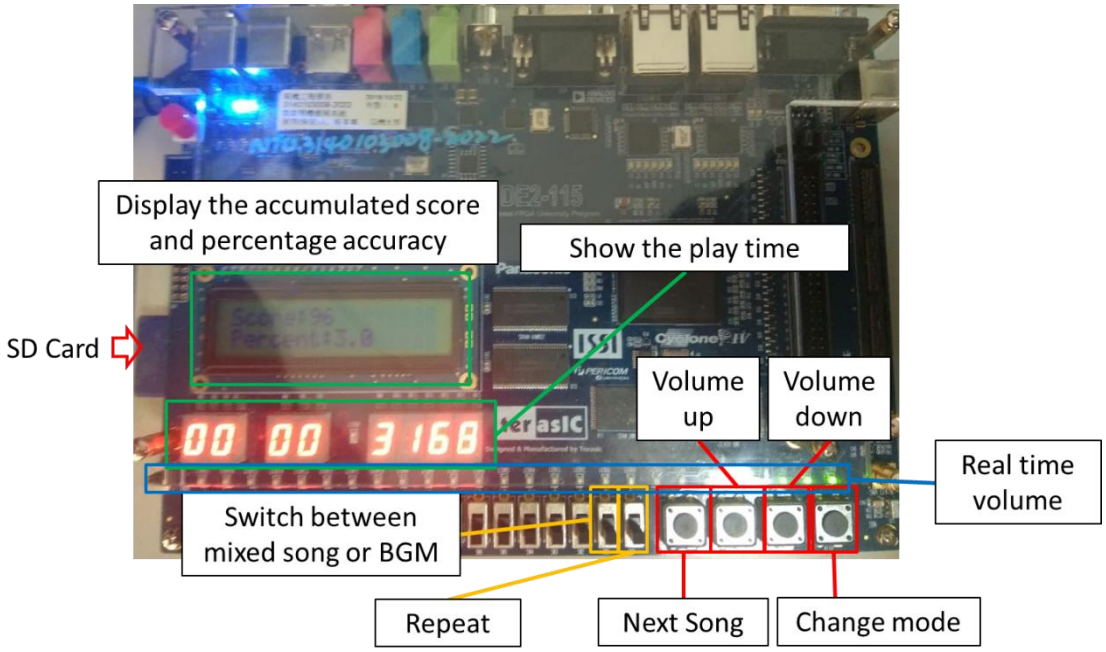
3.拉動 SW1 可以選擇要播放的是原始音檔還是純音樂音檔。

4.按壓 KEY3 可以切換到下一首歌。

5.按下 KEY0 即進入「K 歌模式」(karaoke mode)，此時播放的音檔與在「聽歌模式」(listening mode)時播放的相同，也就是說此系統支援導唱功能。此外，LED 燈同樣會即時顯示聲音大小，七段顯示器會顯示播放時間，而 LCD 面板則改為第一行顯示累加分數，第二行顯示百分比正確率。

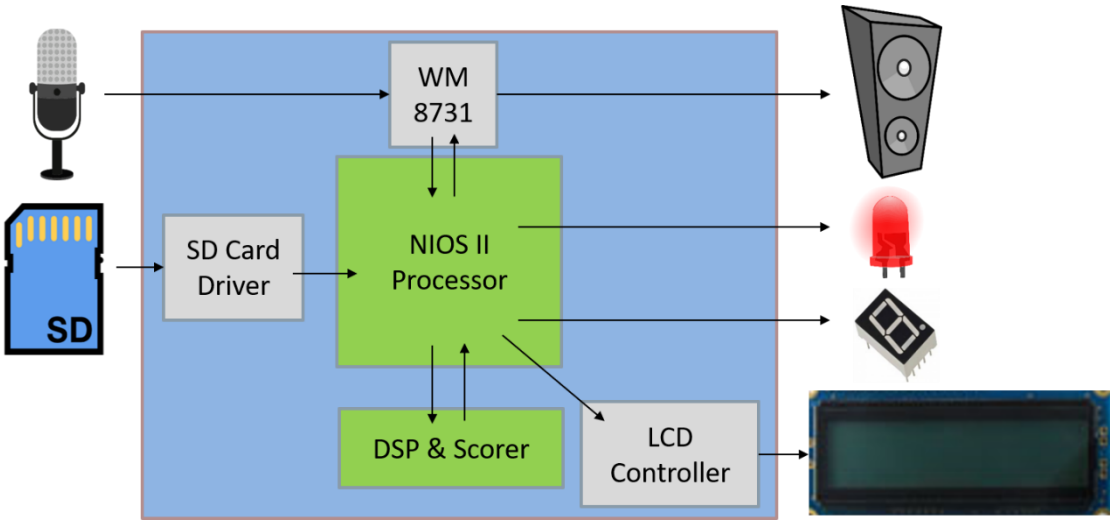


聽歌模式



K 歌模式

四、系統架構



本系統的核心為 NIOS II，藉由 Qsys 的接線對各個 IP core 進行操作，包括 SD 卡的讀取、麥克風音訊輸入、WM8731 音訊處理晶片的初始化與操作、喇叭的音訊輸出、LED 燈、七段顯示器及 LCD 顯示器。以上構成卡拉 ok 機的基本架構，而評分系統則是用 verilog 撰寫，再透過 Qsys 建立的 self-defined IP core 去操作 module 的 I/Oregister。

五、Qsys IP cores

1. Clk_50：產生 50MHz 的 global clock
2. CPU：NIOS II 的中央處理器
3. Timer：計時器，記錄處理器目前運行的時間
4. Onchip_memory：儲存 CPU 指令及資料的記憶體
5. Aptll：產生不同頻率的 local clock
6. Jtag_uart：處理 PC 與 DE2_115 之間的資料傳輸
7. I2C：初始化音訊晶片 WM8731
8. SD_clk, SD_cmd, SD_dat：與 SD 卡的讀取相關
9. KEY, SW, SEG7, LEDG, LEDR, LCD：分別控制 DE2_115 上不同的訊號輸出裝置
10. Score：Self-defined IP core，處理評分的計算

六、SD 卡格式要求

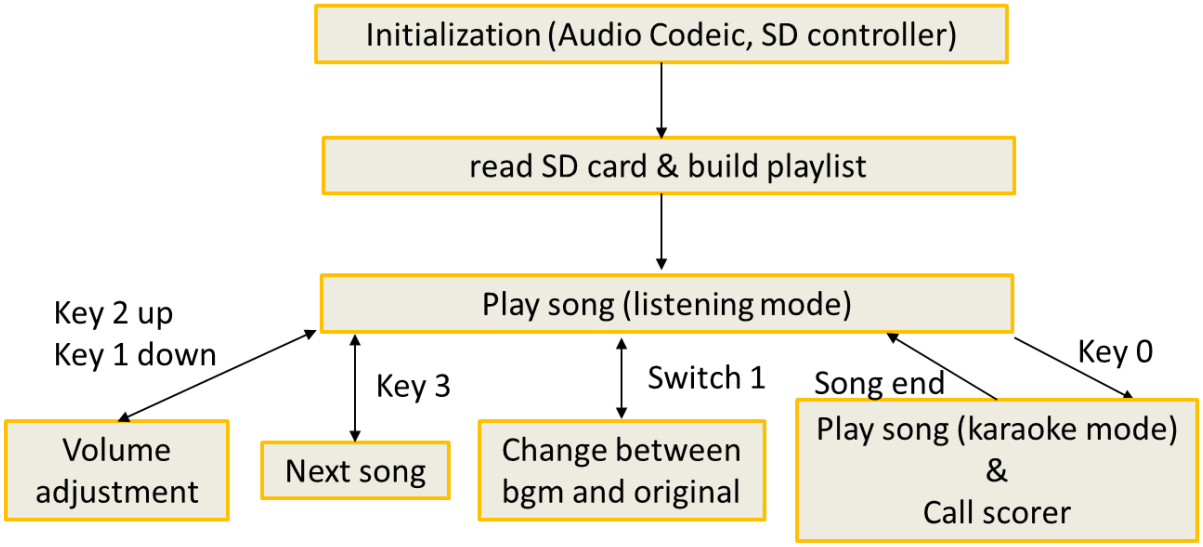
1. SD 卡需格式化為 FAT16/FAT32
2. 音樂檔案格式需為 WAV，並放置於根目錄底下，每一首歌皆需包含原始音檔(xxx.wav)、純音樂音檔(xxx(bgm).wav)與純人聲音檔(xxx(vocal).wav)。
   本實驗音樂分離的方式是使用此網站：https://moises.ai/。
3. WAV 的採樣頻率可為 96k/48k/44.1k/32k/8kHz(於本實驗中我們統一使用32k的採樣頻率,以符合麥克風輸入音訊的採樣頻率，才能進行比較)。
4. WAV 需為立體聲並且每個單位點的大小為 16bits。
   本實驗使用 GoldWave 軟體進行音檔格式轉換。

七、NIOS II

Nios II 是專為 Altera FPGA 系列設計的 32 位元嵌入式處理器架構，簡單來說就是先用 Qsys 在 FPGA 板上產生一個可以編譯並執行 C 語言的電腦，然後再將寫好的 C 程式碼放進它的處理器去運行指令。

在 Nios II 專案底下會有兩個子專案，一個是一般的 C/C++專案，包含 source file 跟 header file，而另一個是 bsp 專案，它的功能是將 Nios II 系統 Qsys 產生的硬體部分包裝起來。

程式運行大致流程：

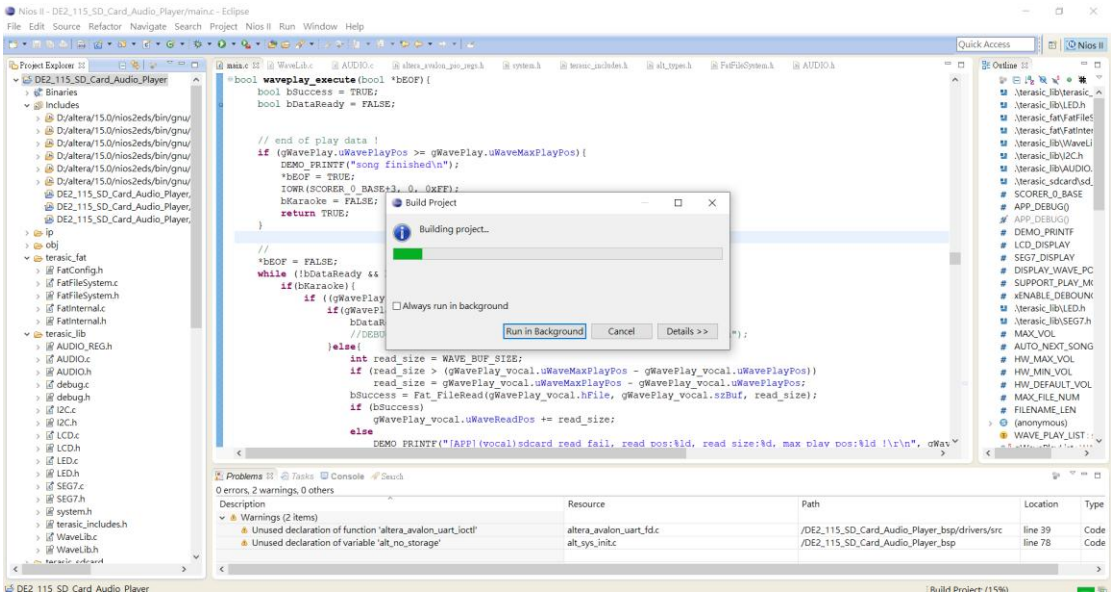


註：本實驗的卡拉 ok 機參考 DE2_115 光碟中的範例 DE2_115_SD_Card_Audio_Player 並進行大幅修改

好處：能夠簡化對每個 IP core 的操作。舉例來說，在實驗一開始的時候我們研究如何寫 verilog 操控 LCD，輸入 LCD controllor 的訊號基本上可以分為指令跟資料，以及告訴 LCDcontroller 說這筆輸入是指令還是資料的控制訊號，而不同的指令的功能又都不一樣，我們希望 LCD 能夠單純根據輸入去顯示字母與數字，於是我們參考了光碟片中 DE2_115_Default 的寫法，它使用了三層 module 去控制 LCD，可是它只能顯示固定的字串，故我們花了一些時間看懂它的寫法然後改成顯示的字串會隨著輸入而跳動，沒想到同組研究 Nios II 的人發現在 Nios II 中可以單用一行類似 C 的 output 指令就做到完全相同的事情，不免覺得 Nios II 在 IP core 操作上實在太過強大了。

難點：儘管 Nios II 簡化了 IP core 的操作，但是在寫程式時仍然不能完全不懂 IP core 的通訊協定，因為這些 C 語言也是建立在 verilog 的硬體描述語言之上，光是 main.c include 的 header file 數量就超級多，很多看不懂的 function 都是經過非常多層的呼叫，往往挖到

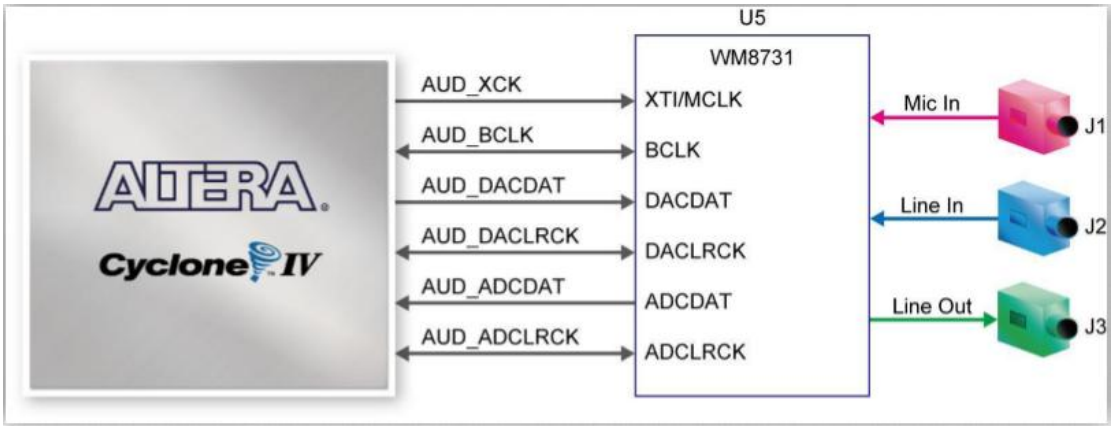
最底下時就會發現跟 verilog 在做的通訊協定是相同的。舉例來說，I2C 的初始化是藉由在不同 register 給定不同的 9bits 二進位數值，每個數值都有各自的含意，當時在做 LAB3 時便有遇到過，而在我們 Nios II code 中，main.c 裡面呼叫的 audio_init()函數做的恰恰也是一模一樣的事情，也就是說 verilog code 跟 C code 在某種程度上是對應的。不過也因為 include 的檔案很多，而且 Nios II 的路徑指定並不友善，所以在 compile 時常常會出現 function undefined 的訊息，這時就得去找尋遺失的 function 在哪裡，花費了不少時間。另一點是在前面的 LAB 中我們不曾使用過 Nios II 這套系統，所以摸索學習也花費了相當多的力氣。



Nios II build project 畫面

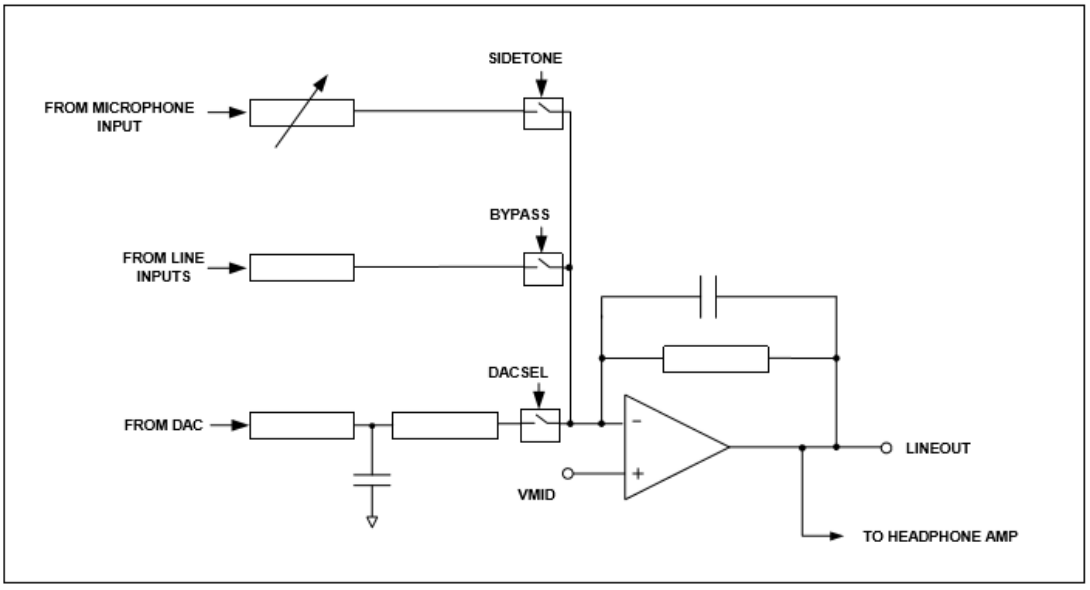
八、Audio Codec

本實驗的卡拉 ok 機不僅需要即時將 Mic In 音訊跟來自 SD 卡的音訊疊加後從 Line Out 輸出，評分系統也需要同時將 SD 卡中純人聲的音訊讀出與 Mic In 音訊進行比較然後計算得分，而由於 Mic In 為類比訊號，需要轉換成數位訊號後才能與 SD 卡中的音訊做疊加跟比較，故會需要使用到 DE2_115 的音訊處理晶片。



文獻閱讀後發現在 I2C 初始化時，對第四個 register 進行 initialize 的訊號會對卡拉 ok 機有所影響，故在此說明。以下為 Analogue Audio Path Control 每個 bit 數值的意義以及電路圖。

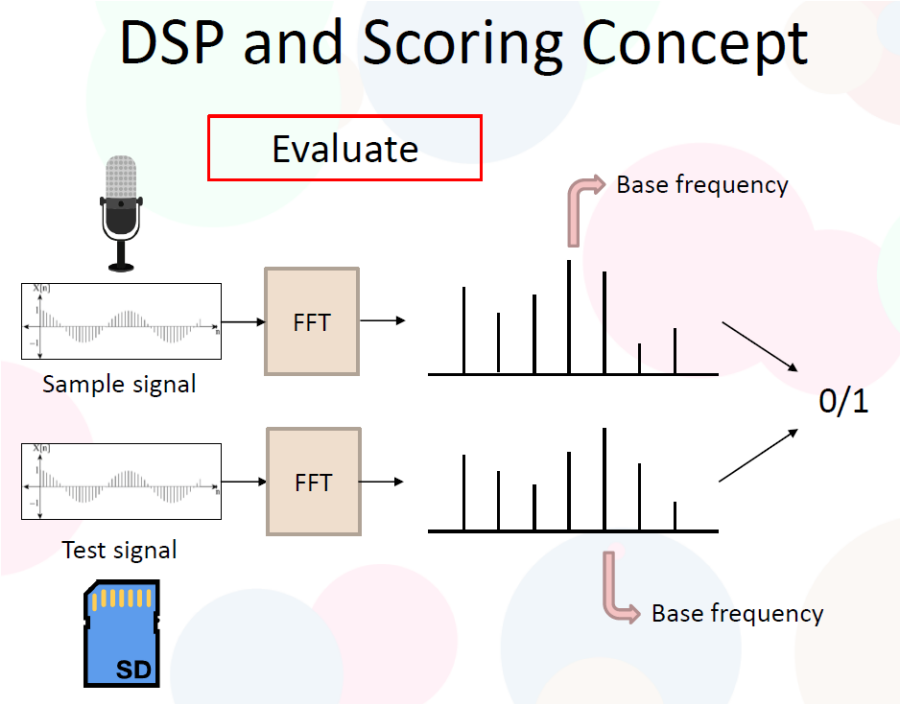
Register	Bit[8]	Bit[7]	Bit[6]	Bit[5]	Bit[4]	Bit[3]	Bit[2]	Bit[1]	Bit[0]
000_0100	0	SIDEATT[1:0]		SIDE TONE	DAC SEL	BY PASS	INSEL	MUTE MIC	MIC BOOST



經過測試後發現就算開啟了 Mic boost，若只有接通 Sidetone 或 Dacsel 其中一邊，那麼 Line out 人唱歌的聲音就會很小聲，所以最後我們設定成兩邊都會接通，Microphone input 提供的是經過 boost 後的原類比訊號，DAC 提供則是 SD 卡音樂與麥克風錄音經過音訊處理晶片做 A-D convert 再 D-A convert 的結果，兩者疊加再一起 Line out 才聽得到人唱歌的聲音。

九、DSP 與評分系統

原先構想：使用 Scorer module

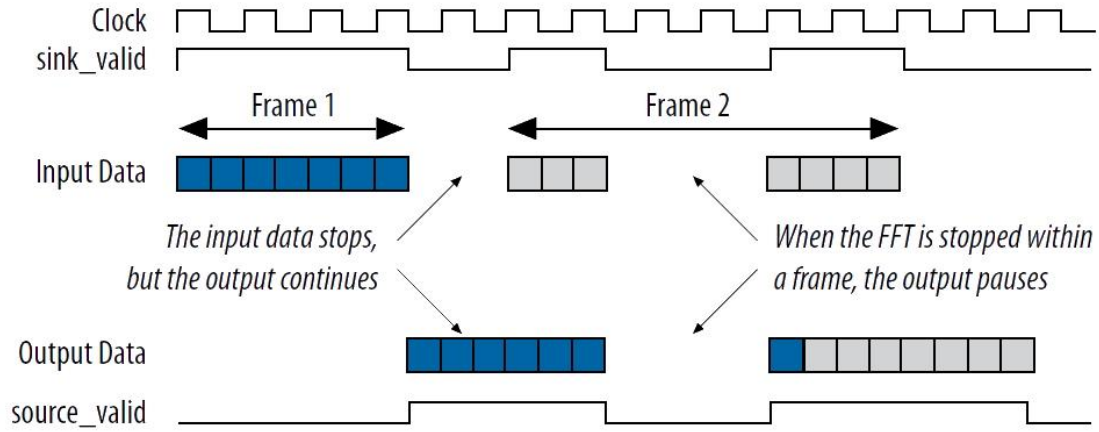


原先計畫使用的評分系統是基於頻率的。無論是麥克風輸入或是 SD 卡已處理好的 vocal 檔，音訊的 sampling rate 都是 32kHz, 基本涵蓋人類聽力的 range. 音訊 data point 之後送進 Altera FFT_ii IP Core. FFT 的 size of points 為 2048, time domain resolution 約是 1/16 秒，約相當於中等速度歌曲三十二分音符的長度。Frequency domainresolution 約為 16Hz,相當於中央 C 附近的一個半音。FFT IP Core 的 I/O 使用 data streaming, 每次吃進 2048 筆資料，就會開始花費 2048 個 clock cycle 去 output frequency domain data.我們之後取前 256 筆 frequency domaindata (相當於 4000Hz 以下), 比對 mic input 及 SD vocal inputfrequency domain 上資料點的最大值，若兩者相同，score 與 number of data points 都 +1, 若不同則只有後者+1. 我們也會即時在 FPGA 的 LCD 面板上，更新 score 及 percentage。

Sample signal 及 test signal 各包含左、右聲道，每聲道 16bit 的

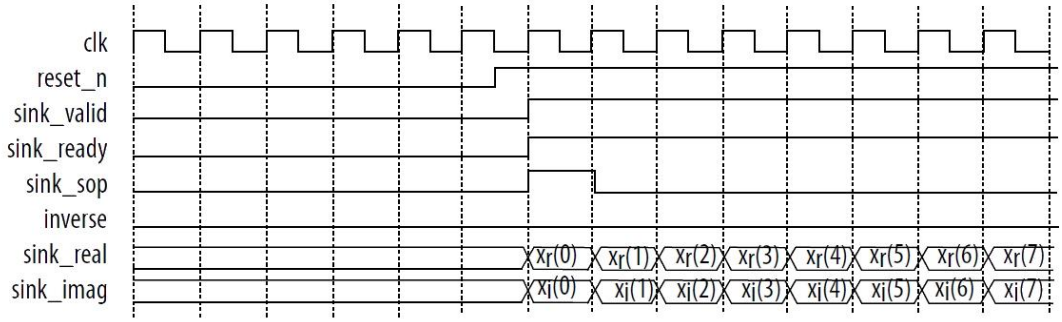
資料，I^2S I/O 最多支援到 32 bits，因此必須分兩個 clock cycle 把一個 Sample Point 的資料，傳入 Scorer, 先傳入麥克風的 time domainsignal，加總左右聲道訊號，並暫存到 buffer, 再傳入 SD Card 的訊號並加總站存，之後把兩筆資料同時 input 到兩個 FFT core, 從而確保 FFT core output 同步，能正確比對。

由於 clock frequency 為 50MHz, 而 sampling rate 則是 32kHz, 即兩次 sampling 之間有 1562.5 clock cycles. Comparing the frequency spectra between the mic input and SD input 需要花費 257 clock cycles, refreshing score and percentage 需另外花 1 個 clock cycle, 因此分數能在下一筆 sample data 抵達前及時更新。



Whensink_valid == 1, the Scorer will input a data per clock cycle and store it in a register.

FFT Streaming Data Flow Input



When reaching xr[2047] and xi[2047], sink_eop will be pulled up for one cycle to indicate the start of the output stream. Then sink_valid is pulled

down for the output process. Another control signal is source_ready. it is pulled up for the entire input/output process.

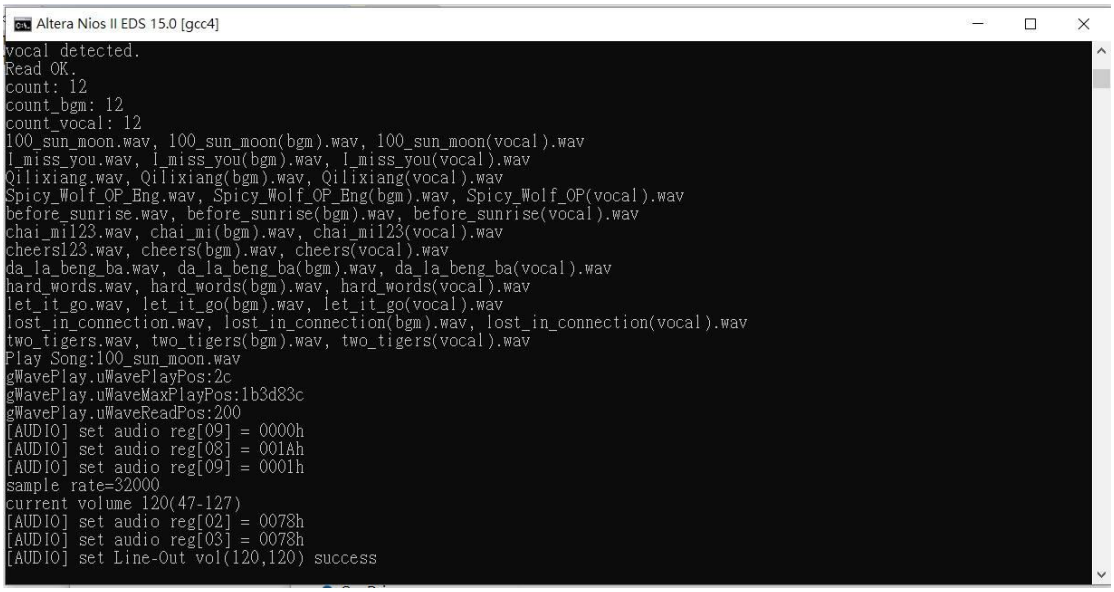
此評分系統的特色，在於扣分多寡取決於拍子不準的程度。例如，若拍子長 1/4 秒，而演唱者慢了 1/16 秒，得分為 3 分(滿分 4 分)。傳訊若有 delay 則以添加 offset 來解決，與多數音樂遊戲的機制相同。當初還考慮了升降 key 及加減速的功能，每升 1 個 key 需把 frequency * $2^{1/12}$,加減速則是先將頻率乘以速度倍數的倒數，轉換回 time domain 以後再 up/down sampling. 因為 frequency resolution 的問題，最終難以進行。

最終方案：使用 Time Domain Signal 評分

由於一些技術上的問題，我們無法順利使用 Nios II 正確 input/output data 到 scorer, 最終我們改成在 Nios II 的 main 裡寫 scorer 進行評分。評分的方式改為比較 SD vocal data and mic input 的振幅大小。我們以每 100sample points 為單位，取平均值，獲得音量大小的資料。每秒有 320 資料點。透過分別計算 SD signal 跟 Mic signal 相鄰兩資料點之間的 first difference，我們可以判斷節拍，評分，並計算正確率。另外，我們取 100 點的 total energy，比較 SD signal 及 Mic signal 的音量大小，作為另外一個評分依據。音量大小使用的是相對大小，這個項目判斷唱歌者是否能掌握歌曲的強弱變化。這套評分系統原先是作為 Frequency domain signal comparison 主系統的輔助評分系統，但最後變成唯一成功的評分裝置。

十、Demo

將 Quatus II 產生的 sof 檔跟 Nios II 產生的 elf 檔放在 DE2_115_SD_Card_Audio_Playe/demo_batch 底下，然後運行 bat 檔進行燒錄並監測程式的運行。



bat 監控視窗

實際操作影片連結：https://youtu.be/obytSkAKaL0

十一、 參考資料

1. DE2_115_User_manual
2. DE2_115_demonstrations/DE2_115_Defalut
3. DE2_115_demonstrations/DE2_115_SD_Card_Audio_Player
4. Lab3_sup1_audiocodec.pdf
5. Lab3_sup3_lcd.pdf
6. DE2_115_datasheets/Audio Codec/WM8731.pdf
7. Intro_Nios_II.pdf