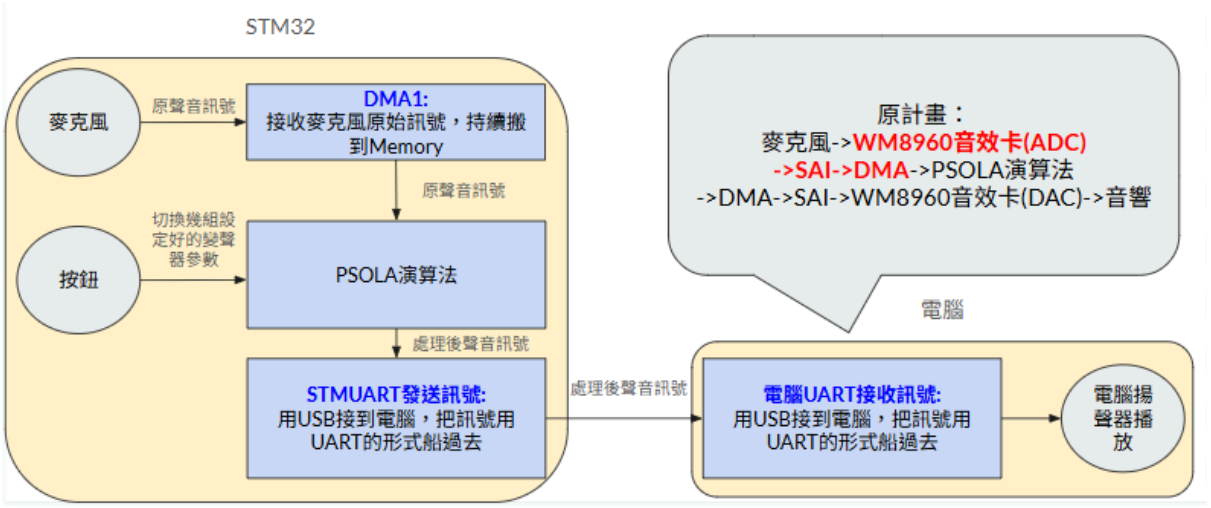
# 嵌入式系統實驗_FinalReport_柯南變聲器

b10611038_余佳融 林祐寬

1. **[連結]**
   所有檔案連結:
   https://drive.google.com/drive/folders/1I3ilQDOKKnBZask6ajzDbbWrkNM-tM-U?usp=sharing
   **Github連結:**
   https://github.com/Eskartur/Embedded_Systems_Lab_Final_Project
   **YouTube連結:**
   https://www.youtube.com/watch?v=LVmtF2DANbE
   https://www.youtube.com/watch?v=supxx6Rj3OY

2. **[Final Project主題]**
   本專案使用STM32為硬體架構，製作出一個能夠Realtime變幻出兩種聲音的變聲器。技術上，本專案使用DSP方法自STM讀取聲音訊號，並經由PSOLA演算法做聲音的處理，最後利用UART的方式將處理後的聲音訊號由STM傳送至電腦播放。
3. **[動機]**
   覺得柯南可以用變聲模仿人很帥，也想自己試著制做一個變聲器。

4. **[系統架構圖]**




5. **[作法]**
   a. 系統1: **STM讀取聲音資料 — DSP**
      i. 收到DMA Buffer半滿/全滿的Interrupt後，讓PSOLA函式讀取並處理Buffer內容
      ii. PSOLA函式處理完，即使用UART Transmit將儲存已變聲的訊號的Buffer內容送到PC端
   b. 系統2: 處裡輸入音訊(變聲) **— PSOLA:**
      i. 我們使用 TD-PSOLA(Time-Domain Pitch Synchronous Overlap and Add)作為變聲器的演算法。此演算法直接在時域中對信號進行操作，可省去對信號進行傅立葉轉換再處理的時間與空間需求。此演算法能分別以基頻(Pitch，決定音高)、共振峰(Formants，決定音色)兩個參數改

變聲音的特質[e]，雖無法直接模仿特定人物的聲音，但能夠改變聲音大致的性別、年齡的印象，而不會出現"鼠來寶聲"等不自然的現象。

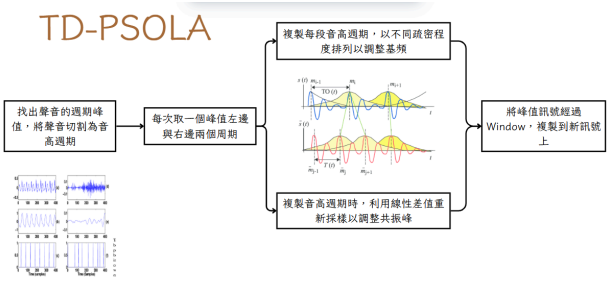
ii. 此演算法需先將聲音依照"音高週期(Pitch period)"切成若干段，且切割的位置必須位於每個週期的峰值。找出音高週期與峰值的演算法有諸多實現方式，例如使用自相關(Autocorrelation)函數去找出可能的週期，而經過我們的實驗發現，KSR Murty等人所使用的Epoch Extraction(找出聲門關閉瞬間)演算法[f]，不僅運算量相對其他方法較少，適合在嵌入式系統實作，用於TD-PSOLA也能達到與其他方法音質更好的變聲。

iii. 我們實作的Epoch Extraction演算法先將訊號進行差分運算以過濾低頻，再將訊號通過協振器(Resonator)，接著減去局部平均值，以最終訊號的負值到正值的轉變時間預測週期峰值。根據人聲的最高到最低頻率，在實作中需限制峰值間的最小與最大距離。原研究所使用的ZFR (Zero Frequency Resonator)協振器為非穩定(Non-Stable)，輸出值會以二次函數增長，在實時運算中會導致精度損失，以及需減去局部平均值兩次。最後我們使用了改良版的ZP-ZFR，為ZFR的穩定版本[g]。

iv. 取得週期峰值後，此演算法將峰值左右各一個週期，經過窗函數(我們使用最簡單的線性窗函數)後，複製並疊加到新的訊號上，每次複製訊號都會有一個週期的重疊。我們以新訊號的位置作為基準，逐漸往右疊加訊號，並以最接近該位置的峰值周圍的週期作為複製來源，並以該峰值右邊的週期決定下一個位置。

v. 若以原訊號的週期決定下一個位置時，以某個特定比例增加或減少，並以新訊號的週期決定複製的窗函數，則新訊號的峰值分布將以該比例變得更疏或更密，因而降低或升高基頻(音高)。但因為原訊號每個週期的波形幾乎沒有受影響，因此共振峰(音色)不變。

vi. 若複製波形時，將原波形按某特定比例縮放(複製的範圍需依比例調整)，但新訊號的峰值分布不變。如此能再不改變基頻(音高)的狀態下，調整每個週期的共振峰(音色)。我們使用運算量最小的線性差值進行縮放。



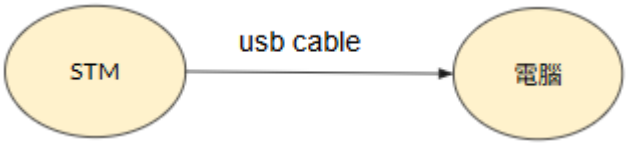

vii. 因此，此演算法能以分開的參數調整音高與音色，而做到性別印象的轉變。例如成年女性的音高約為成年男性的2倍，但共振峰位置約為1.2~1.4倍。若將所有頻率調整成2倍，音高與共振峰皆變成2倍，則會聽起來像幼兒或電影《鼠來寶》中花栗鼠的聲音。透過兩個參數的不同組合，能達到各種不同的聲線，例如若原聲音是女聲，音高降低12個半音，共振峰降低4個半音，能變成年輕男性的聲線，但若共振峰降低6個半音，則變成粗獷大叔的聲線。若不調整共振峰，而在小範圍之內調整音高，則可作為原聲線的音高修正。

viii. 本專案的目標是將此演算法實作為可直接在STM上接收訊號後實時(Real-Time)運算並輸出變聲後的訊號。為了達成實時訊算，我們需面對一些挑戰與考量。TD-PSOLA演算法為非因果(Non-Causal)的系統，一個時間點輸出的訊號可能取決於未來輸入的訊號，故此演算法實時運

算會有不可避免的延遲。另外，因此演算法需使用過去與未來的訊號，且時間差並不固定，實作上需使用緩衝區(Buffer)，且緩衝區的大小需計算edge case來決定。

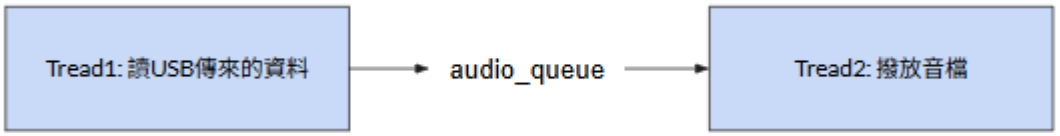
c. 系統3: 傳送處理後的音訊資料至電腦並同步撥放 **— UART**

i. 系統3硬體架構: 利用USB連接STM與電腦。




ii. 系統3軟體架構(**STM**端): 讓STM將處理後的音訊資料不斷的由USB接口利用UART的方式傳送到電腦。當Buffer一半以上的訊號已處理完畢，STM會將一半Buffer大小的訊號作為一個Chunk直接傳送Bytes到UART。為了讓接收端辨識Chunk的開始位置，在傳送Chunk前會先傳送4個Bytes的Marker，並讓接收端比對Marker來同步訊號。

iii. 系統3軟體架構(電腦端): 電腦端將會利用COM來接收傳來的資料，並開啟兩個Thread。Tread1將會讀取自USB傳來的訊號，並將資料儲存到audio_queue中，Tread2會同時讀取audio_queue的資料，並同步撥放到電腦的揚聲器上。




```python
# Serial config
ser = serial.Serial(
    port='COM13',
    baudrate=921600,
    parity=serial.PARITY_NONE,
    stopbits=serial.STOPBITS_ONE,
    bytesize=serial.EIGHTBITS,
    timeout=2
)
```

```python
# 🎙️ 接收音訊 thread
def receive_audio():
    print("▶️ 開始接收 serial 音訊...")
    try:
        while True:
            # print("in recieve while")
            if ser.read(4) != marker:
                wait_for_marker()

            data = ser.read(FRAME_SIZE)
            if len(data) != FRAME_SIZE:
                continue

            samples = struct.unpack('<1650h', data)#'<

            try:
                audio_queue.put(samples, timeout=1)
            except queue.Full:
                print("⚠️ Queue 滿了，資料丟失")
    except Exception as e:
        print("❌ 接收錯誤：", e)
```

(Thread1 code)

```python
# 🔊 播放 thread
def play_audio():
    print("🎧 開始播放音訊...")
    try:
        while True:
            try:
                samples = audio_queue.get(timeout=1)
                stream.write(struct.pack(f"{len(samples)}h", *samples))
            except queue.Empty:
                print("🔇 等待資料中...")
    except Exception as e:
        print("❌ 播放錯誤：", e)
```

(Thread2 code)

6. **[成果]**

a. 專題最終成果: 最終，我們利用調整PSOLA的共振峰與基頻位置，調整了兩種適合我們聲音的變聲器聲線。

i. 聲線1: 男生音(林祐寬聲) → 女生音(小美的聲音)
(基頻高12個半音 → 2倍，共振峰高4個半音 → 1.26倍)

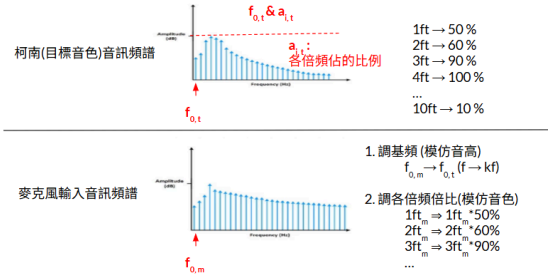
ii. 聲線2: 女生音(余佳融聲) → 男生音(小帥的聲音)
(基頻低12個半音 → 0.47倍，共振峰低3個半音 → 0.84倍)

b. 專題成果展示影片連結:

i. 聲線1(小美): https://www.youtube.com/watch?v=LVmtF2DANbE

ii. 聲線2(小帥): https://www.youtube.com/watch?v=supxx6Rj3OY

7. **[其他嘗試]**
    a. **FFT聲音處理方法:** 在製作的前面，我們也有在python分別嘗試用 FFT來調整聲音與用PSOLA調整聲音的方法，但在比較過後，我們發現PSOLA的效果會比FFT的效果聲音清晰與彈性高。
        i. 原始聲音: https://drive.google.com/file/d/17E2jCKcFZvcmYDMgqdDjSsrQ_-9fG0ph/view?usp=drive_link
        ii. FFT聲音: 在很高音的時候或是很低音的時候會不自然，能調整的音域聲音比較會受限，可能會出現鼠來寶聲音。 https://drive.google.com/file/d/1tEe_7TF7mcdey7FOZqEdqzpLiMfE22Zh/view?usp=drive_link




        iii. PSOLA: 可以分別處理基頻(音高)與共振峰(音色)，更好的還原人類的音色，與比較清晰。 https://drive.google.com/file/d/16rv10EtXg5DpRZAW33NTb14N361oFHCy/view?usp=drive_link

    b. 音效卡: 我們的專案將變聲器演算法直接實作在STM上，本來的理念是希望能不透過PC，只要將麥克風與音響接入嵌入式系統，就能直接達成變聲效果。我們嘗試了使用WM8960音效卡(因原開發版缺乏可用的PIN位，我們需使用沒有內建麥克風的STM32 Nucleo)，並以外接麥克風與音響同時輸入與輸出，也可以達到客製化的麥克風造型(如柯南的蝴蝶結變聲器)。但因WM8960音效卡的設定與資料傳輸非常複雜，我們未能成功同時輸入與輸出聲音，作為折衷方案，我們最終使用UART傳送到PC播放。

8. **[文獻及參考資料]**

    a. https://ithelp.ithome.com.tw/m/articles/10206570
    b. https://blog.csdn.net/zhuoqingjoking97298/article/details/125581253
    c. https://www.ni.com/zh-tw/shop/data-acquisition/measurement-fundamentals/analog-fundamentals/understanding-ffts-and-windowing.html
    d. Mousa, A. (2010). Voice Conversion Using Pitch Shifting Algorithm by Time Stretching with PSOLA and Re-Sampling
    e. http://koigoemoe.g2.xrea.com/koigoe/koigoe.html
    f. Murty, K.S., & Yegnanarayana, B. (2008). Epoch Extraction From Speech Signals. IEEE Transactions on Audio, Speech, and Language Processing, 16, 1602-1613.
    g. Vuppala, D.A. Implementation of epoch detection and its application on FPGA.