生醫工程實驗期末專題書面報告

-可攜式心電訊號顯示與紀錄機

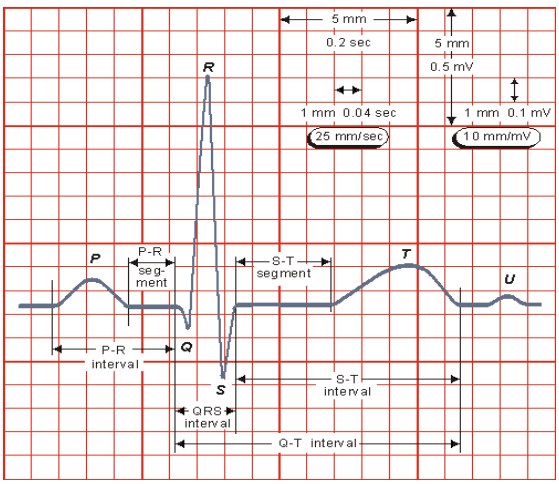
第三組 B92901031 黃俊仁
B92901079 林光威
B93901064 蔡希鈞

一、實驗簡介：

一般醫院常用的心電圖檢查通常需要患者躺在一台大型的心電圖機器旁邊，沒有辦法在家或攜帶式的量測。然而，許多心臟疾病患者的其心臟的跳動不是隨時都不正常，可能平時都很規律，有時候卻突然不正常，因而造成到院檢查時無法量測到不正常心律波形的情況；因此，若是有一個可攜式（portable）並具有長時間（24 hour）監控效能的心電圖機，即可讓心臟病或疑似心臟病患者可以隨身攜帶心電圖機並隨時監控。另外，有些心臟患者平時心律都很正常，突然發病起來卻可能有致命危險；因此，若是有一個可以即時監控並在危急時發出求救訊號的可攜式心電圖機，即有可能在危急時刻挽救一條人命。有鑑於以上兩點，我們這組決定嘗試實做一台可攜式且可長時間監控心律的心電圖機，並且加入心電圖判讀功能以及偵測到異常心電訊號時發出求救訊號的功能。

二、實驗原理：

1. 心電訊號擷取：在心電訊號的擷取方面，我們是採用實驗四的方法，用 OP 和電阻電容兜出儀表放大器和濾波器以及 DRL（Driven Right Leg），然後將得到的心電訊號輸入 EasyARM 學習版的 ADC 輸入，轉成數位訊號後交給 ARM 做訊號處理和判讀。
2. 心電訊號判讀：在做心電圖的判讀時我們要有判讀的標準，而標準的心電訊號圖形與定義請參看下圖一。



圖一、心電訊號各個 interval 的定義

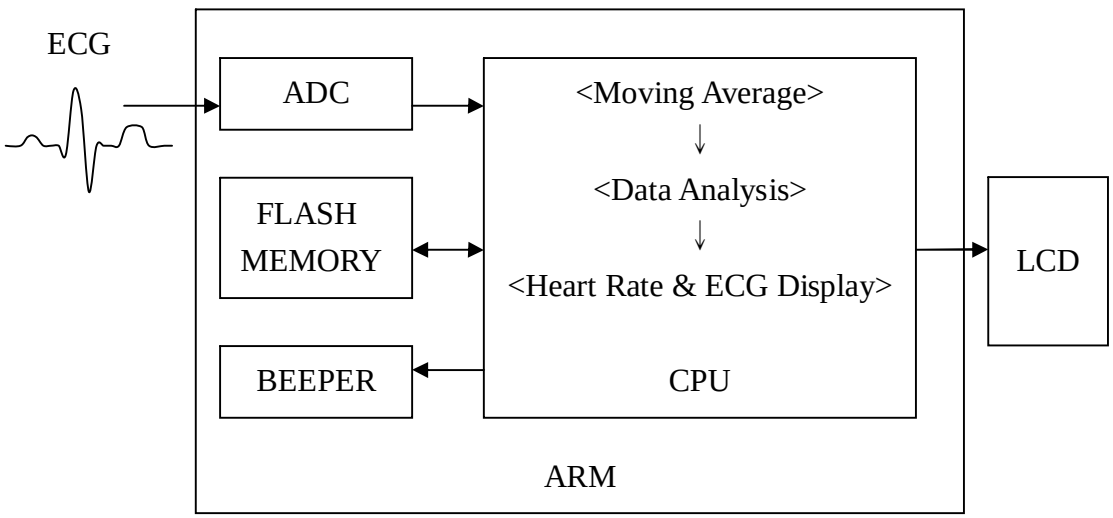
如果我們可以將我們所擷取到的訊號與標準做比較，然後再經由不同處去判斷是否心電圖有反應出心臟的異常。不同的病徵會經由不一樣的異常 ECG 反應出來，因此我們或許可以判斷出異常後還可以再進一步與 ECG 的資料庫作比對以找出病因。其心電盼讀的資料可參看參考資料內的 database。[1][2][3][4][5]

而我們實驗在實際的心電判讀部分，由於要複雜的心電訊號判讀不是一個短短的期末專題可以實現，再加上我們的儀器只是用於簡單的監測和紀錄，因此我們的心電圖判讀部分最主要是判斷 QRS 並計算 R-R interval，然後根據所抓到 baseline 的心跳速率平均和標準差，判斷現在的心跳速率過快或過慢。

3. 心電訊號顯示：在顯示方面，使用 SMG240128A 液晶圖形顯示器與 ARM 的模組相接，藉由 ARM 送訊號給液晶圖形顯示器顯示。顯示方面分三大塊，一部分像示波器一樣即時顯示現在的 ECG 波形圖，一部份顯示現在的心跳速率，一部份顯示現在心跳的狀態。
4. 發出求救訊號：在有危險時，不管是過快過慢，ARM 版都會啓動蜂鳴器發出求救訊號，以提醒使用者本身和呼救身邊的人。另外，原本向外呼救還有考慮要用藍芽模組無線傳給電腦，並由電腦自動啓動 SKYPE 撥號，但由於時間的關係最後沒有完成。

三、細部實行方法：

流程圖：



圖二、流程圖

訊號處理與判讀：

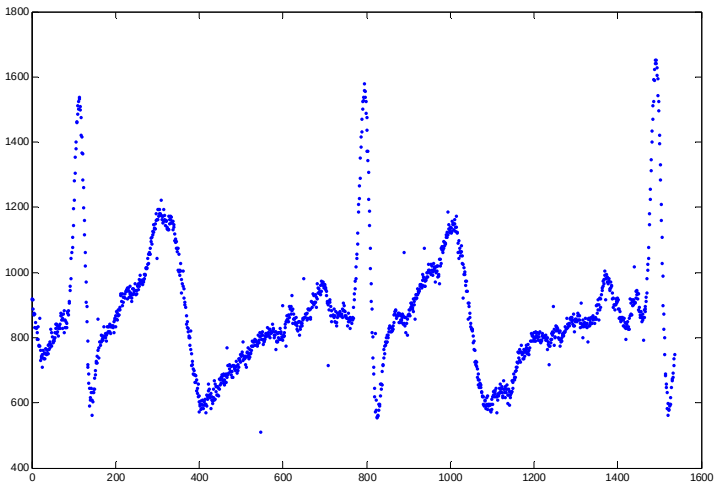
Moving Average

由於直接從 ADC 量測到的信號仍偶有高頻雜訊，所以抓取到訊號後，我們先做一個簡單的 moving average，以去掉高頻雜訊。方法很簡單，如下式：

$$y(n) = \frac{1}{2}[x(n) + x(n-1)]$$

我們也曾做過三點和四點的平均，效果當然比較好，但是看起來兩點的平均已經足夠，加上考慮程式的運算速度，最後使用的是兩點的平均。

(附圖)



圖三、從 ADC 存到 FLASH 的資料用 MATLAB 顯示的圖（已做過 Moving Average）

Data Analysis

分成以下幾個階段：

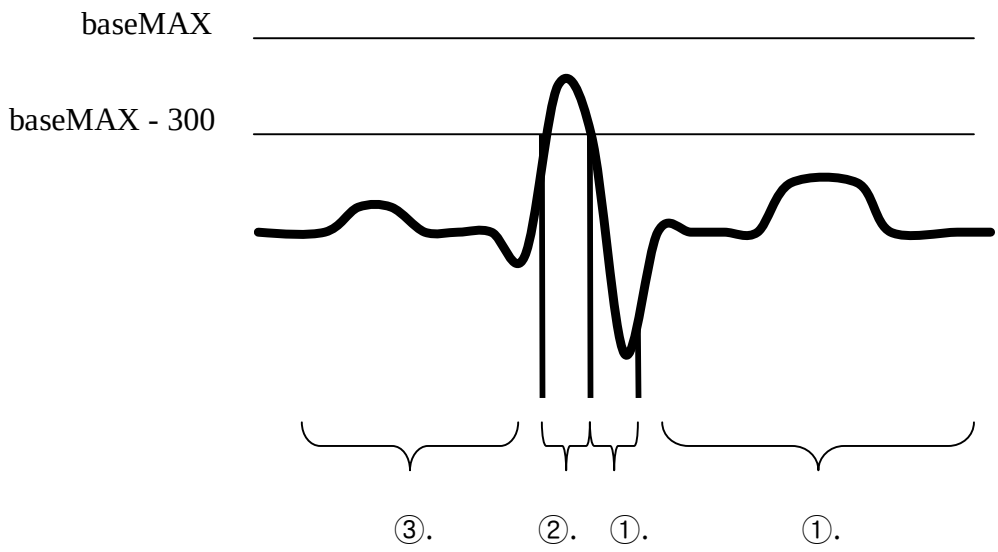
一、MAX Value Acquisition

這是下一步驟的前置步驟，要先抓取一小段信號(約二秒鐘)，找出其最大值。這個最大值必定爲某一次 R 波的數值，我們將之稱作 baseMAX。

二、Baseline Acquisition

求得 baseMAX 後，我們再抓取一段信號(約八秒鐘)，先將這段信號全部做完 moving average 並存入記憶體，再依序讀出並依下列步驟分析。

1.



圖四、心電圖判斷說明圖

①. 不做任何動作，直到輸入的訊號值大於「baseMAX - 300」，即進入下一步驟。

②. 爲避免數值只是因爲雜訊干擾而進入「baseMAX - 300」的範圍，在進入這個步驟後，會先確認接下來三個數值都大於「baseMAX - 300」，才會開始紀錄這一階段的最大值(也就是 R 波的值)。同理，爲避免數值是因爲雜訊干擾而掉出「baseMAX - 300」，會確認接下來三個數值也都小於「baseMAX - 300」，訊號才算真正掉出去。此時，將所紀錄到最大值的¡ 值¡ 與最大值在記憶體裡的¡ 位置¡ 寫入記憶體當中的另一個區塊，並進入下一個步驟。

③. 這個步驟中，只要下一個讀取的信號比上一個的值還要小，就記錄下新的值。未避免雜訊干擾，如果下一個的值比上一個的值連續高十次，才會判斷爲 S 波已經過了。此時，記下最小值的¡ 值¡ 和最小值在記憶體裡的¡ 位置¡ ，並回到步驟①.。

2. 計算 ADC 的速度。之前已經設定好 ADC 的速度大約是每秒 1Khz，每次的實測結果大約是 950Hz 左右。

3. 將各最大值的位置讀出，以計算心跳率。我們的想法是，假設 ADC 的轉換速率固定，CPU 的速度又比 ADC 快很多，這樣一來兩次最大值之間的位置差就代表著經過了那麼多次的 AD 轉換，便可求得兩次最大值之間的時間差，也就是心跳的週期，可進一步求得一分鐘的心跳率。每兩個 R 波可得一個心跳率，將之記在記憶體中另一區塊。
   後來我們開發了 ARM 板上的定時器，於是就把程式改成在找到最大值

時紀錄下計時器目前的時間；兩兩最大值之間的時間差就是週期，省掉了計算的 ADC 速度(步驟 2.)。

4. 將各心跳率的值讀出，求平均值與離差的平均。離差的平均求法如下：

$$d = \frac{1}{M}\sum_{i=1}^{M}\left|x_i - \bar{x}\right|$$

我們將 d 值視做標準差，因爲正常求標準差需要進行開根號的運算，但 ARM 的 C compiler 似乎沒有支援這個函式。

Heart Rate & ECG Display

經過上述兩階段之後，最後這一部份就簡單許多。由於後來不知道能用 S 波的特性做哪些判讀，後來我們就把 S 波這一部分省略掉，只單純求輸入信號的 R 波，判斷方法同二之 1.之②.、①.，並把算出來的心跳率利用液晶圖形顯示器顯示。

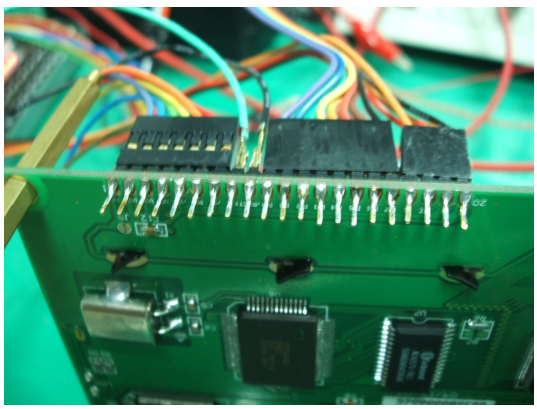
液晶圖形顯示器：

簡介：

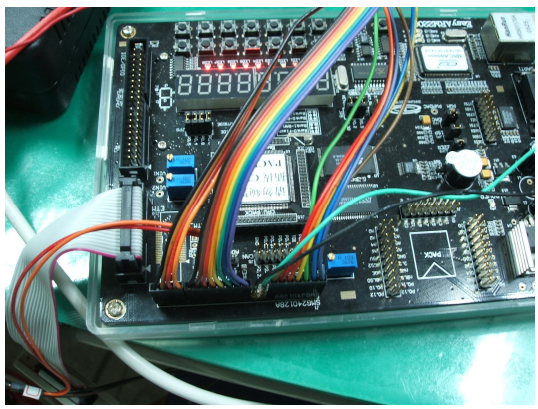
採用 SMG420128A 點陣圖液晶圖形顯示器，該顯示面版主要由 21 個 pin 腳和 ARM 版相接，利用 ARM 版傳送訊號給液晶圖形顯示器，液晶圖形顯示器在根據所傳送的訊號連結。而此液晶圖形顯示器根據參考書的說法，其像素總共是 240*180，然而根據我們實際測試的結果，助教拿給我們的 SMG420128A 其像素爲 240*128。

硬體連結：

主要就是和 ARM 版上的 J1（SMG420128A）模組連接，21 個 pin 腳按照順序相對，細部連接方式則是將一排排針焊在液晶圖形顯示器上，然後再利用排線一根 pin 腳和一根 pin 腳兩兩對接。(我們本來是用杜邦頭自己做排線，結果發現我們自己做的接線不太穩，液晶螢幕輸出不太好，後來助教拿買來的排線讓我們直接接，就可以正常輸出)。請參看下圖五和六。



圖五、液晶圖形顯示器部分的接線



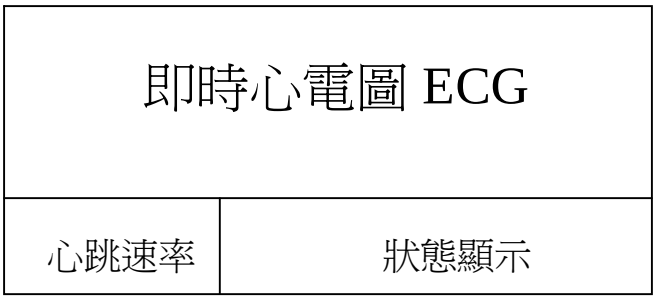
圖六、ARM 的 SMG420128A 模組接線

軟體編寫：

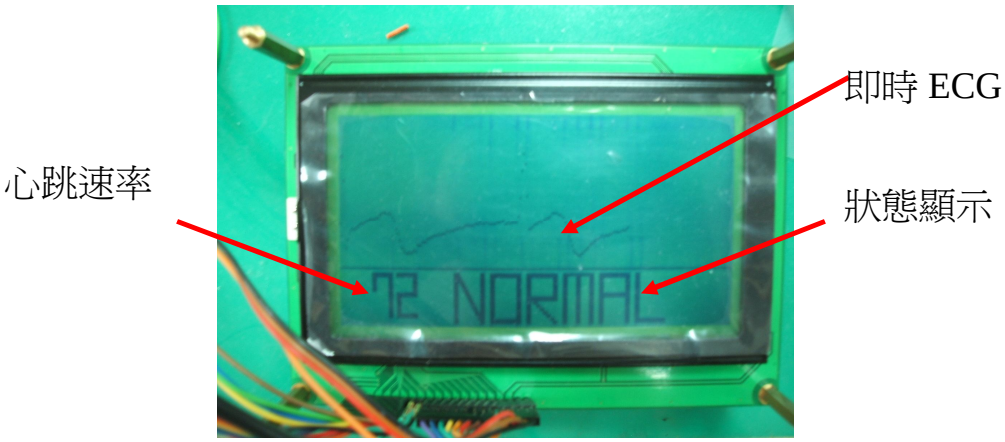
軟體的編寫主要是建立在原先光碟上和書本上所附的參考程式，該程式可以有在顯示器上畫一條水平線、一個點或是將整個螢幕填滿的功能，而我則將附加了可

以部分填充、可以畫垂直線的功能，利用這些功能來實行顯示心電圖、數字、英文字的功能。另外，由於只要有電源在，上次的顯示都會一直停留在螢幕上，因此程式編寫程式時在液晶圖形顯示上不能直接覆蓋，要先使要寫的地方整塊不亮，再讓要亮的地方亮，把字寫上去。

首先，大概簡介版面配置，在整個 SMG420128A 的顯示上，我們將他切成三大塊區域（如圖七和圖八），分別是即時顯示心電圖、心跳速率的顯示、心跳速率狀態的顯示。

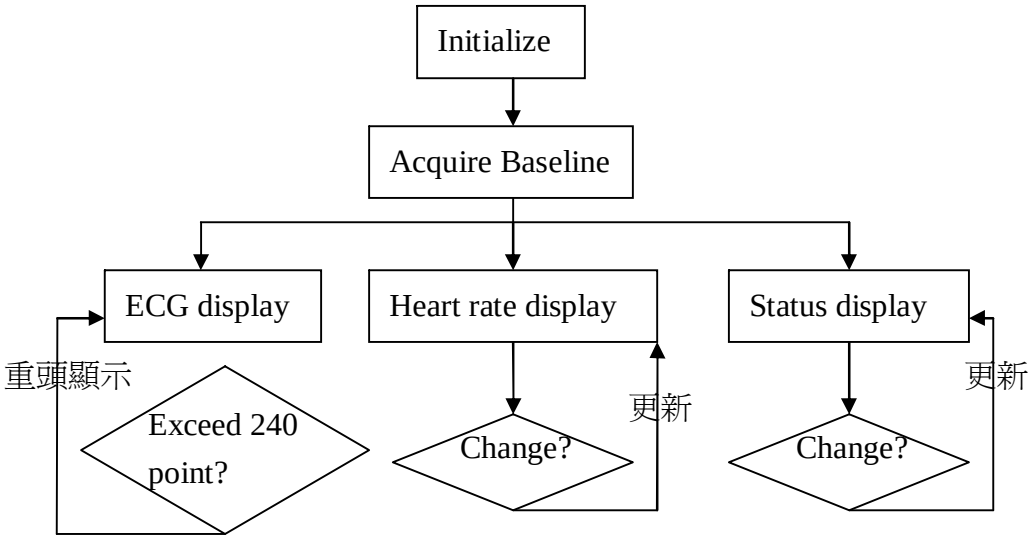


圖七、液晶顯示器三大顯示區塊



圖八、液晶顯示器三大顯示區塊實際圖

接著，簡介顯示的流程圖（請參看下圖九）：



圖九、液晶顯示器顯示流程

1　Initialization：當程式正在初始化時，一開始先將全部螢幕都填上 0（0 爲不亮，1 爲亮）；接著，在螢幕上顯示¡INITIALIZING¡的字樣。如圖四。

2　Acquire Baseline：當程式正在擷取受測者的基礎心電訊號，並計算其心跳速率之平均值和標準差時，會把原來¡INITIALIZING¡的字樣洗掉（即覆蓋爲 0），再顯示¡ACQ. BASELINE¡字樣。

圖十、顯示¡INITIALIZING¡字樣

圖十一、顯示¡ACQ. BASELINE¡字樣

3　即時顯示 ECG 波形和心跳速率與狀態：主程式每次都會傳送現在接收到的心電訊號（傳入現在電位）、現在的心跳速率、現在的狀態給控制顯示器的 function，另外也會傳入此心跳速率和前一次心跳速率是否有不同、此心跳速率和前一次心跳速率是否有不同、以及現在傳入心電電位應該顯示在哪個像素。

3.1　ECG 心電訊號顯示：由於液晶顯示器的水平軸共有 240 個 column，因此主程式會先計算線在傳入的心電訊號應該在哪個 column，一個 column、一個 column 的往後顯示心電圖（就像示波器一樣），若是控制顯示器的 function 收到現在應該顯示在第一個 column，即表示現在是全部 240 個行都顯示過要再重第一個 column 開始顯示，因此該

function 會先將整個 ECG 部分的版面全部清空（全部送 0 的訊號），接著再開始顯示第一個 column 的位置。至於每個時間的心電電位應該顯示在哪個 pixel，此顯示的 function 會根據送進來的電位值（以 mV 爲單位），將其 normalized 之後再乘以 90（顯示 ECG 的 row 共 90 個），然後顯示再上面的 ECG 顯示圖當中。

3.2 心跳速率顯示：由於每抓到一個心跳電位就會傳入一次心跳速率，若是不曉得心跳速率有沒有變的情況下，就得每次都先洗掉螢幕再重新顯示心跳，這樣會造成顯示出來的心跳速度即使是沒有變化，仍然會一直重新顯示造成一直閃爍的情況（心跳速率大部分時間都沒有變化，因爲要抓到下個 peak 才會重新計算一次心跳，但主程式是整個心跳週期都一直傳送 ECG 電位給 function）。因此，程式要能記住原來的心跳速率，顯示時再根據心跳是否有沒有改變來考量是否要先洗掉螢幕再顯示，但由於一個 function 的變數是 local variable 不能一直保留原來的值，因此就從主程式送給 function 心跳是否有改變的訊號（也可以考量設成 global variable 但我們覺得可能浪費記憶體沒有採用），若是心跳速率和原來的一樣，就不做任何事，若是心跳速率有改變，就先洗掉螢幕再顯示新的心跳速率。

3.3 心跳狀態顯示：主程式會將現有的心跳速率和 baseline 抓到的心跳速率比較，若是現有心跳速率大於平均值加三倍的標準差就表示心跳過快，傳送心跳過快信號給 function，function 就顯示¡TOO FAST¡（如圖十二）的字樣；若是現有心跳速率小於平均值加三倍的標準差就表示心跳過慢，傳送心跳過慢信號給 function，function 就顯示¡TOO SLOW¡（如圖十三）的字樣；若是現有心跳速率介於兩者之間就表示心跳正常，傳送心跳正常信號給 function，function 就顯示¡NORMAL¡（如圖十四）的字樣。另外如同 3.2 心跳速率顯示，主程式也會傳送狀態是否有改變的信號給 function，若是沒有改變，此 function 就不做任何事，若是有改變，則先洗掉狀態顯示部分的螢幕再重新顯示。

圖十二、顯示¡TOO FAST¡字樣

圖十三、顯示¡TOO SLOW¡字樣

圖十四、顯示¡NORMAL¡字樣

計時器：

簡介：

在計算心跳速率時，抓到兩次 peak 之後，要能夠量出兩者的時間差才有辦法算出心跳速率，因此我們便採用 ARM 板上的 clock 來計時，並利用參考書上所附的範例程式加以改寫，既可在抓到每次 peak 的瞬間，記錄當下的時間。

硬體連結：

直接使用 ARM 版上的內建 clock，因此沒有在另外使用石英震盪器。

軟體編寫：

參考書的範例程式，有一個計時器的程式，然而它是控制每秒讓蜂鳴器叫一聲，我們將他改寫爲一秒鐘紀錄 100 次（也就是我們的時間精確到 0.01 秒），蜂鳴器改成把一個變數一直往上加，如此一來只要在每次抓到 peak 值都去看現在變數和上次差多少便可知道 R-R interval 的間隔時間爲 0.01 秒的多少倍。至於改成一秒鐘紀錄一百次的方法，在於 ARM 板的 clock 爲 110592Hz，而該程式有個參考變數也是 110592，表示只要算到 110592 次就紀錄爲一秒，而我們的方法就是把這個參數改成 1105.92 = 1106，如此它只要算到 1106 就加一，也就是每 0.01 秒就加一。

蜂鳴器：

簡介：

用來提示使用者心跳是否正常。

硬體連結：

直接使用 ARM 版上的內建蜂鳴器。

軟體編寫：

在計算出心跳率的之後，顯示器除了會顯示當時的心跳率，還會顯示心跳是否過快、過慢或正常。我們利用 ARM 板子上內建蜂鳴器的功能，設定板子在心跳過快和過慢的時候都會發出聲音，心跳正常時則不發出聲音，藉此提醒使用者。

EasyARM 版

我們本此實驗的最主要硬體就是 EasyARM 版，由秉華公司代理的 ARM 學習版，主要是一個 ARM 晶片（LPC2200）再加上許多外加模組組合而成，這些外加模組包括：USB、RS232、IDE、網路線等傳輸工具，以及七段顯示器、LED、按鍵等顯示與輸入硬體，另外還有內建八個 10bitADC、記憶體...等等，是個功能還蠻強大的組合版。而我們最主要則是利用他的 ADC 來 sample 從電路傳進來的心電訊號（已經先經過類比放大和濾波），接著對其做運算處理和存入記憶體當中，並由它判讀心電圖、驅使蜂鳴器發出聲音以及使液晶螢幕顯示器顯示。

該 ARM 組合版主要可以用兩類程式來 program，一類是組合語言，另一類是高階語言（C），而由於我們之前完全沒學過組合語言，再加上其範例程式也是用 C 語言寫成的，最後我們採用 C 語言來編寫我們的程式。

至於 EasyARM 裡面的核心 ARM，以下採用 Wikipedia[1]的網頁來介紹 ARM：

「**ARM** 架構（過去稱作進階精簡指令集機器（**Advanced RISC Machine**），更早稱作 **Acorn RISC Machine**）是一個 32 位元精簡指令集（RISC） 中央處理器（processor）架構，其廣泛地使用在許多嵌入式系統（embedded）設計。由於節能的特點，ARM 處理器非常適用於行動通訊領域，符合其主要設計目標爲低耗電的特性。

在今日，ARM 家族佔了所有 32 位元嵌入式處理器 75%的比例，使它成爲占全世界最多數的 32 位元架構之一。ARM 處理器可以在很多消費性電子產品上看到，從可攜式裝置（PDA、行動電話、多媒體播放器、掌上型電玩，和計算機）到電腦週邊設備（硬碟、桌上型路由器）都有。在此家族中衍伸的重要產品還包括 Marvell 的 XScale 架構和德州儀器的 OMAP 系列。」

四、心得與討論

光威：

由於這個專題只有一個月的時間，整體來說時間有點趕，再加上我們這次使用的儀器 ARM 是我們起前從未使用過的，所以到最後有些我們預設的 BONUS 沒有完成有點遺憾。

在實驗的一開始，我們花了大部分的時間在學習如何使用 ARM，由於從未用過的關係，在看 ARM 的溝通上 suffer 了很久，後來終於弄懂了之後，也接近期末所以進度有點緩慢，不過還是慢慢的可以記錄心電訊號和計算心跳速率。

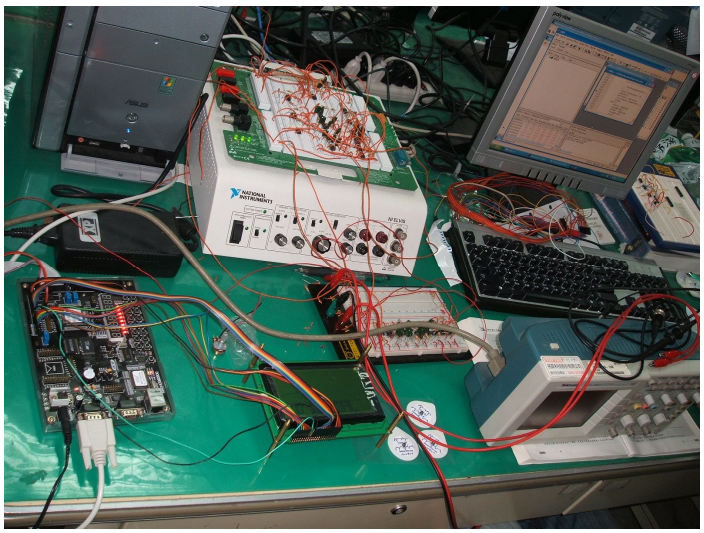
接著我們先嘗試將訊號透過有線的方式傳送到電腦，以期之後能透過無線的方式來傳送，這個部分對我之前有用過 LabVIEW 這個軟體來說還算順利與簡單；再來我們就嘗試想要買傳送，不過在詢問過賣加之後都只有普通的藍芽模組之後，我們就跑去光滑買簡易的 USB 轉藍芽模組，然而對沒有碰過通訊的我們來說，這個部分我們嘗試了許久還是不能成功；後來我們就轉戰了液晶圖形顯示器，在完成了粗部的心電圖、心跳速率和狀態顯示之後，我們就在程式稍微修改，

並讓液晶圖形顯示器顯示的夠精美。後來就差不多期末 demo，就只 implement 到此。

整體來說，我們的可攜式心電圖機是有達到一開始預設的目標，達到簡易的心電記錄、判讀、顯示和發出求救訊號。然而因爲時間的關係，無線傳輸到電腦，並直接啓動 SKYPE 撥號的功能就無法完成，另外，到最後也沒有時間解決 power 的問題，使 ARM 版和 OP 都直接透過可攜式電源，不用外接電源線到市電，這兩個部分是我們覺得比較遺憾的。

俊仁：

這次的實驗是個特別的體驗，因爲我們挑戰了一塊以前從沒碰過的板子，本來想說修過數電實驗以後會比較好上手的，但看來並不是這麼一回事，光是成功讓範例程式在板子上跑起來就花了我們兩個禮拜的時間。好在光威很厲害，摸透了很多的範例實驗，爲我們寫程式省下不少的功夫，因爲只要利用範例實驗所提供的功能，剩下的判讀幾乎就只是跟以前寫程式沒有什麼兩樣。只可惜還是有 bug 出現，而且最後的結果跟我們原先的構想其實大相逕庭，不過也算是走出另一種風格啦¡ 。感謝助教的指導，讓我們漂浮在 debug 的大海裡時還有一塊浮木可以倚靠；感謝另外兩位組員的幫忙，不然我實在是搞不懂這麼多範例實驗與那本簡體字的說明書；順便還有其他一起修課的學弟，多虧他們讓生醫實驗室多了不少的歡笑，也讓我大學的修課生涯劃下一個完美的句點。



圖十五、整體實驗硬體連結圖

五、參考資料

[1] http://butler.cc.tut.fi/~malmivuo/bem/bembook/15/fi/1504.gif

[2] http://butler.cc.tut.fi/~malmivuo/bem/bembook/15/15.htm

[3] http://www.ecglibrary.com/

[4] http://library.med.utah.edu/kw/ecg/image_index/index.html#Diagrams

[5] http://www.fleshandbones.com/readingroom/pdf/505.pdf

[6] http://zh.wikipedia.org/