國立臺灣師範大學
資訊工程研究所碩士論文

指導教授： 黃文吉 博士

以管線化 GHA 電路實作棘波分類之硬體架構設計

Pipelined GHA Hardware Implementation
for Spike Sorting

研究生： 陳 昊 撰

中華民國 一百零一 年 八 月

國立臺灣師範大學資訊工程研究所
博碩士學位論文

以管線化 GHA 電路實作棘波分類之硬體架構設計
Pipelined GHA Hardware Implementation
for Spike Sorting

經考試合格特此證明

審查教授：

指導教授：

系主任：

中華民國 101 年 7 月

中文摘要

本論文針對快速棘波分類設計了一套專用的架構，並於硬體中實現此架構。本論文採用 Generalized Hebbian Algorithm (GHA) 來擷取棘波的特徵值，搭配 Fuzzy C-Means (FCM) 演算法將擷取到的棘波特徵值進行分類。GHA 演算法可高速計算主成分特徵值供後續分群演算法進行運算。同時，利用 FCM 演算法對於初始質心選取好壞不敏感的特性可獲得較佳的分類結果。為了加速執行時間與運算速度，針對 GHA 架構進行了管線化設計，使各單元運算能併行運作，提升產能輸出，而 FCM 採用逐步增量計算權重係數與質量中心點，這可以避免原本需要大量儲存空間儲存權重係數矩陣所造成的空間消耗。因此，本論文所提出的架構同時擁有低資源消耗(area cost)與高輸出產能(throughput)的優點。為了驗證本論文所提出的架構有效性，我們於現場可程式邏輯閘陣列 (Field Programmable Gate Array , FPGA) 中實作出本架構，並於嵌入式 System-On-Programmable-Chip (SOPC) 平台中進行實際效能量測。實驗結果證明針對棘波分類本論文所提出的架構同時具有低判斷錯誤率、低資源消耗與高速計算的優點。

關鍵字：棘波分類、GHA 演算法、管線化 GHA。

誌謝

首先要感謝我的指導教授 黃文吉 教授，在碩士班兩年間對我不辭辛勞地指導，讓我有機會去學習研究與待人處世的態度，當我遇到困難時與我討論並引導我正確方向，使得在這段學習的時間可以獲得很多的成長與體驗，在此獻上最誠摯的謝意。同時也感謝中央大學 張寶基教授、輔仁大學 葉佐任教授以及台灣科技大學 夏至賢教授，撥冗來參加學生的碩士論文口試，且給予論文上的評閱與建議。

此外，我要感謝已畢業的學長姐：銘彥、士彰、斯閔、宗懋、侃翰及坤宏，還有博士班的學長姐們，謝謝你們的教導，同時也感謝這兩年內與我一起奮鬥的同學們：偉豪、浩聲、哲誠、嘉翎以及子昕，另外也感謝可愛的學弟妹建廷、清志、國璿、聖穎、任軒和翰逸，於我研究生活帶來歡樂及學業上的協助與切磋，在此一併致謝。

最後，更要感謝家人與所有關心我的朋友，有了你們的支持、關懷與鼓勵，讓我有勇氣去面對各種的挫折與壓力，並順利地完成學業。謹將此論文成果獻給所有關心我的人，希望大家與我分享這份喜悅與榮耀。

陳　昊　謹誌

台灣師大資訊工程研究所

中華民國一百零一年八月

目錄

附圖目錄

附表目錄

第一章 緒論

1.1 研究背景與動機

棘波序列(Spike Train)是一連串由腦神經元細胞所發出的動作電位(Action Potential)訊號。在人腦內植入探針來探測大腦神經元細胞所發出的訊號稱之為棘波偵測，由於探針所偵測到的訊號為附近神經元細胞之總和，因此棘波分類這項技術被提出來解決將不同神經元所發出的訊號加以分離，但是體內雜訊與周遭神經元細胞所發出的干擾，使得棘波分類並不如想像中容易。

傳統的棘波分類演算法包含了特徵擷取與分類這兩部分[1]。針對棘波分類的應用目前已經有不少解決方法被提出，其中常見的做法為將探針偵測到的訊號透過無線傳輸，由外部設備進行特徵的擷取與訊號的分類。以外部設備進行分析計算固然是方法之一，但是考量腦波訊號的龐大資料量，若將原始資料以無線網路傳輸，所需的頻寬以及造成的延遲會是個不容忽視的議題。傳輸時的品質、雜訊干擾等等也是待解決的議題，所以許多針對腦機介面(Brain Machine Interface, BMI)應用所提出的棘波分類系統多得依靠有線裝置的連結，然而卻大大侷限了使用者移動的便利性以及多元應用的可能性。

1.2 研究目的與方法

將棘波分類演算法硬體化可有效避免傳輸速度與品質的各種議題。硬體化的優點不僅能使得 BMI 有更高的效能與應用，也因為神經元間的訊號傳遞都在幾百個毫秒內，對於需要因應腦神經細胞元各種訊號而做出不同對應的人工義肢、輔助裝置等等，即時計算棘波分類的結果更顯重要。

目前許多新的棘波分類演算法被持續地提出，在硬體化的過程中要能滿足法則的精進與修正是一必要考量，因此本論文選用可程式化邏輯閘陣列(Field Programmable Gate Array, FPGA)做為硬體化的設備。FPGA 的低 NRE(Non-Recurring Engineering) cost 以及可重覆燒錄修改、容易驗證的特點，相較於特殊應用積體電路(Application-specific integrated circuit, ASIC)一經生產便難以修改的缺點，FPGA 提供了實踐棘波分類法則很好的平台與較高的彈性。

傳統的棘波分類演算法包含了特徵擷取與分類這兩步驟，在特徵擷取的方面本論文利用 Generalized Hebbian Algorithm(GHA)[2,3]進行高速運算與資料的降維，避免了傳統主成分分析法則(Principal Component Analysis, PCA)需要計算共變異數矩陣(covariance matrix)的複雜過程。透過 GHA 擷取棘波特徵值將原本複雜龐大的訊號資料維度降低，然後交由後續的分類單元進行分類。

在分類的步驟中採用模糊分群演算法(Fuzzy C-Means, FCM)能快速將棘波特徵值分群，並且省去儲存權重係數矩陣所需要的龐大儲存空間，而以質心逐步調整的方式達到棘波分類的效果。

一般的 GHA 法則分為主成分計算與突觸權重更新兩步驟，常見的硬體化 GHA 先計算訓練資料的主成分，再進行突觸權重的更新，這種循序的運算方式其訴求為共享硬體資源，但計算時間較為緩慢。本論文提出管線化 GHA 之硬體架構，使主成分計算與突觸權重更新可併行運算，達到更快速的計算效率。

由於 FCM 使用模糊理論，每筆初始資料在分群的過程中可以歸屬於不同群集，因此 FCM 有著對於初始質心選擇較不敏感的特性，適合棘波分類法則這種非監督式的應用，同時也能達到高準確率的分類結果。

圖 1.1 為本系統之運作流程，經由探針偵測到的棘波序列，透過 GHA 單元的特徵值擷取與 FCM 單元的資料分群後，可以將原始訊號分類，進而判斷出該棘波訊號是由哪一個腦神經元所發出，使 BMI 能於後續做各類型的延伸應用。

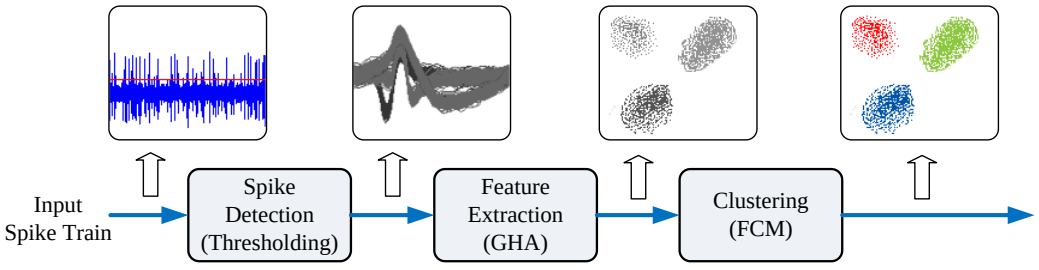


圖 1.1　棘波分類系統運作流程

為了實際驗證本論文所提出的架構能有效實踐棘波分類法則，本文於System-On-Programmable-Chip(SOPC)平台實作所提出的棘波分類系統，藉由SOPC 平台中的軟核處理器控制不同元件間的資料傳遞。

本文同時比較了傳統的 PCA 法則、循序式 GHA 運算單元與本論文提出之管線化 GHA 運算單元所需的運算時間(Computation Time)、功率消耗(Power Consumption)與產能輸出(Throughput)；以及本文所採用之 FCM 分群法則與其他分群法則所需的計算時間；另外也比較了於 FPGA-Based 與 ASIC-Based 上實作棘波分類法則之功率消耗與產能。實驗結果證明本文提出的棘波分類架構，有著低判斷錯誤率、低功率消耗、高產能以及相較其他架構更為快速的即時運算。

第二章 基礎理論與背景介紹

本章節將針對棘波分類之背景進行介紹。

2.1 文獻探討

論文[4]提出一個運作於 SmartDust 平台的植入式棘波分類系統，並以軟體實現了特徵值擷取與分類，但在低時脈運作的情況下處理器很難達到高速運算的要求，而加快運作時脈又將造成功率消耗的提升。

目前有許多以 FPGA 為基礎的硬體架構提出，如架構[5,6,7,8,9,10]等。其中架構[5,6]可執行 online PCA/GHA 特徵擷取。架構[7,8]則以離散小波轉換(Discrete Wavelet Transform , DWT)為基礎進行特徵值擷取。架構[9]實現了一個擷取棘波過零特徵(Zero-Crossing Features)的電路。架構[10]實現以自組織映射網路(Self-Organizing Map , SOM)為基礎的分類運算。這些架構普遍有著無法同時進行特徵擷取與分類運算的缺點，對於應用至棘波分類法則而言並無法提供足夠的產能輸出。

在論文[11,12,13]中所提出的 GHA 架構，其應用多針對紋理圖辨認與臉部辨識等範疇，雖然有些架構可被直接應用於棘波分類領域當中，但部分架構因為資源消耗過高或執行時間過長，因此在硬體化實作上並不適用於棘波分類法則中。架構[11]於 GHA 訓練過程中可同時計算所有輸入向量所對應的輸出結果，雖然提供了較高的產能輸出，但也造成該架構隨著輸入向量維度增加，資源消耗將呈線

性成長。而另一與其相反地架構[12]提出一次只傳送一筆輸入向量，雖然該作法能讓該架構有著較低的資源消耗，但隨著輸入向量維度增加其執行時間亦將呈現線性成長。架構[13]將輸入向量切成幾個較小的區塊一次僅運算一個區塊，雖然該架構有著低資源消耗的優點，但對資料分區塊進行運算，將使得記憶體管理趨於複雜，實屬不必要，且執行時間仍過長。因此本論文提出管線化 GHA 電路架構，捨去架構[13]複雜的記憶體管理，將各單元的運算管線化，透過管線化的方式有效減少延遲，大幅降低運算所需的時間。

對於擷取後的特徵向量將交由 FCM 進行分群以完成棘波分類法則。現行的許多 FCM 架構其應用層面多屬於影像處理[14,15,16]，要將此類架構延伸應用至棘波分類領域當中並不適合。架構[14]的設計僅適用於兩個叢集的分類結果，考量棘波分類領域中的應用，其分類結果經常會需要大於兩個叢集，因此本論文並不使用此架構。架構[15]雖然允許所有訓練向量平行計算其權重係數，達到高產能輸出，但也同時造成了高硬體資源消耗的負擔，要將 FCM 電路與 GHA 電路整合於同一晶片當中，硬體資源的消耗會是值得考量的議題，使用架構[15]將大幅增加其設計的困難度。

因此本論文採用[16]提出的辦法，不同權重係數間所需的共同因子事先使用同一塊電路運算好，並且對於權重係數的調整採逐步計算，這不僅有效的降低了硬體資源消耗，也因為事先運算共同因子的步驟讓後續權重係數的調整得以加快，藉此達到即時運算的效果。

2.2 GHA 演算法

令

$$\mathbf{x}(n) = [x_1(n), \dots, x_m(n)]^T, n = 1, \dots, t \quad (1)$$

$$\mathbf{y}(n) = \left[y_1(n), \dots, y_p(n)\right]^T, n = 1, \dots, t \quad (2)$$

分別為 GHA 第 n 筆輸入與輸出向量。而 m, p 以及 t 為分別為向量維度，主成分個數、輸入向量與輸出向量筆數。而輸出向量 $\mathbf{y}(n)$ 與輸入向量 $\mathbf{x}(n)$ 關係如下：

$$y_j(n) = \sum_{i=1}^{m} w_{ji}(n) x_i(n) \quad (3)$$

其中 $w_{ji}(n)$ 代表第 n 次迭代過程中第 j 個神經元的第 i 筆突觸權重值。

令

$$\mathbf{w}_j(n) = \left[w_{j1}(n), \dots, w_{jm}(n)\right]^T, j = 1, \dots, p \quad (4)$$

為第 j 筆突觸權重向量。每筆突觸權重向量 $\mathbf{w}_j(n)$ 根據赫賓學習法則 (Hebbian learning rule) 進行調整，如下：

$$w_{ji}(n+1) = w_{ji}(n) + \eta \left[y_j(n) x_i(n) - y_j(n) \sum_{k=1}^{j} w_{ki}(n) y_k(n) \right] \quad (5)$$

其中 η 為學習率。經過多次迭代計算調整後 $\mathbf{w}_j(n)$ 將趨近於輸入向量之共變異數矩陣的第 j 筆特徵值 λ_j。為了降低計算時的複雜度，公式(5) 可以改寫成公式(6)：

$$w_{ji}(n+1) = w_{ji}(n) + \eta y_j(n) \left[x_i(n) - \sum_{k=1}^{j} w_{ki}(n) y_k(n) \right] \quad (6)$$

更多關於 GHA 細節的討論可詳見[2,3]。

2.3 FCM 演算法

令 $F = \{\mathbf{f}_1, \ldots, \mathbf{f}_t\}$ 為欲分類的訓練向量所形成的集合，t 為訓練向量的個數。FCM 計算 $\mathbf{v}_i, i = 1, \ldots, c$，並將 F 分成 c 個群集，其中 $\mathbf{v}_i$ 表示第 i 個叢集的質量中心點。FCM 目標為最小化公式(7)：

$$J = \sum_{i=1}^{c} \sum_{n=1}^{t} u_{i,n}^2 \|\mathbf{f}_n - \mathbf{v}_i\|^2 \tag{7}$$

$u_{i,n}$ 為 $\mathbf{f}_n$ 對於第 i 個叢集的權重係數。而 FCM 計算過程中公式(7)被分成兩步驟迭代計算。第一步固定 $\mathbf{v}_1, \ldots, \mathbf{v}_c$，計算最佳權重係數矩陣 $\{u_{i,n}, i = 1, \ldots, c, n = 1, \ldots t\}$ 根據下式：

$$u_{i,n} = \left(\sum_{j=1}^{c} \left(\|\mathbf{f}_n - \mathbf{v}_i\| / \|\mathbf{f}_n - \mathbf{v}_j\| \right)^2 \right)^{-1} \tag{8}$$

接著，固定權重係數矩陣，透過下列式子獲得新的質量中心點 $\mathbf{v}_i$：

$$\mathbf{v}_i = \left(\sum_{n=1}^{t} u_{i,n}^2 \mathbf{f}_n \right) \Big/ \left(\sum_{n=1}^{t} u_{i,n}^2 \right) \tag{9}$$

反覆迭代計算至 J 值收斂為止。

2.4 應用於棘波分類之 GHA 與 FCM

GHA 與 FCM 被用於擷取棘波特徵值與分類當中。而公式(1)中 $\mathbf{x}(n)$ 為第 n 筆棘波。向量維度 m 為每個棘波之取樣點個數。

令

$$\mathbf{w}_j = \left[w_{j1}, \dots, w_{jm}\right]^T, j = 1, \dots, p \tag{10}$$

為 GHA 訓練完成後的突觸權重向量。由這些已訓練完成之突觸權重向量 $\mathbf{w}_j, j = 1, \dots, p$，依據下列式子 GHA 擷取訓練向量 $\mathbf{x}(n)$ 的特徵向量稱為 $\mathbf{f}_n$：

$$\mathbf{f}_n = \left[f_{n,1}, \dots, f_{n,p}\right]^T \tag{11}$$

當

$$f_{n,j} = \sum_{i=1}^{m} w_{ji} x_i(n) \tag{12}$$

為 $\mathbf{f}_n$ 中第 j 筆元素。而獲得的特徵向量集合 $F = \{\mathbf{f}_1, \dots, \mathbf{f}_t\}$ 後續被當作 FCM 的訓練集合。當 FCM 按照 2.2 節所述訓練完畢後所獲得的質量中心點 $\mathbf{v}_1, \dots, \mathbf{v}_c$ 會被用來將棘波訊號分類。而棘波訊號 $\mathbf{x}(n)$ 分類的結果可由下列判斷式獲得：

$$i = \arg\min_{1 \le j \le c} d\left(\mathbf{f}_n, \mathbf{v}_j\right) \tag{13}$$

其中 $d\left(\mathbf{f}_n, \mathbf{v}_j\right)$ 為 $\mathbf{f}_n$ 與 $\mathbf{v}_j$ 的距離平方。

第三章 棘波分類系統架構實現

圖 3.1 為本論文提出的棘波分類法則系統架構。其中 GHA 單元用於計算突觸權重向量 $\mathbf{w}_j, j = 1, \ldots, p$ 以及特徵向量 $\mathbf{f}_n, n = 1, \ldots, t$。接著 FCM 單元會依據 GHA 單元運算後的結果，計算分類過後的質量中心點 $\mathbf{v}_j, j = 1, \ldots, c$。之後可依據這些質量中心點將原始棘波訊號分類，藉此判斷訊號來源來自於哪個腦神經元細胞。

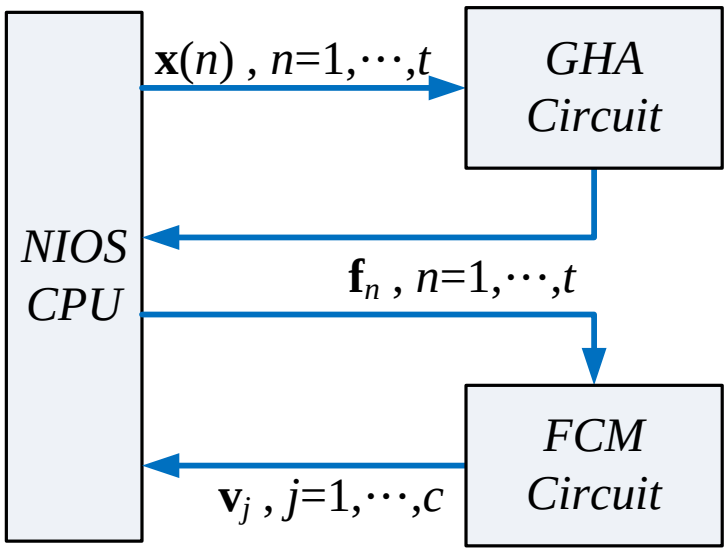
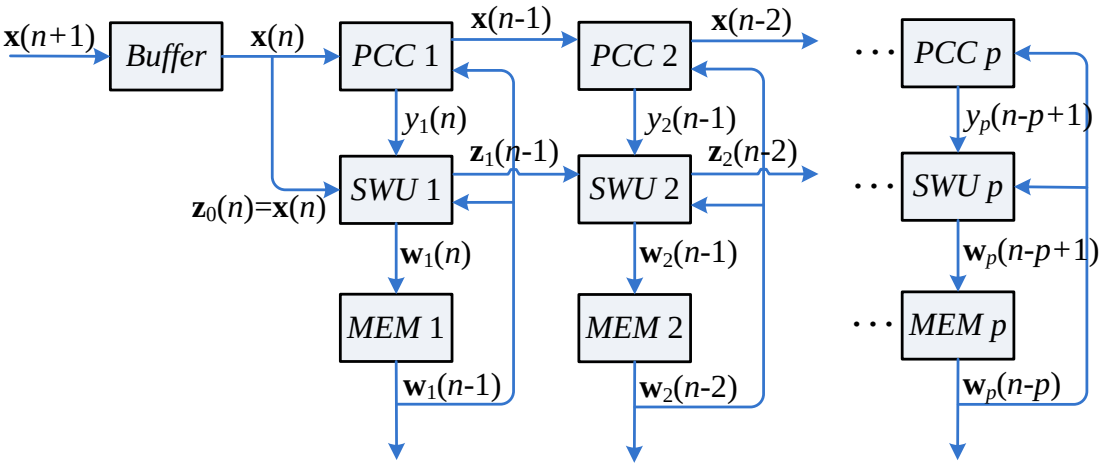


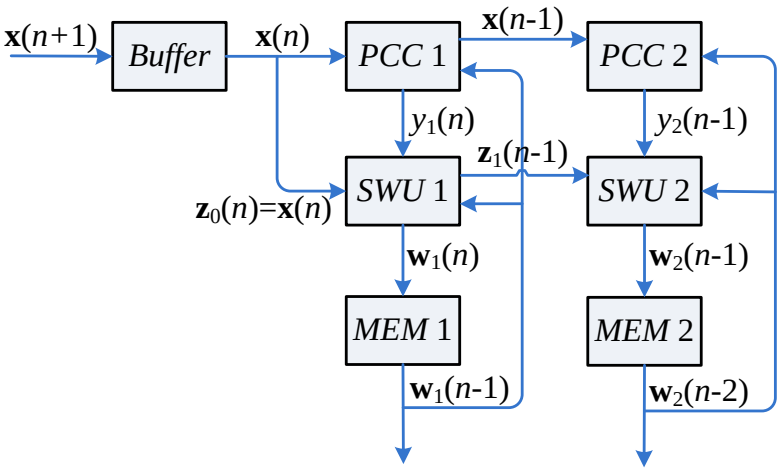
圖 3.1　棘波分類系統架構

3.1 GHA 管線化架構

為了達到管線化的目的，使得主成分計算與突觸權重更新可以併行運算，提高整體電路的產能，本論文特別提出了一套管線化 GHA 架構如圖 3.2，其中可再細分為：記憶體(Memory, MEM)單元、突觸權重更新(Synaptic Weight updating, SWU)單元以及主成分計算(Principle Components Computing, PCC)單元。值得注意的是，本論文針對棘波分類法則之需求，於特徵擷取時僅需計算 2 筆主成分，因此根據此需求設計之架構如圖 3.2(b)；而圖 3.2(a)乃一般化之 GHA 管線化電路，適用於主成分個數為 p 個的情況。後面各小節將針對不同單元做討論，最後說明如何利用此架構達成 GHA 管線化運算之目的。



(a)



(b)

圖 3.2　GHA 管線化系統架構：

(a)一般化之 GHA 管線化電路，主成分個數為p個；

(b)針對棘波分類法則所設計之電路架構，主成分個數為 2 個。

3.1.1 突觸權重更新單元

突觸權重更新單元主要是基於公式(6)來設計。雖然直接採用公式(6)來實作是可行的，但這將會造成大量硬體資源的消耗。一個可以降低硬體資源消耗的辦法為將公式(6)改寫如下：

$$w_{ji}(n+1) = w_{ji}(n) + \eta y_j(n) z_{ji}(n) \tag{14}$$

當

$$z_{ji}(n) = x_i(n) - \sum_{k=1}^{j} w_{ki}(n) y_k(n) \, , j = 1, \dots, p \tag{15}$$

且 $\mathbf{z}_j(n) = \left[z_{j1}(n), \dots, z_{jm}(n)\right]^T$ 。而根據下式 $z_{ji}(n)$ 可由 $z_{(j-1)i}(n)$獲得，如下：

$$z_{ji}(n) = z_{(j-1)i}(n) - w_{ji}(n) y_j(n) \, , j = 2, \dots, p \tag{16}$$

當 $j = 1$ 時，根據公式(15)與公式(16)可得：

$$z_{0i}(n) = x_i(n) \tag{17}$$

因此本論文實作了公式(14)與公式(16)其運算結果等價於公式(6)。

圖 3.3 描述了硬體當中公式(14)與公式(16)的運作情形。如圖所述，SWU Unit 1 單元一次計算一筆新的突觸權重值。以 $\mathbf{z}_0(n)$ 、 $y_1(n)$ 、 $\mathbf{w}_1(n-1)$ 為輸入，計算出第 n 次迭代時的第 1 筆權重向量 $\mathbf{w}_1(n)$ ，同時 SWU Unit 1 單元亦會計算出 $\mathbf{z}_1(n)$ 供 SWU Unit 2 單元後續計算 $\mathbf{w}_2(n)$ 時使用。於此同時 SWU Unit 2 單元接收$\mathbf{z}_1(n-1)$ 、 $y_2(n-1)$ 、 $\mathbf{w}_2(n-2)$為輸入，計算出第 $n-1$ 次迭代時的第 2 筆

權重向量$\mathbf{w}_2(n-1)$，透過這樣的做法突觸權重更新單元將可以管線化，有效地提升運算速度。

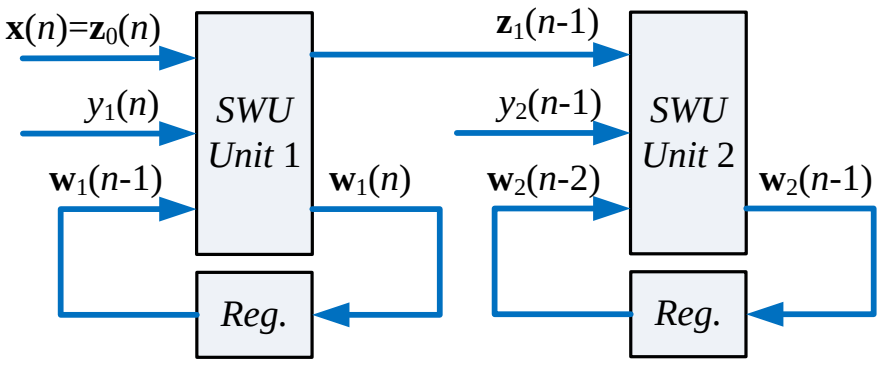


圖 3.3　管線化硬體實現公式(14)與公式(16)

一個 SWU 單元的實現辦法為同時計算 $\mathbf{w}_j(n+1)$ 與 $\mathbf{z}_j(n)$。由於資料維度為 m，因此必須使用 m 個相同的模組來分別獨立計算，每個模組的內容如圖 3.4 所示。

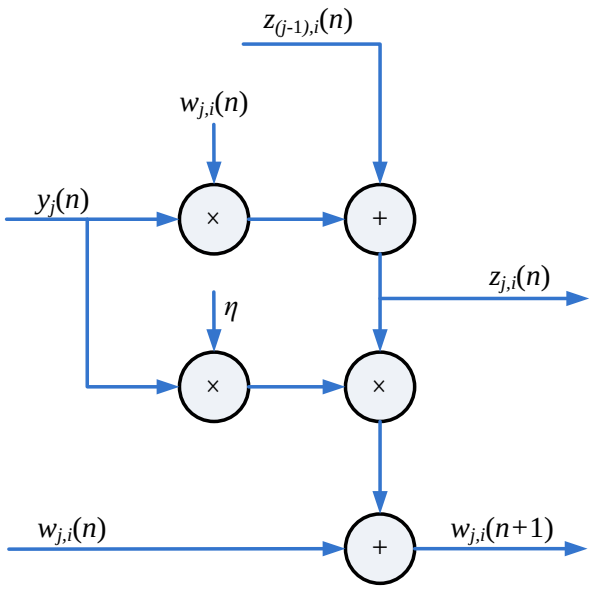


圖 3.4　SWU 單元中每個 module 內部架構

3.1.2 主成分計算單元

主成分計算單元架構是基於公式(3)來設計，此架構中包含了乘法器與平行加法器。圖 3.5 描述了硬體當中公式(3)的運作情形。如圖所述，PCC Unit 1 單元以$\mathbf{x}(n)$為輸入，計算出第n次迭代時的第 1 筆主成分$y_1(n)$，並且輸入的$\mathbf{x}(n)$會通過暫存器保存，留待 PCC Unit 2 單元計算$y_2(n)$。於此同時 PCC Unit 2 單元會接收$\mathbf{x}(n-1)$為輸入，計算出第$n-1$次迭代時的第 2 筆主成分$y_2(n-1)$，透過這樣的資料流將主成分計算單元管線化，可大幅地提升運算速度。

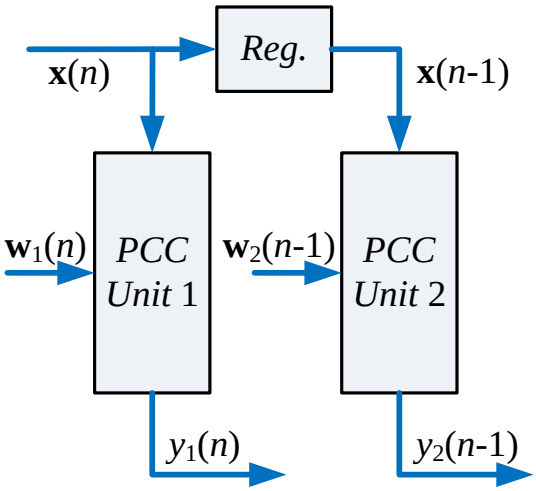


圖 3.5　管線化硬體實現公式(3)

圖 3.6 為 PCC 單元細部的運作情形。由於資料維度為 m，因此必須使用 m 個乘法器，再透過平行加法器算出對應於$\mathbf{x}(n)$之主成分$y_j(n)$。

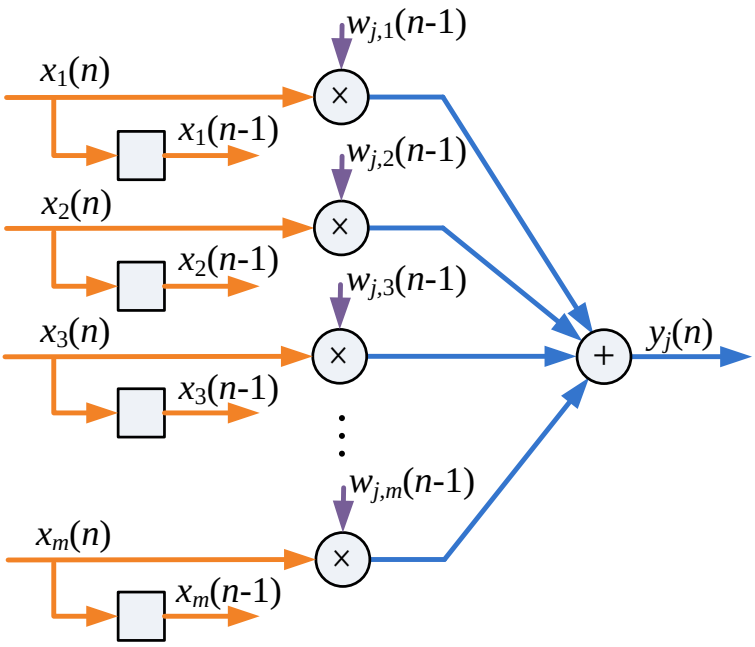


圖 3.6　PCC 單元內部架構

3.1.3 記憶體單元

記憶體單元主要由四個 Buffer 所組成：Buffer A,B,C,D。Buffer A,B,C,D 皆為 shift registers。

Buffer A 保存由主記憶體中獲得的棘波訊號 $\mathbf{x}(n)$。Buffer B 存放 $\mathbf{x}(n-1)$供 PCC 單元與 SWU 單元進行運算。Buffer C 存放突觸權重值向量 $\mathbf{w}_j(n)$。最後，計算出來的特徵值向量 $\mathbf{f}_1, \ldots, \mathbf{f}_t$ 被放置於 Buffer D 供後續 FCM 單元計算使用。

3.1.4 管線化 GHA 電路運作流程

本論文於棘波分類實作中設定每個棘波被取樣為 36 個取樣點，並由這 36 個取樣點中擷取出 2 筆特徵值以滿足棘波特徵擷取的需求[1]。為了符合棘波分類的需求，本文在實作 GHA 單元時定單元維度 $m = 36$ ，主成分個數 $p = 2$ ，意即 GHA 單元中每筆訓練資料$\mathbf{x}(n)$包含 36 筆訓練值$x_1(n), \ldots, x_{36}(n)$。圖 3.7 說明了 GHA 電路當 $m = 36, p = 2$ 時的電路架構。

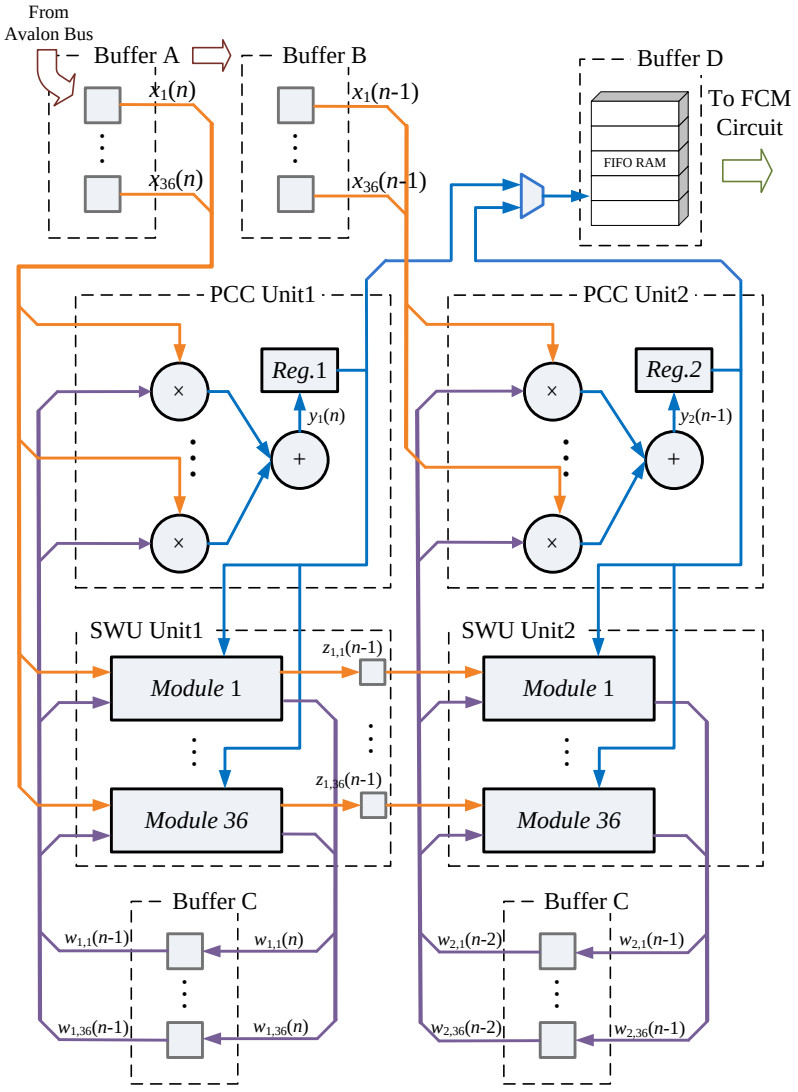


圖 3.7　管線化 GHA 電路架構 (m=36 , p=2)

圖 3.8 描述了此電路的時序圖。以 Cycle 10 至 18 為例，當 Buffer A 從 Avalon Bus 獲取新一筆的訓練資料$\mathbf{x}(n+2)$的同時，會將訓練資料$\mathbf{x}(n+1)$交由 PCC Unit 1 單元計算出主成分$y_1(n+1)$，當 PCC Unit 1 單元計算出主成分$y_1(n+1)$後會送至 SWU Unit 1 單元與$\mathbf{x}(n+1)$以及 $\mathbf{w}_1(n+1)$進行運算，得到新的突觸權重$\mathbf{w}_1(n+2)$以及計算$\mathbf{w}_2(n+1)$所需的$\mathbf{z}_1(n)$，同時會將得到的$\mathbf{w}_1(n+2)$存於 Buffer C 中供下一次的迭代使用。

於此同時 PCC Unit 2 單元接收於 Buffer B 中的前一筆訓練資料$\mathbf{x}(n)$計算出主成分$y_2(n)$，當 PCC Unit 2 單元計算出$y_2(n)$後會送至 SWU Unit 2 單元與 SWU Unit 1 單元提供的$\mathbf{z}_1(n)$以及$\mathbf{w}_2(n)$進行運算，得到新的突觸權重$\mathbf{w}_2(n+1)$，並且存於 Buffer C 中供下一次的迭代使用。

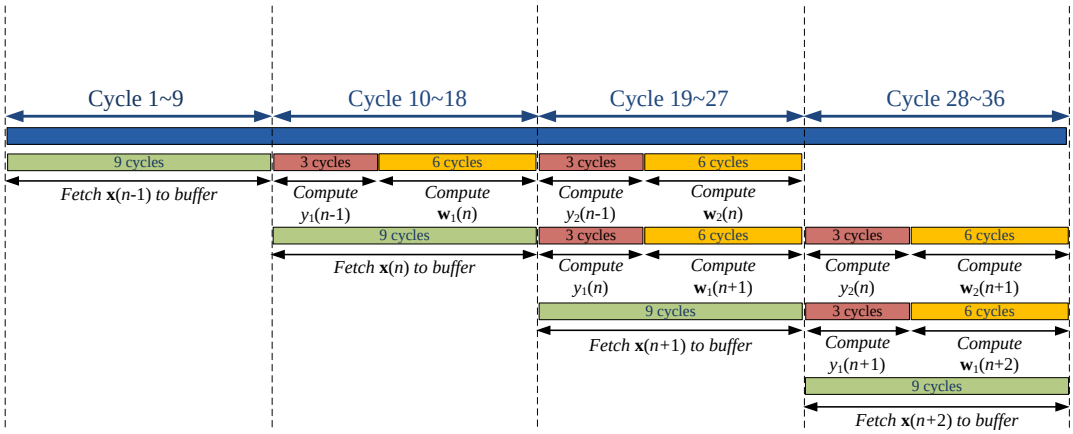


圖 3.8　管線化 GHA 電路架構之時序圖

圖 3.9 描述了以管線化角度來看，同一時間點下電路內部各單元運作情形。舉例當$T = 3$時，Buffer A 正從 Avalon Bus 獲取$\mathbf{x}(3)$，且 Stage 1 中的 PCC Unit 1 與 SWU Unit 1 接收了上一個時間點$T = 2$中 Buffer A 的資料正在計算$y_1(2)$與$\mathbf{w}_1(2)$，而位於 Stage 2 中的 PCC Unit 2 與 SWU Unit 2 也藉由 Stage 1 提供的資料計算$y_2(1)$與$\mathbf{w}_2(1)$。

GHA 單元分為兩個運作模式：訓練模式、特徵擷取模式。訓練模式會依據輸入的來源棘波訊號經由迭代計算後調整突觸權重值；而特徵擷取模式會計算出來源棘波訊號所對應的兩筆主成分值，特徵擷取模式的運作如圖 3.6 PCC 單元之運算流程，而計算出來的特徵向量會被存放於 Buffer D 當中。

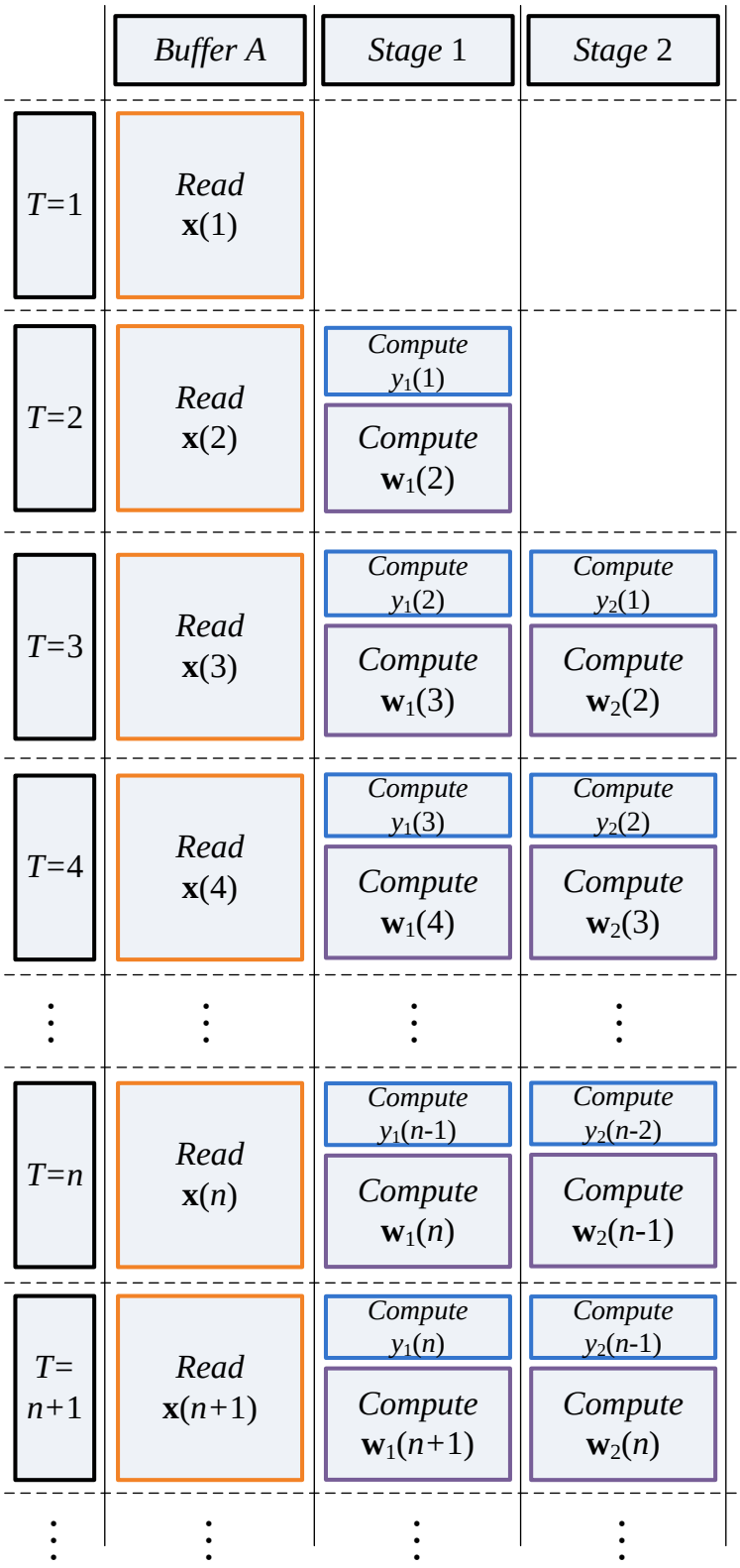


圖 3.9　管線化 GHA 電路架構各單元運作情形

3.2 FCM 架構

FCM 單元架構中包含了六個子單元：pre-computation 單元、membership co-efficients updating 單元、center updating 單元、cost function computation 單元、FCM memory 單元以及 FCM 控制單元。每個單元的運作流程將於後續各小節中說明。

3.2.1 Pre-computation 單元

為了降低後續計算權重係數時的複雜度，pre-computation 單元將事先計算不同權重係數間的共同因子，因此可將公式(8)改寫為：

$$u_{i,n} = \|\mathbf{f}_n - \mathbf{v}_i\|^{-2} {P_n}^{-1} \tag{18}$$

其中

$$P_n = \sum_{j=1}^{c} \left(1/\|\mathbf{f}_n - \mathbf{v}_j\|^2\right) \tag{19}$$

輸入 $\mathbf{f}_n$ 以及質量中心點 $\mathbf{v}_1, \ldots, \mathbf{v}_c$，計算權重係數時皆使用相同的 P_n。因此透過 pre-computation 單元事先計算出 P_n 可降低於後續 membership coefficients updating 單元計算權重係數時的複雜度。根據公式(19) pre-computation 單元可分成兩個階段來計算 P_n，第一階段計算 $\|\mathbf{f}_n - \mathbf{v}_j\|^2$。第二階段則將前一階段所計算出的 $\|\mathbf{f}_n - \mathbf{v}_j\|^2$ 進行倒數運算，並將該值與 $\sum_{j=1}^{i-1} 1/\|\mathbf{f}_n - \mathbf{v}_j\|^2$ 進行累加運算。

3.2.2 Membership coefficients updating 單元

根據公式(18) membership coefficients updating 單元使用 pre-computation 單元計算出來的 P_n 來計算權重係數。給定一筆 $\mathbf{f}_n$，本單元計算出 $u_{i,n}^2$, $\mathrm{i} = 1, \ldots, \mathrm{c}$。本單元電路亦被分作兩階段計算，第一階段計算 $\|\mathbf{f}_n - \mathbf{v}_i\|^2 P_n$ ；第二階段將第一階段計算完成後的結果進行倒數與平方運算獲得 $u_{i,n}^2$ 。

3.2.3 Center updating 單元

本單元逐步更新計算每個叢集的質量中心點，這最大的優點是在計算質量中心點時不需要保存龐大的權重係數矩陣。定義公式(20)，逐步更新計算對於每一筆訓練向量 $\mathbf{f}_k$, $k = 1, \ldots, n$ 之第 i 個叢集的質量中心點。

$$\mathbf{v}_i(n) = \left(\sum_{k=1}^{n} \mathbf{f}_k u_{i,k}^2\right) \Big/ \left(\sum_{k=1}^{n} u_{i,k}^2\right) \tag{20}$$

當 $n = t$ 時， $\mathbf{v}_i(n)$ 即與公式(9) 中目標計算出的質量中心點 $\mathbf{v}_i$ 相同。

Center updating 單元根據公式(20) 來設計，其包含了一個乘法器、一個 cell 陣列與一個除法器。而 cell 陣列中分為兩組 cells：第一組 cell i 保存 $\sum_{k=1}^{n-1} \mathbf{f}_k u_{i,k}^2$ 累加結果；第二組 cell i 則保存 $\sum_{k=1}^{n-1} u_{i,k}^2$ 累加結果。因此，cell 陣列中每一個 cell 內部實際上為累加器。最後，該陣列的輸出結果會交由除法器計算出 $\mathbf{v}_i(n)$ 。

3.2.4 Cost function computation 單元

如同 center updating 單元一般，cost function computation 單元逐步計算出 cost function J。定義 cost function $J(i,k)$ 為：

$$J(i,n)=\sum_{k=1}^{n}\sum_{j=1}^{i}u_{j,z}^{2}\left\|\mathbf{f}_k-\mathbf{v}_j\right\|^2 \tag{21}$$

本電路接收來自 membership coefficients updating 單元的計算結果 $u_{i,n}^2$ 與 $\|\mathbf{f}_n-\mathbf{v}_i\|^2$，並計算出兩者的乘積 $u_{i,n}^2\|\mathbf{f}_n-\mathbf{v}_i\|^2$。接著根據公式(21) 累加計算出 $J(i,n)$。當 $i=c$ 且 $n=t$ 時，$J(i,n)$ 將與公式(7) 中目標計算出的 J 值相同。因此，當所有訓練向量都傳送完畢後本電路單元即會輸出完整的 cost function J 值。

3.2.5 FCM Memory 單元

此單元用來存放 FCM 各叢集的質量中心點，於 on-chip center memory 單元中被分為兩個記憶體區塊(Memory Bank 1、Memory Bank 2)。Memory Bank 1 存放現在 FCM 計算過程中被使用到的 $\mathbf{v}_1, \ldots, \mathbf{v}_c$ 。Memory Bank 2 則存放由 center updating 單元逐步更新計算出的 $\mathbf{v}_1, \ldots, \mathbf{v}_c$ 。

僅只有存放於 Memory Bank 1 的質量中心點會被 pre-computation 單元與 membership coefficients updating 單元所使用，用來計算出權重係數。而由 center updating 單元所計算出新的質量中心點則會暫存於 Memory Bank 2 中。

特別注意的是，除非所有輸入的特徵向量 $\mathbf{f}_n, n = 1, \ldots t$ 都已計算完畢，否則 Memory Bank 2 中所存放被更新後的質量中心點將不會更新取代至 Memory Bank 1 中。

3.2.6 FCM 運作流程

為了能夠簡化說明 FCM，圖 3.10 為針對兩個叢集($c = 2$) 所設計的 FCM 電路架構圖，該電路可將來源棘波所擷取到的特徵向量區分為兩個叢集。針對未來可能需要支援更多的叢集數可輕易地透過增加 center updating 單元中 cell 數以及 Memory 單元中 memory banks 大小來進行延伸。

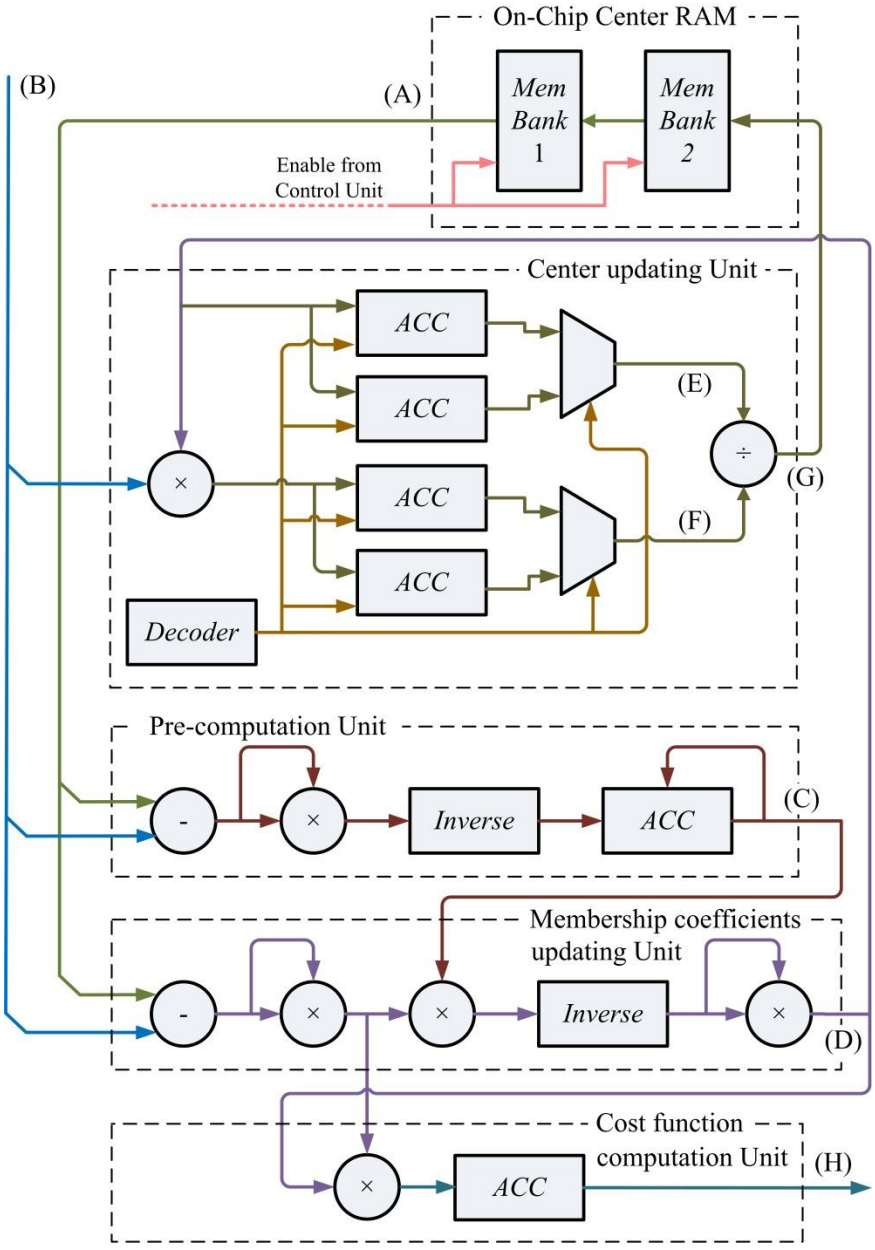


圖 3.10　FCM 電路架構($c = 2$)

針對叢集數 $c = 2$ 時，FCM 電路運作可被分為四個階段來進行。圖 3.10 說明了 controller 與 FCM 電路所對應的狀態圖。表 3.1 說明了本電路於每個階段不同 Node 間所產生的對應訊號。表 3.1 中 8 個 Nodes (Nodes A-H) 所對應的位置標示於圖 3.10 中。為了方便說明，表 3.1 僅舉例說明前兩筆特徵向量 $(\mathbf{f}_1, \mathbf{f}_2)$ 的運作流程，其餘特徵向量的運作模式亦比照此方式運行。

表 3.1　針對前兩筆特徵向量 $(\mathbf{f}_1, \mathbf{f}_2)$ FCM 單元運作流程

State	**Node Name**							
	A	B	C	D	E	F	G	H
State 1	$\mathbf{v}_1$	$\mathbf{f}_1$	×	×	×	×	×	×
State 2	$\mathbf{v}_2$	$\mathbf{f}_1$	P_1	×	×	×	×	×
State 3	$\mathbf{v}_1$	$\mathbf{f}_1$	P_1	$u_{1,1}^2$	×	×	×	×
State 4	$\mathbf{v}_2$	$\mathbf{f}_1$	P_1	$u_{2,1}^2$	$u_{1,1}^2$	$u_{1,1}^2\mathbf{f}_1$	$\mathbf{v}_1(1)$	$J(1,1)$
State 1	$\mathbf{v}_1$	$\mathbf{f}_2$	×	×	$u_{2,1}^2$	$u_{2,1}^1\mathbf{f}_1$	$\mathbf{v}_2(1)$	$J(2,1)$
State 2	$\mathbf{v}_2$	$\mathbf{f}_2$	P_2	×	×	×	×	$J(2,1)$
State 3	$\mathbf{v}_1$	$\mathbf{f}_2$	P_2	$u_{1,2}^2$	×	×	×	$J(2,1)$
State 4	$\mathbf{v}_2$	$\mathbf{f}_2$	P_2	$u_{2,2}^2$	$\sum_{k=1}^{2} u_{1,k}^2$	$\sum_{k=1}^{2} u_{i,k}^2 \mathbf{f}_k$	$\mathbf{v}_1(2)$	$J(1,2)$

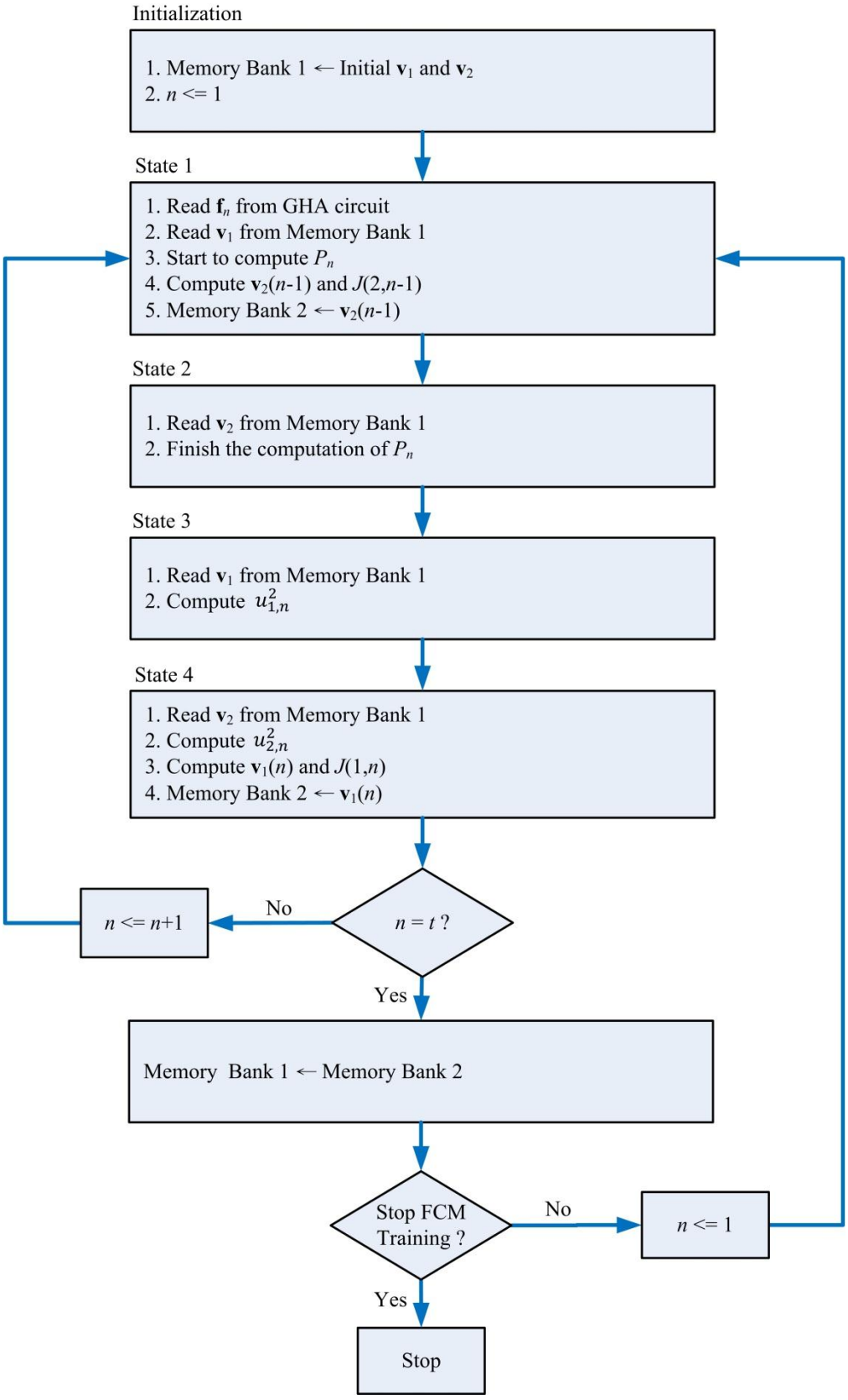


圖 3.11 FCM 單元運作流程圖

FCM 電路中，每筆由 GHA 傳送來的特徵向量 $\mathbf{f}_n$ 將於 Node B 處停留 4 個 stages。而 Node A 為每筆從 Memory 單元獲得的質量中心點 $\mathbf{v}_i$ 。

表 3.1 中給定一筆特徵向量 $\mathbf{f}_n$ ，state 1 與 state 2 經由 pre-computation 單元計算出 P_1 (Node C 處)，接著 state 3 與 state 4 membership coefficients updating 單元分別計算出權重係數 $u_{1,1}^2, u_{2,1}^2$ ，並於圖 3.10 中 Node D 處輸出。

此外根據由 state 3 中獲得的 $u_{1,1}^2$ ，center updating 單元與 cost function computation 單元將會分別於 Node G 與 H 逐步計算出新的質量中心點 $\mathbf{v}_1(1)$ 與 $J(1,1)$ 。最後 state 4 中獲得的 $u_{2,1}^2$ 將會於計算 $\mathbf{f}_2$ 時之 state 1 計算出 $\mathbf{v}_2(1)$ 與 $J(2,1)$。此外，P_2 亦會於該 state 中進行計算。而後續針對 $\mathbf{f}_2$ 計算時 state 2-4 之運作流程與計算 $\mathbf{f}_1$ 時相同。詳細的運作模式流程圖可見圖 3.11。

3.3 GHA 與 FCM 電路之整合

本論文在實踐棘波分類法則時利用 NIOS CPU 來負責操控 GHA 與 FCM 電路間的資料傳輸，如圖 3.1。棘波訊號 $\mathbf{x}(n), n = 1, \dots, t$ 被傳送至 GHA 電路，訓練出突觸權重向量 $\mathbf{w}_1, \dots, \mathbf{w}_p$ ，待突觸權重向量訓練完畢後相同的棘波訊號 $\mathbf{x}(n), n = 1, \dots, t$ 再度被傳送至 GHA 電路當中計算出對應的特徵向量 $\mathbf{f}_n, n = 1, \dots, t$ 。

當 GHA 電路計算完所有特徵向量 $\mathbf{f}_n, n = 1, \dots, t$ 並存放於 Buffer D 後，NIOS CPU 以讀寫的方式將每一筆特徵向量送至 FCM 電路，此時 FCM 電路藉由一次一筆的特徵向量訓練調整出新的質量中心點 $\mathbf{v}_1, \dots, \mathbf{v}_c$ 。持續重複傳送相同的特徵向量 $\mathbf{f}_n, n = 1, \dots, t$ 直到 FCM 訓練完畢，最後 FCM 單元訓練完成後所獲得的質量中心點 $\mathbf{v}_1, \dots, \mathbf{v}_c$ 將可被用來分類棘波訊號。

3.4 FPGA-Based 棘波分類系統

本論文提出的架構為建置於 SOPC 系統中使用者客製化邏輯電路(Custom User Logic)，圖 3.12 為本系統與 SOPC 平台之關係圖。系統中包含了 NIOS CPU[18]，DMA controller 與 on-chip RAM。利用 DMA 將所有存放於 on-chip RAM 中的棘波訊號經由 Avalon bus 傳送至客製化電路當中，這可使得本系統對於記憶體存取的 overhead 最小化。軟核 CPU 僅執行簡單的指令供 GHA 與 FCM 訓練時使用，這些指令僅負責 SOPC 平台中不同元件間的溝通。NIOS CPU 本身並不參與 GHA 與 FCM 的計算。當所有訓練向量傳送完畢後，軟核 CPU 從客製化電路中接收訓練結果 $\mathbf{w}_1, \ldots, \mathbf{w}_p$ 與 $\mathbf{v}_1, \ldots, \mathbf{v}_c$。

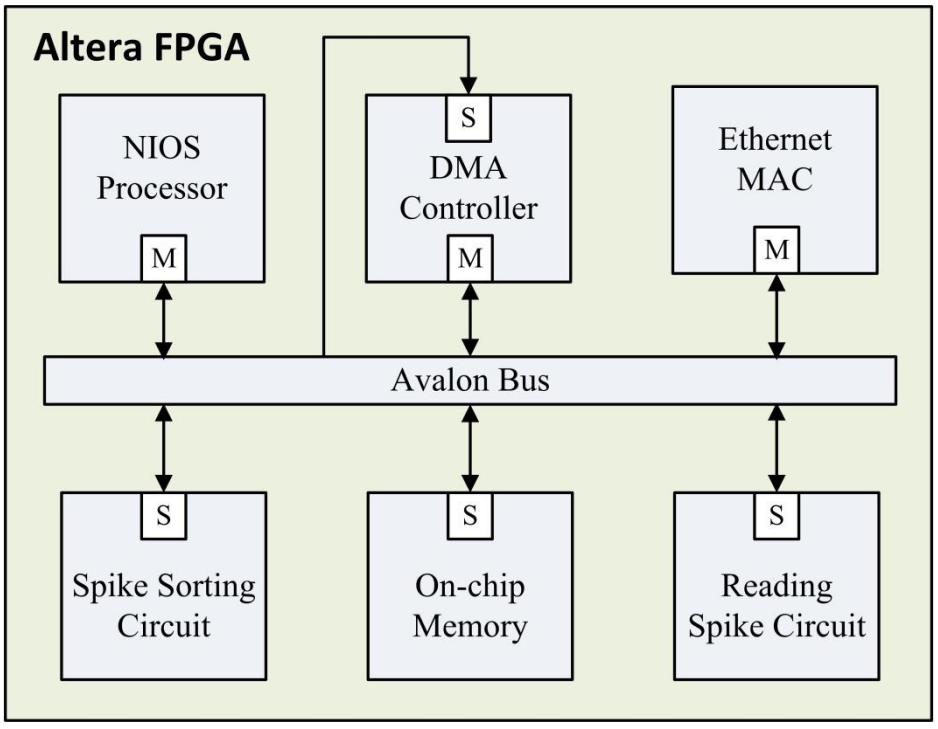


圖 3.12 FPGA-Based 棘波分類系統

第四章 實驗數據與效能比較

4.1 開發平台與實驗環境

本論文提出的棘波分類硬體架構實現於 Altera Cyclone IV GX EP4CGX150DF31 FPGA 開發板中，如圖 4.1 所示。Cyclone 系列為 Altera 公司針對低階市場所推出 FPGA 晶片，其特點為低功率消耗以及低開發成本，非常符合棘波分類晶片的需求。同時，拜製程進步所賜 Cyclone IV FPGA 採用 65 奈米製程，使得功率消耗較以往 FPGA 降低約 30%。

圖 4.1　FPGA 開發板

而利用 FPGA 可重複修改電路設計與快速驗證的特性，使得棘波分類晶片可以快速且容易的更新演算法則，同時低開發成本也使 FPGA 變得更具優勢。表 4.1 整理了 Cyclone IV GX 開發板中所提供的相關硬體資源。也因為其提供了大量的

硬體資源以及與周邊元件豐富的溝通介面，使得使用者得以輕鬆的將許多複雜的演算法則整合至開發板當中，成為一套完整且獨立的嵌入式系統。

表 4.1 軟硬體實驗平台規格整理

Hardware FPGA Development Board	**Process**	**LEs**	**Embedded Memory (Kbits)**	**18-bit x 18-bit Multipliers**	**User I/O**	**Package Type**
	65 nm	149760	6480	360	475	896-pin FBGA
Software Intel i5-2467M	**Process**	**# of Cores**	**# of Threads**	**Clock Speed**	**L3 Cache**	**Max TDP**
	32 nm	2	4	1.6 GHz	3 MB	17 W

本論文使用 Altera Quartus II 10.1 版本搭配 Verilog 硬體描述語言進行設計與開發。並且透過 Altera SOPC Builder 建立客製化 SOPC 系統，於 SOPC 中加入軟核 CPU，DMA Controller，on-chip memory 以及自行開發撰寫的棘波分類系統。棘波分類系統於 SOPC 系統中被視為客製化邏輯電路，透過 Altera 所提供的匯流排 (Avalon bus) 與其它周邊元件溝通。同時，使用 Altera NIOS II IDE 撰寫 C 語言程式碼，並利用其所提供的函式庫、驅動程式與 FPGA 開發板溝通，加速 SOPC 系統的開發，SOPC 系統開發流程圖可見圖 4.2。

軟體部分本論文使用 C 語言實現 GHA 與 FCM 演算法，並運作於 CPU Intel i5-2467M 當中。 i5-2467M 為 Intel 所開發低電壓版本的處理器，採用雙核心架構，主打低功率消耗，這部分與棘波分類系統的基本需求雷同。因此，本論文將其效能與本論文架構納入比較。詳細 Intel i5-2467M 規格內容可見表 4.1。

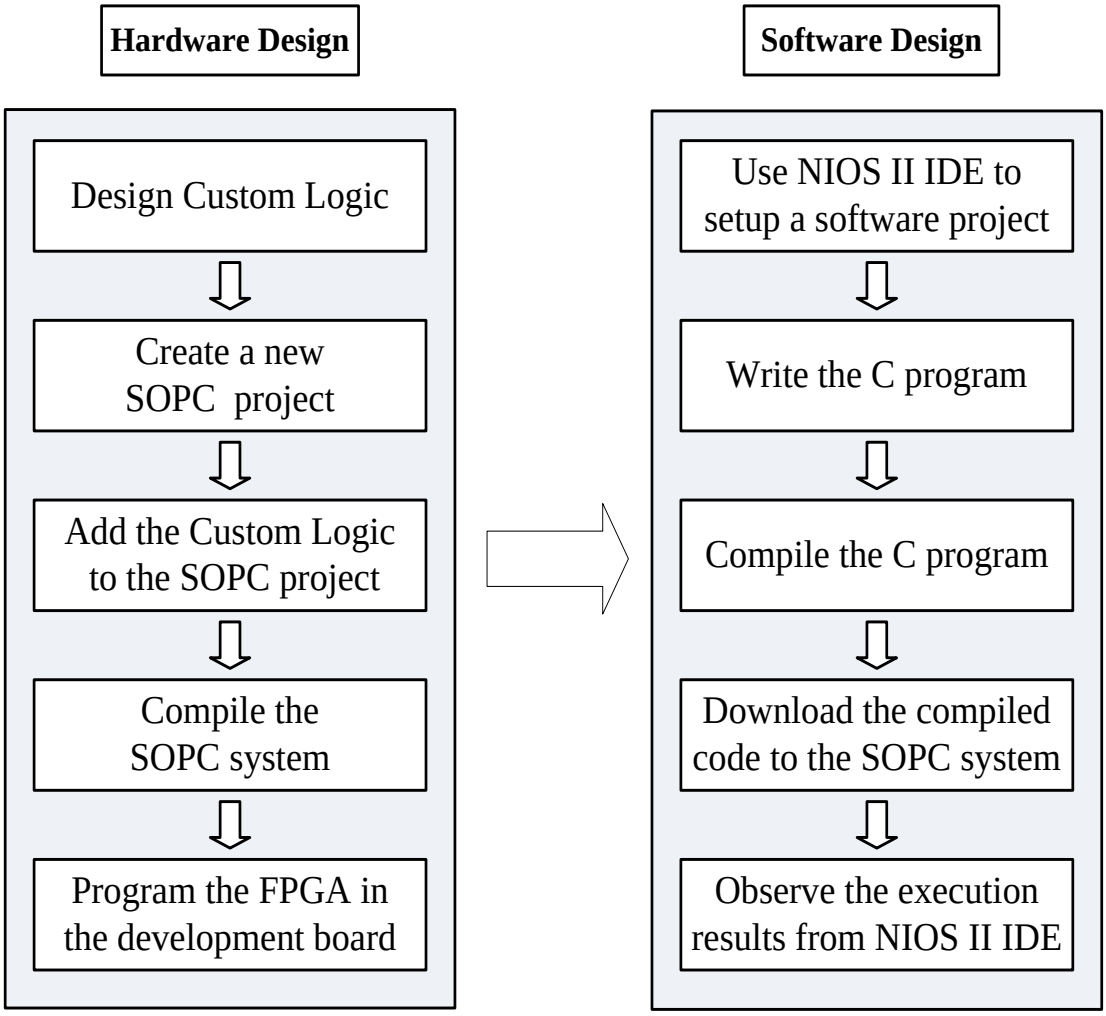


圖 4.2　SOPC 系統開發流程

4.2 實驗數據呈現與討論

為了量測本論文針對棘波分類所提出的架構之實際效能，本論文採用由[17]所開發用來產生神經元訊號的模擬器。此模擬器提供了大量的參數模擬棘波活動情形，也因為其可針對許多環境變數進行設定與調整，使得實驗結果能更進一步的進行各種分析與比較。本實驗數據利用該模擬器產生出於不同信噪比(signal-to-noise ,SNR)下之棘波序列，並將取樣率設定為 13500 取樣點/秒，每個棘波訊號長度為 2.67 毫秒，所以每個棘波訊號有 36 個取樣點。根據上述設定，本論文於電路設計當中定 GHA 訓練向量維度 $m = 36$ ，主成分個數 $p = 2$ 。

電路中 GHA 與 FCM 的運算資料皆採用定點數格式。在有限精度的情況下不同精度的定點數對於分類結果影響的比較可見表 4.2。表 4.2 中採用的棘波序列為 3 個神經元所產生的訊號總和，偵測到的棘波數目為 800，並分別比較於信噪比為 SNR=10,SNR=20,SNR=100 時本電路的分類正確率(classification correct rate ,CCR)，分類正確率定義為所有棘波訊號中被正確分類的棘波訊號筆數。從表 4.2 中可清楚看出 GHA 於 16-bits 的定點數格式與雙倍精確度浮點數格式有著相似的分類正確率；相比之下，8-bits 定點數格式對於 FCM 則有著足夠的精準度來進行分類運算。根據實驗比較結果，本論文對於 GHA 與 FCM 兩塊電路分別採用 16-bits 與 8-bits 定點數運算。

表 4.2　不同精度之定點數影響分類結果正確率

Cluster	**Precision**		**Interference**		
	GHA	FCM	SNR=10	SNR=20	SNR=100
c=2	Floating	Floating	99.5%	99.5%	99.5%
	16 bits	Floating	99.5%	99.5%	99.5%
	16 bits	8 bits	99.5%	99.5%	99.5%
c=3	Floating	Floating	96.2%	96.3%	96.3%
	16 bits	Floating	96.2%	96.3%	96.3%
	16 bits	8 bits	96.2%	96.2%	96.3%
c=4	Floating	Floating	92.3%	94.1%	94.1%
	16 bits	Floating	92.3%	94%	94.1%
	16 bits	8 bits	92.3%	94%	94%

為了更進一步說明本論文架構於不同條件下 CCR 的表現。圖 4.3 說明了於不同 SNR 情況下對於來源棘波訊號的影響。根據圖 4.3 可以清楚的看到當 SNR 小於 10 後，因為大量雜訊的干擾使得分類變得更加困難。圖 4.4 說明了 GHA 特徵向量散佈情形以及 SNR=10 時有限精度 16 bits GHA 與 8 bits FCM 的分類結果。根據表 4.2 與圖 4.4、圖 4.5 與圖 4.6 可以看出本論文所提出的 GHA 與 FCM 對於大量雜訊干擾時仍可以正確地將棘波訊號給分類出來。

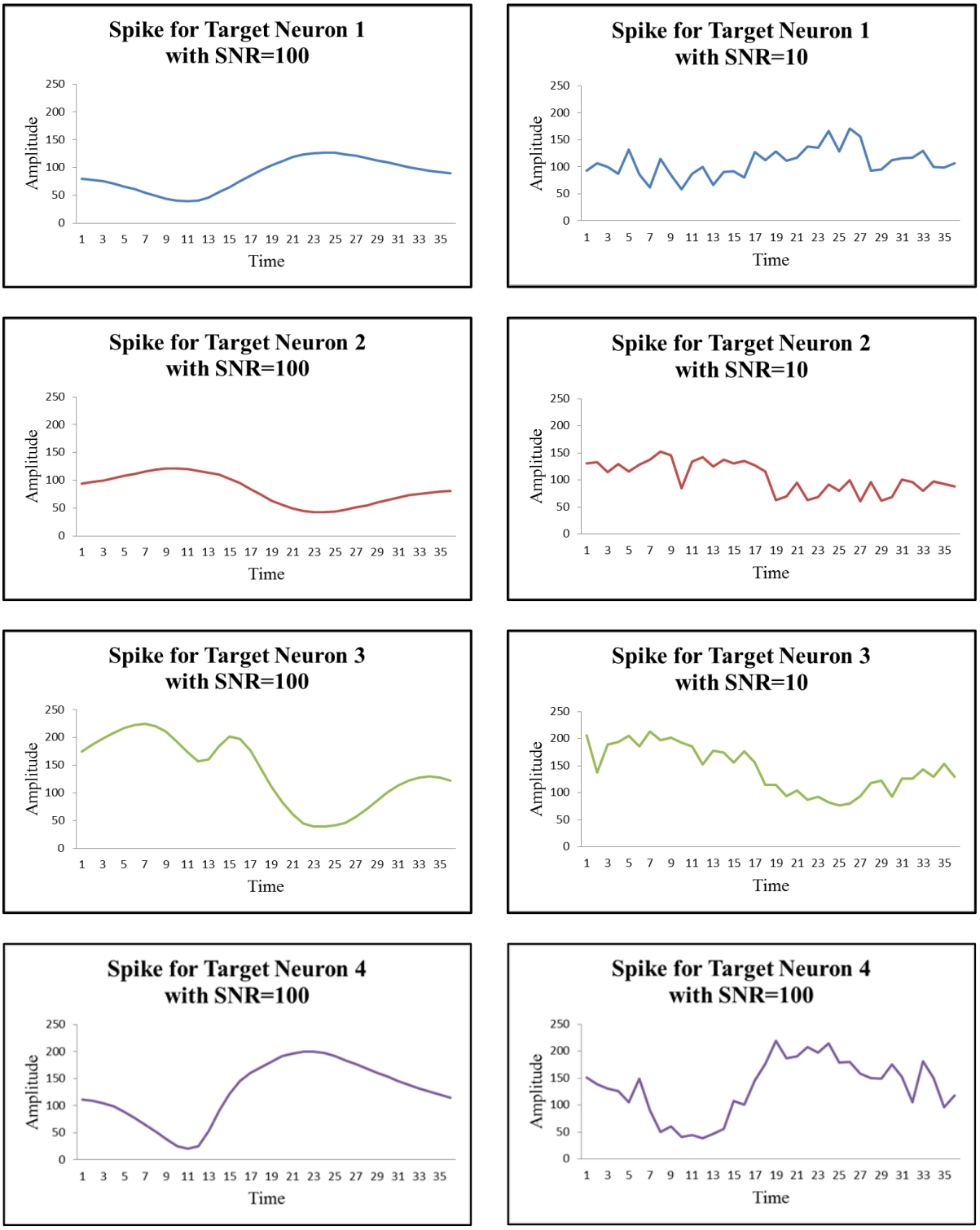
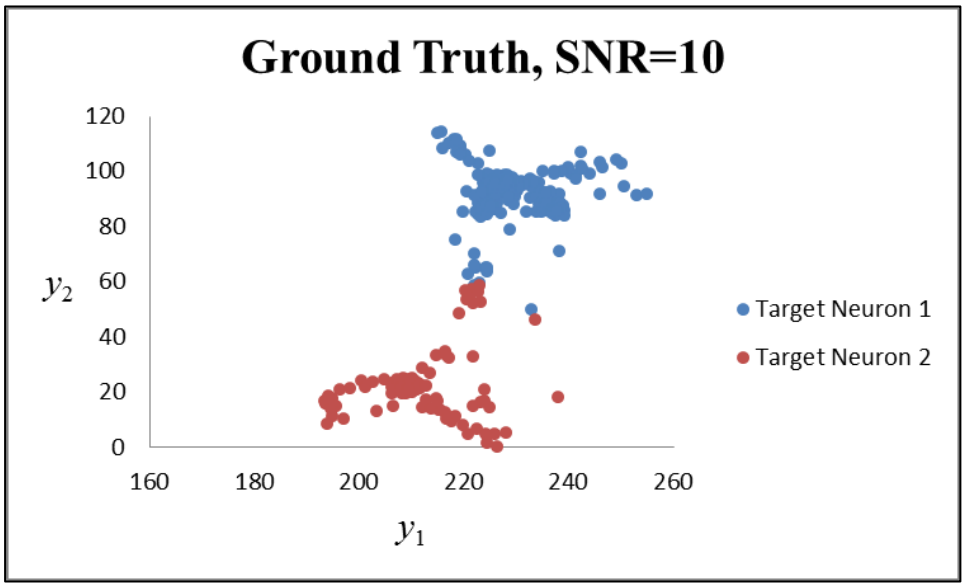
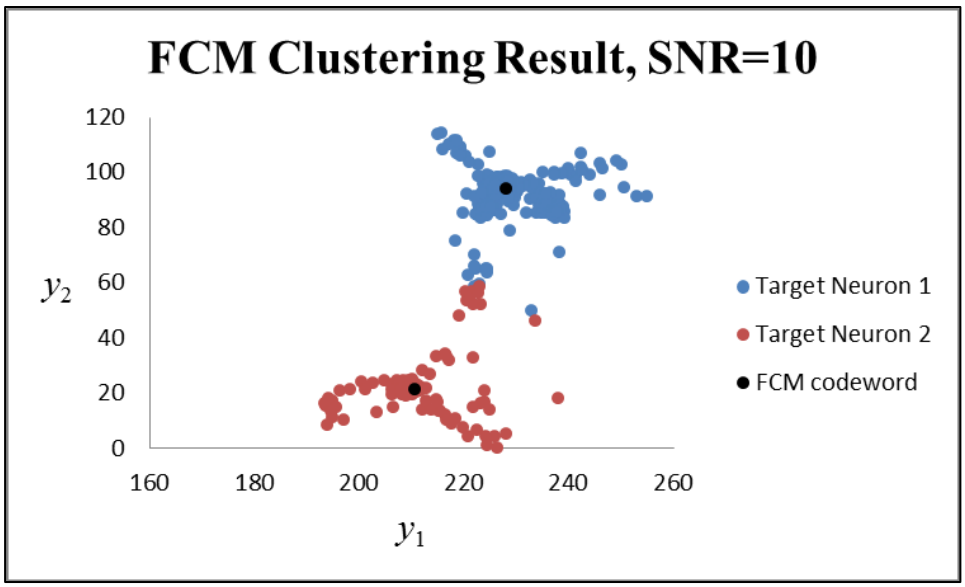


圖 4.3　於不同信噪比下三個不同神經元之棘波訊號比較



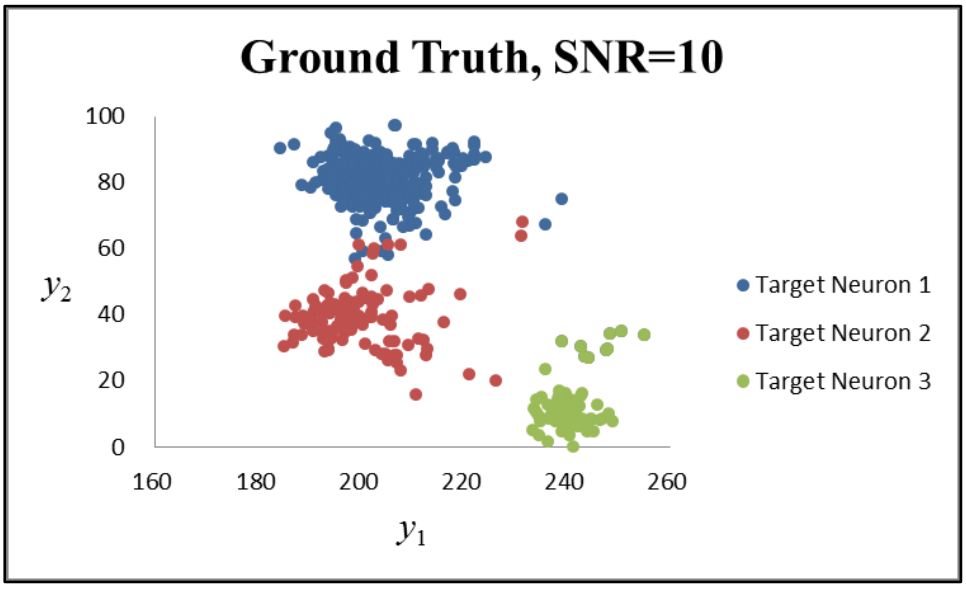
(a)



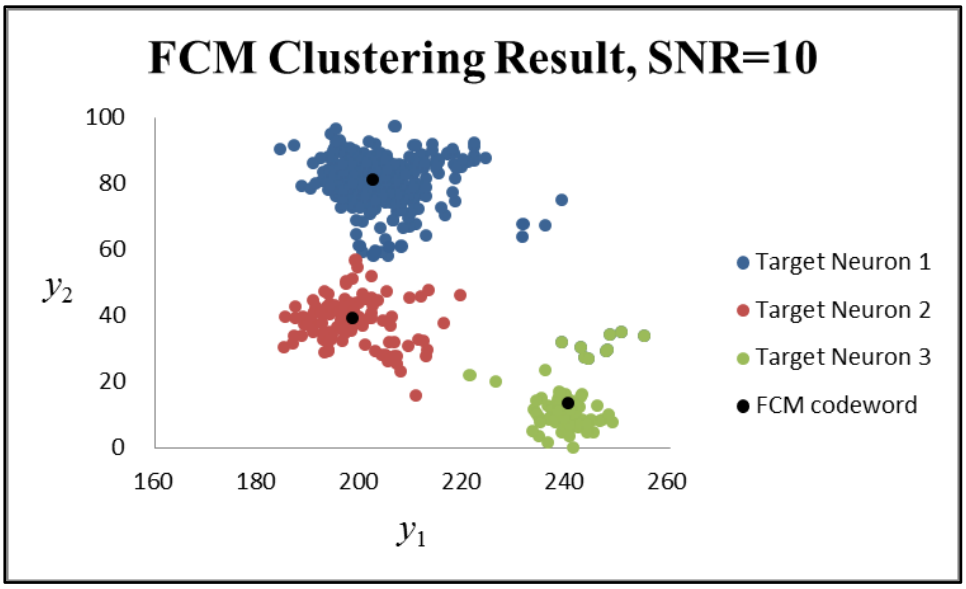
(b)

圖 4.4 棘波主成分值y_1, y_2投影至特徵平面之散佈圖($c = 2$)

(a) 棘波正確分類結果 (b) FCM 分類結果



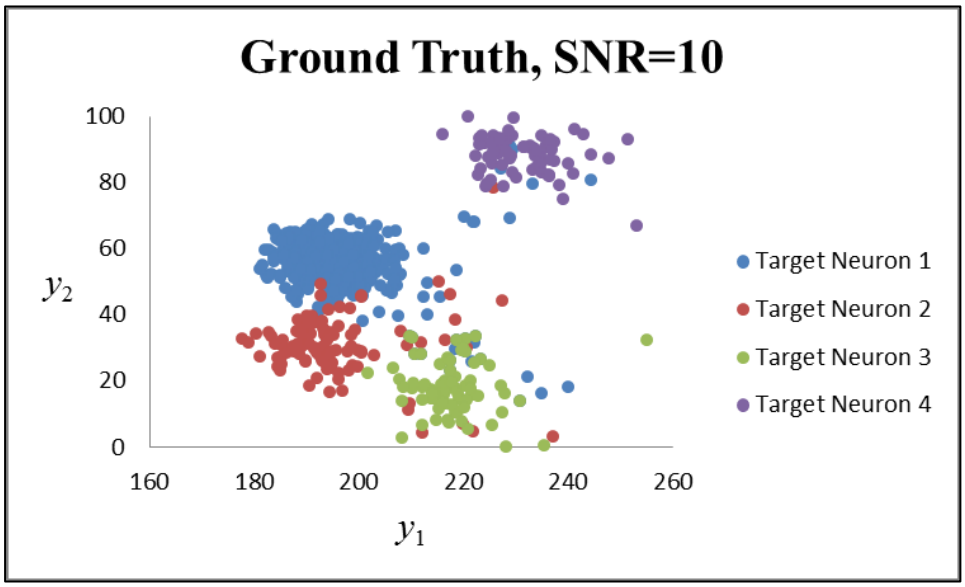
(a)



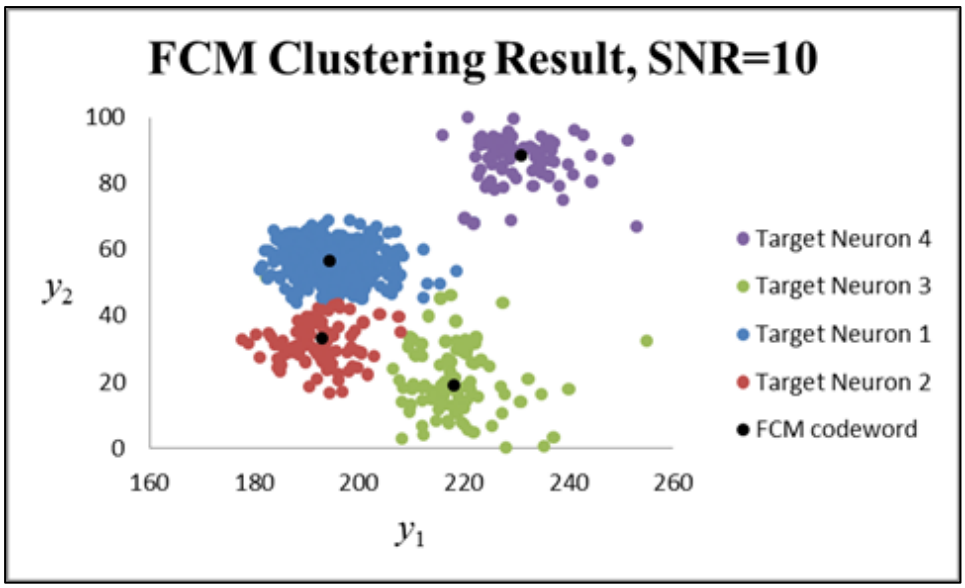
(b)

圖 4.5　棘波主成分值y_1, y_2投影至特徵平面之散佈圖($c = 3$)

(a) 棘波正確分類結果 (b) FCM 分類結果



(a)



(b)

圖 4.6　棘波主成分值y_1, y_2投影至特徵平面之散佈圖($c = 4$)

(a) 棘波正確分類結果 (b) FCM 分類結果

此外，本棘波分類電路不僅擁有高分類正確率，同時亦保有低面積複雜度，表 4.3 說明了本電路的面積複雜度。因為加法器、乘法器與暫存器為 GHA 與 FCM 架構中的基本區塊，因此表 4.3 分成了三個類別：加法器數量、乘法器數量、暫存器數量來進行比較。從表 4.3 中可以看出加法器與乘法器數量將與主成分個數 p 以及向量維度 m 呈線性成長關係，與叢集個數 c 以及訓練向量個數 t 無關。

表 4.3　棘波分類電路之面積複雜度

	GHA	**FCM**	**Total**
Adders	$O(pm)$	$O(p)$	$O(p(m+1))$
Multipliers	$O(pm)$	$O(p)$	$O(p(m+1))$
Registers	$O(pm+pt)$	$O(pc)$	$O(p(m+t+c))$

因為棘波分類電路必須存放來源棘波訊號、突觸權重向量以及特徵向量。因此，暫存器數量將與 p, m, c, t 呈線性成長。表 4.4 說明了針對棘波分類應用 $p = 2, m = 36, c = 3$ 與 $t = 800$ 時的硬體資源消耗。本實驗設計於 Altera Quartus II with SOPC Builder[19]與 NIOS II IDE 平台中。所使用的 FPGA 開發板為 Altera Cyclone IV EP4CGX150DF31C7。表 4.4 顯示了三種不同的資源比較：Logic Elements (LEs)，embedded memory bits 以及 embedded multipliers。

表 4.4 棘波分類系統硬體資源消耗

	GHA Circuit	FCM Circuit	Entire SOPC
Logic Elements	9144/149760 (6.10%)	4441/149760 (2.97%)	24175/149760 (16.14%)
Embedded Multiplier	432/720 (60.00%)	23/720 (3.19%)	459/720 (63.75%)
Memory bits	63488/6635520 (0.96%)	113584/6635520 (1.71%)	858438/6635520 (12.94%)

其中 GHA 與 FCM 架構中 LEs 被用來實現加法器、乘法器與暫存器。而 LEs、embedded multiplier 與 memory bits 被用來實現 SOPC 系統中的 NIOS CPU。另外，GHA 與 FCM 電路中的乘法器使用 embedded multiplier 來實現。從表 4.4 中可以看出單獨本電路架構所消耗的硬體資源，本電路架構加入 SOPC 後硬體資源消耗僅微幅成長，詳細各元件之資源消耗比較可見表 4.4。

本論文所提出的 GHA 與 FCM 電路與現存既有的相關電路設計比較後有著高計算速度(Computation Time)與低功率消耗(Power Consumption)的優點。由於各種架構所採用之演算法與實作平台的差異，使得要直接地將各類架構拿來比較是有其困難的。而表 4.5 中本論文所提出的 GHA 架構與實現於 SmartDust 中的架構[4]比較，有著明顯較低的 Computation Time，並有著相近的 Power Consumption。而

本論文所提出的 GHA 電路架構與軟體演算法實現於 Intel i5 CPU 中比較後，也有著低 Computation Time 與低 Power Consumption 的優勢。

表 4.5　GHA 電路與 Software-Based 之特徵擷取法則比較

	Proposed GHA Architecture	**PCA Software Implementation [4]**	**PCA Software Implementation [4]**	**GHA Software Implementation**
Number of Spikes	800	800	800	800
Clock Rate	50 MHz	13 MHz	416 MHz	1.60 GHz
Computation Time (ms)	2.914	214521.45	8333	36.41805
Power Consumption (mW)	209.37	92.4092	943.71	1116
Platform	FPGA Cyclone IV EP4CGX150	SmartDust iMote2	SmartDust iMote2	General Purpose CPU Intel i5-2467M

除了表 4.5 與軟體設計比較之外，本論文於表 4.6 亦針對輸出產能(Throughput)與功率消耗(Power Consumption)與 ASIC-based 以及 FPGA-based 的硬體設計作比較。ASIC-based 架構[5]基於 PCA 演算法設計。該表中 Throughput 定義為每分鐘可以訓練的 channel 數目。由於本論文運作於較高的時脈底下，於表 4.5 中可以看出本論文所提出的架構與 ASIC-based Implement 比較後有著較高的

Throughput。為了於同一基準下比較，另外定義 Normalized Throughput 為每 MHz 下每分鐘可訓練的 channel 數目。針對 Normalized Throughput 比較後可以發現本論文所提出的架構依舊有著較高的 Normalized Throughput。此外，ASIC-based Implement 使用 PCA 演算法必須先計算訓練資料的共變異數矩陣 (Covariance Matrix)，然而其計算時間並未被包含進比較數據當中。相較之下對 GHA 而言並不需要去計算共變異數矩陣，使得 GHA 對於特徵擷取有著較佳的效率。

表 4.6 GHA 電路與 Hardware-Based 之特徵擷取法則比較

	Proposed GHA Architecture	**FPGA-Based GHA [13]**	**ASIC-Based PCA [5]**	**FPGA-Based GHA [6]**
Clock Rate	50 MHz	50 MHz	1 MHz	70 MHz
Throughput (channels/min)	20590	12381	90	11928
Normalized Throughput (channels/min/MHz)	441.8	247.62	90	170.4
Power Consumption (mW)	209.37	217.12	0.521	N/A
Platform	FPGA Cyclone IV EP4CGX150	FPGA Cyclone IV EP4CGX150	ASIC 90 nm 1P9M	FPGA Xilinx Virtex6 XC6VSX315T

而 FCM 的部分如同 GHA 一般，本論文亦針對執行時間(Computation Time)與功率消耗(Power Consumption)進行了比較，詳細內容請見表 4.7。

表 4.7 FCM 電路與 Software-Based 之分群法則比較

	Proposed FCM Architecture	**K-Means Software Implementation [4]**	**K-Means Software Implementation [4]**	**FCM Software Implementation**
Number of Feature Vector	800	800	800	800
Clock Rate	50 MHz	13 MHz	416 MHz	1.60 GHz
Computation Time (ms)	26.28	11597.359	333.33	57.4726
Power Consumption (mW)	199.26	85.809	933.99	672.22
Platform	FPGA Cyclone IV EP4CGX150	SmartDust iMote2	SmartDust iMote2	General Purpose CPU Intel i5-2467M

架構[4]採用 K-Means 演算法並於 SmartDust 平台中實作，由表 4.7 可以看出本論文架構與其相比有著較低的 Computation Time 以及相似的 Power Consumption。同時，本論文採用 FCM 亦避免了傳統 K-Means 容易落入局部最佳化的缺點。而與實現於 Intel i5 中的 FCM 軟體演算法作比較，本論文所提出的架構不論在計算時間或是功率消耗上都有著明顯地優勢。

表 4.8 棘波分類系統功率消耗與功率密度

Platform	**Chip Area**	**Power Consumption**			**Power Density**
		GHA Circuit	FCM Circuit	Spike Sorting System (GHA+FCM+NIOS CPU)	Spike Sorting System
Cyclone IV GX150	9.61 cm^2	209.37 mW	199.26 mW	284.39 mW	29.59 mW/cm^2

最後，本論文於表 4.8 整理了所提出的棘波分類系統功率消耗之相關資訊。考量棘波分類晶片必須植入於體內，因此在高速計算的情況下所帶來的熱消散也必須在設計的考量範圍內。為了避免於高速計算時所產生的廢熱對腦細胞組織造成永久的傷害，本論文參考了由論文[20]針對於功率密度(Power Density)設定了一個安全閥值(62 mW/cm^2)，且功率密度的定義為每單位面積內所消耗的功率。由表 4.8 可看出本論文所提出運作於 FPGA 中的棘波分類晶片其功率密度小於安全閥值一半以下，對於人體細胞而言屬於安全範圍內。因此，本論文所提出的架構不僅擁有高速計算，低功率消耗等優點，同時低功率密度的消耗也使得晶片植入體內後並不會對人體細胞造成損害。

第五章 結論

近年來生物學家、醫學家對於研究人腦的運作以及分析不遺餘力，目前各領域學家也提出了許多針對腦波訊號分析以及處理的相關研究。本論文以 FPGA 為基礎實現了一套棘波分類系統，透過管線化的方式將運算速度大幅縮短，因此就算於低時脈運作下本系統仍然能擁有著卓越的輸出產能。同時，本系統在合理的資源消耗下，對於運算時間、準確率與功率消耗取得了一個較佳的平衡點。因此本論文兼具著低資源消耗與即時運算的優點。

根據前一章比較的結果，可以知道本論文所提出的架構不論在分類正確率或計算速度上與現有提出的設計[4,5,6]相較都有著不錯的表現，雖然採用 FPGA 來實現本系統可能會使功率消耗比 ASIC 來得大，但整個系統的功率密度仍舊在安全範圍內，因此雖然功率消耗不如 ASIC 低，但對於植入體內的需求與安全考量亦能滿足。

總結來說，本論文所提出的電路架構在棘波分類的應用上確是有其助益，而且也具有充分的延伸性與可調整性，同時與現有的架構相比更有著不少的優勢。因此，本論文所設計的棘波分類系統確是有其需求和效率的電路架構。

參考文獻

[1] M.S. Lewicki,“A review of methods for spike sorting: the detection and classification of neural action potentials,” Network Computer Neural System, Vol. 9, pp. R53R78, 1998.

[2] S. Haykin, Neural Networks and Learning Machines, 3rd ed.; Pearson: Upper Saddle River, NJ, USA, 2009.

[3] T.D. Sanger, “Optimal unsupervised learning in a single-layer linear feedforward neural network,” Neural Network, Vol. 12, pp.459-473, 1989.

[4] Y. Sun, S. Huang, J. J. Oresko, and A. C. Cheng, “Programmable Neural Processing on a SmartDust for Brain-Computer Interfaces,” IEEE Trans. Biomedical Circuits and Systems, Vol. 4, pp.265-273, 2010.

[5] T.-C. Chen, W. Liu and L.-G. Chen,“VLSI Architecture of Leading Eigenvector Generation for On-chip Principal Component Analysis Spike Sorting System,” Proc. 30th Annual International IEEE EMBS Conference Vancouver, British Columbia, Canada, pp.3192-3195, 2008.

[6] B. Yu, T. Mak, X. Li, F. Xia, A. Yakovlev, Y. Sun, C.-S. Poon,“A Reconfigurable Hebbian Eigenfilter for Neurophysiological Spike Train Analysis,” Proc. International Conference on Field Programmable Logic and Applications, pp.556-561, 2010.

[7] M. Aghagolzadeh, F. Zhang, and K. Oweiss,“An Implantable VLSI Architecture for Real Time Spike Sorting In Cortically Controlled Brain Machine Interfaces,” Proc. 32nd Annual International Conference of the IEEE EMBS Conference, Buenos Aires, Argentina, pp.1569-1572, 2010.

[8] A. M. Kamboth, M. Raetz, K. G. Oweiss, and A. Mason, “Area-power efficient VLSI implementation of multichannel DWT for data compression in implantable neuroprosthetics,” IEEE Trans. Biomedical Circuits and Systems, Vol. 1, pp.128-135, 2007.

[9] A. M. Kamboh, and A. J. Mason, “On-Chip Feature Extraction for Spike Sorting in High Density Implantable Neural Recording Systems,” Proc. IEEE Biomedical Circuits and Systems Conference, pp.13-16, 2010.

[10] Y. Yang, and A. J. Mason,“On-Chip Spike Clustering & Classification using Self Organizing Map for Neural Recording Implants,” Proc. IEEE Biomedical Circuits and Systems Conference, pp.145-148, 2011.

[11] S.-J. Lin, Y.-T. Hung, and W.-J. Hwang, “Efficient hardware architecture based on generalized Hebbian algorithm for texture classification,” Neurocomputing, pp.3248-3256, 2011.

[12] N. Sudha, A.R. Mohan, P.K. Meher, “A self-configurable systolic architecture for face recognition system based on principal component neural network,” IEEE Trans. Circuits Syst. Video Technol, Vol. 21, pp.1071-1084, 2011.

[13] S. -J. Lin, W. -J. Hwang, andW. -H. Lee, “FPGA Implementation of Generalized Hebbian Algorithm for Texture Classification,” Vol. 12, pp.6244-6268, Sensors, 2012.

[14] J. Lazaro, J. Arias, J.L. Martin, C. Cuadrado, A. Astarloa, “Implementation of a modified fuzzy c-means clustering algorithm for realtime applications,” Microprocessor and Microsystems, Vol. 29, pp.375-380, 2005.

[15] H. -Y. Li, W. -J. Hwang, and C. -Y. Chang, “Efficient Fuzzy C-Means Architecture for Image Segmentation, Sensors, Vol. 11, pp.6697-6718, 2011.

[16] Y.-J. Yeh, H.-Y. Li, C.-Y. Yang, and W.-J. Hwang, ”Fast Fuzzy C-Means Clustering Based on Low-Cost High-Performance VLSI Architecture in Reconfigurable Hardware,” pp.112-118, IEEE International Conference on Computational Science and Engineering, 2010.

[17] L. S. Smith and N. Mtetwa, “A tool for synthesizing spike trains with realistic interference,” Vol. 159, pp.170-180, Journal of Neuroscience Methods, 2007.

[18] Altera Corporation. NIOS II Processor Reference Handbook ver 10.0. 2010. Available online: http://www.altera.com/literature/lit-nio2.jsp (accessed on 3 May 2012).

[19] Altera Corporation. SOPC Builder User Guide. 2011. Available online:http://www.altera.com/literature/lit-sop.jsp (accessed on 3 May 2012).

[20] W. Reichert, “Indwelling neural implants: Stategies for contending with the in vivo environment,” in BMI-Related Thermal Studies. Boca Raton, FL: CRC, 2007.