

國立交通大學

電機與控制工程學系

碩士論文

智慧型 DMA 應用於成本導向之語音處理器
晶片設計



A Cost-Effective Speech Processor Design and
Implementation Using Smart DMA

研究生：廖峻谷

指導教授：林進燈 教授

中華民國九十五年七月

智慧型DMA應用於成本導向之語音處理器晶片設計
A Cost-Effective Speech Processor Design and
Implementation Using Smart DMA

研究生：廖峻谷

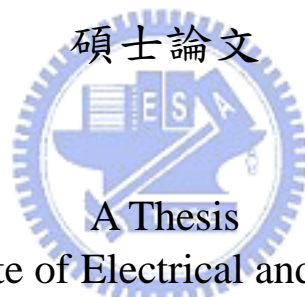
Student: Chun-Ku Liao

指導教授：林進燈 教授

Advisor: Chin-Teng Lin

國立交通大學

電機與控制工程學系



Submitted to Institute of Electrical and Control Engineering
College of Electrical Engineering and Computer Science
National Chiao Tung University
in partial Fulfillment of the Requirements
for the Degree of
Master
in
Electrical and Control Engineering

July 2006

Hsinchu, Taiwan, Republic of China

中華民國九十五年七月

智慧型 DMA 應用於成本導向之語音處理器晶片設計

學生：廖峻谷

指導教授：林進燈 博士

國立交通大學電機與控制工程研究所

中文摘要

近年來嵌入式系統與消費性電子產品的迅速發展，造成產品週期愈來愈短，而效能需求卻愈來愈高，如何兼顧成本及效能是個很重要的議題。目前定點數 16-bit DSP 的產值高達七成以上，低成本成為消費性電子產品很重要的一個關鍵。這裡提供一個兼顧成本與效能的解決方案。本篇論文在以開發的智慧型 DMA 上(擁有乘加運算器的 DMA)，改進定址模式與運算資料路徑，將它提昇為 DSP 運算器，與已開發的通用嵌入式處理器整合成為 RISC/DSP 單核心處理器，並針對特定的演算法，加入硬體加速器 IP，擴充處理器的應用範圍。此架構的優點在於單核心處理器具有低成本的特性，相對於雙核心處理器而言系統複雜性低。為了增強單核心處理器的運算效能，對於不同的演算法可搭配不同的硬體加速器，本論文為了加速語音壓縮的演算法，特別針對向量量化的部份，設計一個能處理多級編碼簿的向量量化單元，設計特色在於：(1)加入 M-L 搜尋法，增加量化的精確度；(2)四級運算管線設計，提高計算吞吐量(throughput)；(3)採用低成本方式設計，運算元與智慧型 DMA 共用，且利用記憶體暫存量化資料，減少暫存器的使用；(4)可由使用者參數設定量化環境，符合 IP 可重複使用的要求。本論文提出一個兼顧成本與效能，語音運算導向的 RISC/DSP 單核心處理器。此晶片採用 UMC 0.18 μm 製程，晶片面積約 $3.5 \times 3.5 \text{ mm}^2$ ，預估最大操作頻率在 100MHz。

A Cost-Effective Speech Processor Design and Implementation Using Smart DMA

Student: Chun-Ku Liao

Advisor: Chin-Teng Lin

Department of Electrical and Control Engineering

National Chiao-Tung University

Abstract

Recently the rapid growth of embedded system and consumer electronics has resulted in the decreasing product life cycle; the demand of efficiency, however, kept increasing. Therefore, how to balance the efforts spent between cost and efficiency became an urgent issue. In the present time, the 16-bit fixed-point DSP has achieved more than 70 percent of that of total DSP, hence low cost has turned into a key target of consumer electronics. This study presents a solution to look after both cost and efficiency. On the developed smart DMA(DMA with MAC), this thesis improves addressing mode and data path ,and promotes smart DMA to a DSP unit , then integrated it with developed general purpose embedded processor. This can achieve a unit-core processor with paralleled structure of RISC and DSP. In addition, this thesis presents hardware accelerator IP in accordance to a certain algorithm to expand the applied scope of the processor. The benefit of this structure is that unit-core processor possessed properties of low cost and one memory system with one set of developer, compared to dual-core one, the unit-core processor is less complicated in system. To improve the computing efficiency of unit-core processor, different algorithm can be collocated different hardware accelerator. Moreover, in

order to accelerate the algorithm of the speech compression, this thesis designed a unit of vector quantization which can process multistage codebook, specifically. The main features of this design are: (1)added M-L search methods to improve the accuracy of quantization; (2)Four-stage computing pipeline to raise throughput; (3)with the consideration of low cost, shared computing unit with smart DMA, and took advantage of memory to register quantification data to decrease the use of register; and (4)capability to set environment of quantification with parameters of user to conform to the demand of reusable of IP. In this thesis, we presented a speech RISC/DSP unit-core processor, which balanced the efforts spent between cost and efficiency. The chip has been integrated in the total area of $3.5 \times 3.5 \text{mm}^2$ by using UMC $0.18 \mu\text{m}$ CMOS technology. The maximum clock frequency is 100MHz with a single 1.8V supply.



誌謝

兩年的研究所生涯隨著論文的完成劃上了句號，這兩年間，要感謝許多人的鼓勵和幫忙，使我獲得充實的專業能力並順利完成研究所的學業。

首先要感謝的是我的指導教授-林進燈老師。林老師是國內十分傑出的一位教授，在不同領域內都有相當好的研究成果。感謝老師提供了很理想的研究環境及正確的引導，使我在研究上非常順利。在老師悉心的指導下，讓我學習到解決問題的能力及做研究應有的態度。

另外，最感謝資工系范倫達教授及鐘仁峰學長的教導，尤其是面臨畢業主題方向模糊的壓力時，范教授在這上面給予我相當大的助力，而教授親切的態度及學識的也讓我在討論論文時感到輕鬆而無壓力，獲益良多。此外在實驗室中，不管大小疑難雜症，常常去請教仁峰學長，非常感謝學長不厭其煩地教導，使我增進了對積體電路設計上的專業知識，開拓了我的視野。也感謝實驗室所有的夥伴，得正學長、家昇、有德、庭緯、經翔、紹航、德璋、智文、靜瑩、肇廷等，感謝大家在研究上的互相扶持及鼓勵。

也感謝我的家人，父母廖見郎先生及洪玉蓉女士，你們的支持一直是最溫暖的後盾；哥哥姊姊峻毅、家沁謝謝你們包容我的煩躁，你們的鼓勵是我信心的來源。

最後要感謝沛柔，陪伴我渡過這段辛苦的日子，這段時間有妳真的很重要，謝謝！

目 錄

中文摘要	iii
Abstract	iv
誌謝	vi
目 錄	vii
圖 目 錄	x
表目錄	xii
第一章 緒論	1
1.1 簡介	1
1.2 論文架構	7
第二章 智慧型 DMA 控制器介紹	8
2.1 智慧型 DMA 傳輸模式	8
2.2 智慧型 DMA 雙通道運算	8
2.3 智慧型 DMA 控制器架構	9
2.3.1 通道控制器 (Channel Controller)	10
2.3.2 暫存器組 (Register Bank)	11
2.3.3 優先權仲裁器 (Prioritizing Arbiter)	13
2.3.4 中斷處理 (Interrupt)	13
2.3.5 算數運算邏輯單元 (ALU)	14
2.3.6 多級向量量化單元 (Multi-stage Vector Quantization) ..	15
2.3.7 蝴蝶運算單元 (Butterfly Unit)	16
2.3.8 記憶體選擇器 (Memory Mux)	17

2.3.9 周邊匯流排 (APB - Advanced Peripheral Bus)	19
2.4 智慧型 DMA 控制暫存器	20
2.4.1 來源暫存器 (Source Register)	20
2.4.2 目的暫存器 (Destination Register)	21
2.4.3 控制暫存器 (Control Register)	21
2.4.4 配置暫存器 (Configuration Register)	22
2.4.5 狀態暫存器 (Status Register)	23
2.4.6 輸出暫存器 (Output Register)	24
2.4.6 功能暫存器 (Function Register)	24
2.4.7 智慧型 DMA 使用流程.....	24
第三章 智慧型 DMA 運算控制器設計	26
3.1 多級向量量化器設計	26
3.1.1 原理	26
3.1.2 設計架構.....	29
3.1.3 控制輸入暫存器.....	30
3.1.4 流程控制單元(Address Generator Unit).....	31
3.1.5 運算管線設計(Pipeline design).....	33
3.1.6 比較器(Comparator).....	34
3.1.7 數值表單元(Write Table).....	34
3.1.8 最佳路徑選擇單元(Select M_Paths).....	37
3.1.9 運作流程.....	37
3.2 蝴蝶運算器單元設計	38
3.2.1 蝴蝶運算理論.....	38
3.2.2 架構.....	39
3.2.3 程式流程設定.....	41
第四章 智慧型 DMA 控制器與 RISC 處理器的整合.....	43
4.1 RISC 處理器	43
4.1.1 RISC 處理器核心.....	43
4.1.2 RISC 處理器指令集.....	45
4.2 軟體開發環境	48
4.3 整合	50

第五章 驗證結果與晶片實現	53
5.1 設計驗證與應用結果	53
5.1.1 MELP 語音壓縮介紹	53
5.1.2 MELP 語音壓縮演算法複雜性分析	54
5.1.3 搜尋音高(Pitch Search)驗證	56
5.1.4 量化線性預估係數(LPC)驗證	59
5.1.4 快速傅立葉轉換(FFT)驗證	61
5.1.5 結論	63
5.2 晶片製作	65
5.2.1 設計流程	65
5.2.2 合成結果	66
5.2.3 佈局與封裝	66
5.3 效能比較	68
5.3.1 資料傳輸效能	70
5.3.2 資料運算效能	71
5.3.3 其他處理器比較	72
第六章 結論	75
參考文獻	77



圖目錄

圖 1-1：RISC 與 DSP 規劃圖	2
圖 2-1：雙通道運算情形	9
圖 2-2：智慧型 DMA 的架構	10
圖 2-3：通道控制器架構圖	11
圖 2-4(a)：智慧型 DMA 暫存器組的讀取時序圖	11
圖 2-4(b)：智慧型 DMA 暫存器組的存取時序圖	11
圖 2-5：暫存器組示意圖	12
圖 2-6：優先權仲裁器 (Prioritizing Arbiter) 控制兩組匯流排	13
圖 2-7：中斷控制器連接狀態暫存器	14
圖 2-8：乘法器的使用情形	15
圖 2-9：ALU 單元	15
圖 2-10：運算管線與 ALU 使用情形	16
圖 2-11：蝴蝶運算單元與周邊的溝通	17
圖 2-12：透過記憶體選擇器的資料傳輸	18
圖 2-13(a)：記憶體讀取時序圖	18
圖 2-13(b)：記憶體存取時序圖	18
圖 2-14：APB 匯流排受控端介面	19
圖 2-15(a)：APB 匯流排寫入時序圖	20
圖 2-15(b)：APB 匯流排讀取時序圖	20
圖 2-16：智慧型 DMA 的使用流程	25
圖 3-1：三級向量量化器 [18]	27
圖 3-2：三級向量量化器利用 M-L 法搜尋編碼簿 (這裡 M=4)	27
圖 3-3：多級向量量化器使用 M-L 搜尋流程圖	28
圖 3-4：多級向量量化器架構圖	29
圖 3-5：多級向量量化器流程圖	30
圖 3-6：流程控制單元狀態圖	31
圖 3-7：運算管線時間圖	33
圖 3-8：數值表單元狀態圖	35
圖 3-9：數值表單元記錄下一級的輸入圖	36
圖 3-10：數值表單元輸出圖	37
圖 3-11：多級向量量化器與周邊運作流程圖	38
圖 3-12：8 點 FFT 圖	39
圖 3-13：蝴蝶運算圖	39
圖 3-14：多級向量量化器架構圖	40

圖 3-15：蝴蝶運算單元架構.....	41
圖 3-16：FFT 運算分工情形.....	42
圖 3-17：利用 SDMA 計算 FFT 運算.....	42
圖 4-1：組譯器 (Assembler) 的用途.....	49
圖 4-2：圖形化介面組譯器 (Assembler).....	50
圖 4-3：RISC 處理器與智慧型 DMA 的整合.....	51
圖 4-4：RISC 處理器存取智慧型 DMA 暫存器-記憶體對映.....	52
圖 5-1：語音壓縮演算法特性比較[29].....	54
圖 5-2：MELP 編碼解碼比例.....	54
圖 5-3：MELP 編碼各單元複雜度.....	55
圖 5-4：Quantization 內各單元運算複雜度.....	55
圖 5-5：處理器與智慧型 DMA 處理 Pitch 的分工情形.....	57
圖 5-6：Pitch Search C 程式與 Post-sim 結果 I.....	58
圖 5-7：Pitch Search C 程式與 Post-sim 結果 II.....	58
圖 5-8：多級向量量化 C 程式執行結果.....	60
圖 5-9：多級向量量化 Post-sim 結果.....	60
圖 5-10：512-FFT 各點數值表較.....	62
圖 5-11：晶片設計流程.....	65
圖 5-12：佈局及腳位圖.....	67
圖 5-13：208 CQFP 打線圖.....	67



表目錄

表 1-1：向量量化架構比較表.....	5
表 1-2：向量量化參數表.....	7
表 2-1：數位訊號處理運算的定址法表.....	9
表 2-2：來源暫存器 (Source Register).....	20
表 2-3：目的暫存器 (Destination Register).....	21
表 2-4：控制暫存器 (Control Register).....	22
表 2-5：配置暫存器 (Configuration Register).....	23
表 2-6：狀態暫存器 (Status Register).....	23
表 2-7：輸出暫存器 (Output Register).....	24
表 2-8：功能暫存器 (Function Register).....	24
表 3-1：控制輸入暫存器.....	31
表 3-2：控制輸入暫存器.....	32
表 3-3：控制輸入暫存器.....	32
表 3-4：架構修改前後誤差表.....	34
表 3-5：數值表單元狀態工作內容.....	35
表 3-6：控制輸入暫存器.....	36
表 4-1：資料搬移指令列表.....	45
表 4-2：算數邏輯運算指令列表.....	46
表 4-3：跳躍指令列表.....	47
表 4-4：其他指令列表.....	47
表 4-5：智慧型 DMAC 控制指令列表.....	48
表 5-1：Pitch Search 效能表.....	59
表 5-2：多級向量量化效能表.....	61
表 5-3：M Paths 量化路徑結果表.....	61
表 5-4：512-FFT 含位元反轉效能表.....	63
表 5-5：智慧型 DMA 化簡複雜度表.....	64
表 5-6：合成後的資訊.....	66
表 5-7：晶片設計規格.....	68
表 5-8：與市面上 DMAC 做比較.....	69
表 5-9：資料傳輸效能表.....	70
表 5-10：位元反轉資料傳輸效能表.....	71
表 5-11：資料運算效能表.....	71
表 5-12：FFT 效能比較表.....	72
表 5-13：多級向量量化效能比較表.....	72

表 5-14：多級向量量化效能比較表[33]..... 73
表 5-15：與 TI 的規格比較表..... 74
表 5-16：效能與成本比較表..... 74



第一章 緒論

1.1 簡介

近年來嵌入式系統與消費性電子產品的迅速發展，造成產品週期愈來愈短，而效能需求卻愈來愈高，如何兼顧成本及效能是個很重要的議題。目前定點數 16-bit DSP 的產值高達七成以上，低成本成為消費性電子產品很重要的一個關鍵。傳統上演算法的處理都是在雙核心系統上進行的，系統中的通用處理器用於控制和管理功能，DSP 核心用於語音編解碼和高品質的演算法處理。雙核心處理器如 OMAP 將 RISC 處理器與 DSP 數位訊號處理器整合在一顆晶片中，透過高效率的分工合作，達成系統晶片的效能需求。這種架構常見於需要大量迴圈、向量運算等應用中。將這類的運算交由 DSP 處理，可以發展較佳之平行化與最佳化演算法，來提高整體效能。

在單一處理器上結合控制和訊號處理功能的單核心架構，相較於傳統雙核心系統可以降低功耗和成本。如圖 1-1 所示，使用一個處理器可以簡化設計、減少整合問題、縮短開發時間並降低風險，同時還能減少晶片週邊的數據傳輸。反之，將 DSP 獨立出來的雙核心架構將涉及多個記憶體系統管理，以及 DSP 和處理器核心之間的封包交換，且佔用的每個時脈週期都要消耗功率。而單核心處理器意味著只有一個記憶體系統。使用開發環境時，若能只處理單一個工具鏈(tool chains)工具和工時方面以減少設計費用。此外，採用一個核心而不是兩個核心可以減少核心和記憶體系統的晶片面積，因而降低晶片的成本，單核額外的成本來自於硬體加速器或特殊功能的 IP，比起雙核心仍要少的多。本論文將採用單核心的方式進行處理器設計來簡化系統，針對語音處理器需要的低成本要求。

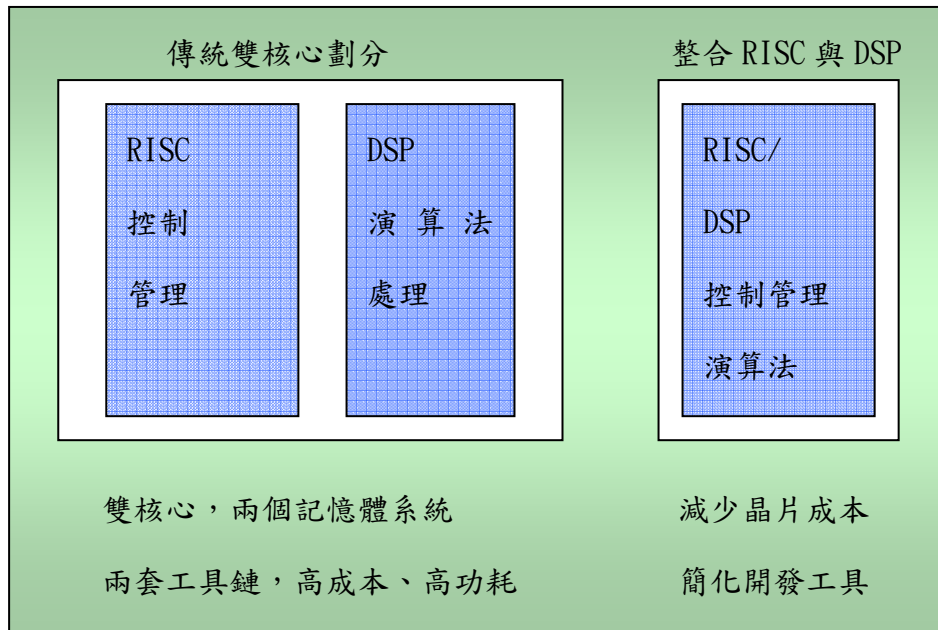


圖 1-1：RISC 與 DSP 規劃圖

精簡指令集處理器(RISC)架構特色在於區別存取指令和計算的指令，並擁有多級管線架構，以一個時脈週期為單位執行指令，達到高時脈的處理速度。以運算為導向的數位訊處理器(DSP)則不同於RISC的架構，指令集以記憶體相關與運算效率為導向，例如濾波器的運算與快速複立葉轉換演算法。此外還擁有自己的暫存器組，位址單元與運算器，數位訊號處理器有相當多特別的架構及定址法[1]，如環狀定址(Circular Addressing)、位元反轉定址模式(Bit-reversed Addressing)、零負擔(Zero overhead)迴圈、單一指令週期乘加運算等，並與RISC間有少量的暫存器共用。

為了支援日新月異的多媒體技術，如使用VOIP和WiFi技術的手機，商業應用總量在2009年將達到1.41億部，許多設計公司爭相推出或研究新的解決方案以滿足這種即將到來的市場需求。為了降低數位訊號處理器開發成本和複雜度[2]，因此，本論文在發展出的智慧型DMA控制器(Smart DMA Controller)，加入數位訊號處理的運算器，利用DMA有控管大量資料的特性，結合DMA資料傳輸與數位訊號處理能力，可幫助處理器進行運算前資料排序[3]和記憶體相關的指令運算，如向量內積[4]，快速複立葉轉換等；此外，利用智慧型DMA的各種

定址模式，設定傳輸資料數，可以模擬 DSP 對固定 LOOP 的處理，節省處理器跳躍指令的個數，減少因跳躍指令所帶來的負擔(penalty)。在 DSP 指令解碼方面，使用 DMA 的控制介面來設定運算器的運作方式，仍屬於 RISC 架構的指令方式，並不增加指令集複雜度。處理器與智慧型 DMA 的結合，為一個單核 RISC/DSP 架構數位訊號處理器[2]。

一般數位訊號處理器(General Purpose DSP)在面對特定且複雜演算法時，須對大量數據進行處理與計算，例如 3D 聲音處理和影像、語音的辨識等多媒體演算法，但高階的 DSP 成本過高，其成效不如專業的特定用途 IC(Application-Specific IC; ASIC)，因此需要特定的輔助運算器或硬體加速器[5][6]，以提高處理器的層級。利用矽智產觀念來輔助設計，矽智產的重複使用可有效降低設計單晶片系統的複雜度，縮短晶片設計時程，提昇晶片設計的成功率。在智慧型 DMA 中，設計了一個能處理多級編碼簿的向量量化器，設計上符合 IP 的重覆使用性與低成本目標[7]，以解決處理器處理編碼簿時，需反覆且規律存取記憶體所帶來的負擔。處理器在加上硬體加速器後，成為能快速處理特定演算法的高效能處理器，兼顧成本與效能。

向量量化(Vector Quantization, VQ)的應用，隨著網際網路的蓬勃發展及電腦相關科技的進步，已被廣泛地應用在語音及影像的編碼系統中，包含各種影像、聲音、圖片等多媒體產品[8]。多媒體資料量龐大，資料壓縮愈突顯重要且無法避免。向量量化技術在語音壓縮方面：網路電話VoIP的通訊協定G.723.1，已實現為產業界產品，而美國國防部發展的保密通訊協定MELP[9]與MELPe[10]，分別有2.4Kbps與1.2Kbps的高壓縮率。在影像壓縮方面：已運用在多媒體系統、高畫質電視、電傳系統(Teleconferencing Systems)、影像資料庫管理(Image Database Management)、影像壓縮(Image Compression)、影像辨識(Identify)等。

向量量化需從編碼簿中，比對每一個碼向量與輸入的向量，找出最接近輸入向量的碼向量，然後將此碼向量的相對應編碼簿的索引值輸出，達到壓縮的目的。

的，而解碼端只要依據索引值去編碼簿找出相對應的碼向量，即能還原訊號，為失真的壓縮方式(Lossy Compression)。此種逐一比較的方式 稱完全搜尋(Full Search)法，此法雖然簡單，且相較於其他的搜索法精確度最高，但是卻也是最耗時、運算需求量最大的方法。近年來已有許多文獻提出不同的方式來加速碼簿的搜尋，但缺乏經濟及有效率的使用實現在硬體電路。回顧硬體設計關於向量量化的文獻，最早由Allen Gersho [11]提出7級管線的向量量化處理器架構，只能處理完全搜尋，而Wang[12]利用心臟收縮電路(systolic architecture)，用陣列式的運算單元一起運算，達到高平行度與高吞吐量(throughput)的目的，但這個方法卻也受到面積與精確度的限制，當編碼簿越大，需要的運算陣列也越大，而陣列中每個運算單元受限於硬體複雜度，選用最小絕對值誤差(Mean Absolute Error)當失真誤差的量測，精確度不如最小均方誤差(Mean Square Error)。Lee[13]利用有限狀態實現向量量化(Finite State VQ)的硬體，並用多條計算路徑(Multipath)增加量化的精確度，但面積相當大。Zunino [14]利用階層式的搜尋增進搜尋速度，在硬體設計上用4個最小絕對值誤差單元。綜觀以上的向量量化架構與收尋法，硬體複雜度相當高，並且當量化條件不同時，架構也勢必做調整，並不符合IP的特性與要求。

這裡提出一個向量量化的架構，能進行完全搜尋或搜尋多級編碼簿的能力，多級編碼簿的優點在於化簡編碼簿的大小；在失真量測方面，可以處理加權最小均方誤差(Weighted Mean Square Error)，並利用M-L搜尋法[15]，加強多級向量量化的精確度。此外，為了達到IP的使用性，可由使用者自行設定量化個數(Order)、編碼簿級數(Stage)、編碼簿各級碼向量數(Level)與M-L搜尋路徑數(M Paths)，以上參數的調整能使本論文的向量量化器應付各種量化的要求。最後為了增加硬體運算吞吐量(throughput)，設計四級的運算管線，相較於其他文獻的設計方式，如表1-1。

表 1-1：向量量化架構比較表

Method	C.L. Wang [12]	C.Y. Lee [13]	Zunino [14]	Proposed
Architecture	Systolic	Parallel HW compute	Parallel HW compute	4-Pipeline Design
Distance Measure	MAE	MSE	MAE	WMSE
VQ Type	Full Search	FiniteState VQ with MultiPath Search	Hierarchical VQ	MultiStage VQ with M-L Search
Hardware Throughput	16 pixel/cycle	1 pixel/cycle	4 pixel/cycle	1 sample/cycle
Gate count	300K (N=256, K=16) (256 MAE)	420K	42K (4 MAE unit)	26K

綜合以上，智慧型 DMA 的主要設計特點如下：

1. 應用上支援廣泛的 I/O 系統，並有效率地處理資料流：

多媒體訊號處理需要有大量的資料流做為輸入輸出，因此必須設計一個 DMA 模組，支援匯流排結構，並對 I/O 進行大量資料移動、緩衝及控管的工作。對於一些特殊應用，如影像語音壓縮(Image/Speech Compression)，空間迴響器(Reverberator)等演算法，需要大量的記憶體空間存放編碼簿或暫存濾波器的資料，所以智慧型 DMA 也支援外部記憶體的連接。另外提供了五種定址模式，加快傳輸的速率：

- 遞增/遞減定址法 (Increasing/Decreasing Addressing)

- 環狀定址法 (Circular Addressing)
- 鏡射定址法 (Mirror Addressing)
- 索引定址法 (Index-based Addressing)
- 位元反轉法 (Bit-Reverse Addressing)

2. 內建 ALU 單元，達到數位訊號處理器的效能：

加入 ALU 單元，配合各種定址法與資料路徑，效能可達數位訊號處理器 (DSP)。智慧型 ALU 運算單元與處理器的 ALU 做區隔，集中在需要多個指令週期的運算，能與處理器平行運作。ALU 內部有多級除法器 (DIV)，乘加器 (MAC)，柱狀位移器 (Barrel Shift)，輔助處理器運算特殊的指令。運算元搭配各種定址法與資料路徑可進行各種數位訊號常見的運算[16]：

- DCT : Mirror + Index-based + Increasing Addressing + MAC
- FIR : Increasing + Decreasing Addressing + MAC
- FFT : Bit-Reverse Addressing + Butterfly datapath + Shifter

3. 向量量化 IP，擴充處理器的應用範圍：

智慧型 DMA 加入向量量化 IP，使的處理器處理的範圍可延伸到各種影像與語音方面的應用，提高處理器的層級。向量量化 IP 用有 IP 可攜性與重覆使用性，可由使用者設定各種量化狀況，支援的向量量化參數如表 1-2。而與其他文獻設計[12][13][14]比起來成本相當低廉。另外使用多條路徑，在精確度上可以增進 2dB(SNR 比)。

表 1-2：向量量化參數表

	Max. Support
Distance Measure	Weighted Mean-Square-Error
Order	10
Stage	4
Level	256 (Every stage)
CodeBook Size	1024 Codewords
M Paths	8

1.2 論文架構



本篇論文中，第二章介紹智慧型 DMA 控制器的介紹，從硬體架構設計到如何使用有詳細的說明。第三章說明智慧型 DMA 控制器的特殊運算設計，包括加速 IP 的設計。第四章說明 VLIW 處理器的整合方式，及相關軟體發展及如何應用。第五章詳述語音應用分析，包含模擬驗證方法及結果，晶片製作，及測試效能的比較。最後，在第六章做總論。

第二章 智慧型 DMA 控制器介紹

在智慧型 DMA 中有兩種特性，分別為負責傳輸記憶體與周邊資料的 DMA，另一個為負責算術運算的數位訊號運算器。在傳輸方面，有連續資料傳輸及透過 LLI (Linked List Item) 不連續資料傳輸，並且支援四種傳輸模式搭配匯流排結構，可進行記憶體到記憶體、記憶體到周邊、周邊到記憶體、周邊到周邊的溝通。在運算方面，提供柱狀位移(Barrel shift)、乘累加、除法等資料運算路徑，可與處理器的 ALU 平行運算，為一個數位訊號處理器(DSP unit)。DSP 的輸入資料由處理器設定兩個 32-bit 內部暫存器，當運算完會將結果存在通用暫存器中，由智慧型 DMA 發出中斷給處理器取得。

智慧型的 DMA 的傳輸特性和運算特性可以結合，(1)利用智慧型 DMA 傳輸資料的定址法，與 ALU 的乘加器合作，可進行雙通道資料記憶體向量運算，實現濾波器的運算；(2)利用智慧型 DMA 的位元反轉傳輸模式，在進行快速複立葉轉換前先安排係數，再經由智慧型 DMA 裡 ALU 的資料路徑作蝴蝶運算；(3)進行編碼簿搜尋時，利用智慧型 DMA 對外部記憶體周邊的溝通能力，協助向量量化加速器傳輸編碼簿做計算。

本章介紹智慧型 DMA 控制器傳輸模式，運算模式及各單元的架構。

2.1 智慧型 DMA 傳輸模式

與一般的 DMA 相同，智慧型 DMA 支援四種傳輸模式：記憶體到記憶體、記憶體到周邊、周邊到記憶體、周邊到周邊。

2.2 智慧型 DMA 雙通道運算

智慧型 DMA 支援四種定址模式包含遞增/遞減定址法、環狀定址法、鏡射定址法、索引定址法。搭配內建運算邏輯單元(ALU)的使用，可進行各種數位訊號

處理的運算。如圖 2-1 所示，設定通道控制器的傳輸模式與定址法，智慧型 DMA 便能同步從兩顆內建的記憶體讀取資料，讀入乘加器做運算。

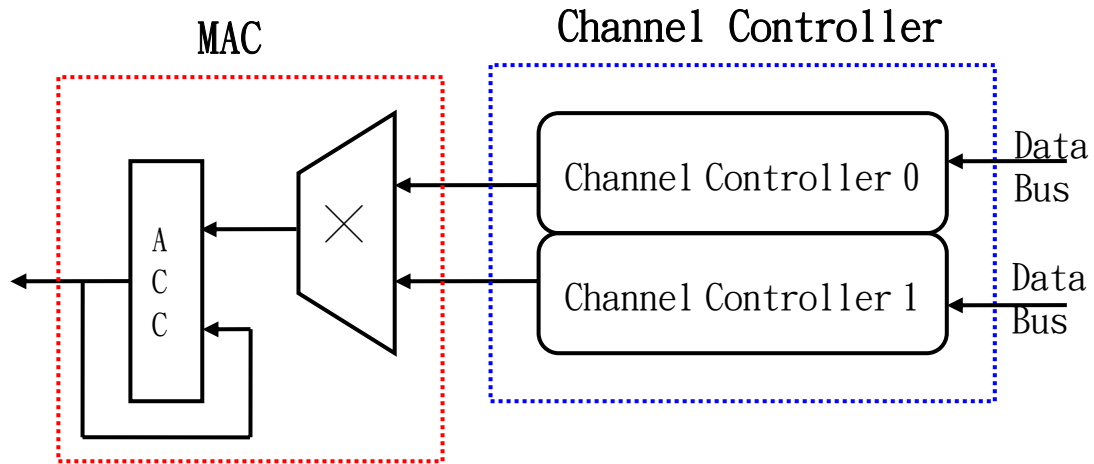


圖 2-1：雙通道運算情形

底下介紹雙通道運算如何實現各種數位訊號處理，如表 2-1 所示

表 2-1：數位訊號處理運算的定址法表

運算	使用的定址法
DCT	Mirror + Index-based + Increasing Addressing
DWT	Increasing + Decreasing Addressing
FIR	Increasing + Decreasing Addressing

2.3 智慧型 DMA 控制器架構

智慧型 DMA 整體架構如圖 2-2，包含通道控制器、優先權仲裁器、暫存器組、算術運算邏輯單元、中斷控制器及記憶體介面與多工器等，其說明如下：

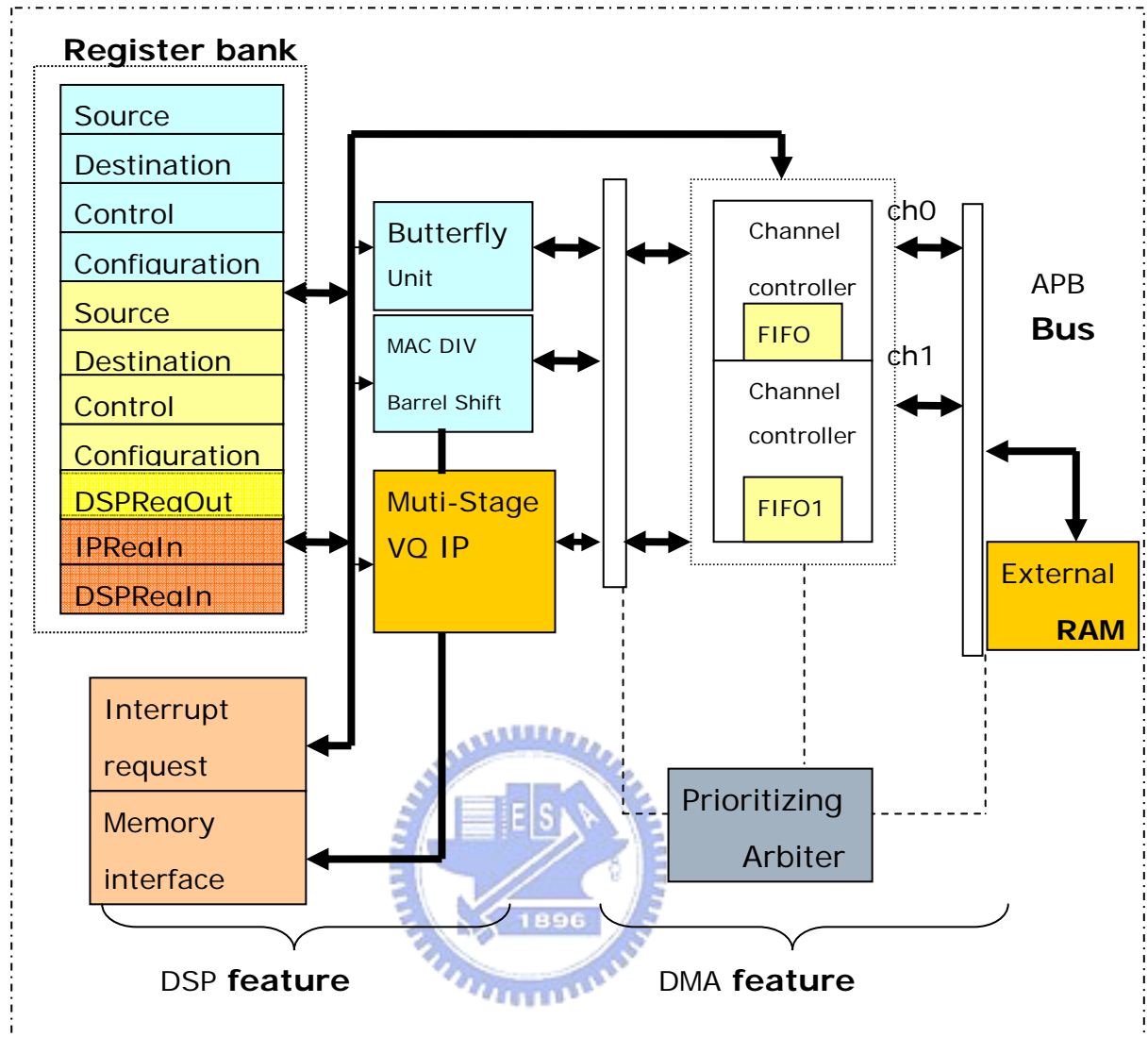


圖 2-2：智慧型 DMA 的架構

2.3.1 通道控制器 (Channel Controller)

通道控制器為 DMA 裡的主要部份，所有工作皆由通道控制器來控制。智慧型 DMA 內建兩組相同的通道控制器，只要匯流排不衝突，所有通道可同時運作；反之，由匯流排仲裁器來決定優先權。若有需要，可增加多組的通道控制器，以提高效率。通道控制器架構如圖 2-3 所示，由一組讀取控制器、寫出控制器和緩衝器 (FIFO) 所組成，讀寫控制器採用有限狀態機 (Finite State Machine, FSM) 實現，判斷緩衝器內是否已滿或有資料，來進行讀取或寫入的動作。

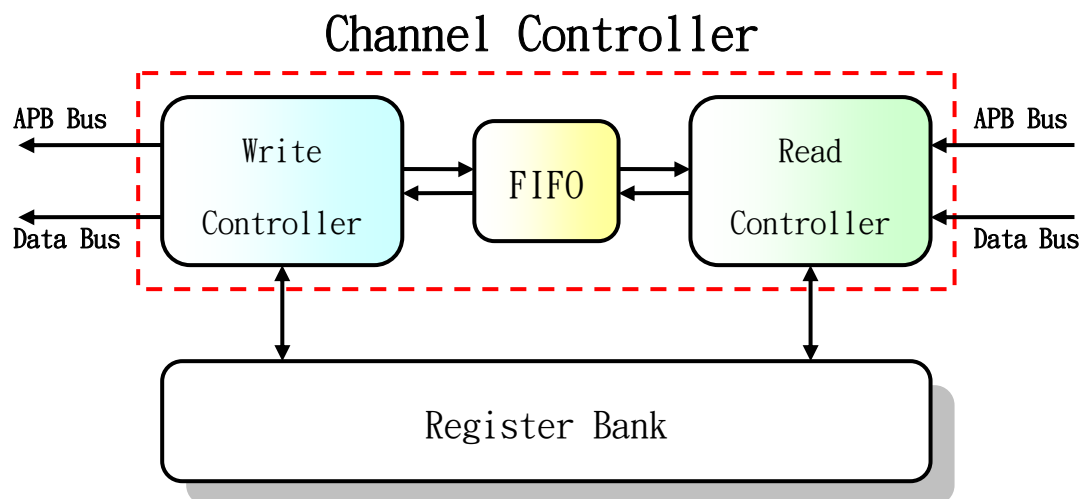


圖 2-3：通道控制器架構圖

2.3.2 暫存器組 (Register Bank)

智慧型 DMA 內部擁有十二組 32-bit 的暫存器組，而外界對智慧型 DMA 視為一組 16x16 的記憶體。外界對內部存取的方式與記憶體相同，可採用記憶體映射 (Memory-mapping) 的方式整合。詳細每個暫存器的用法，會在下一節說明。

智慧型 DMA 暫存器組的讀取/寫入時序如圖 2-4(a)(b)，所有的讀/寫動作由時脈正緣觸發，配合 CEN 訊號 (Low-active)，當 WEN=1 時由給定的位址取出暫存器的值；當 WEN=0 時，由暫存器輸入值寫入指定暫存器內。

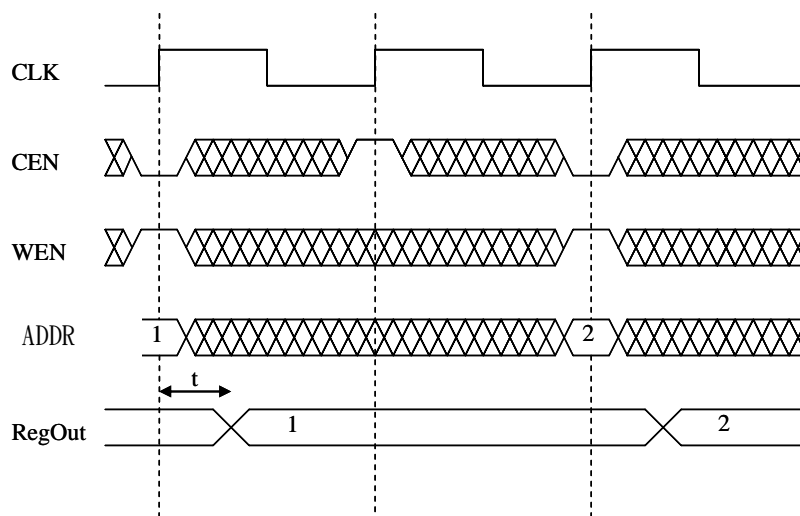


圖 2-4(a)：智慧型 DMA 暫存器組的讀取時序圖

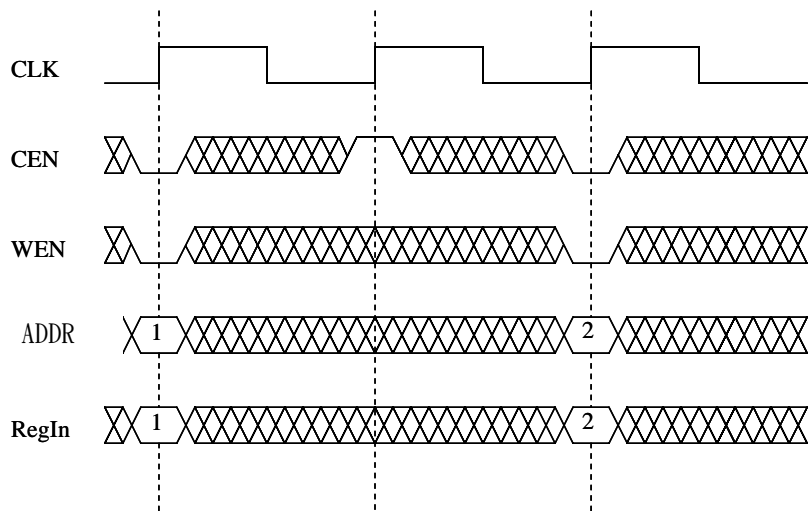


圖 2-4(b)：智慧型 DMA 暫存器組的寫入時序圖

暫存器組的周邊連接如圖 2-5 所示，左邊與處理器連接，此暫存器組被視為處理器中記憶體的一部分，右邊連接通道控制器、算數運算邏輯單元與各個 IP 單元。

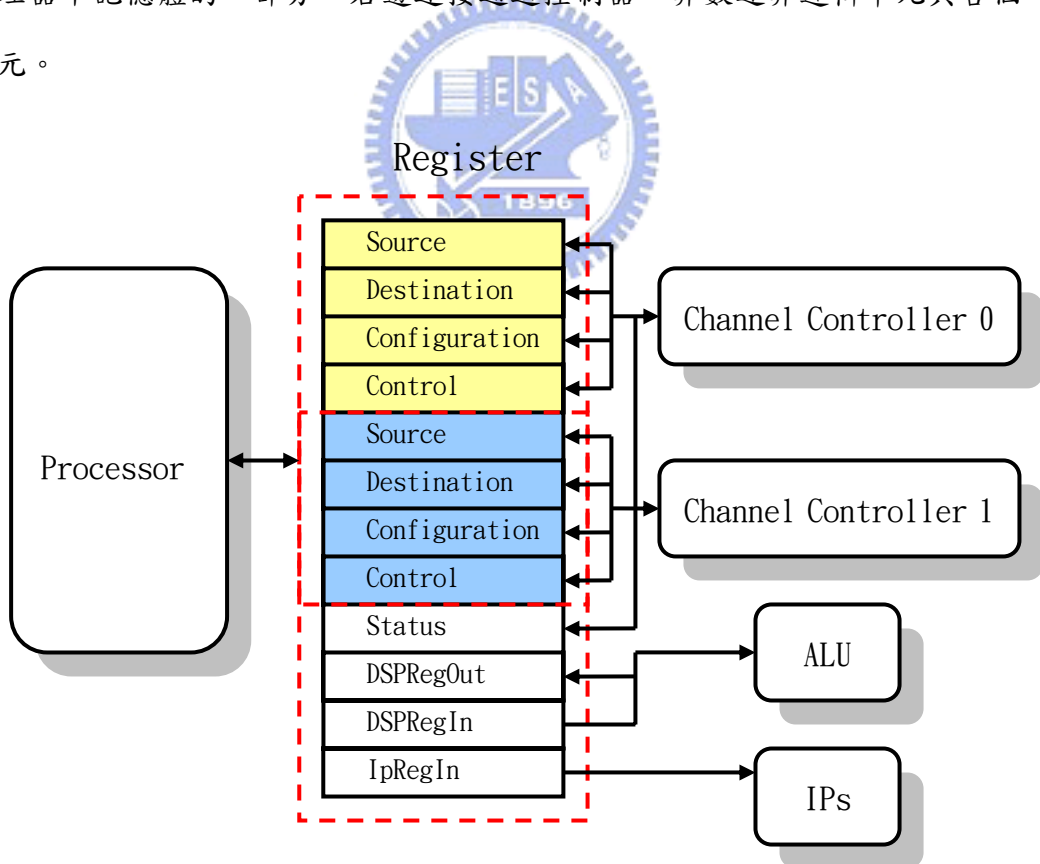


圖 2-5：暫存器組示意圖

2.3.3 優先權仲裁器 (Prioritizing Arbiter)

由於通道控制器共用兩種匯流排 (Data Bus and Peripheral Bus)，若同時使用匯流排會造成衝突的情況，因此，優先權仲裁器協調衝突的情況，決定優先的順序。如圖 2-6 所示，兩組通道控制器 Channel Controller 0、Channel Controller 1，以 Channel Controller 0 的優先權較高。因此，當同時取用相同匯流排的資料時，仲裁器會將優先權低的 Channel Controller 1 先暫停，待優先權較高的通道控制器工作結束後，再將匯流排控制權交還給 Channel Controller 1。

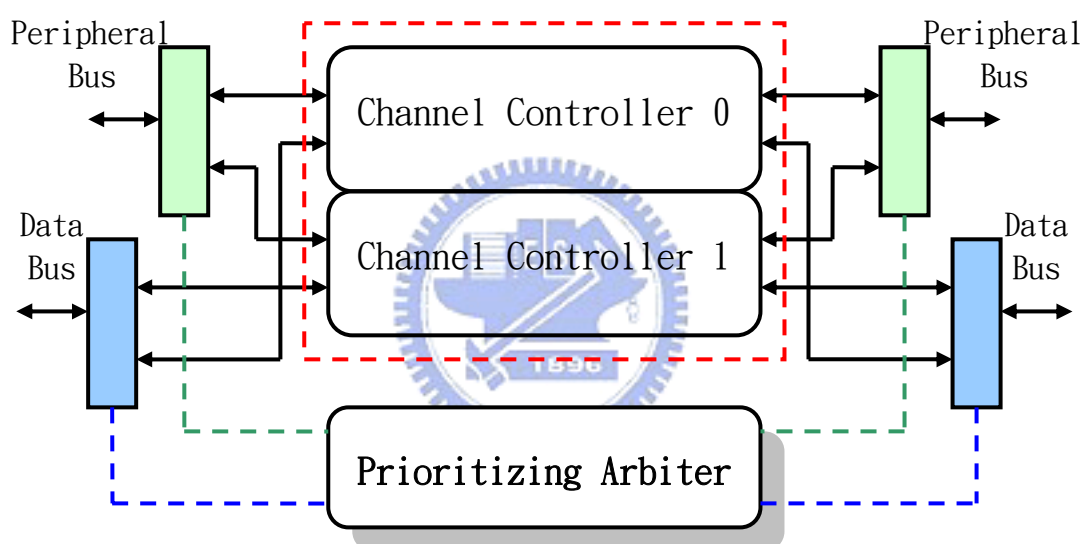


圖 2-6：優先權仲裁器 (Prioritizing Arbiter) 控制兩組匯流排

2.3.4 中斷處理 (Interrupt)

在暫存器組中的配置暫存器 (Configuration Register)，可設定中斷致能 (Interrupt Enable)。當中斷致能設定後，若通道控制器或算數運算單元工作處理完畢，將立即發出中斷請求給處理器，並關閉通道處理器，中斷訊號為 Low-Active，並持續維持二個時脈周期。通道控制器和算術運算單元連接中斷控制器，當寫出資料已結束，中斷控制器發出中斷要求，並寫入狀態暫存器中，如圖 2-7 所示。

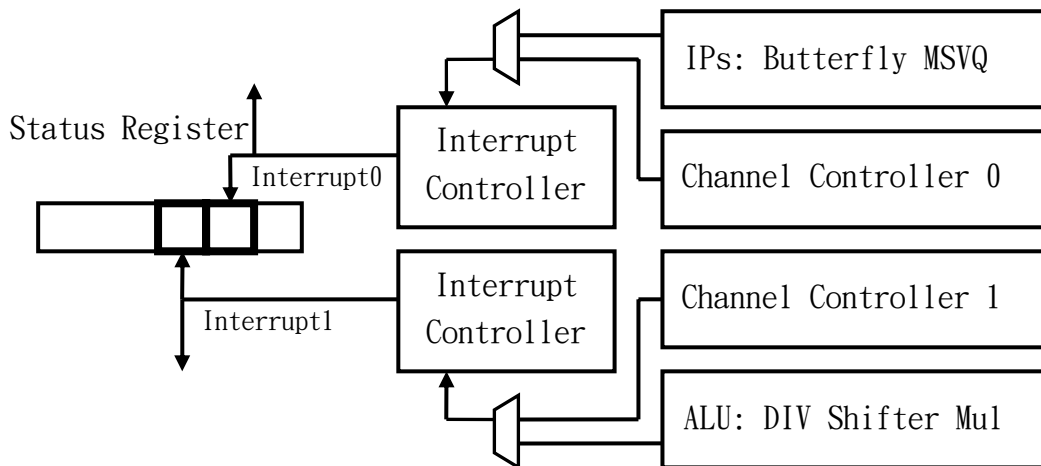


圖 2-7：中斷控制器連接狀態暫存器

2.3.5 算數運算邏輯單元 (ALU)

為了達到具有 DSP 的運算能力，在智慧型 DMA 內建算數運算邏輯單元(ALU)。它與處理器 ALU 的區別是智慧型 DMA 裡的算數運算邏輯單元為幫助處理器計算多個指令週期的運算，與和記憶體直接存取有關的指令。ALU 包含了四級除法器 (4-stage DIV)、柱形移位器 (Barrel Shifter) 及乘加器與乘法器 (MAC and Multiplier) 的設計。下面介紹 ALU 裡各個運算單元，與在運算中所扮演的角色。

ALU 擁有兩顆乘法器，與一個累加器。兩顆乘法器可以針對演算法應用，設計相對應的資料路徑，有效率的處理各種特殊運算。如圖 2-8 所示，加權最小均方誤、與蝴蝶運算裡的複數乘法、及 32-bit 乘法都是需要多個乘法的運算，使用兩個乘法器可以縮短運算時間。

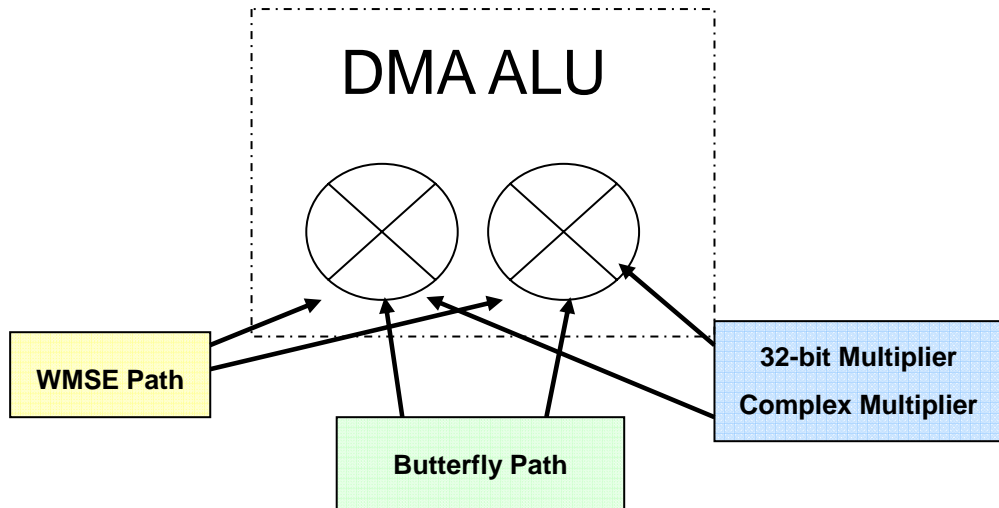


圖 2-8：乘法器的使用情形

另一種運算模式為乘加器的使用，配合前面提到的定址法說明，可處理大量繁複的數位訊號處理。乘法器為 16×16 bits 的有號數 (Signed) 快速乘法器，輸出為 32-bit 的有號數。累加器寬度為 40-bit，以確保在加法過程中不會因溢位導致結果錯誤。若溢位發生，錯誤會顯示在狀態暫存器內，並發出中斷通知處理器。圖 2-9 表示 ALU 裡各個運算單元。

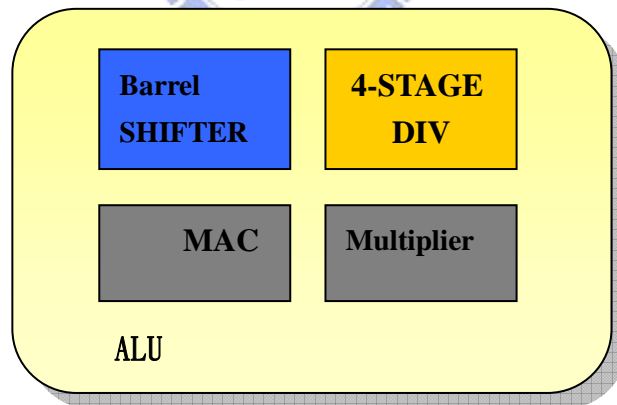


圖 2-9：ALU 單元

2.3.6 多級向量量化單元 (Multi-stage Vector Quantization)

智慧型 DMA 配合內建的 ALU，設計了兩種種資料路徑：分別為多級向量量化路徑與蝴蝶運算路徑。多級向量量化單元 (Multi-stage Vector Quantization) 可以針對各種影像與語音壓縮編碼進行處理，並可由使用者設定參數，適合各種

量化要求的向量量化編碼。

多級向量量化單元的核心為一個四級的運算管線，實現權重失真量測 (Weighted Distortion Measure) 的運算，運算中使用到 ALU 裡的兩顆 16x16 bits 的有號數 (Signed) 快速乘法器，實現乘上權重 (Weight) 與平方 (Square) 的運算，最後由一 40-bit 的累加器，計算失真結果。單元內的其他功能方塊為暫存器組 (Register Bank)、比較器 (Comparator)、數值表單元 (Write Table)、最佳 M_Path 輸出單元 (Select M_Path)，與流程控制單元 (Address Generator Unit)。流程控制單元與數值表單元分別與外部記憶體 (編碼簿) 以及內部記憶體 (數值表) 作溝通。當運算結束後發中斷給處理器取值。圖 2-10 為多級向量量化運算管線與 ALU 使用情形。

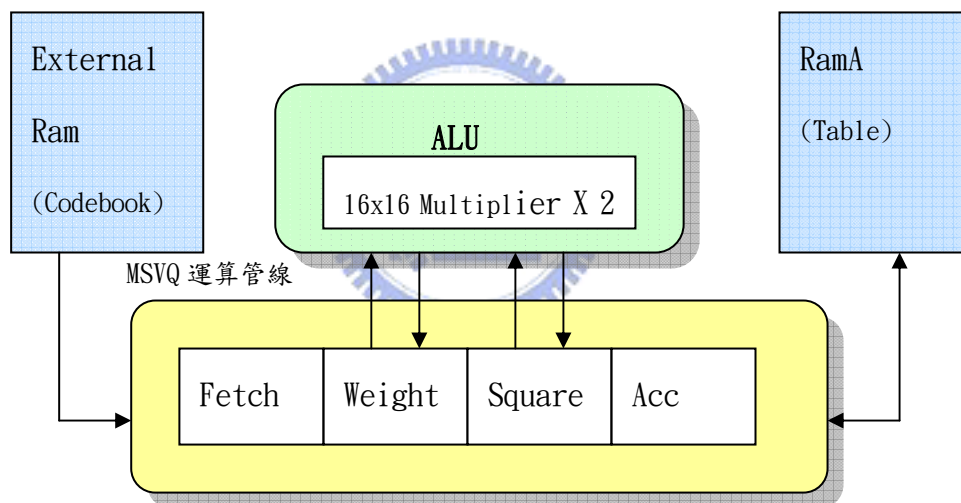


圖 2-10：運算管線與 ALU 使用情形

2.3.7 蝴蝶運算單元 (Butterfly Unit)

另一個智慧型 DMA 的資料運算路徑為蝴蝶運算單元 (Butterfly Unit)。蝴蝶運算單元可取代快速複立葉轉換 (Fast Fourier Theorem) 中，規律且繁複的蝴蝶運算，有效的對許多常用到快速複立葉轉換的多媒體運算作加速。

蝴蝶運算單元的核心是複數乘法，利用 ALU 裡的兩顆乘法器可以在兩個指令週期下完成運算；此外為了定點化程式的使用性，可由使用者設定複數乘法後

的 Scaling 位移量，有效的處理資料範圍的問題。蝴蝶運算單元的輸入由兩顆內部記憶體傳輸，內部記憶體 A 為輸入的資料，內部記憶體 B 為暫存正弦與餘弦函數的係數，當運算完後將結果存回記憶體 A 準備給下一級做運算，並發中斷給處理器。圖 2-11 為蝴蝶運算單元與周邊的溝通。

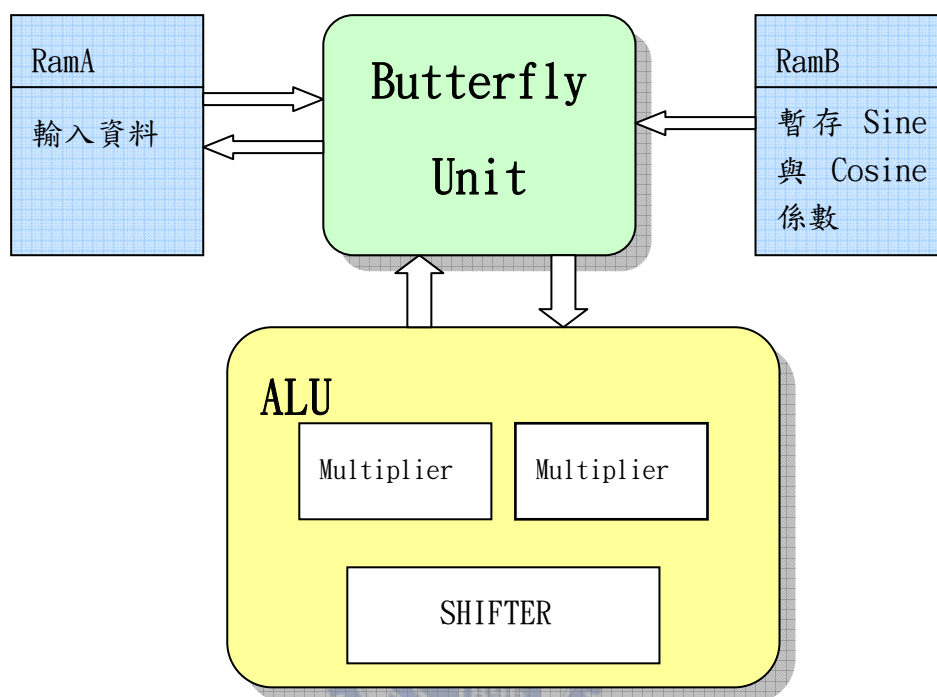


圖 2-11：蝴蝶運算單元與周邊的溝通

2.3.8 記憶體選擇器 (Memory Mux)

智慧型 DMA 直接支援兩個 2KB (512×32-bit) 的單埠高速同步記憶體，速度可達 500MHz，每個記憶體包含 9 位元位址線，32 位元資料輸入/輸出線及讀寫控制線，記憶體介面與內部記憶體資料傳輸連接方式如圖 2-12，記憶體讀取/寫入使用相同的記憶體介面。記憶體多工器由控制暫存器的設定，觸發工作的模式；當控制暫存器設定為通道致能時，智慧型 DMA 操作在傳輸模式或是雙通道運算，記憶體掌控權交付給通道控制器；當控制暫存器設定通道除能時，智慧型 DMA 操作在運算模式，記憶體直接由算術運算邏輯或 IP 來溝通，如圖 2-12 所示。

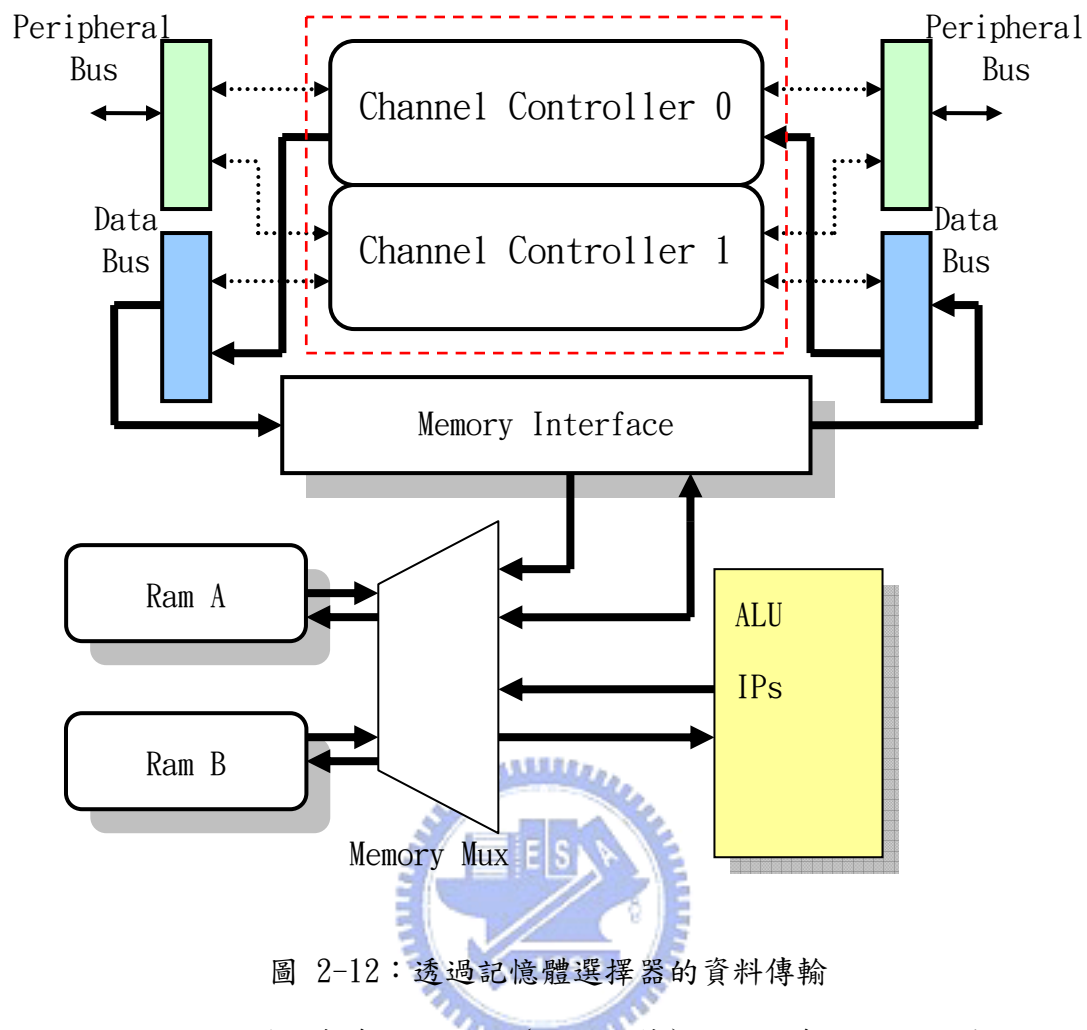


圖 2-12：透過記憶體選擇器的資料傳輸

通道控制器可透過暫存器的設定（見 2.2 節）選擇要存取的記憶體。因此，記憶體介面必須符合圖 2-13(a)(b)的存取時序。

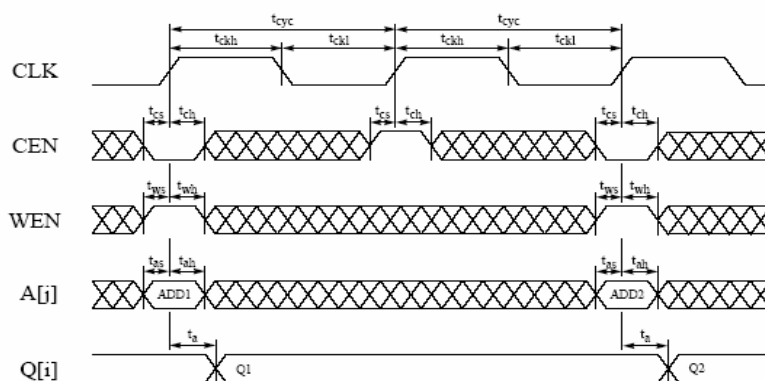


圖 2-13(a)：記憶體讀取時序圖

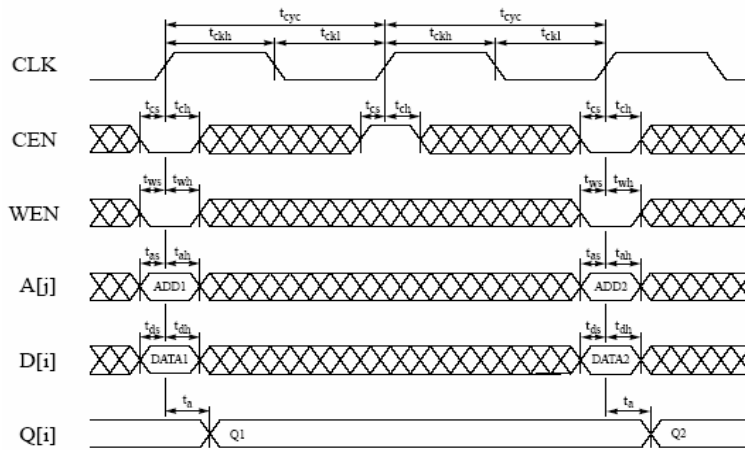


圖 2-13(b)：記憶體寫出時序圖

2.3.9 周邊匯流排 (APB - Advanced Peripheral Bus)

APB 匯流排為 ARM 的 AMBA (Advanced Microcontroller Bus Architecture) [17] 架構下的一部分，其受控端介面訊號如圖 2-14 所示，並有簡單易用、省功率的好處，主要應用在資料量不大，低頻寬的周邊裝置。內建一組音源介面 IC Sound Bus (I²S)、與外部記憶體介面，最高支援八組周邊裝置，可擴充符合 APB 介面的周邊 IP，延伸處理器的應用範圍。

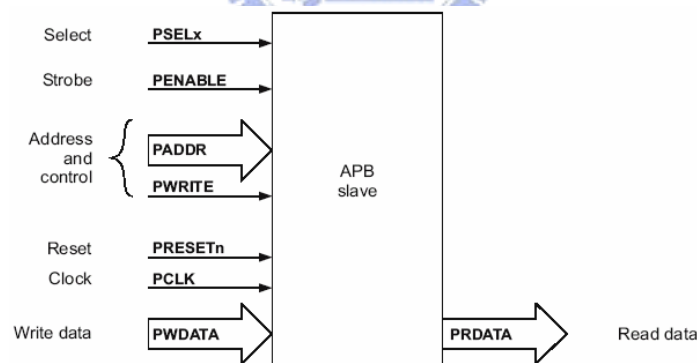


圖 2-14：APB 匯流排受控端介面

APB 匯流排時序如圖 2-15，一筆資料傳輸需要二個時脈周期，通道控制器直接支援 APB 匯流排介面的訊號。

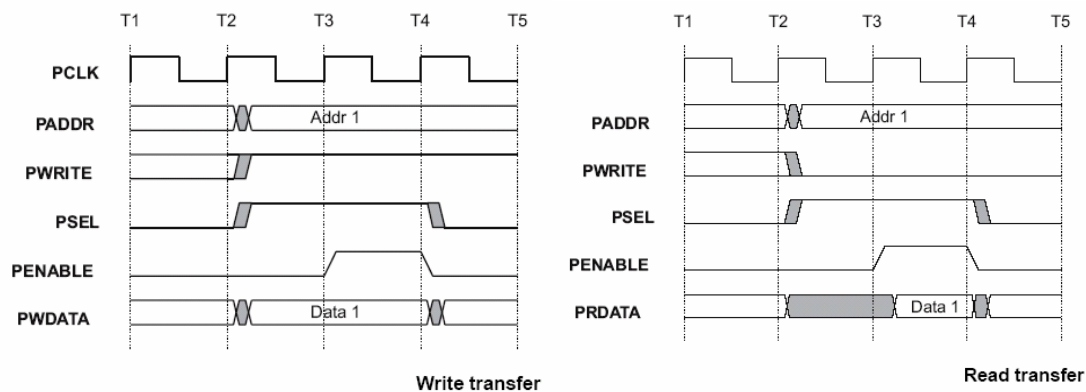


圖 2-15(a)：APB 匯流排寫入時序圖 圖 2-15(b)：APB 匯流排讀取時序圖

2.4 智慧型 DMA 控制暫存器

為了發揮智慧型 DMA 最大的效能，使用者必須設定其控制暫存器以達成傳輸或運算功能，以下分別說明控制暫存器群組內各暫存器的功能與設定方式。

2.4.1 來源暫存器 (Source Register)

來源暫存器內部為 32-bit，對外為兩組 16-bit 可讀寫的暫存器。只要設定來源暫存器後，其值會被保留，但來源位址會隨著實際的讀取位址而改變。來源暫存器中高 16 位元用做特殊定址，低 16 位元用做資料定址，如表 2-2，

表 2-2：來源暫存器 (Source Register)

Bits	High Source Register	
15	Source Circular	0:circular / 1:mirror
14:7	Source Base	遞增 (或遞減) 位址數
6:0	Source Offset	區塊起始位置
Bits	Low Source Register	
15	Source Device	來源裝置 0:Ram A / 1:Ram B
14:0	Source Address	來源位址

2.4.2 目的暫存器 (Destination Register)

設定功能與來源暫存器相同，目的暫存器用在寫出的目標上。目的暫存器經過設定，會保持設定過的值，但目的位址會隨著實際的寫出位址而改變。暫存器位元排列如表 2-3 所示。

表 2-3：目的暫存器 (Destination Register)

Bits	High Destination Register	
15	Destination Circular	0:circular / 1:mirror
14:7	Destination Base	遞增 (或遞減) 位址數
6:0	Destination Offset	區塊起始位置
Bits	Low Destination Register	
15	Destination Device	目的裝置 0:Ram A / 1:Ram B
14:0	Destination Address	目的位址

2.4.3 控制暫存器 (Control Register)

控制暫存器對外為兩組 16-bit 可讀寫的暫存器，只要設定控制暫存器，其值會被保留，但傳輸資料數會隨著實際的寫出數目而減小。控制暫存器的高 16 位元部分為來源/目的區塊大小，低 16 位元為 DMA 控制，如表 2-4。

表 2-4：控制暫存器 (Control Register)

Bits	Source/Destination Region Register	
15:8	Source Region	來源區塊大小
7:0	Destination Region	目的區塊大小
Bits	Control Register	
15	Source Increase	來源位址遞增
14	Source Decrease	來源位址遞減
13	Destination Increase	目的位址遞增
12	Destination Decrease	目的位址遞減
11	Source Width	來源寬度 (Reserved) Default =32 bit
10	Destination Width	目的寬度 (Reserved) Default =32 bit
9:0	Transfer Size	傳輸資料數

2.4.4 配置暫存器 (Configuration Register)

配置暫存器對外為一組 16-bit 可讀寫的暫存器，只要設定配置暫存器，其值會被保留，但累加器清除 (High-Active) 會在設定後一個時脈周期改變成 Low；通道致能後，會在傳輸結束後，自動關閉。如表 2-5，其中 Function 為設定智慧型 DMA 的運算功能，設定如下：

- 001:MAC: 雙通道乘加模式。
- 010:Shifter: 柱狀位移模式。
- 011:DIV: 多級除法器模式。
- 100:FFT: 蝴蝶運算單元模式。
- 101:32-bit Multiplier: 32 位元乘法模式。
- 110:MSVQ: 多級向量量化單元模式

表 2-5：配置暫存器 (Configuration Register)

Bits	Configuration Register	
15	Halt	暫停
14	Interrupt Enable	中斷致能
13:11	Source Peripheral	來源周邊 (記憶體傳輸模式時無效)
10:8	Destination Peripheral	目的周邊 (記憶體傳輸模式時無效)
7:6	Transfer Type	傳輸模式
5	Sequence Transfer	連續傳輸
4:2	Function	特殊功能
1	ACC Clear	累加器清除
0	Channel Enable	通道致能

2.4.5 狀態暫存器 (Status Register)

狀態暫存器對外為一組 16-bit 僅能讀取的暫存器，包含兩個通道的資訊，高位元為通道 1，低位元為通道 0。如表 2-6。

表 2-6：狀態暫存器 (Status Register)

Bits	Status Register	[15:8] Channel 1 / [7:0] for Channel 0
7	Interrupt	中斷
6	FIFO Full	DMA FIFO 資料狀態已滿
5	FIFO Empty	DMA FIFO 內無資料
4	FIFO Half	DMA FIFO 資料狀態已達半數
3	Channel Select	Channel 被選擇在工作模式
2:0	Error	錯誤狀態 (累加器溢位發生)

2.4.6 輸出暫存器 (Output Register)

智慧型 DMA 裡的 IP 如 MSVQ 單元以及 ALU 單元裡計算後的輸出，都存放在輸出暫存器中；當進行雙通道的乘加運算時，輸出暫存器為一組 32-bit 可讀寫累加器，僅能以 16-bit 寫入，以 32-bit 有號數讀出，內部為 40-bit 的暫存器。如表 2-7 所示。

表 2-7：輸出暫存器 (Output Register)

31:0	DSPRegOut	SDMA 輸出
------	-----------	---------

2.4.6 功能暫存器 (Function Register)

功能暫存器對外為四組 16-bit 寫入的暫存器，負責設定智慧型 DMA 裡的運算功能：當智慧型 DMA 進行 ALU 單元的運算時，功能暫存器為運算的輸入數值；當智慧型 DMA 進行 MSVQ 單元運算時，功能暫存器為設定參數的輸入資訊；當智慧型 DMA 進行 Butterfly 單元運算時，功能暫存器為記憶體輸入位址，以及位移量的設定。如表 2-8 所示。

表 2-8：功能暫存器 (Function Register)

Bits	Function Register	
15:0	DSPIn1	<p>The diagram shows a block labeled 'Function Register' containing four 16-bit input registers: DSPIn1, DSPIn2, IPIn1, and IPIn2. These registers are connected to a central block labeled 'ALU' and 'IPs'. The output of this block is connected to a register labeled 'DSPReg Out'.</p>
15:0	DSPIn2	
15:0	IPIn1	
15:0	IPIn2	

2.4.7 智慧型 DMA 使用流程

智慧型 DMA 的設定方式如圖 2-16，將來源暫存器、目的暫存器、控制暫存

器及功能暫存器設定後，配置暫存器必須最後一步設定，用以將通道致能。此時可去處理其他工作，待中斷發生後，智慧型 DMA 處理的工作跟著結束。在設定暫存器後，其值多會被保留下來，因此，若每次設定僅需選擇有改變的暫存器設定即可。

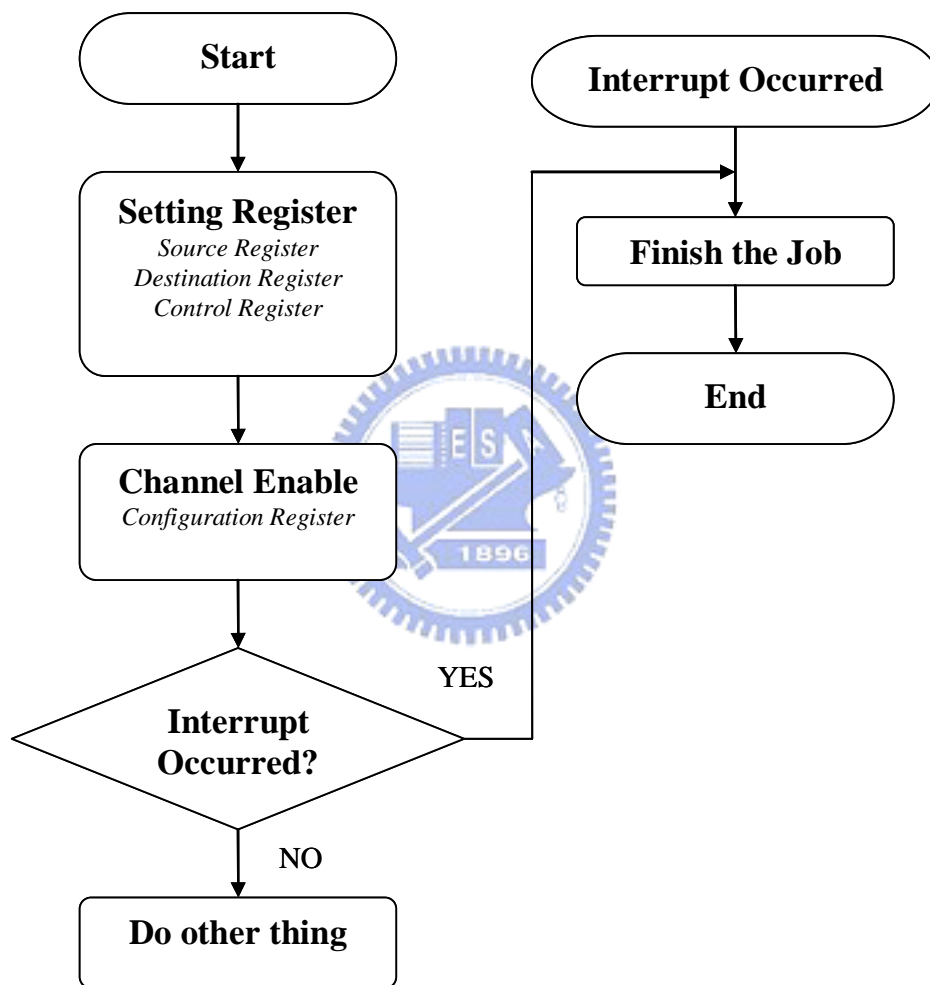


圖 2-16：智慧型 DMA 的使用流程

第三章 智慧型 DMA 運算控制器設計

本章介紹智慧型 DMA 與其他 DMA 相異的特色，利用內部的算術運算邏輯單元 (ALU)，搭配各種傳輸定址模式與資料路徑，達到具有數位訊號處理 (DSP) 的效能。以下將介紹智慧型 DMA 的特殊運算功能，包括多級向量量化資料路徑設計及蝴蝶運算資料路徑設計。

3.1 多級向量量化器設計

本節介紹多級向量量化的設計，包含多級向量量化理論、M-L 搜尋法及設計架構及硬體使用方法。

3.1.1 原理

多級向量量化器 (Multistage Vector Quantization) 在 1982 年由 Juang 與 Gray 提出 [18]。每個向量 x 先由一個粗略的向量量化器量化 $\hat{x}_1 = Q[x]$ ，量化的誤差即為 $e_1 = x - \hat{x}_1$ 。接著量化的誤差 e_1 由另一個向量量化器量化 $\hat{e}_1 = Q[e_1]$ ，重複上述的過程。根據使用幾級的向量量化器，每一級皆產生出一個編碼簿的索引值，這些索引值即為量化後的結果。在解碼端，解碼器利用這些索引值，在碼簿中找出相對應的向量，即可得到重建後的訊號。圖 3-1 表示 3 級向量量化的編解碼過程。

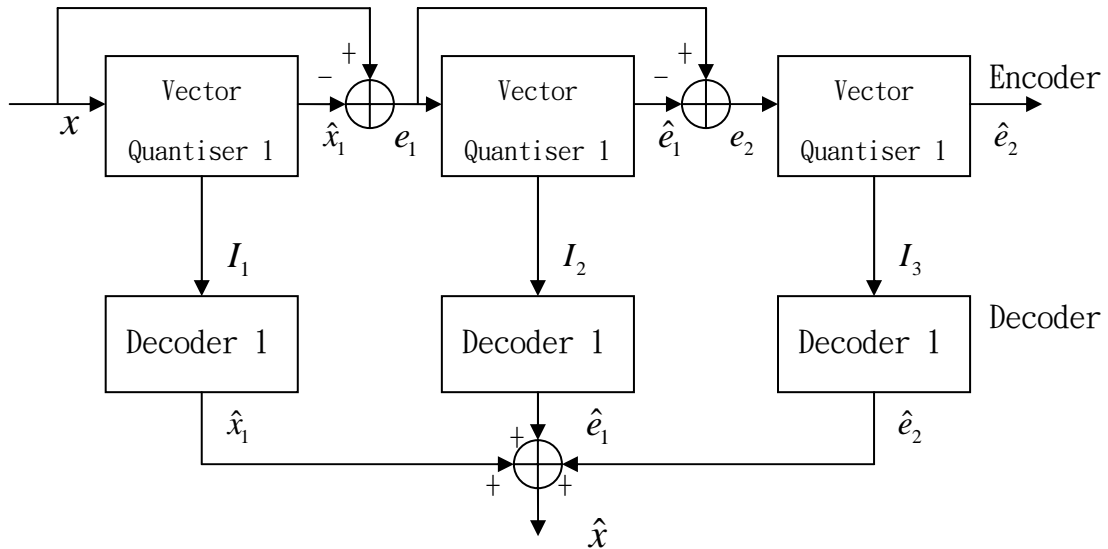


圖 3-1：三級向量量化器 (I_i 定義為第 i 個量化器 index) [18]

M-L 搜尋法多級向量量化器，也稱為樹狀搜尋多級向量量化器 (tree-searched multistage vector quantiser)，在 1993 由 LeBlanc 提出[19]。相較於傳統循序的收尋，他使用更佳的搜尋演算法。在 MSVQ 的第一級量化器，選擇擁有最小失真測量的 M 個碼向量，將 M 個碼向量與原輸入係數的差值即為第一級量化器的量化誤差，這 M 個碼向量誤差將傳送至第二級的量化器，即產生出 M 條量化路徑。在量化器的最後一級，這 M 條路徑將得到 M 個最後的量化誤差值，此時最佳的量化路徑即為誤差最小者。圖 3-2 與圖 3-3 表示 M-L 搜尋過程及流程圖。

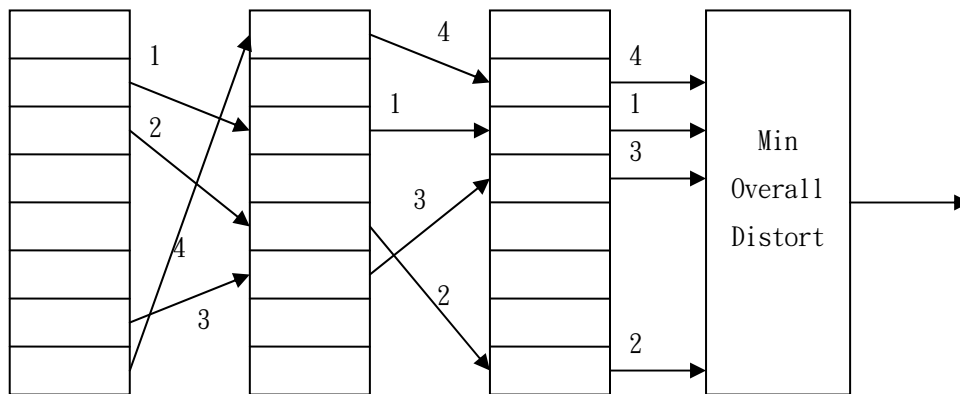


圖 3-2：三級向量量化器利用 M-L 法搜尋編碼簿 (這裡 $M=4$)

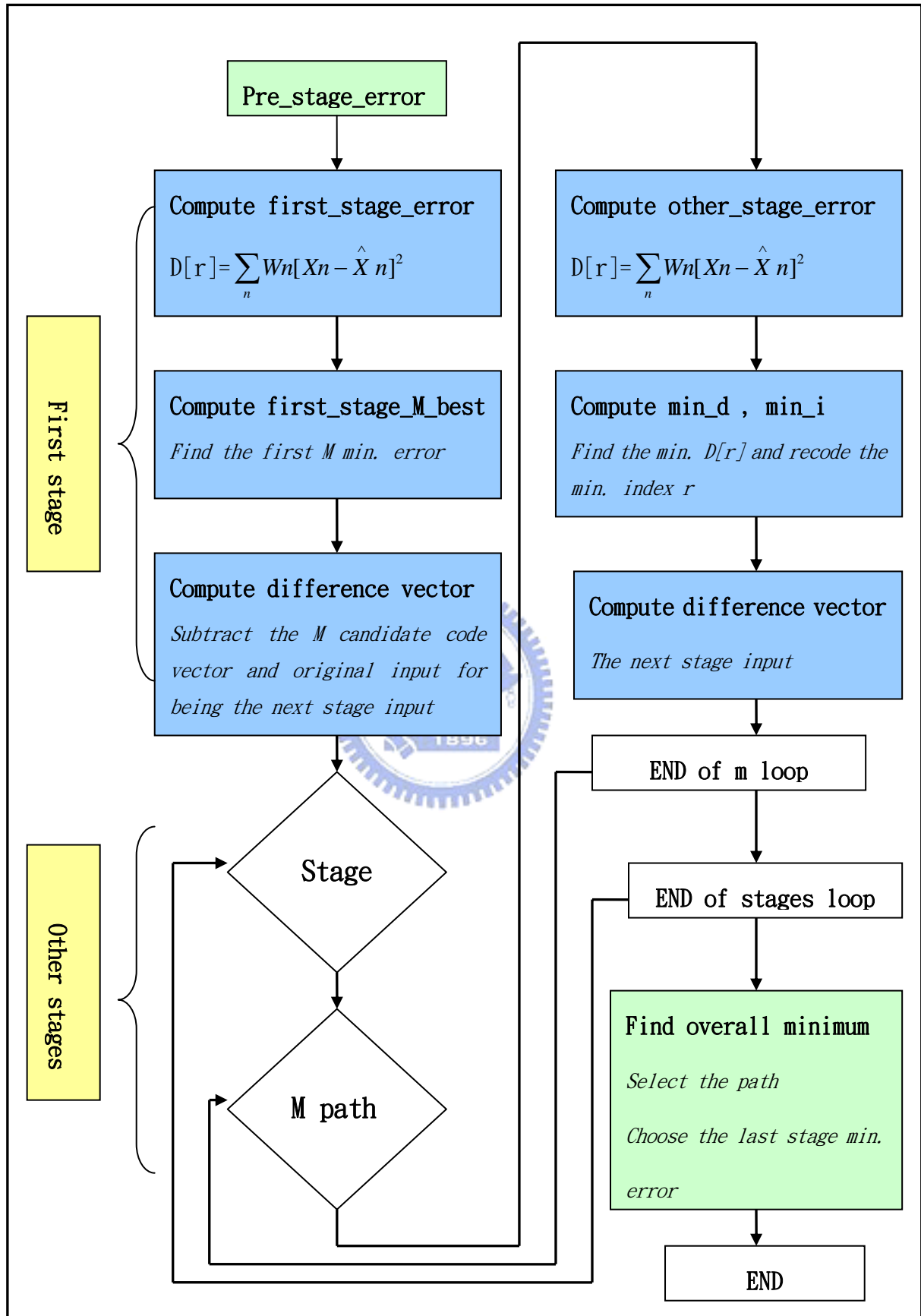


圖 3-3：多級量量量化使用 M-L 搜尋流程圖

3.1.2 設計架構

圖 3-4 為多級向量量化單元，架構圖包含流程控制器、運算管線、比較器、控制輸入暫存器、數值表單元、最佳路徑選擇單元，其說明如下：

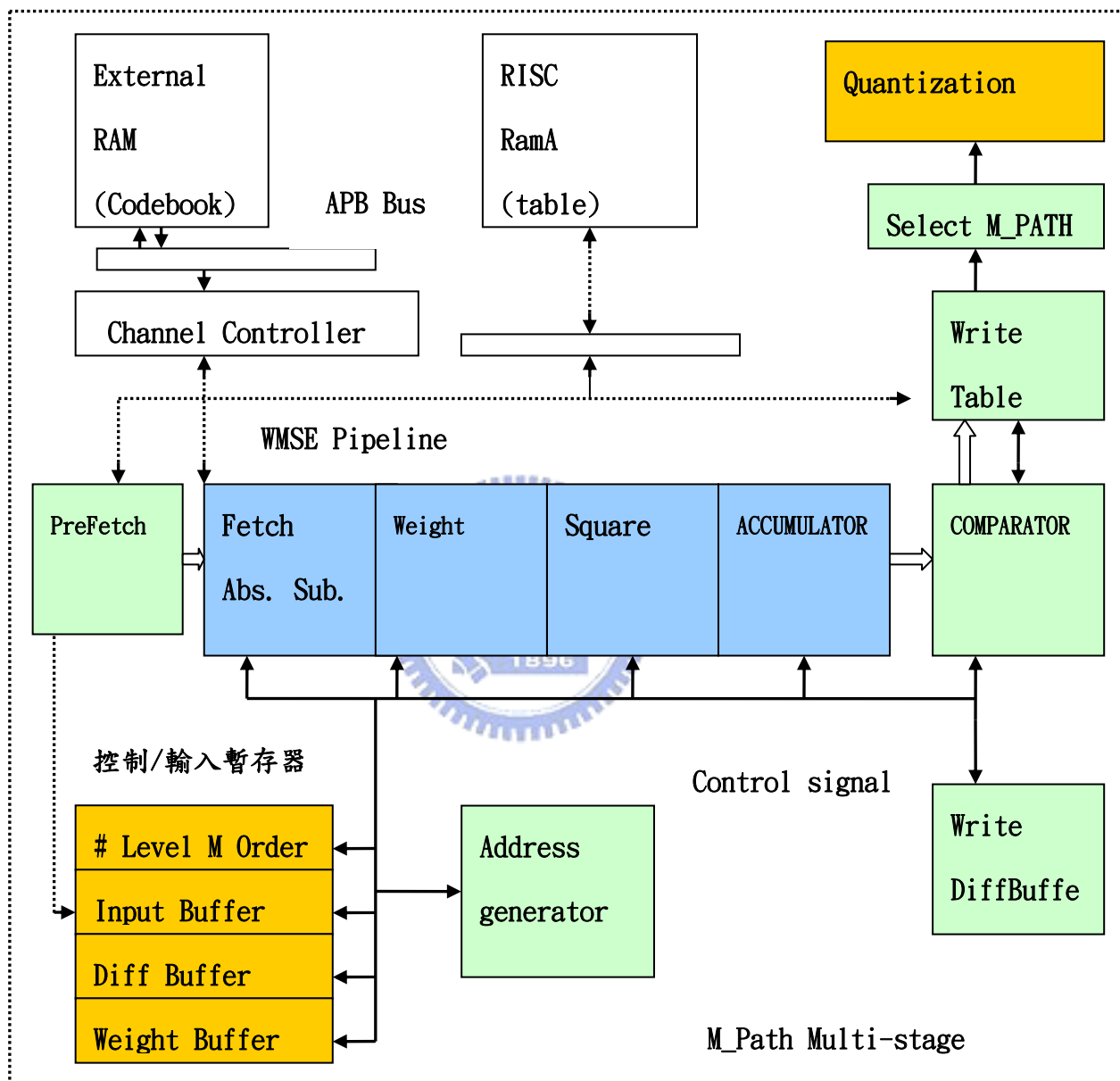


圖 3-4：多級向量量化器架構圖

圖 3-5 表示多級向量量化器流程圖，使用者在控制暫存器設定量化環境後，由流程控制器解碼，再進行位址的發配。流程控制器控制 4 個迴圈的流程，內層迴圈實現加權均方誤差(WMSE)；Level 迴圈循序抓取編碼簿的碼向量進行運算；Stage 迴圈抓取下一級的編碼簿；Path 迴圈進行 M-L 搜尋的實現。比較器選擇

WMSE 最小者，將誤差存到記憶體中，當成下一級的輸入。數值表單元負責暫存資料，由圖可知，他控制下一級輸入與量化路徑表的暫存，暫存到記憶體中，以減少暫存器的使用。最後由最佳路徑選擇單元選出誤差最小的路徑，交由數值表單元輸出。

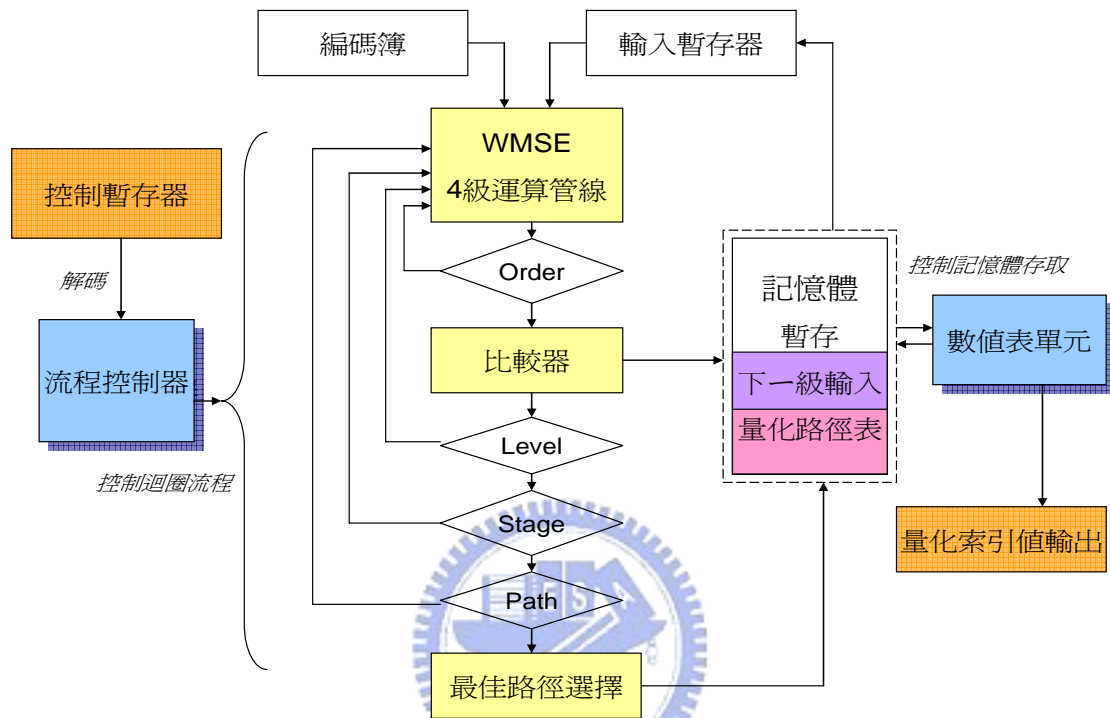


圖 3-5：多級向量量化器流程圖

3.1.3 控制輸入暫存器

控制輸入暫存器為 4 組 16-bit 暫存器組，負責設定系統的參數與待處理的資料。各暫存器的內容如表 3-1 所示，Next_Stage Buffer 用在多級編碼簿的情形，代表下一級的輸入；Weight Buffer 則用在加權失真量測上(weighted mean squared error)，若系統為 MSE(mean squared error)，則 Weight Buffer 預設值為 1。

表 3-1：控制輸入暫存器

控制參數	
Control Register	使用者自定 Order 數、Level 數、M Path 數
輸入暫存器	
Input Buffer	向量量化係數
Next_Stage Buffer	下一級的向量量化係數
Weight Buffer	權重係數

3.1.4 流程控制單元(Address Generator Unit)

流程控制單元為一個位址產生器(Address Generator)，負責控制系統的各個運算流程，由一個 FSM(Finite State Machine)實現，共有 8 個狀態，狀態圖如 3-6。各個狀態的工作說明在表 3-2。

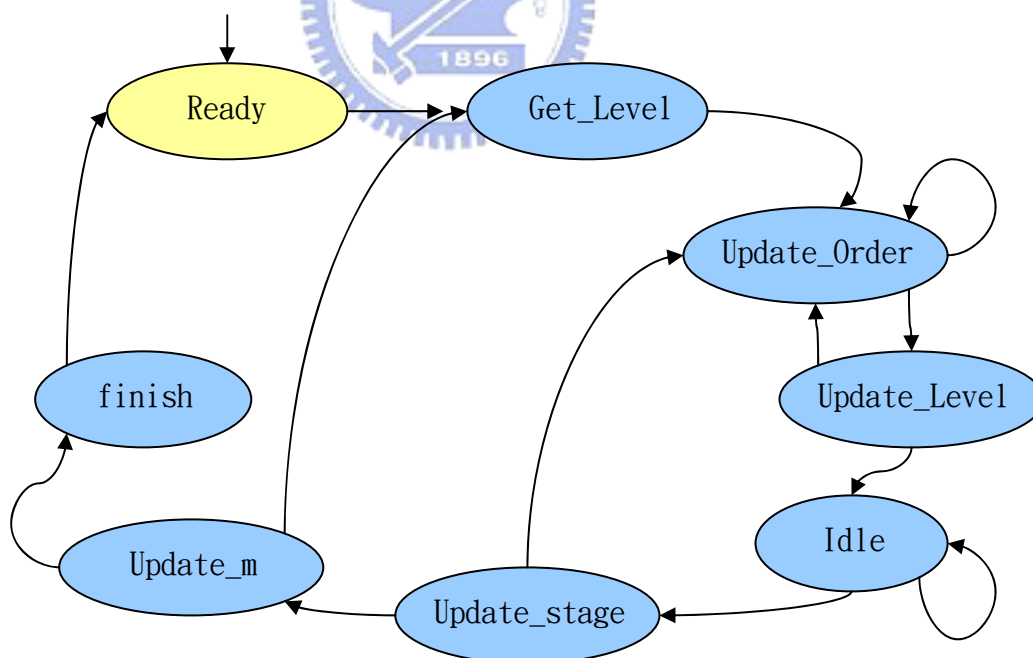


圖 3-6：流程控制單元狀態圖

表 3-2：控制輸入暫存器

狀態名稱	工作內容
Get_Level	對控制暫存器解碼，得到 Level 數的資訊
Update_Order	產生 Oder 位址
Update_Level	產生 Level 位址
Idle	等待忙碌訊號，數值表單元正在記錄
Update_stage	產生 Stage 位址
Update_M	產生 M paths 位址
Finish	送出完成訊號，結束工作

流程控制單元有 4 個計數器，依據控制暫存器設定的 4 個參數值：Order、Level、Stage 及 M，決定 4 個迴圈的次數，當最內層迴圈 Update_Order 的計數器數到 Order 次時，狀態就會跳至 Update_Level，將 Update_Level 狀態裡的計數器加一，以此類推至 Update_Stage 與 Update_M 狀態。Idle 狀態發生在換下一級 Update_Stage 時，數值表單元會記錄上一級的各種資訊及下一級的輸入來源，此時流程控制單元進入 Idle 狀態等待數入資料準備完成。

為了掌控各單元間的操作，流程控制單元共產生 6 組控制訊號，分別控制與驅動各個功能方塊，6 組控制訊號的內容如表 3-3 所示。

表 3-3：控制訊號

控制訊號	
Enable_acc	累加器開始累加
Enable_compare	驅動比較器
Enable_recode	開始紀錄數值表
Enable_final	趨動比較器選擇最佳 M_Path
Enable_output	輸出最後的索引值
Enable_m	進行下一條 Path，紀錄數值

3.1.5 運算管線設計(Pipeline design)

多級向量量化單元的核心為一個四級運算管線(4-stage pipeline)的設計，為實現 WMSE(weighted mean squared error)計算，每一級在一個 clock 週期進行一個運算，並將結果儲存在暫存器中準備給下一級使用，資料及控制訊號經由管線進行同步傳送。利用管線同步運算的架構，以及每級運算時間平均的特性，多級向量量化單元可達到較高的資料處理能力，相對於在相同時脈下以循序電路實現。

運算管線運作與暫存器傳送時間圖如圖 3-7 所示。

Time	0	1	2	3	4
Fetch	$ A_1 - B_1 $	$ A_2 - B_2 $	$ A_3 - B_3 $	$ A_4 - B_4 $	$ A_5 - B_5 $
Weight	-	$w_1 A_1 - B_1 $	$w_2 A_2 - B_2 $	$w_3 A_3 - B_3 $	$w_4 A_4 - B_4 $
Square	-	-	$W_1 A_1 - B_1 $	$W_2 A_2 - B_2 $	$W_3 A_3 - B_3 $
Acc	-	-	-	$\sum_{i=1}^1 W_i A_i - B_i $	$\sum_{i=1}^2 W_i A_i - B_i $

註： $W = w^2$ A 與 B 分別為輸入向量與碼向量

圖 3-7：運算管線時間圖

Fetch 級運算兩個輸入來源的絕對值差；Weight 級運算差值的權重，這裡與演算法不同的是，在 Square 級前先做了權重的運算，這是因為考量到數值的特性，當 16-bit 絕對值差直接傳送到 Square 級運算，輸出須以 32-bit 儲存及傳送給 Weight 級做乘法，如此則需一個 32-bit 乘法器或是多浪費一級作 32-bit 的乘法。為減少設計面積及考量到權重的係數特性，即使經過定點化後仍很小，各係數可正歸化在 1 到 2 間，誤差最小，所以先乘上權重並傳送其號數(Sign)至 Acc 級，再經過平方，這樣只需做 16-bit 的乘法動作。得到平方值後，依權重係數的正負號，進行累加或是累減運算，輸出為 40-bit 的 WMSE 值。

將 Weight 提前(Weight modify)所造成的 WMSE 結果與未修改前(Original Input)的比較如表 3-4，誤差值在 5%之內，與 SNR 的影響很小。

表 3-4：架構修改前後誤差表

	SNR(dB)
Original Input	57.243
Weight modify	56.335

3.1.6 比較器(Comparator)

當運算管線將 WMSE 數值計算完畢後，經由比較器選擇最小的誤差值，比較器的設計為有號數(Sign)的比較，另外為了配合 M Paths 的演算法，這裡設計了一個 threshold 暫存上一條 Path 的誤差值，下一條 Path 則需大於 threshold 下進行比較。

3.1.7 數值表單元(Write Table)

數值表單元記錄各種數值暫存的路徑，由一個 FSM(Finite State Machine)實現。其狀態流程圖如 3-8 所示，狀態工作內容如表 3-5 所示。

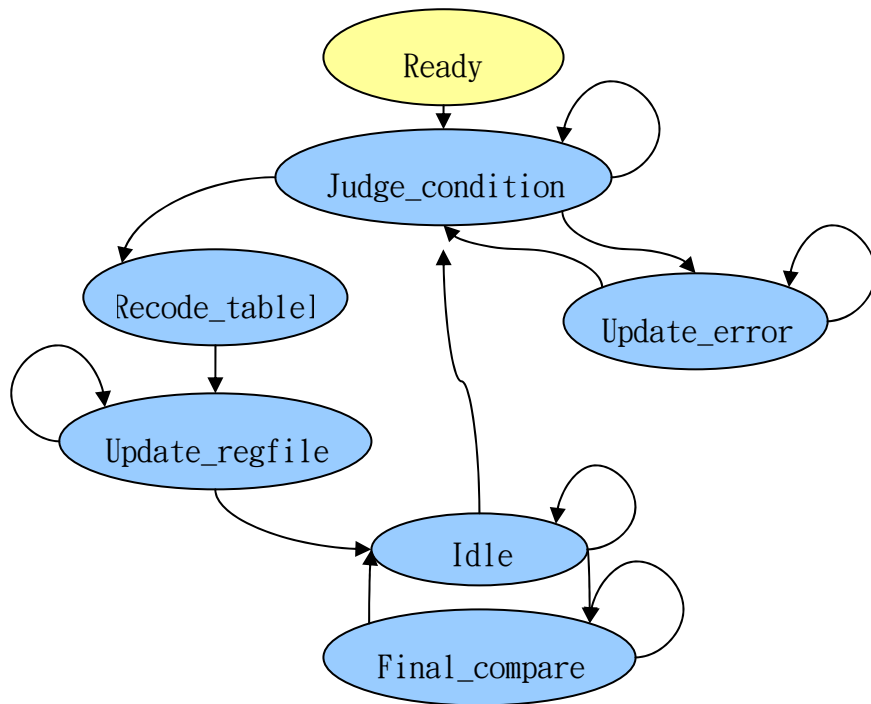


圖 3-8：數值表單元狀態圖

狀態名稱	工作內容
<i>Ready</i>	初始值設定
<i>judge_condition</i>	判斷該進入哪個 stage
<i>update_error</i>	更新誤差值
<i>recode_table(index)</i>	紀錄各 stage 的索引值
<i>update_regfile</i>	將最小 WMSE 的誤差當作下一級的輸入
<i>idle</i>	忙碌狀態
<i>Final_compare</i>	紀錄表完畢後，比較最後量化誤差大小，選擇一條最好的 PATH，再將他的各級索引值輸出

表 3-5：數值表單元狀態工作內容

為了增加多級向量量化的精確度，例如表 3-6 所示，假設 M paths 數為 3，Stage 數為 2，為了求最佳的量化路徑，可用一張表紀錄各個路徑的索引值(index)與最後一級的誤差值(error)，比較每個誤差值的大小，選出最小的誤差值就是最佳的路徑。假設 Error1 數值最小，可知最佳的量化路徑在 M1 上，此時輸出的

索引值即為 Index1、Index2。

	M1	M2	M3
S1	Index1	Index3	Index5
S2	Index2	Index4	Index6
Error	Error1	Error2	Error3

表 3-6：控制輸入暫存器

數值表單元即記錄各個 M Paths 的資訊，為了節省暫存器的使用，這裡直接將表的內容暫存到內部記憶體。數值表單元依據系統現在的狀態，寫入 M 個量化路徑的索引值，與上一級的誤差值。當系統跑完一個 Stage 時，數值表單元會將暫存在記憶體的誤差資訊取出，存在暫存器組裡當做下一個 Stage 的輸入。如圖 3-9 所示。

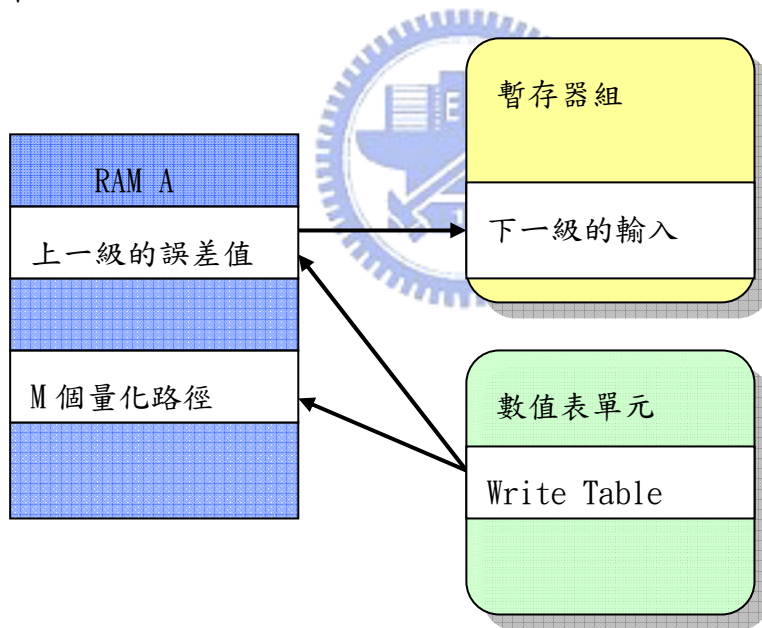


圖 3-9：數值表單元記錄下一級的輸入圖

當系統運算完畢後，由最佳路徑選擇單元(Select_M_Paths Unit)選出最佳的向量量化路徑，數值表單元便將這個路徑上的索引值包裝成 32-bit 的數值，輸出到智慧型 DMA 的暫存器組，由處理器取值，如圖 3-10 所示。

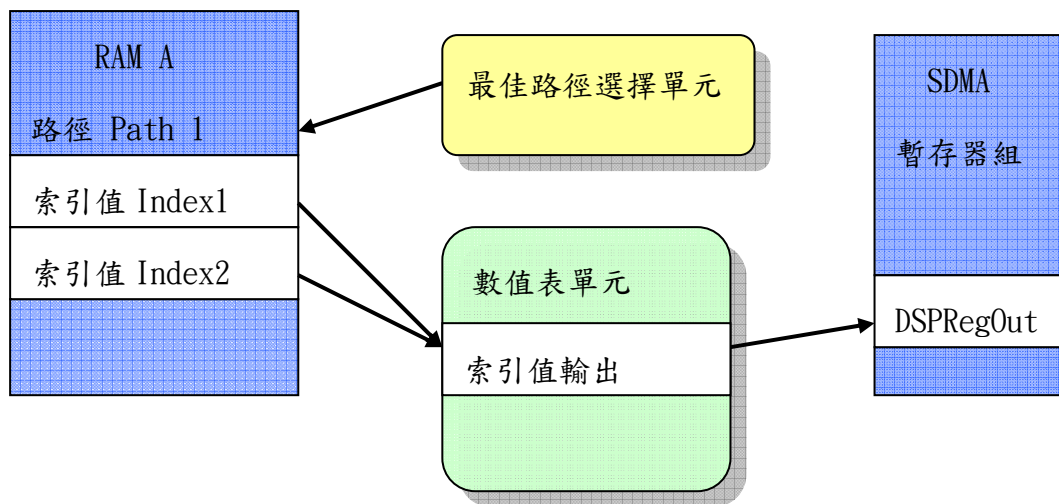


圖 3-10：數值表單元輸出圖

3.1.8 最佳路徑選擇單元(Select M_Paths)

將每條 Path 的最後一級誤差暫存，比較 M 組 Paths 的大小，輸出最小誤差的路徑。

3.1.9 運作流程

多級向量量化一個重要的步驟在於編碼簿的搜索，龐大的編碼簿不可能存放在內部記憶體中，所以需要一個外部記憶體的周邊。智慧型 DMA 有一般 DMA 溝通周邊的能力，對外的溝通介面為 APB 匯流排，傳輸符合標準協定。在編碼簿的設計上，由於 APB Bus 的傳輸方式為兩個時脈週期傳送一筆資料，為了符合多級向量量化單元中運算管線的機制，將兩筆碼向量存成 32 位元存放在一個記憶體位置上，經由 DMA 傳送至向量量化單元中，如此運算管線就可以每個時脈週期抓取一筆的碼作運算，如圖 3-11 所示。

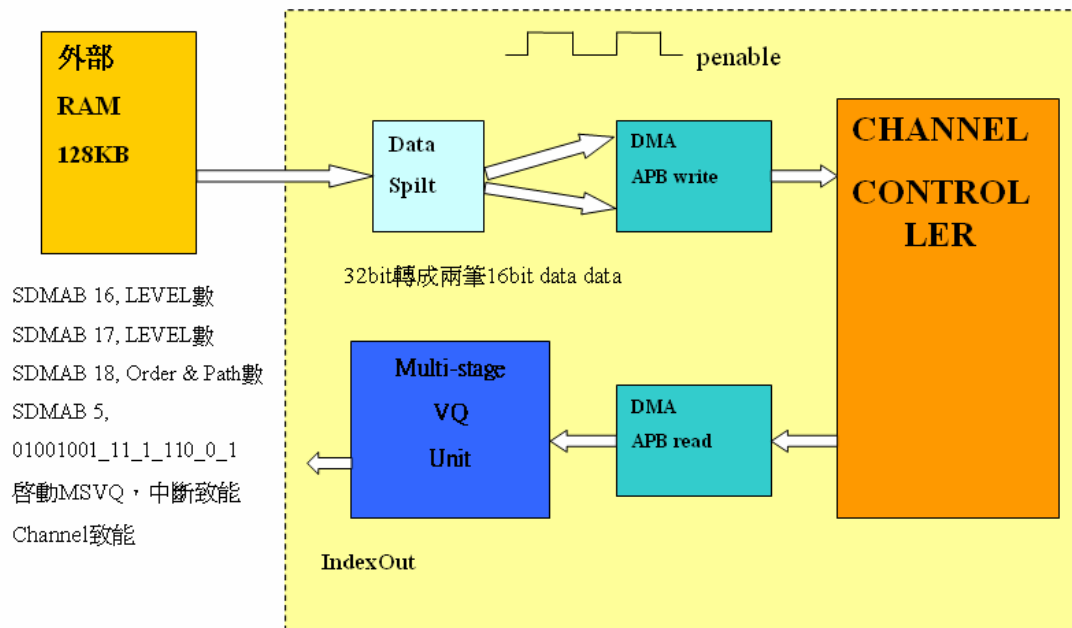


圖 3-11：多級向量量化器與周邊運作流程圖

在處理器的設定上，如圖 3-11 所示的程式碼，利用指令設定智慧型 DMA 的暫存器組，寫入 Level、Stage、Order 與 Paths 等參數，最後設定 MSVQ 功能致能參數，系統便開始運作，此時處理器不必理會智慧型 DMA 的運作，可以進行其他演算法的運算，當智慧型 DMA 運算完畢後，會發中斷給處理器取值。

3.2 蝴蝶運算器單元設計

本節將介紹蝴蝶運算器單元的設計。包括蝴蝶運算的理論、架構及其使用方法。

3.2.1 蝴蝶運算理論

快速複立葉轉換(Fast Fourier transform;FFT)在多媒體應用上非常常見，詳細的 FFT 理論詳見[16]，本章節介紹 FFT 理論的核心：蝴蝶運算。對於一個 8 點的時域分離的 FFT 形式(Decimation in Frequency)如圖 3-12 所示，共有 3 級，每級皆有 4 個蝴蝶的運算元，所以 8 點的 FFT 共有 12 個蝴蝶運算元，蝴蝶運算元形式如圖 3-13 所示，此時 $W_N^r = e^{-j2\pi r/N}$ 。

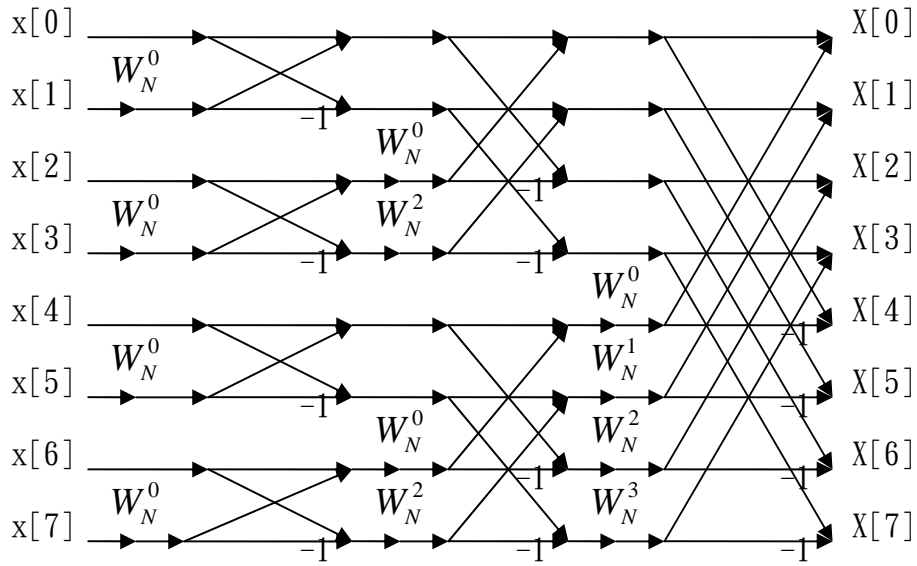


圖 3-12：8 點 FFT 圖

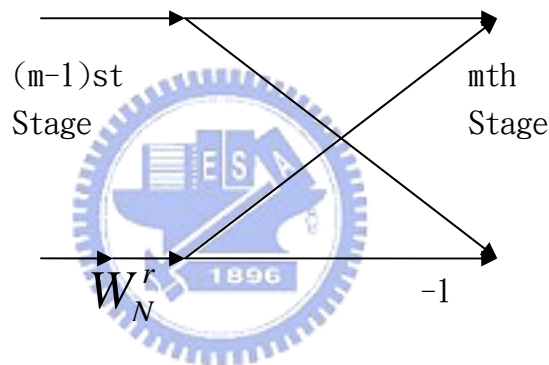


圖 3-13：蝴蝶運算圖

3.2.2 架構

設計一個蝴蝶運算的路徑，使用者只需輸入三個記憶體來源位址，代表蝴蝶運算的兩個輸入，與正弦餘弦的係數，設定完 DMA 的功能鍵後開始運算，此時處理器負責控制與掌管 FFT 的流程，由智慧型 DMA 負責內部迴圈的計算。

蝴蝶運算單元裡的資料型態為 32-bit，其中低 16 位元為實部，高 16 位元為虛部，這樣分配的優點在於節省記憶體空間，與加快存取的效率：對於一個 512 點的 FFT 運算，只需要 512 條記憶體位址線儲存複數的資料，一個時脈週期可以得到實部和虛部的數值，處理完後再存成一個 32 位元的封包，在一個時脈週期下存回記憶體。此外，為了符合定點化程式的需要，正弦餘弦係數常會做定

點格式的調整，在與輸入進行複數乘法完後，必須要將格式復原，捨去額外的資料。所以這裡設計用位移器來達到格式調整的目的，並可由使用者自行設定位移量，增加蝴蝶運算單元的使用性。蝴蝶運算單元流程如圖 3-14。

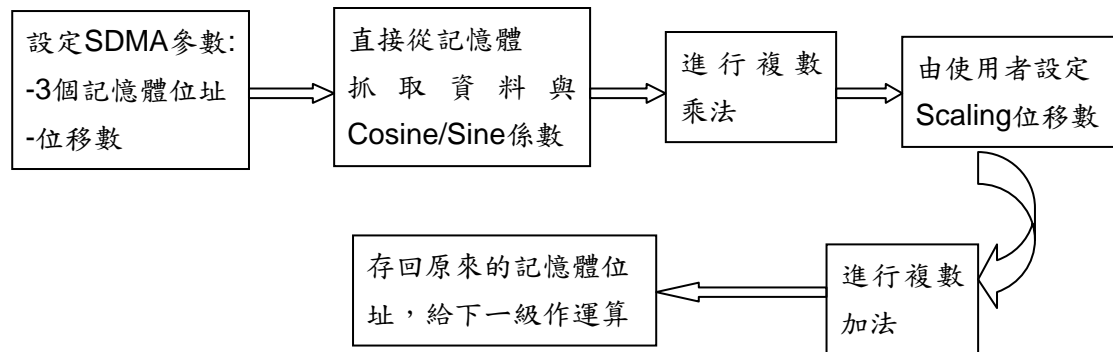


圖 3-14：多級向量量化器架構圖

蝴蝶運算單元的架構圖如圖 3-15 所示，進行一次的運算只需 9 個時脈週期。



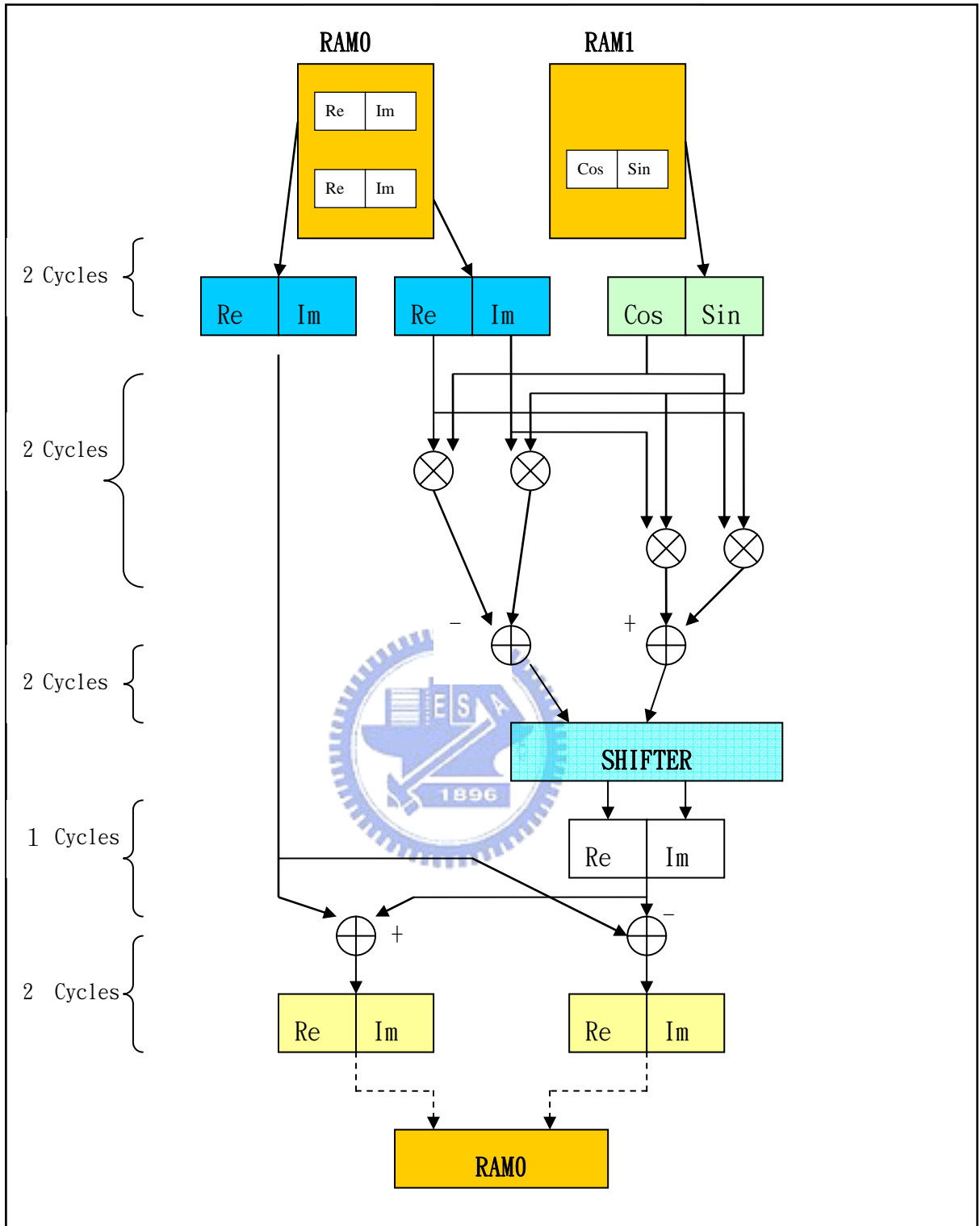


圖 3-15：蝴蝶運算單元架構

3.2.3 程式流程設定

蝴蝶運算單元在 FFT 運算中，由智慧型 DMA 操作，處理器負責流程的控管，如圖 3-16 及圖 3-17 所示，處理器負責控制三層迴圈，以及蝴蝶運算輸入位址的

計算，當蝴蝶運算在進行時，處理器開始計算下一筆的輸入位址以及計數器的運作，達到處理器與計算單元分工的效果，大幅提升效能。

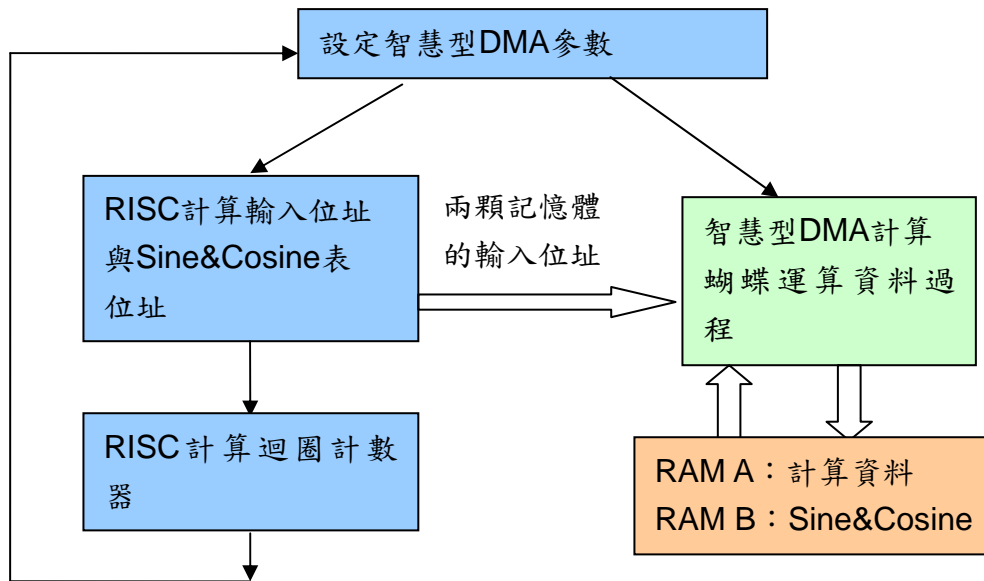


圖 3-16：FFT 運算分工情形

```

    LABEL:FFT                (主迴圈:計算 FFT 級數)
    LABEL:LOOPI              (次迴圈:一級裡有幾個蝴蝶運算)
    LABEL: Butterfly        (內迴圈:蝴蝶運算)
    SDMAR 16, R4              (蝴蝶 Input 1)
    SDMAR 17, R8              (蝴蝶 Input 2)
    SDMAR 18, R5              (SinCos 位址)
    SDMAB 19, R11             (設定位移量)
    SDMAB 5, 01000000_00_0_100_0_0 (蝴蝶單元致能)
    DMAOK                     (DMA 設定完畢)
    .....                    (計數器計算)
    .....                    (位址計算)
    JMBR R15, R4, Butterfly
    JNER R3, R1, LOOPI
    JNER R9, R10, FFT
  
```

圖 3-17：利用 SDMA 計算 FFT 運算

第四章 智慧型 DMA 控制器與 RISC 處理器的整合

智慧型 DMA 控制器搭配 RISC 處理器可以增強處理器的數位訊號處理效能。因此，本論文利用實驗室發展的一個 32 位元精簡指令集架構 (RISC) 處理器核心，並有快取記憶體控制器，與智慧型 DMA 控制器整合，為一個單核具有 DSP 平行架構的嵌入式處理器。以下章節將說明此精簡指令集架構處理器的特性、指令集、軟體開發環境及如何與智慧型 DMA 做整合。

4.1 RISC 處理器

實驗室發展的 RISC 訊號處理器擁有九級管線架構的設計[20][21]，為一個 Unit-core 和 Multi-Ram 的處理器 IP[22][23]。當一個指令透過程式計數器 (Program counter) 抓取進入處理器內部後，會先對該指令進行解碼，接著到暫存器提取所需要的資料到 ALU 中執行，最後將結果存回暫存器或記憶體中。周邊裝置的存取透過智慧型 DMA 來規劃，組譯器及處理器直接支援智慧型 DMA 的設定及使用[24]，周邊裝置(或 IP)可利用標準的 APB 規格掛上匯流排[25]，將外部的取樣訊號透過智慧型 DMA，將資料儲存在內部記憶體，或內部運算的結果送到周邊裝置上輸出[26]。

4.1.1 RISC 處理器核心

RISC 架構處理器擁有 9 級管線架構，9 級管線包含程式計數與跳躍判斷模組、指令抓取模組、程式碼記憶體模組、指令預測模組、指令解碼模組、暫存器模組、及算數邏輯單元模組，以下分別描述其主要的工作簡述如下：

1. PC Counter/Branch Protect：在硬體架構最上層，目的是把程式計數

器加一，並處理跳躍，最後決定程式計數器的值，並送到下一級指令抓取 (Instruction fetch)。

2. **Cache**：將 PC 位址傳到快取記憶體去抓指令，其中 PC 的位址線為 16-bit，傳回 RISC 指令為 32-bit，快取記憶體控制器架構為三級，因此若搭配此 RISC 訊號處理器，則總級數為九級。
3. **Instruction Fetch**：這一級的目的是把程式計數器的值，轉成程式記憶體的位址，送進下一級的程式記憶體去抓取指令。其中，由於有些指令在 ALU 級時會需要處理中指令的程式計數器值，所以必須把程式計數器值一級一級地傳下去，因此，程式計數器值會傳進下一級的暫存器。在處理跳躍指令時，為了避免浪費跳躍指令發生後一個指令被抓取，設定兩個訊號做為防止跳躍發生時的指標，用以表示現在的管線是否在 Stall 狀態，決定要不要執行該級動作。
4. **Program Memory**：將程式記憶體的位址經由硬體最上層的腳位傳到外部的唯讀記憶體去抓指令，其中位址線為 16-bit，傳回 RISC 指令為 32-bit。
5. **Instruction Decoder**：這一級的目的是把指令依照對應的指令碼去做解碼，指令中有兩個來源暫存器的位置，一個目的暫存器的位置。這些位置可以對下一級的暫存器組取得來源運算元並提供未來 ALU 級寫回的目的位置。內建兩組處理核心，所以也提供了指令去從兩個處理核心中互相提取對方暫存器組的資料，方便資料的直接交換。
6. **Register File**：處理核心有 32 個通用暫存器。設計有 13 個中斷暫存器來處理外部中斷，暫存器組為 2 讀 1 寫的格式，負責提供解碼器所解碼出的來源運算元值，並將 ALU 級算出的目的運算元寫回。在此模組中，有很多連接 ALU 級模組的輸出信號，目的是把這些 ALU 級要用到的信號經由管線級送往 ALU 級中，利用這些信號來完成 Data forwarding 的動作。

7. **ALU**: 本級功能是計算出邏輯或運算值，為了 Data forwarding 的實作，Data memory 及 Write back 這兩級隱含在 ALU 級內。Data forwarding 的機制是利用前面一級一級傳回的信號實作而成，目的是為了減少 RAW hazard。

除了以上基本的管線模組設計外，還有一些特殊硬體設計需求被強調出來 [27]，包含以下五種說明：

1. 一個指令週期完成乘加運算。
2. **Regular Loop Prediction**: 數位訊號處理運算中有很多固定次數迴圈的運算，利用一個良好的跳躍預測 (Branch prediction)，使處理器不會有 Control hazard 造成的不必要 Stall。
3. 良好的 **Data Forwarding** 機制。
4. **Condition Branch**: 預測採用 Prediction-untaken 方法設計。

4.1.2 RISC 處理器指令集

RISC 處理器指令集共分五大類：資料搬移、算數邏輯運算、跳躍指令、其他類指令、智慧型 DMAC 控制類，列表如下：

■ 資料搬移指令：

表 4-1：資料搬移指令列表

Instruction	Opcode	Example	Mode
MOVRC	000001	MOV rd, data	Direct
MOVRR	000010	MOV rd, rs	Reg-Reg
MOVVM	000011	MOV rd, address	Direct
MOVMR	000100	MOV address, rs	Direct
MOVMR	000101	MOV @rs2, rs	Indirect

MOVRRM	000110	MOV rd, @rs	Indirect
MOVB	101111	MOVB rd, base(rs)	Displacement
MOVI	110000	MOVI rd, rs1(rs2)	Index
MOVRRMB	110001	MOV rd, address	Direct-Second Ram
MOVMRB	110010	MOV address, rs	Direct-Second Ram
MOVRRB	110011	MOV @rs2, rs	Indirect- Second Ram
MOVRRMB	110100	MOV rd, @rs	Indirect- Second Ram

RISC 處理器共提供 Direct、Reg to Reg、Indirect、Displacement (base add)、Index 五大類的定址模式。

■ 算數與邏輯運算指令：

表 4-2：算數邏輯運算指令列表

Instruction	Opcode	Example
ADDRR	001000	ADD rd, rs1, rs2
SUBRR	001010	SUB rd, rs1, rs2
MULRR	001100	MUL rd, rs1, rs2
ADDRC	000111	ADD rd, data
SUBRC	001001	SUB rd, data
MULRC	001011	MUL rd, data
MACR	100111	MAC rd, rs1, rs2
MACC	110001	MAC rd, rs1, data
ANDRR	001110	AND rd, rs1, rs2
ORRR	001111	OR rd, rs1, rs2

XORRR	010000	XOR rd, rs1, rs2
INVR	010001	INV rd, rs

■ 跳躍指令：

表 4-3：跳躍指令列表

Instruction	Opcode	Example
JMP	010010	JMP address
JMPR	010011	JMP @rs
JBE	010100	JBE rs1, address
JNE	010101	JNE rs1, address
JMB	010110	JMB rs1, address
JLB	010111	JLB rs1, address
JBER	011000	JBER rs1, rs2, address
JNER	011001	JNBR rs1, rs2, address
JMBR	011010	JMBR rs1, rs2, address
JLBR	011011	JLBR rs1, rs2, address
CALL	100011	CALL address
RET	011110	RET

Address 的部分也可以是 Label，組譯器會自動轉換成對應的位址。

■ 其他指令：

表 4-4：其他指令列表

Instruction	Opcode	Example
SET	011100	SET A, rs
INTOK	011101	INTOK

SHR	100000	SHR rs
SHL	100001	SHL rs
Instruction	Opcode	Example
ENDC	011111	ENDC

SET 指令是當 RISC 處理器沒有連接匯流排時，利用這指令可以設定兩個 16-bit 的 I/O port，與外界溝通；INTOK 是處理軟體中斷的指令，利用這指令可發出軟體中斷；SHR 將 rs 向右移一位元；SHL 將 rs 向左移一位元。

■ 智慧型 DMA 控制相關指令：

表 4-5：智慧型 DMAC 控制指令列表

Instruction	Opcode	Example
SDMAD	100100	SDMAD data
SDMAR	100101	SDMAR rs
GDMA	100110	GDMA rd
DMAOK	101001	DMAOK
GDMAR	101110	GDMAR rd, address

4.2 軟體開發環境

硬體設計外，處理器的軟體支援是相當重要，為此發展了圖形化介面的組譯器 (Assembler)，提供機械碼 (Machine code) 的轉譯、程式記憶體 (Program rom) 的產生、及錯誤資訊 (Debug information)，讓使用者能夠利用以上資訊來除錯及產生 Testbench。

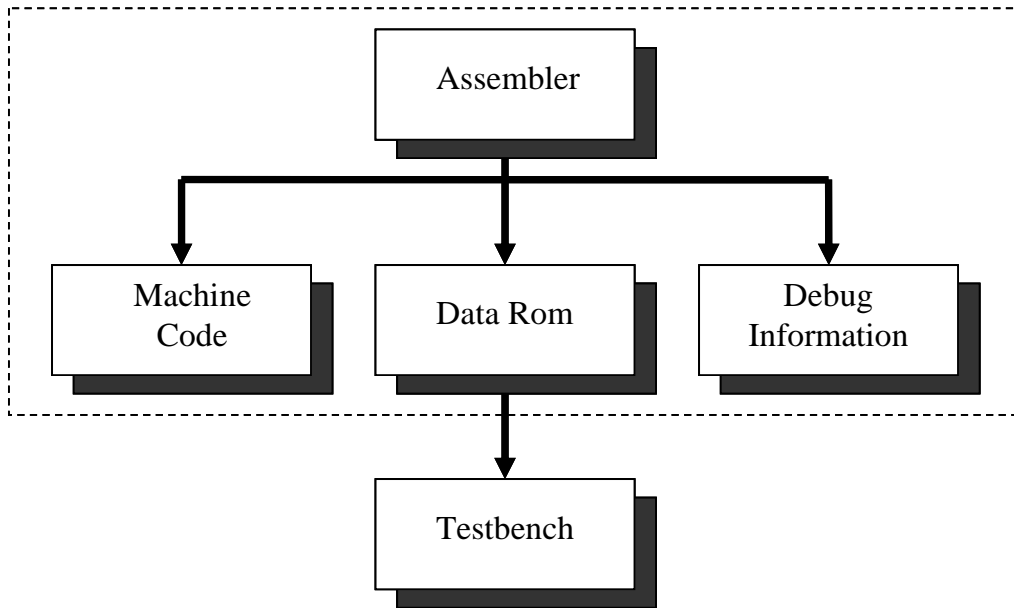


圖 4-1：組譯器（Assembler）的用途

圖型化介面組譯器如下圖，使用 Visual C++完成演算法部分，使用 VB做組譯器的圖形介面，並用 dll 作連結，並提供編輯檔案的能力。當完成編輯檔案後，須先經過 compile 的動作，確保無錯誤產生。最後經由 Build 的動作，產生所需的檔案：

- pop.txt：產生十六進位的程式碼，提供晶片測試的資料。
- bin.txt：產生程式記憶體體的資料，提供模擬及後續測試必備的資料。

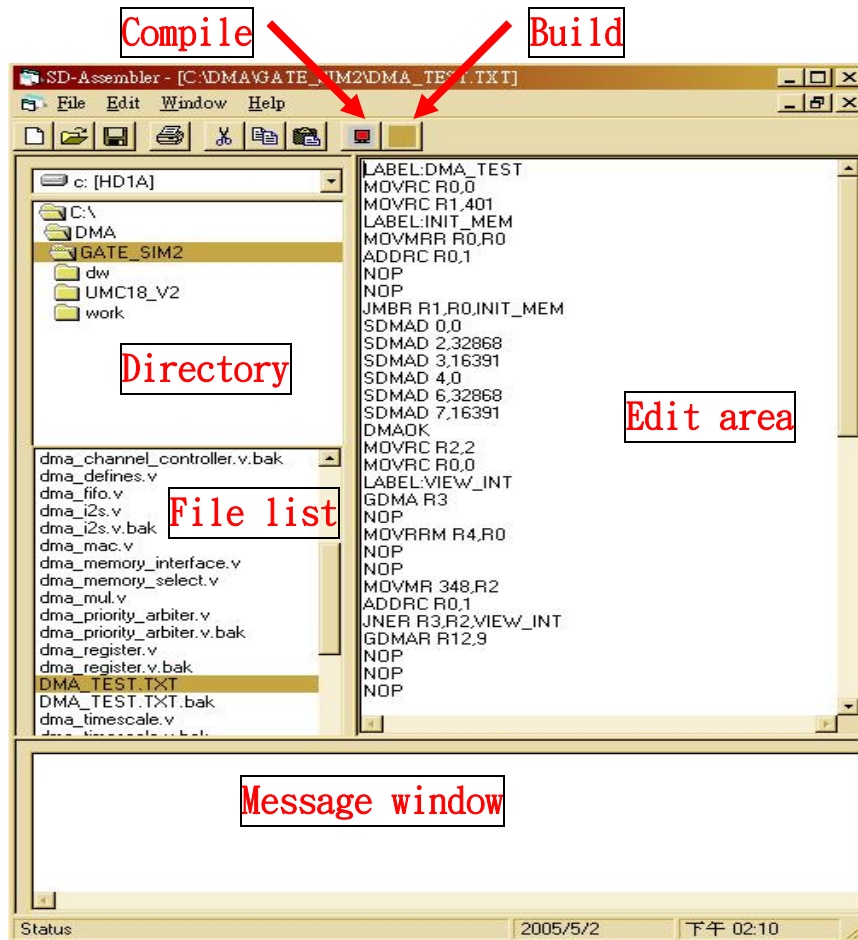


圖 4-2：圖形化介面組譯器 (Assembler)

4.3 整合

本論文將智慧型 DMA 和 RISC 處理器做一個整合，用以驗證智慧型 DMA 的功能，及整個系統的正確性，並驗證處理器加上智慧型 DMA 的功能，測試其效能。如圖 4-3，RISC 處理器整合智慧型 DMA 成為一個系統，擁有多個共用匯流排。程式記憶體為外接方式，內部兩個記憶體與智慧型 DMA 共用兩個資料匯流排。智慧型 DMA 並支援周邊匯流排，增加整體的擴充性。

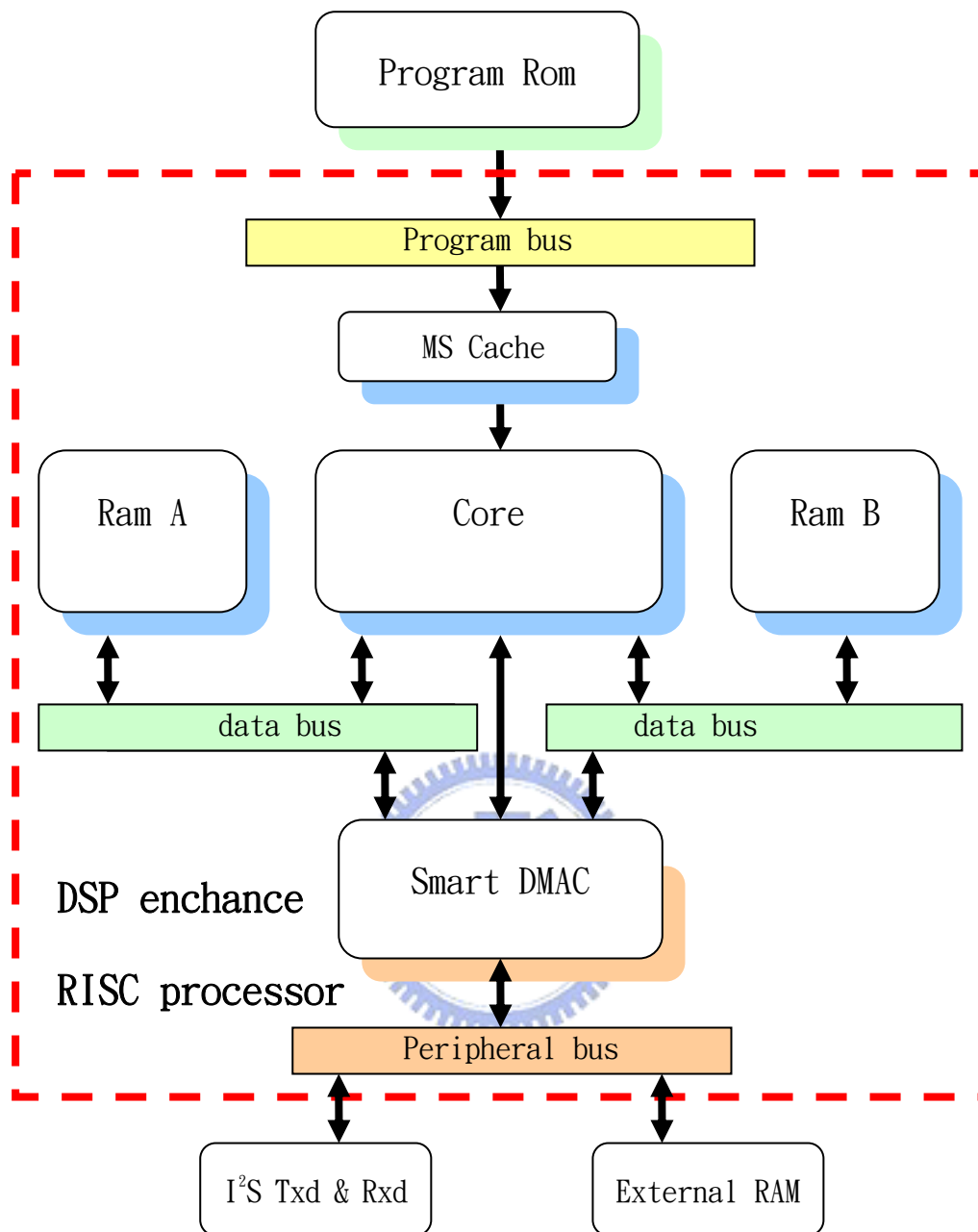


圖 4-3：RISC 處理器與智慧型 DMA 的整合

整個系統共有四條分開的匯流排，智慧型 DMA 與處理器共用一條資料匯流排，透過資料匯流排讓處理器與智慧型 DMA 使用兩個資料記憶體。資料匯流排的管制與衝突，由 RISC 處理器解決，當沒有使用到記憶體相關的指令，記憶體的主控權會釋放給智慧型 DMA，以解決衝突的情況。

除了共用匯流排外，為了設定智慧型 DMA 的動作，處理器必須存取智慧型 DMA 的暫存器。整合兩個 IP 的方式有兩種，第一，是從硬體電路上直接支援，有對應的指令解碼，存取智慧型 DMA 暫存器。第二，可以採用記憶體對映的方式，

更動電路較小，也不需增加額外的指令集，如下圖：

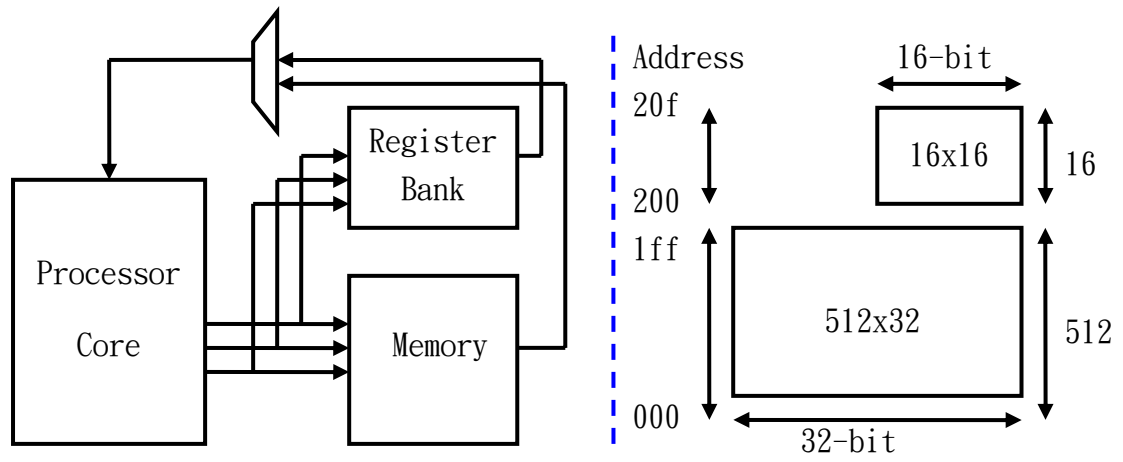


圖 4-4：RISC 處理器存取智慧型 DMA 暫存器-記憶體對映



第五章 驗證結果與晶片實現

本章詳述智慧型 DMA 整合在 RISC 處理器中，其功能上的驗證方法及結果。

5.1 設計驗證與應用結果

本論文的架構設計適合於語音和音訊訊號的處理，因為這種處理需要組合的數位訊號處理和控制功能。本節以 MELP 語音壓縮[9][10]當應用，在組譯器上寫一系列測試程式，驗證 RISC 處理器加上智慧型 DMA 的功能和效能，並分別針對演算法運算量集中的地方，進行加速，詳述如下：

5.1.1 MELP 語音壓縮介紹

語音傳輸是目前最主要也最普遍的通訊傳輸服務。在數位語音下的傳輸更有彈性，且能夠降低價格、維持品質、並提供保密的功能。由於使用者的增加與有限的頻寬。新的語音編碼傳輸位元率已由 8Kbps (CELP) 與 4.8Kbps (CS-ACELP) 發展至 2.4Kbps (MELP) [9]與 1.2Kbps (MELPe) [10]。也因為傳輸位元率的降低，語音品質就只能由更複雜的演算法來提升，這使得實現快速語音編碼相當困難。圖 5-1 比較了各個語音壓縮演算法的特徵，觀察 MELP 可知為壓縮率低且複雜性高的演算法。接著分析 MELP 各個單元的複雜性。

Speech compression algorithms' characteristics

Standard	kb/s	MOS	DRT	DAM	Seg. (ms)	MIPS	Adoption
PCM	128	4.5	-	-	0.125	0	
G.711 PCM	64	4.3	95	73	0.125	0.01	1972
G.726 ADPCM (G.721, G.723)	16,24, 32,40	4.1	94	68	0.125	2	1990 (1988,1988)
G.728 LD-CELP	16	4.1	-	-	0.625	19	1992
G.729 CS-ACELP	8	4.1	-	-	10	20	1995
G.729 CS-ACELP Annex A DSVD	8	4.1	-	-	10	10.5	1996
G.723.1	5.3, 6.4	3.51,4	-	-	30	16	1995
IS-641	8	4.0	-	-	10	14	1996
RPE-LTP (GSM)	13.0	3.51	-	-	20	6	1987
GSM EFR (ACELP)	12.2	4.1	-	-	20	15.4	1997
IS-54 VSELP (TIA)	7.95	3.51	-	-	20	13.5	1990
JDC VSELP	6.7	3.51	-	-	20	7.8	1990
IS-96 QCELP(TIA)	8/4/2/0.8	3.5	-	-	20	23/21/17/11	1993
IS-127 RCELP	8/4/0.8	4.1	-	-	20	20	1997
JDC/2 PSI-CELP	3.45	3.5	-	-	40	18.7	1993
GSM/2 VSELP	5.6	3.51	-	-	20	17.5	1997
USFS1016 CELP	4.8	3.1	91.0	64.6	30	17.1	1991
USFS1015 LPC10e	2.4	2.2	85.7	52.5	22.5	8.7	1984
MELP	2.4	3.1	92.1	64.5	22.5	20.43	1997

圖 5-1：語音壓縮演算法特性比較[28]

5.1.2 MELP 語音壓縮演算法複雜性分析

這裡針對壓縮率高但複雜性相對高的 MELP 編碼進行討論，利用本論文提出的 RISC 處理器加上智慧型 DMA 進行驗證與效能分析。圖 5-2 及圖 5-3 為 MELP 解碼與編碼複雜性分析，及編碼的複雜性分析，利用 Visual C 編輯軟體裡的系統時脈(System Clock)對 C 源程式碼，量測演算法各方塊執行的系統時脈周期數，最後除上 CLOCKS_PER_SEC 常數，就是執行的時間，單位為毫秒(ms)。

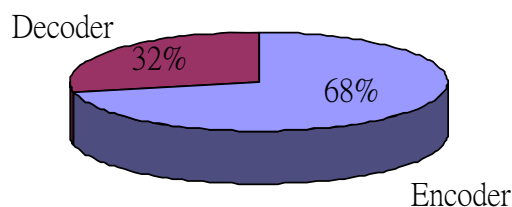


圖 5-2：MELP 編碼解碼比例

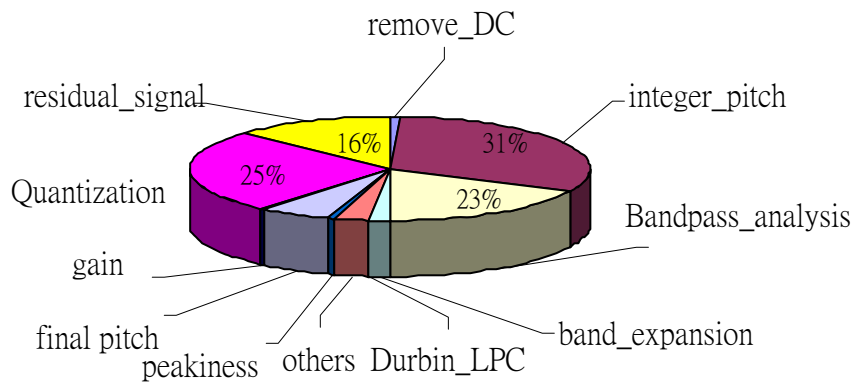


圖 5-3：MELP 編碼各單元複雜度

由圖 5-3 可見，Integer_Pitch 與 Quantization 方塊經分析各佔 31% 與 25% 的計算時間，為運算量最密集的两个地方，另外針對 Quantization 作分析，如圖 5-4 所示，可發現量化 LPC 係數(Q_LPC)與 FFT 各佔了 61% 與 18% 的 Quantization 計算時間。

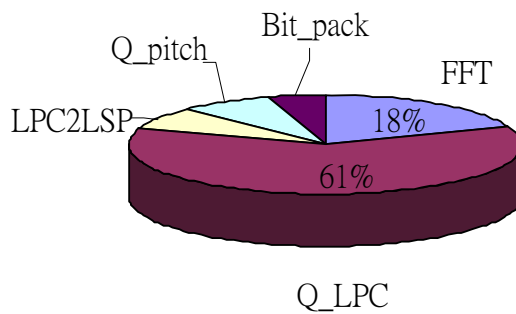


圖 5-4：Quantization 內各單元運算複雜度

針對 MELP 的這些運算密集的地方，利用智慧型 DMA 來進行最佳化，簡化其運算量，底下將對 Integer Pitch、Q_LPC 及 FFT 部分來進行效能分析與比較。

5.1.3 搜尋音高(Pitch Search)驗證

計算自相關函數 $r(\tau)$ ， $\tau=40, 41, \dots, 160$ ，在這 121 點中最大者即為所求的 Integer Pitch(P1)，其中 $r(\tau)$ 及 160 點自相關單元 C_τ 如下列式子：

$$r(\tau) = \frac{c_\tau(0, \tau)}{\sqrt{c_\tau(0, 0) * c_\tau(\tau, \tau)}} \quad , \quad c_\tau(m, n) = \sum_{-\lfloor \tau/2 \rfloor - 80}^{-\lfloor \tau/2 \rfloor + 79} s_{k+m} * s_{k+n}$$

Pitch search 裡包含了大量的自相關(autocorrelation)計算，在運算的過程中，需要將記憶體裡的值存到暫存器中，對於一個 160 點的自相關運算，需要對記憶體的資料搬移 320 次，且每個 C_τ 單元皆需要 160 個乘加器的運算。所以對於每個 P1 的決定，其中 $\tau=40、41、\dots、160$ ，所需的運算量 $(160-40)*3*160=57600$ 個乘法，與 $320*3*(160-40) = 115200$ 次資料搬移，如此大量資料搬移且運算規律的資料路徑，可以利用智慧型 DMA 作輔助運算，將與記憶體有規律且大量存取的單元，交給智慧型 DMA 作為輔助運算的功能，運作過程描述如下：

設定 SDMA 的定址模式：遞增(遞減)位址數、區塊起始位址、記憶體的來源位址與目的位址、開啟乘加器旗標。當做完設定後，智慧型 DMA 便會自動的進行如下動作：在從兩顆記憶體抓取兩個值進暫存器時，智慧型 DMA 便直接對他們做相乘的動作，放入累加器中。在 SDMA 自動的運算過程中，處理器可以進行其他功能的運作。如圖 5-5 所示。

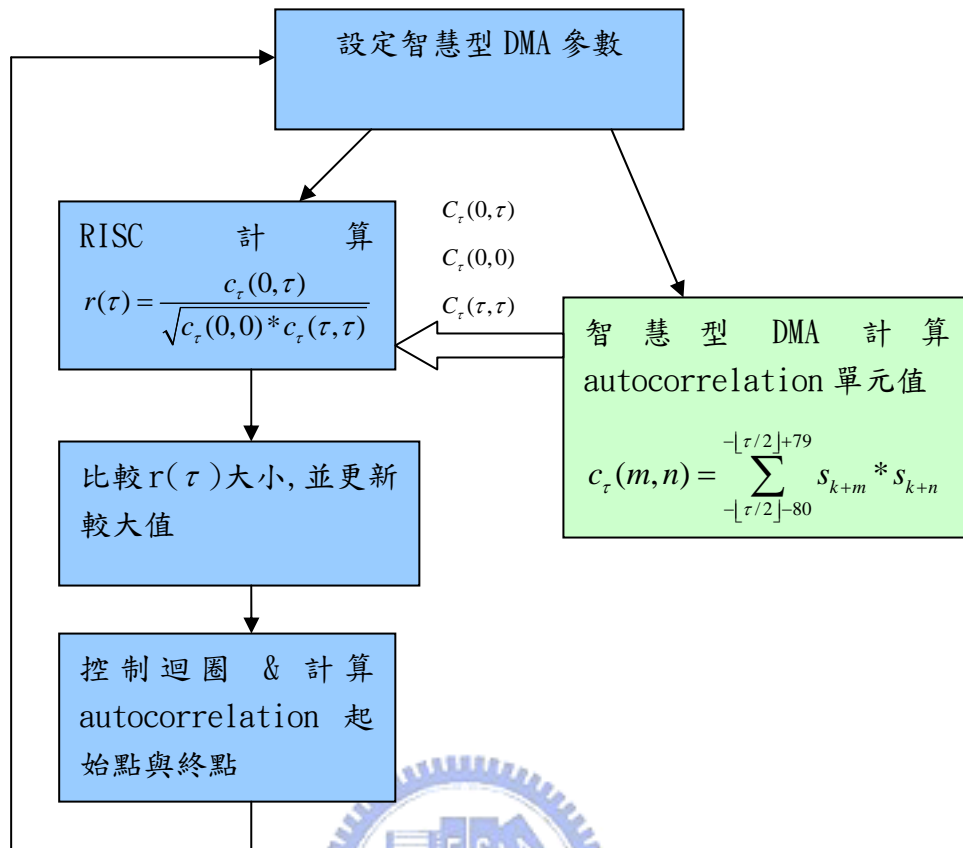


圖 5-5：處理器與智慧型 DMA 處理 Pitch 的分工情形

結果驗證：

搜尋 3 個自相關函式的值其中 $\tau=40$ 到 160，分析資料分佈的情形，利用 21 段線性曲線去近似開根號的曲線，最後相除求 $r(\tau)$ 的值比較大小，當結果一樣時進一步比較餘數。圖 5-6 及圖 5-7 所示為 Pitch Search 的情形，列出更新較大函式數值的情形(update bigger $r(\tau)$)。比較為定點化 C 程式與 post-sim 結果數值，驗證無誤。

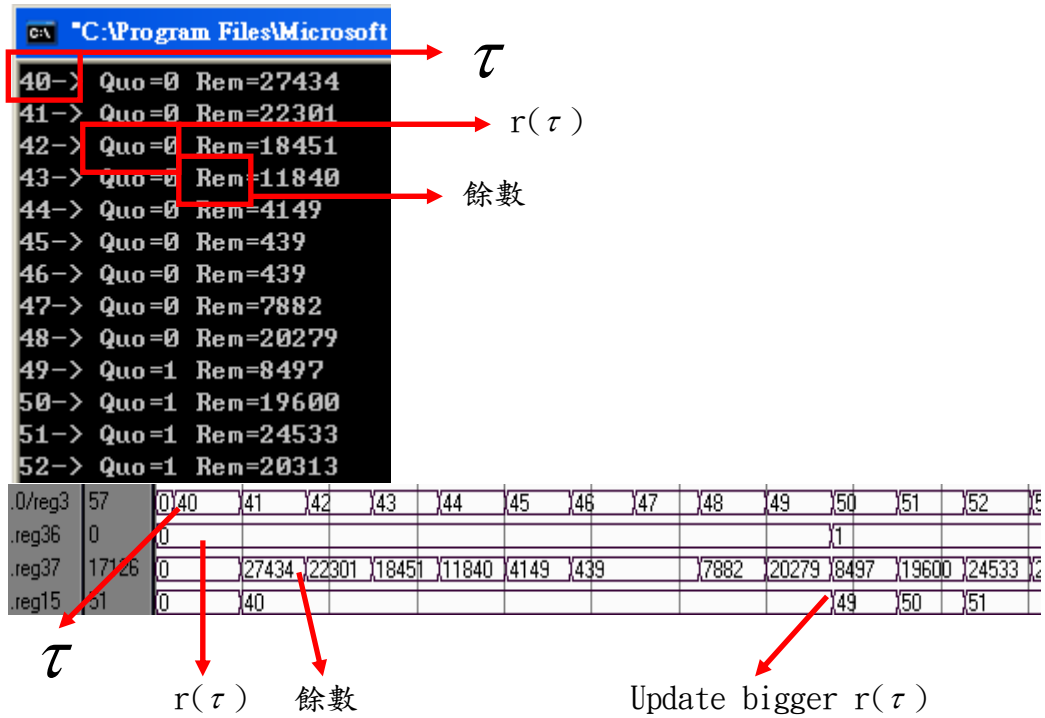


圖 5-6：Pitch Search C 程式與 Post-sim 結果 I

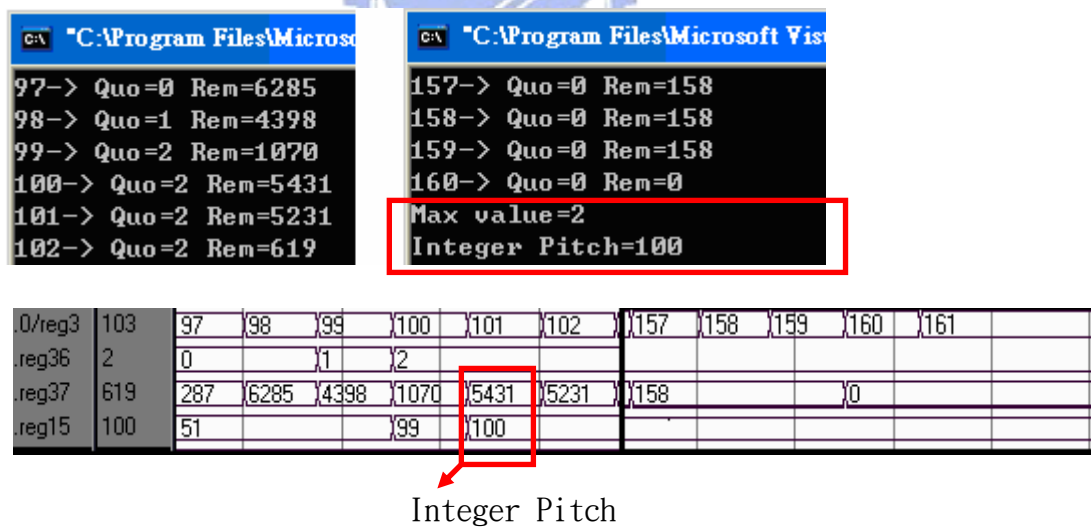
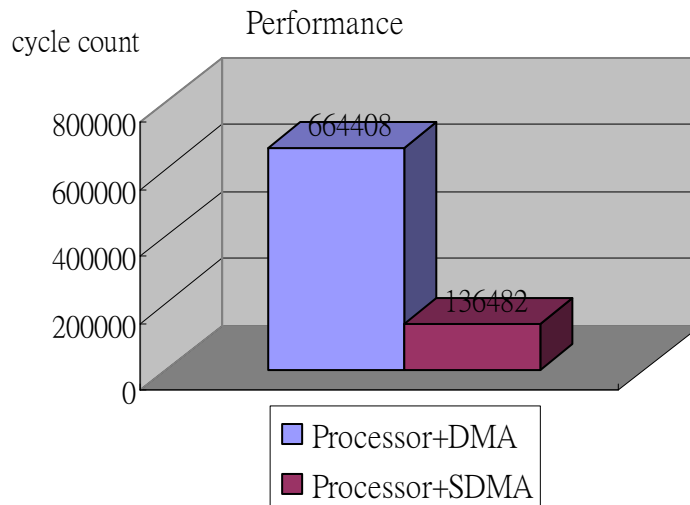


圖 5-7：Pitch Search C 程式與 Post-sim 結果 II

效能分析：經由智慧型 DMA 的加速，與處理器分工平行運算，可以化簡運算量 80%，如表 5-1 所示。

表 5-1：Pitch Search 效能表



5.1.4 量化線性預估係數(LPC)驗證

當量化線性預估(Linear prediction)係數時，MELP 採用多及向量量化法 (Multi-stage Vector Quantization)化簡編碼簿的大小，並使用 M-L 搜尋法加強 MSVQ 的精確度。

在實現方面，不管是用全域收尋(full search)，二元樹狀搜尋(binary search)或是多級(Multi-stage)的量化架構，無可避免的都必須對龐大的編碼簿(Codebook)作存取，在這之中會浪費很多時間在做資料搬移的工作，利用智慧型 DMA 在幫忙處理器搬移資料的同時，直接用硬體來處理向量量化裡 WMSE (weight mean square error)的計算，如此可節省處理器作長而規律的工作。

結果驗證：圖 5-8 為定點化 C 程式的模擬結果，與圖 5-9 相比，驗證無誤。

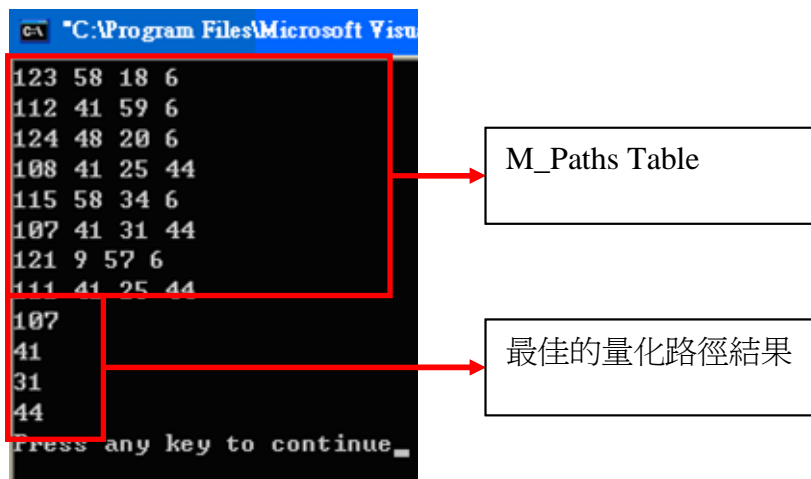


圖 5-8：多級向量量化 C 程式執行結果

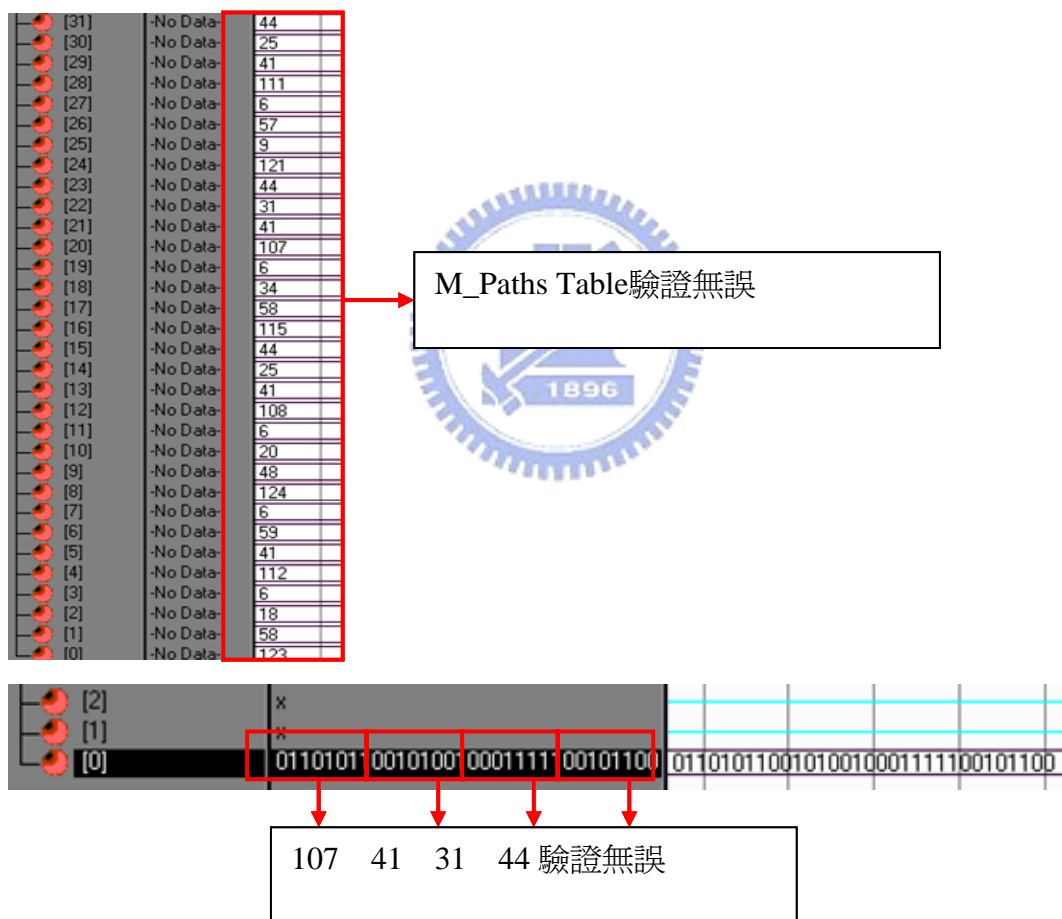
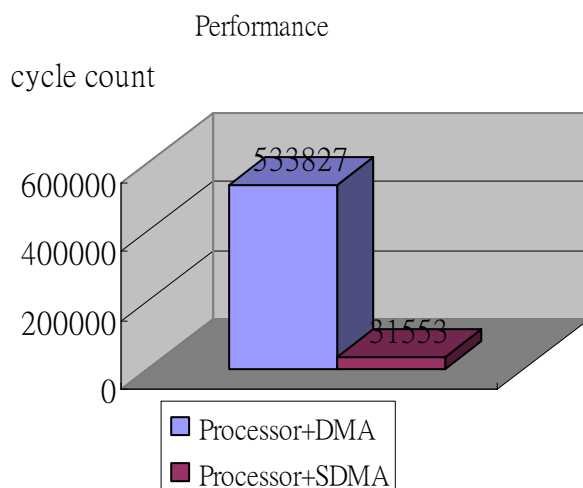


圖 5-9：多級向量量化 Post-sim 結果

效能分析：分析 Multi-stage Vector Quantization 組語，與記憶體相關的指令共有 53052 個，但可觀的是這些指令造成的額外負擔(overhead)，計算整個記憶體搬移的指令與 NOP 指令(overhead)佔整體運算量的 51%，換言之，多級向量量

化有超過一半的運算時間在做重複性的資料搬移。所以若利用智慧型 DMA 來搬移記憶體資料，在搬運的同時提供一條運算加權最小均方誤差(WMSE)的資料路徑，將可以避開這些運算累贅的地方，大幅提高效能，如表 5-2 所示，化簡了 94%的運算複雜度。

表 5-2：多級向量量化效能表



多級向量量化利用 M-L 搜尋法增加量化的精確度，如表 5-3 所示，M=1 與 M=8 SNR 相差約 2dB，M-L 搜尋法可找出最佳的量化路徑，使的誤差減到最低。

表 5-3：M Paths 量化路徑結果表

Schemes	Test Speech File (SNR)		
	TEST. PCM (335K)	TEST2. PCM (246K)	TEST3. PCM (500K)
MSVQ with M=1	55.43	56.065	55.873
MSVQ with M=4	56.92	56.695	57.168
MSVQ with M=8	57.38	57.197	57.303

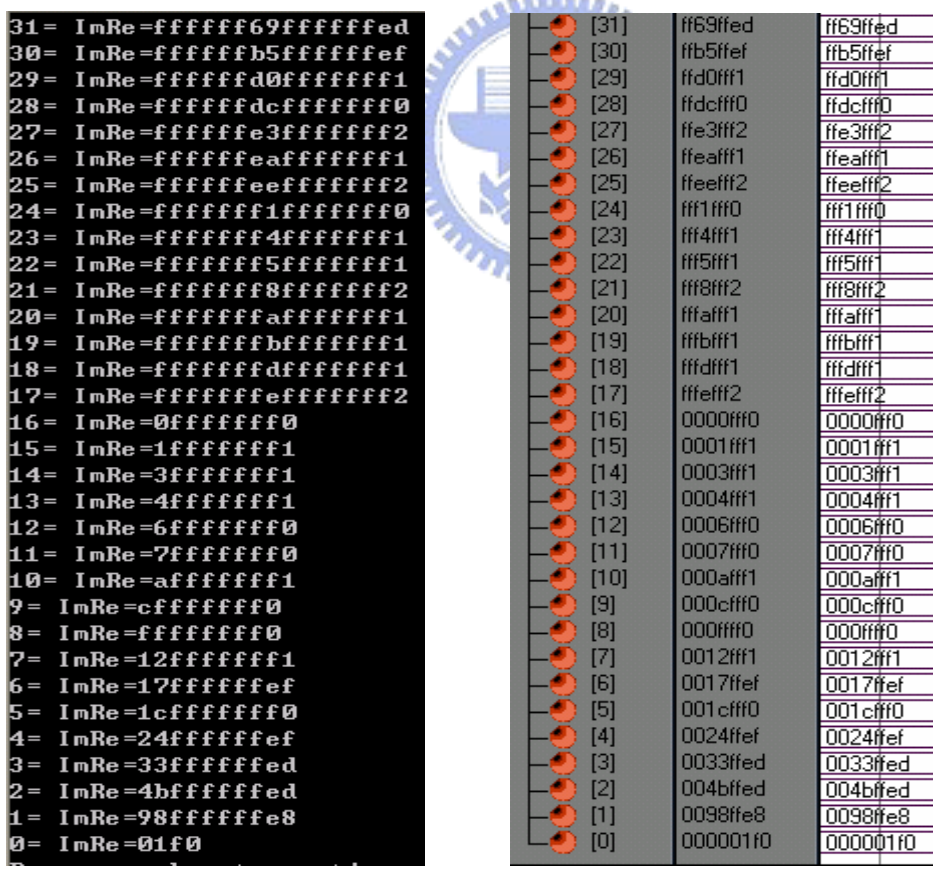
5.1.4 快速傅立葉轉換(FFT)驗證

MELP 對於前 10 個 prediction residual signal 的 pitch harmonics 值計算它的傅立葉值，這裏使用 512 點的 FFT。分析 FFT 的演算法，在進行運算前須

將係數作位元反轉(bit-reverse)的動作，利用智慧型 DMA 的位元反轉的定址模式，可以不經處理器快速的完成轉換。

另外，對於一個 2 點的蝴蝶運算，計算上需要進行複數的加法及乘法，並且重複進行從記憶體取資料出來，作蝴蝶運算運算，再存回記憶體，規律且重複性的做此動作。這裏設計了一個 Butterfly 單元，利用智慧型 DMA 裡的 ALU，增加一條計算路徑，利用 DMA 抓取資料的同時進行複數加法及乘法，再存回記憶體，如此只需要 8 cycles；而處理器只需要設定智慧型 DMA 的參數值，並控制 FFT 的流程，。分工排程將大幅提升計算能力。

結果驗證：底下驗證了 512 點 FFT 結果，列出前 32 點，分別為定點化 C 程式模擬與 Post-sim 結果，如圖 5-10，驗證無誤。



(a)C Simulator

(b)Post-sim 值

圖 5-10：512-FFT 各點數值表較

效能比較：

利用智慧型 DMA 搬移 FFT 的係數，搬移的效率為 $2*(n-1)$ ，其中 n 為 FFT 的點數，相較於利用處理器實現，智慧型 DMA 可以節省 96% 的運算量。

此外，經觀察一次兩點的蝴蝶運算光搬移資料與額外負擔就需要 28 個指令週期，複數運算與迴圈計算只有 17 個週期，所以一次的蝴蝶運算將會花超過一半的時間在等待資料。這裏利用智慧型 DMA 來處理，在搬移資料的同時利用內建的蝴蝶運算資料路徑，可以大幅減少因搬移資料所造成的額外負擔。如表 5-4 所示，進行 512 點 FFT 含搬移係數的準備動作，可以減少運算量約 76%。

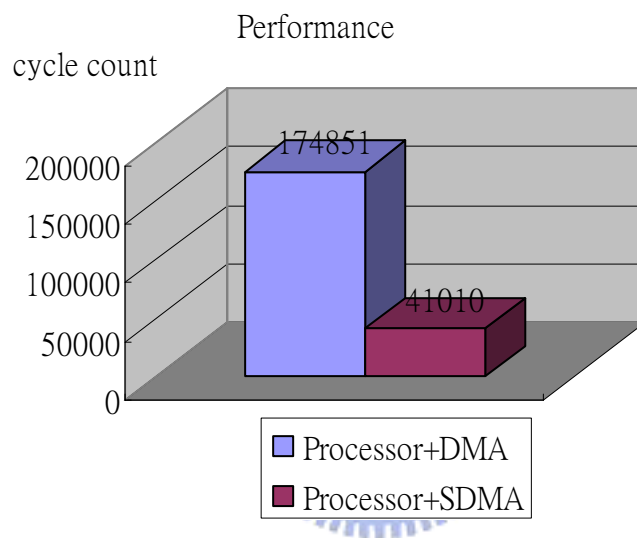


表 5-4：512-FFT 含位元反轉效能表

5.1.5 結論

利用智慧型 DMA 的定址模式，包括遞增遞減定址法、位元反轉定址法，與資料路徑配合內建的 ALU 使用，只要設定智慧型 DMA 的暫存器組，就可以幫助處理器進行各種數位訊號處理的運算。另外對於語音與影像常見的向量量化運算，MELP 演算法使用 4 級向量量化器與增加 8 條搜尋路徑，智慧型 DMA 加入向量量化的資料路徑單元，可以有效的處理 MELP 中 10 個線性預估參數的量化。表 5-5 表示經由智慧型 DMA 可以成功的簡化運算複雜度，所以處理器加入智慧型 DMA 不僅可以獨立進行記憶體資料搬移的功能，配合內建的運算單元更可以達到協助處理器運算的能力。

表 5-5：智慧型 DMA 化簡複雜度表

	Pitch Search	Quantize LPC	512 Complex FFT
Processor+DMA	664408	533827	174851
Processor+SDMA	136482	31553	41010
Reduction Ratio	80%	94%	76%



5.2 晶片製作

5.2.1 設計流程

依照標準的CIC Cell-Based Design Flow設計硬體，設計流程如下：

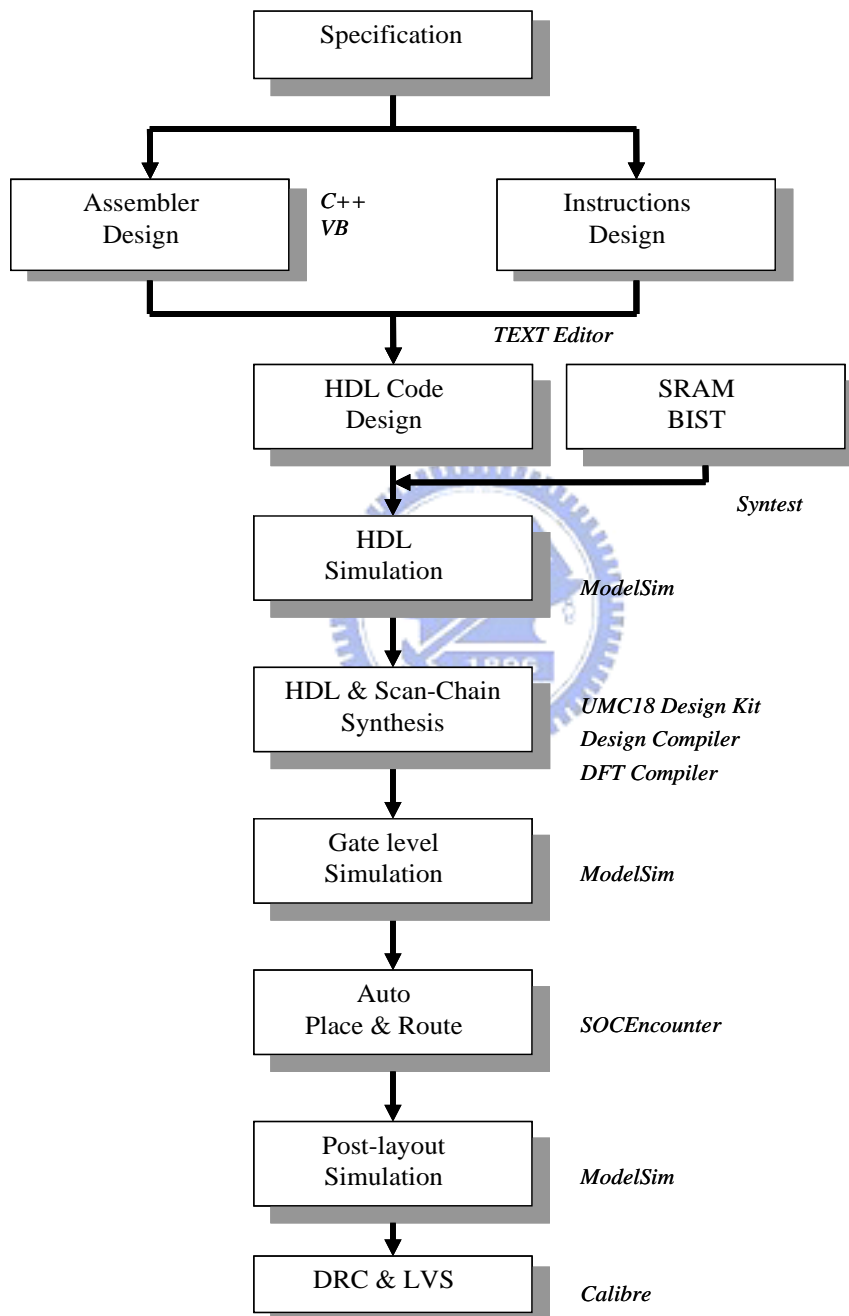


圖 5-11：晶片設計流程

5.2.2 合成結果

初期功能的設計，以 Verilog 硬體描述語言實作[29]，搭配 Mentor 公司出的 ModelSim 進行功能驗證。待功能驗證正確後，以 Synopsys 公司出的 Design Compiler 進行電路合成[30]，Library 使用 UMC 0.18um 製程。經由合成軟體 Design Compiler 合成後，其結果如下：

表 5-6：合成後的資訊

ITEM	Area (mm ²)	Timing	Fault coverage
Processor+SDMA	1355414.875	7.53 ns	97.35 %

5.2.3 佈局與封裝

採用 Cadence 公司出的 SOC Encounter 佈局軟體[31]，將合成後的電路放在晶片上，並依據速度的要求，自動最佳化並繞線，結果如下：

CHIP name : SD297

Technology : UMC 0.18um 1P6M CMOS

Package : 208 CQFP

Chip Size : 3.5× 3.5 mm²

Gate Count : 126K gate count

Power Dissipation : ~200mW

Max. Frequency : 128.5 MHz

使用 Prime Power 量測功率，跑驗證功能的測試程式，得到平均消耗功率約 400mW。如圖 5-12，為佈局及腳位圖，圖 5-13 為晶片打線圖，以 208-CQFP 的方式包裝。

而而在後續的佈局驗證部分，使用Calibre的DRC (Design Rule Check) 及 LVS (Layout VS Schematic) 也驗證無誤。

晶片設計規格如表 5-7。

表 5-7：晶片設計規格

Entry	Description
Architecture	9-stage pipeline and RISC
Clock(expectant/Pre/Post)	100MHz/102.3MHz/128.5MHz
Process / Supply	UMC 0.18 μ m 1P6M Mixed Signal / 1.8V
Power Consumption	200mW
Embedded memory	RAM(512x32)x2 RAM(1024x64)x2 RAM(1024x6)x2
Gate count / Die size	126k / 3.8 \times 3.8 mm ²
I/O Bus	AMBA Ver. 2.0 only APB
Audio Interface	I2S (Inter-IC Sound)/ External mem. on APB
DMA gate count	41k
DMA channels	2 (offer 8 I/O requirements)
DMA special function	ALU with Multiple Addressing and DataPath

5.3 效能比較

市面上有種類相當多的 DMA 控制器[31][32]，多數以 ARM 的 AMBA 匯流排為主，以 IP 化為訴求，因此，設計者多半為 IP 設計公司，強調在傳輸模式的支援，及連續傳輸的能力。ARM 本身對於 DMA 控制器設計，多了不連續資料 (LLI) 的傳輸方式。本論文的智慧型 DMA 控制器，與一般 DMA 控制器同，具有進行資料傳輸與多種傳輸模式的特性，此外還增加多樣的定址能力，包括位元反轉等，可以在資料被取出進行運算前，對資料先作搬移的動作。

表 5-8：與市面上 DMAC 做比較

	NCTU	FARADAY	GLOBAL UNICHIP	
	This Work	FTDMAC020	UAPC-5110	UAPC-5100
Channel	2	8	3	2(8)
Request	8	8	3	4(32)
Transfer Type	M-to-M M-to-P P-to-M P-to-P	M-to-M M-to-P P-to-M	M-to-P P-to-M	M-to-M M-to-P P-to-M P-to-P
Addressing	Incrementing Non-Increment Circuit Mirror Index-Based Bit-reverse	Chain Transfer	Incrementing Non-Increment Wrap-incrementing	
Operating Function	DIV Barrel shift MAC	None	None	
Data Path	Butterfly VQ	None	None	

另一個智慧型 DMA 的特點在於運算能力，內建 ALU 運算單元，與處理器分工，負責幫進行多個指令週期的運算，配合定址模式，可以處理各種數位訊號處理的運算，而內建的蝴蝶運算資料路徑與位元反轉定址法，大幅提升處理器運算 FFT 的效能，處理器結合智慧型 DMA 將可達到數位處理器晶片的運算能力。

為了應付語音與影像的應用，智慧型 DMA 處理器掛一個多級向量量化的 IP，由使用者可以設定量化器的參數，如 Order、Level、Stage 與 M_Paths 數，對於不同的處理情況可以加速處理器的運算。

以下各小節列出處理器結合智慧型 DMA 的資料傳輸效能比較、處理器結合智慧型 DMA 的資料運算效能比較，最後是處理器結合智慧型 DMA 和其他處理器效能的比較。

5.3.1 資料傳輸效能

比較 5 種不同傳輸狀況，使用處理器與一般 DMA 和處理器搭配智慧型 DMA 的結果。如表 5-9，其中 A 表示為第一個內部記憶體，B 表示為第二個內部記憶體。下列各種情況下，DMA 循序搬移資料所需的時間相當一致，而在使用不同定址模式的傳輸後，處理器必須花額外的時間去處理，因而造成在使用定址方式時，大幅降低傳輸效能。表 5-10 表示位元反轉(Bit-Reverse)的定址模式，橫軸代表反轉位元數，分別比較 6 種位元的情形，當反轉的位元數越高，相較於一般 DMA 搬移的效率越高。

表 5-9：資料傳輸效能表

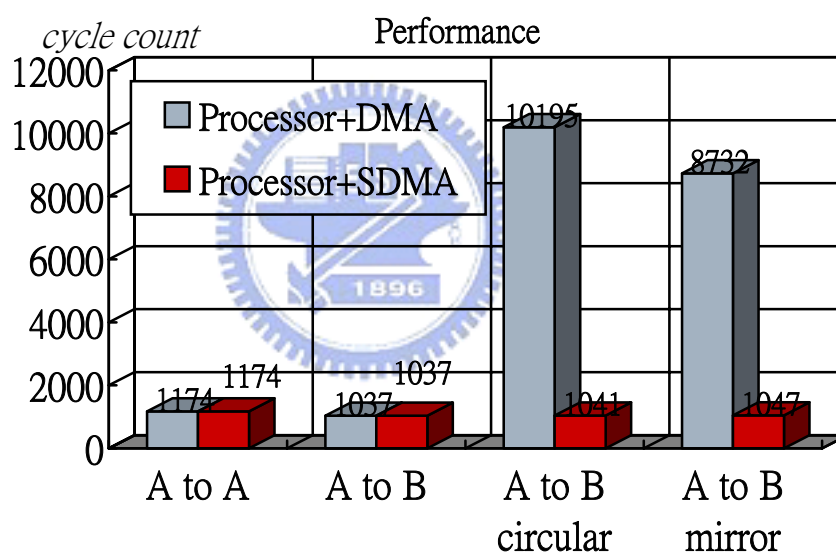
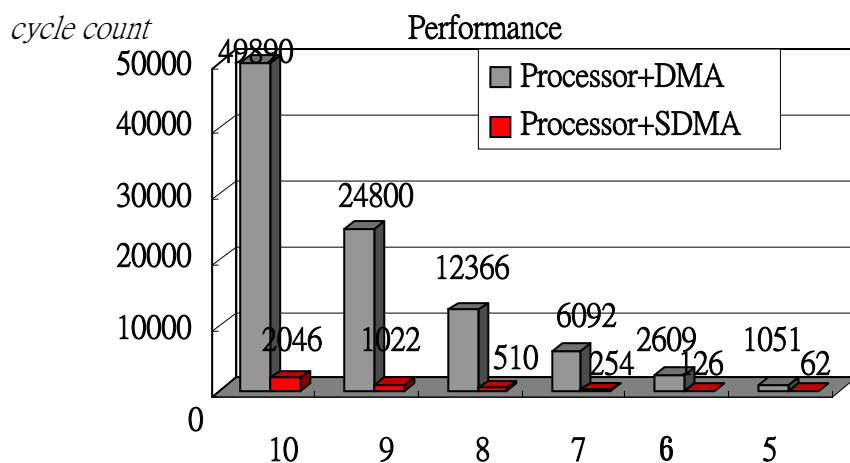


表 5-10：位元反轉資料傳輸效能表



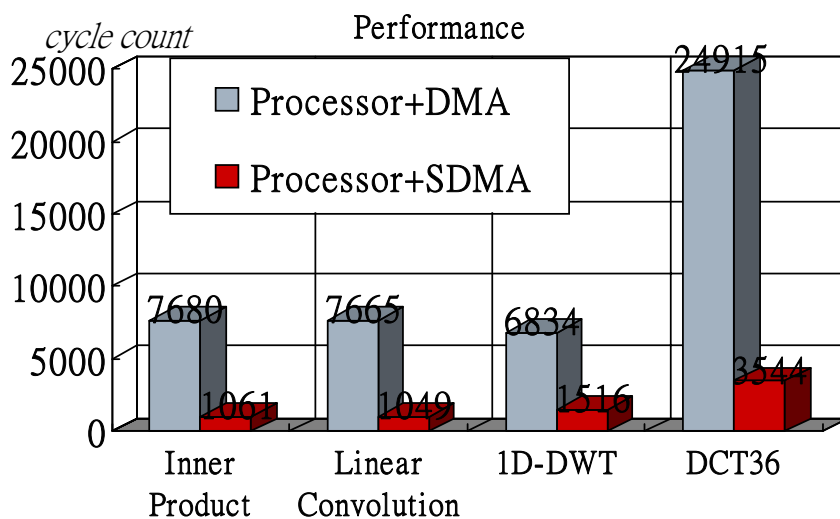
5.3.2 資料運算效能

以下將比較 3 種資料運算的效能。分別為利用智慧型 DMA 定址法搭配內建的乘加器進行數位訊號運算、利用智慧型 DMA 的蝴蝶運算資料路徑與位元反轉定址模式輔助運算實現 FFT，與多級向量量化器加速的效能比較。

■ 數位訊號運算

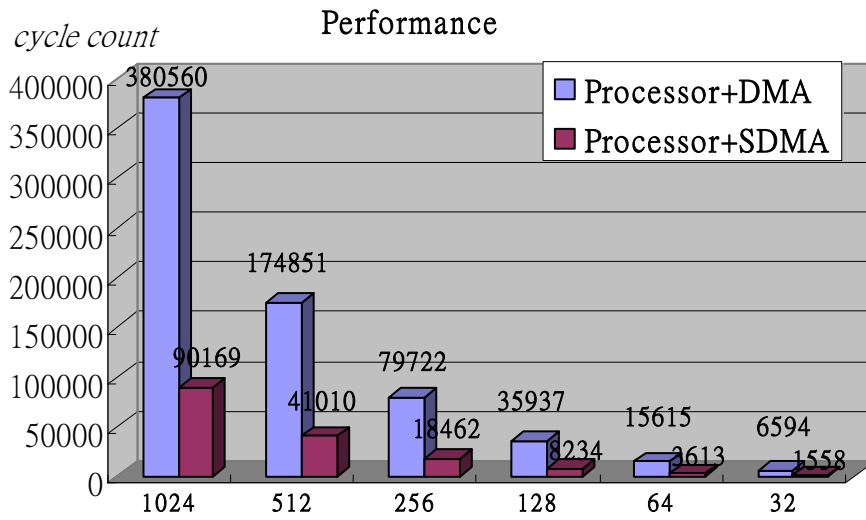
相較於處理器用一般的 DMA 進行資料搬移後再計算，運算量大幅減少，如表 5-11 所示。

表 5-11：資料運算效能表



■ 各點 FFT 數值

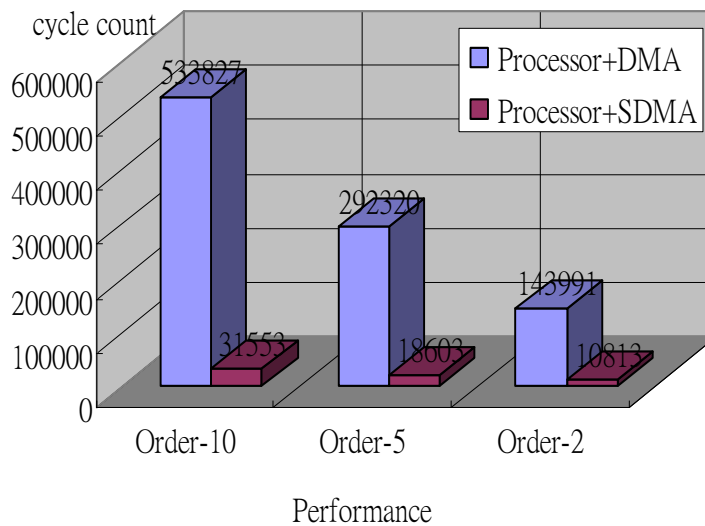
表 5-12：FFT 效能比較表



■ 多級向量量化比較

表 5-13 所示，當量化的 Order 數越高，運算管線的效用越大。







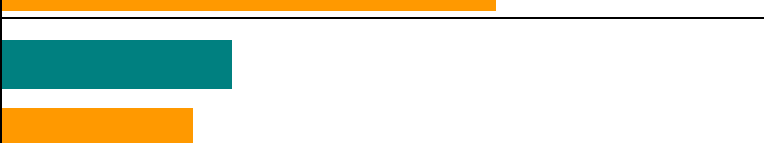


表 5-13：多級向量量化效能比較表



5.3.3 其他處理器比較

如表 5-14 所示，與其他處理器比較 256 點 FFT 效能與速度。

表 5-14：多級向量量化效能比較表[33]

Device	Speed MHz	256-point FFT benchmark
ADSP-218X	75	
DSP16410	170	
TMS320C54X	160	
' 320VC549	100	
ARM7TDMI/ Piccolo	70	
DSP1620	120	
This work	125	
Execution time		20 40 60 80 100 120 140 160 180 200 (us)
Execution cycle		2K 4K 6K 8K 10K 12K 14K 16K 18K 20K (cycles)

另外與 TIC54x 的規格比較如表 5-15。

表 5-15：與 TI 的規格比較表

	TIC54X	Proposed Design
S-P - Com MAC	8 cycle	1 cycle
S-P - Real MAC	1 cycle	1/2 cycle
R2 Butterfly	8 cycle	9 cycle *
S-P Com Bit-Revise	3 cycle	2 cycle
Convolution	1 cycle	1/2 cycle
Hardware Accelerators	Image/Video Extension	MSVQ Paths
External RAM Type Support	Async SDRAM	SRAM

*: include Scale shift & load/store memory

與其他 paper[2] 比較，處理器加上智慧型 DMA 後效能已逼近且成本非常低。

表 5-16：效能與成本比較表

ITEM	50-taps FIR 100 samples	50-taps complex FIR, 100 samples	1K Complex FFT	Gate Count
This Work	11200 (cycles)	24000 (cycles)	53427 (cycles)	136K (Include VQ IP)
A 32-b RISC/DSP Microprocessor[2]	12200 (cycles)	22000 (cycles)	45000 (cycles)	210K

第六章 結論

通用的 32-bit RISC 處理器與平行的 16-bit/32-bit DSP 整合，為一個兼顧成本與效能單核心處理器的解決方案，本論文提出語音導向的智慧型 DMA 搭配處理器可成為 RISC/DSP 平行處理的架構，掛載的語音向量量化加速器符合 IP 重覆使用的特性，對特定運算可大幅提升處理器的效能。智慧型 DMA 擁有傳輸與運算的模式，傳輸方面從周邊到記憶體擁有多種傳輸組合，並有多種的定址模式提升傳輸效率；在運算方面內建 ALU，能處理複數乘法、柱狀位移等資料路徑，且指令集模式符合 RISC 架構，不影響處理器管線的運作。

智慧型 DMA 的特色在於能結合傳輸與運算的功能，在計算中有效率的安排資料與運算，提高處理器的處理的效率。以下說明三種合作方式：

- (1) 利用智慧型 DMA 傳輸資料的定址法，與 ALU 的乘加器合作，可進行固定迴圈的乘累加運算。
- (2) 利用智慧型 DMA 的位元反轉傳輸模式，在進行快速複立葉轉換前先安排好係數，再經由智慧型 DMA 裡 ALU 的資料路徑作蝴蝶運算。
- (3) 進行編碼簿搜尋時，利用智慧型 DMA 對外部記憶體周邊的溝通能力，協助向量量化加速器傳輸編碼簿，進行加權均方誤差的計算。

本論文提出的 RISC/DSP 處理器特別適合於語音的運算，語音的運算常需要對訊號做頻譜轉換來進行分析；另外也常用線性預測模型來合成語音的訊號，對此，這個處理器特別適合作以上兩個動作，如 FFT 及 FIR 的實現，可用在語音壓縮、語音辨識等。

為了應付其他語音應用或影像處理，智慧型 DMA 可以針對演算法掛載特定的硬體加速器 IP，因此未來可增加一條高速的匯流排如 AHB bus，各種加速 IP 與外部記憶體皆可透過匯流排與處理器內部作溝通，由智慧型 DMA 掌管優先權順序。另外在智慧型 DMA 與記憶體處理能力上，為了成本導向的目標，未來在智慧

型 DMA 內可增加處理動態隨機存取記憶體的單元(DRAM)，相較於靜態隨機存取記憶體(SRAM)，需支援高低位址的資料交互讀取的機制，雖然在速度上不如 SRAM，但擁有高密度成本低的優點。

本論文設計與一顆通用 RISC 處理器做整合，並應用在 MELP 語音壓縮演算法上，能解決 MELP 三個運算集中的地方，成功減少運算量 70%以上。本設計將在國家晶片中心(CIC)下線，未來此顆晶片可用做低成本的單核心 RISC/DSP 處理器來使用，也可以 IP 的方式，將系統整合起來，成為一個 SOC 的系統。



參考文獻

- [1] Vijay K. Madiseti, "VLSI Digital Signal Processors: An Introduction to Rapid Prototyping and Design Synthesis," *IEEE Press*, 1995.
- [2] Michael Dolle, Satwinder Jhand, Walter Lehner, Otto M'uller and Manfred Schlett "A 32-b RISC/DSP microprocessor with reduced complexity," *IEEE Journal of Solid-State Circuits*, vol. 32, no. 7, 1997.
- [3] Dave Comiskey, Sanjive Agarwala, and Charles Fuoco, "A scalable high-performance DMA architecture for DSP application," *Proceedings of the IEEE International Conference on Computer Design (ICCD)*, pp. 414-417, 2000.
- [4] Luca Breveglieri and Luigi Dadda, "A VLSI inner product macrocell," *IEEE Trans. on VLSI*, vol. 6, no. 2, pp. 292-298, 1998.
- [5] Vassilios A. Chouliaras and Jose Nunez, "Scalar Coprocessors for Accelerating the G723.1 and G729A Speech Coders," *IEEE Transactions on Consumer Electronics*, vol. 49, no. 3, 2003.
- [6] V.A.Chouliaras, J.L. Nunez, K. Koutsomyti, D.J. Mulvaney and S. Datta, S.R. Parr, "Development of custom vector accelerator for high-performance speech coding," *Electronics Letters 25th*, vol. 40, no. 24, 2004.
- [7] Han Qi, Zheng Jiang and Jia Wei, "IP reusable design methodology," *ASIC, 2001. Proceedings. 4th International Conference*, pp. 756-759, 2001.
- [8] 戴顯權編著,資料壓縮, 紳藍出版社, 2002.
- [9] L. Supplee, R. Cohn, J. Collura, and A. McCre., "MELP: The New Federal Standard at 2400 bps," *IEEE ICASSP-97 Conference*, pp.1591-1594, 1997.
- [10] Tian Wang, Kazuhito Koishida., Vladimir Cuperman, Allen Gersho and John S. Collura, "A 1200/2400 bps coding suite based on MELP," *Speech Coding, 2002*,

IEEE Workshop Proceedings, pp.90-92, 2002.

- [11] Grant Davidson, Allen Gersho, “Application of a VLSI Vector Quantization Processor to Real-Time Speech Coding,” *IEEE Journal on Selected Areas in Communications*, vol. sac-4, no. 1 , 1986.
- [12] Chin-Liang Wang and Ker-Min Chen , “A New VLSI Architecture for Full-Search Vector Quantization,” *IEEE Transactions on Circuit and Systems for Video Technology*, vol. 6, no. 4, 1996.
- [13] Chen-Yi Lee, Shih-Chou Juan and Yen-Juan Chao, “Finite State Vector Quantization with Multipath Tree Search Strategy for Image Video Coding ,”*IEEE Transactions on Circuit and Systems for Video Technology*, vol. 6, no. 3, 1996.
- [14] Massimiliano Bracco, Sandro Ridella and Rodolfo Zunino, “Digital Implementation of Hierarchical Vector Quantization,” *IEEE Transactions on Neural Networks*, vol. 14, no. 5, 2003.
- [15] F. Lahouti and A. K. Khandani, “Reconstruction of Multi-Stage Vector Quantized Sources Over Noisy Channels- Applications to MELP Codec Technical Report,” Department of Electrical & Computer Engineering University of Waterloo Waterloo, Ontario, Canada, 2004.
- [16] Alan V. Oppenheim, Ronald W. Schaffer, and John R. Buck, *Discrete Time Signal Processing*, 2nd Edition, Prentice Hall, 1999.
- [17] ARM Ltd, AMBA Specification, rev. 2.0, <http://www.arm.com>, 1999.
- [18] B.H. Juang and A.H. Gray, “Multiple stage vector quantization for speech coding,” *Acoustics, Speech, and Signal Processing, IEEE International Conference on ICASSP '82*, vol. 1, pp. 597–600, 1982.
- [19] W.P. LeBlanc, B. Bhattacharya, S.A. Mahmoud, V. Cuperman, “Efficient search and design procedures for robust multi-stage VQ for LPC parameters for 4 kb/s

- speech coding,” *IEEE Transactions Speech Audio Process.*, vol 1, no.4, pp.373-385, 1993.
- [20] John L. Hennessy and David A. Patterson, *Computer Architecture*, 3rd Edition, Morgan Kaufmann, 2003.
- [21] John L. Hennessy and David A. Patterson, *Computer Organization & Design : The Hardware / Software Interface*, 2nd Edition, Morgan Kaufmann Publishers, 1998.
- [22] Bill S.-H.Kwan, Bruce F.Cockburn, and Duncan G. Elliott, “Implementation of DSP-RAM: architecture for parallel digital signal processing in memory,” *Proceedings of IEEE Canadian Conference on Electrical and Computer Engineering*, pp. 341-346, 2001.
- [23] Corinna. G. Lee and Mark. G. Stoodley, “Simple vector microprocessors for multimedia applications,” *Proceedings of 31st Annual ACM/IEEE International Symposium*, pp. 25-36, 1998.
- [24] Asawaree Kalavade and Edward A. Lee, “A hardware-software codesign methodology for DSP application,” *IEEE Design & Test of Computers*, vol. 10, pp. 16-28, 1993.
- [25] Steve Fuber, *ARM System-on-Chip Architecture*, 2nd edition, Addison-Wesley Professional, 2000.
- [26] Hans-Joachim Stolberg, Mladen Berekovic, Lars Friebe, Sören Moch, Mark Bernd Kulaczewski and Peter Pirsch, “HiBRID-SoC: A Multi-Core System-on-Chip Architecture for Multimedia Signal Processing Applications,” *Proceedings of International Conference on Very Large Scale Integration of System-on-Chip*, pp. 155-160, 2003.
- [27] Jae Sung Lee and Myung H. Sunwoo, “Design of new DSP instructions and their hardware architecture for high-speed FFT,” Kluwer Academic Publishers, pp.

247-254, 2003.

- [28] Markovic, M.Z., “Speech compression - recent advances and standardization,” *Telecommunications in Modern Satellite, Cable and Broadcasting Service, 2001. TELSIS 2001. 5th International Conference*, vol.1, pp.235 – 244, 2001.
- [29] Donald E. Thomas and Philip Moorby, *The Verilog Hardware Description Language*, Kluwer Academic Publishers, 1994.
- [30] Pran Kurup, et al., *Logic Synthesis Using Synopsys*, 2nd Edition, Kluwer Academic Publishers, 1997.
- [31] Nian Shyang Chang, *Cell-Based IC Physical Design and Verification*, Chip Implementation Center, July, 2004.
- [32] GlobalUnichip, UAPC-5110 DMA Controller Lite,” <http://www.globalunichip.com.tw>, 2002.
- [33] W.Smith and J.Smith, *Handbook of Real-Time Fast Fourier Transforms*. New York: IEEE Press, 1995

